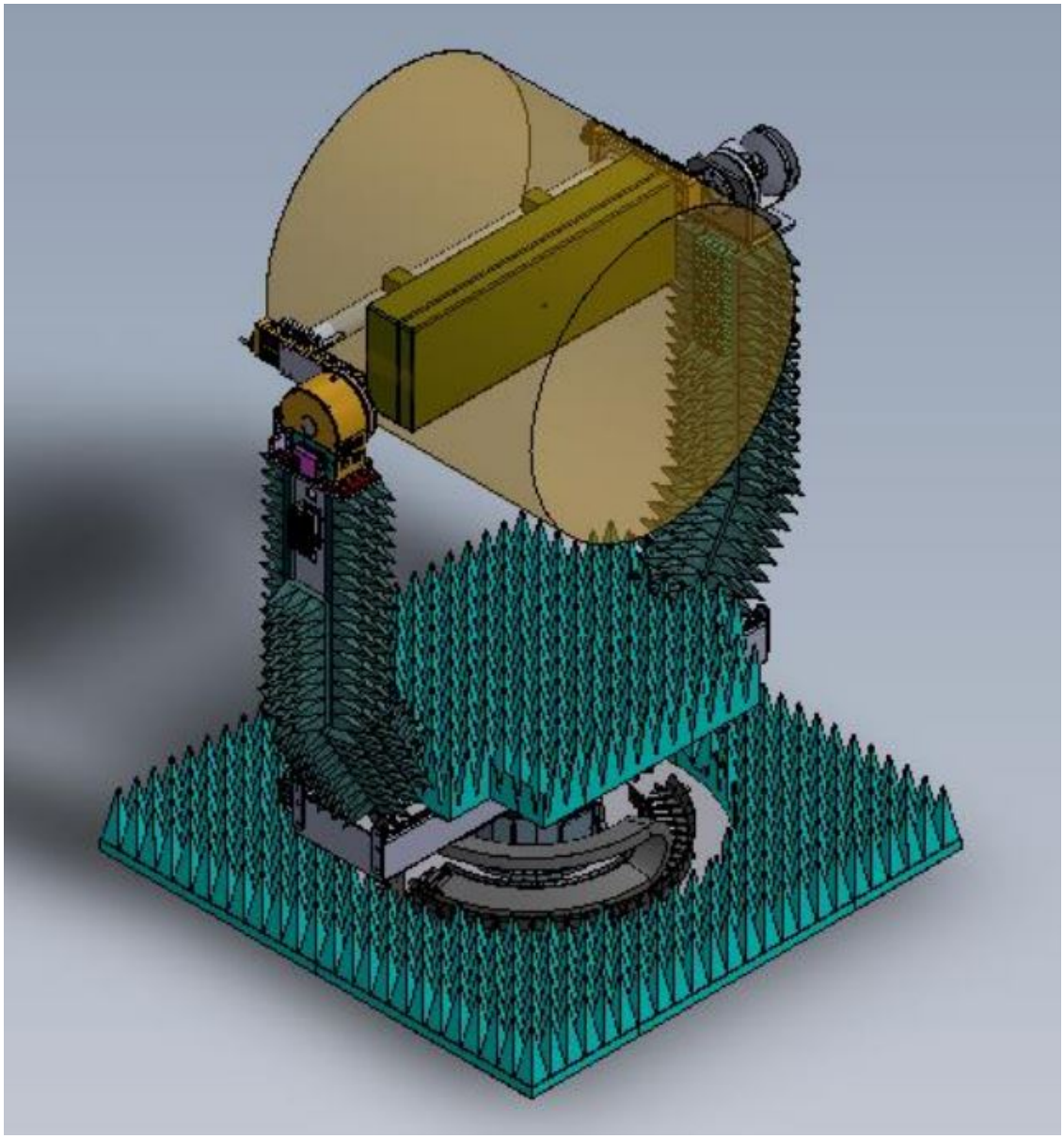
CHALMERS
UNIVERSITY OF TECHNOLOGY



Spherical Probing of Quiet Zone

Master's thesis in Electrical engineering

Filip Aldebrink
Rikard Carlsson

Department of Electrical engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024

Master's thesis 2024

Spherical Probing of Quiet Zone

Filip Aldebrink
Rikard Carlsson

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical engineering
Chalmers University of Technology
Gothenburg, Sweden 2024

Spherical Probing of Quiet Zone
Filip Aldebrink
Rikard Carlsson

Supervisor: Jian Yang, Electrical Engineering
Advisor: Anders Jernberg, Ericsson AB
Examiner: Jian Yang, Electrical Engineering

Master's Thesis 2024
Department of Electrical engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 (0)31-772 10 00

Cover: Quiet Zone volume (yellow) at the barbeque. Courtesy of Ericsson AB.

Typeset in LaTeX
Gothenburg, Sweden 2024

Spherical Probing of Quiet Zone
Filip Aldebrink & Rikard Carlsson
Department of Electrical engineering
Chalmers University of Technology

Abstract

This report furthers research antenna development, focusing on the crucial aspect of field probing in Compact Antenna Test Ranges. Traditional methods of field probing using Field Probe Scanners are examined, highlighting their limitations in terms of complexity, time consumption, and the inability to capture the full testing zone accurately. The study aims to explore the feasibility and advantages of employing Spherical Probe Scanning as an alternative to Field Probe Scanner-based techniques. By utilizing Spherical Probe Scanning, the report aims to streamline the measurement process, enhance accuracy, and enable comprehensive evaluation of antenna performance within a designated sphere. The methodology applied to process the measurement data, probe compensation and to analyze the Quiet Zone, which is of interest when using Field Probe Scanner or Spherical Probe Scanning, are presented. Relevant figures of the result, both theoretical and real measurements, and findings during the thesis, are discussed. The method at this stage of development did not allow the Quiet Zone to be analyzed from measurements, only from theoretical data. The problem was most likely the probe compensation, but the method shows promising features to be able to analyze the Quiet Zone using Spherical Probe Scanning.

Keywords: Quiet Zone, Spherical Probe Scanning, Spherical Wave Expansion, Probe Compensation, CATR, Anechoic Chamber, Field Probe Scanner, Ericsson.

Acknowledgements

We would like to thank our supervisor Anders Jernberg at Ericsson AB for all the help and support during our Master's thesis.

We would also like to thank Jonas Fridén for all the input and expertise on Spherical Probe Scanning and MATrix LABoratory.

Many thanks to Jian Yang, our supervisor and examiner at Chalmers University of Technology, for the guidance and support during our Master's thesis.

Special thanks to Dan Roos, who has done the measurements in collaboration with Anders Jernberg and has given us insight into how the chamber and the measurements worked.

Thanks to Jean-Marc Pfeiffer, and the rest of Ericsson AB, for enabling us to do the Master's thesis and for all the support and help given by our colleagues at Ericsson AB.

Filip Aldebrink & Rikard Carlsson, Gothenburg, 2024-06-14

Acronyms

AUT Antenna Under Test. 11–13, 23, 61, 62

CATR Compact Antenna Test Range. v, xvii, xviii, 1, 2, 9, 11, 12, 14, 18, 23, 24, 26, 27, 29, 51, 52

DUT Device Under Test. 1, 11

FPS Field Probe Scanner. v, xvii, 1, 2, 12, 30

GNSS Global Navigation Satellite Systems. 2

LHS Left Hand Side. 5

MATLAB MATrix LABoratory. vii, xvii, 2, 12, 13, 23, 25, 26, 29, 31, IX

MVG Microwave Vision Group. xviii, xix, 23, 25, 27, 52–56, 61

OEW Open Ended Waveguide. 23, 25, 29, 63

QZ Quiet Zone. v, 1, 2, 11–13, 23, 29, 30, IX

RF Radio Frequency. 11

SPS Spherical Probe Scanning. v, vii, xvii, 1, 2, 12, 13

SWE Spherical Wave Expansion. v, xvii, xviii, 13, 21, 22, 29, 49, 50, 62

VNA Vector Network Analyzer. 11

Nomenclature

Physics Constants

c_0	Speed of light in a vacuum	$299\,792\,458\,\mathrm{m/s}$
ϵ_0	Vacuum permittivity	$8.854\,187\,817\,620\,391 \cdot 10^{-12}\,\mathrm{F/m}$
μ_0	Permeability of vacuum	$1.256\,637\,061\,435\,917 \cdot 10^{-6}\,\mathrm{H/m}$

Number Sets

$\mathbb{N}$	Natural numbers	{1,2,3,...}
$\mathbb{Z}$	Integers	{...,-2,-1,0,1,2,...}
$\mathbb{R}$	Real numbers	$\{\ldots, -2, -1.816, 0, 1, \sqrt{2}, \pi, \ldots\}$
$\mathbb{C}$	Complex numbers	$\{\ldots, -2, -1.718 - \mathrm{j}, 0, \mathrm{j}, \sqrt{2} + \mathrm{j}\pi, \ldots\}$

Other Symbols

$\vec{E}$	Electric field	[V/m]
$\vec{B}$	Magnetic flux density	[T]
$\vec{H}$	Magnetic field	[A/m]
$\vec{D}$	Electric flux density	[C/m^2]
$\vec{M}$	Magnetic current density (fictitious)[1]	[V/m^2]
$\vec{J}$	Electric surface current density	[A/m^2]
ρ	Electric charge density	[C/m^3]
Q	Electric charge	[C]
ϵ	Absolute permittivity	[F/m]
μ	Permeability	[H/m]
j	Imaginary unit	

[1]The magnetic current $\vec{M}$ is a theoretical construct used for mathematical convenience. In reality, the source of magnetic current is always a loop of electric current or a similar type of magnetic dipole, rather than the flow of an actual magnetic charge, since magnetic monopole charges have not been observed [1].

f	Frequency	[Hz]
ω	Angular frequency	[rad/s]
λ	Wavelength	[1/m]
$\vec{k}$	Wave vector	rad/m
k	Wave number	rad/m

Contents

List of Figures

1

Introduction

This chapter presents the background about the thesis. It describes the aim, limitations, and the specific issues being investigated.

1.1 Background

Antenna development in the last decades has increased a lot, digital communication is everywhere. To satisfy the increase in data communication, and also the available bit rate, bandwidth, et cetera, the demand for new innovative antennas are high. When developing the antennas, they are measured in a Compact Antenna Test Range (CATR) to be able to characterize the electrical performance of the Device Under Test (DUT). But a well-defined Quiet Zone[2] (QZ) is required in the CATR, which is the volume where a DUT can be placed during measurements and produce accurate test results. Traditionally, field probing is done with a Field Probe Scanner (FPS), which is installed on the DUT positioner at the center of the QZ. The FPS is a heavy device for moving the probe on a linear axis, and complicated to handle. It also requires the positioner to be adapted from the normal setup used for measurements. All CATRs need a field probing after installation, to validate the alignment of the reflector, feeder, and positioner. This process is highly time-consuming and usually only consists of a few cuts, and does not measure the full QZ.

1.2 Aim

The aim of the thesis is to study the possibility of using Spherical Probe Scanning (SPS) instead of the classical way of field probing with FPS. The advantage would be that the same positioner configuration that is used for the DUT can be used for SPS (that is, by mounting the probe with a radial offset from the azimuth and elevation axes). The probe, in this configuration, would be able to cover a full sphere at a selected radius. From this measurement data, the hope is to be able to calculate the field in any point or geometry within the sphere. In the end, this would allow for better estimation of the measurement uncertainty for a DUT that is measured in a CATR.

[2]The ideal Quiet Zone is where the electromagnetic wave has a constant phase in the transversal direction, and the phase varies with the wave propagation in longitudinal direction.

1.3 Limitations

This thesis was limited to investigate the problem for only one frequency. This was chosen because there were only limited amounts of measurement data available, which was due to the high demand of the CATR and SPS measurements taking a long time to complete.

The measurement uncertainty was not only dependent on the antenna characteristics, but also on the chamber itself, for example, cables, reflector, et cetera. It could be possible to analyze measurement uncertainties with SPS, but this was not studied.

The main focus of this thesis was on the mathematical model and evaluating whether it is possible to extract the desired information from the measurements. As mentioned earlier, much more information could be extracted from spherical probing, but due to time constraints, and as a first step in exploring SPS, this was not analyzed.

1.4 Specification of the issue being investigated

This master's thesis involved investigation of:

- Background study of the QZ, that is, theory, and its relevance in antenna measurements.
- The existing measurement methods of the QZ.
- Mathematical model for evaluation of the QZ using SPS.

The ambition of the thesis was to develop a mathematical model for evaluation of the QZ using SPS and to compare it with the existing methods today, for example, FPS. If successful, the model should be implemented in MATrix LABoratory (MATLAB) and should be validated by real measurement data from spherical probing in CATR.

1.5 Societal, ethical and ecological aspects

If spherical probing is proven as a suitable option to evaluate the QZ, the measurement time will be substantially reduced. It is hard to be certain about the outcome of this, but more available time would lead to more measurements, which in turn can aid the development of new antenna technologies. The reduction of the measurement time would also reduce the impact the measurements has on the environment, by using less electricity.

The challenging part is to know how these new antenna technologies would be used. Perhaps, it can improve our way of life by enhancing the way we communicate, leading to a more unified world and bringing humanity closer. But, these new antennas could also be used for destruction, for example, Global Navigation Satellite Systems (GNSS) which is an important part of weapon guidance system [2].

If the SPS outperforms FPS, then there would be no need to use FPS. Hence, no FPS would need to be produced, which could lead to people losing their jobs since they are no longer needed.

2

Theory

This chapter will present the theory used in this report, largely drawn from the books *Spherical Near-Field Antenna Measurements* by Hansen, [3] and *Foundations of Antenna Engineering: A Unified Approach for Line-of-Sight and Multipath* by Kildal [4].

2.1 Coordinate systems

The standard coordinate systems used in this report for measurements are spherical coordinate system, which will be transformed to Cartesian coordinate system. The following parts are to describe these systems in the anechoic chamber, where the electromagnetic fields will be orientated to propagate in the z-direction, to give the same coordinate system used in the book *Spherical Near-Field Antenna Measurements* by Hansen, [3].

2.1.1 Spherical coordinate system

The spherical coordinate system implemented in this thesis uses the notation where θ is defined from the z-axis downwards towards the xy-plane, φ is defined from the x-axis around the z-axis and r is defined from origin outwards to the point. The angles θ and φ vary in the range $[0, \pi]$ and $[-\pi, \pi]$, giving a full sphere. An illustration of the coordinate system can be seen in Figure 2.1.

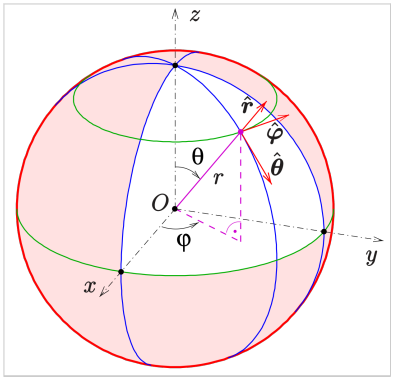


Figure 2.1: Illustration of spherical coordinate system [5].

$$x = r \sin(\theta) \cos(\varphi) \tag{2.1a}$$
$$y = r \sin(\theta) \sin(\varphi) \tag{2.1b}$$
$$z = r \cos(\theta) \tag{2.1c}$$

2.1.2 Cartesian coordinates

The Cartesian coordinate system used in this thesis is a regular right-handed Cartesian coordinate system, the x- ,y- and z-axis can be seen in Figure 2.1. Note that

boresight, that is, the axis pointing at the incoming wave (at the reflector), is the x-axis.

2.2 Maxwell's equations

Maxwell's equations are the foundation for electromagnetic waves. The following sections are derived from Maxwell's equations, hence a proper introduction to the basics of these equations are in order. Maxwell's equations can be seen in Equations (2.3a) to (2.3d) [see 1, chapter 1 page 6]. This is a rewrite of the classical representation of Maxwell's equations, where we have it expressed in $\vec{E}$ and $\vec{H}$ instead of $\vec{E}$ and $\vec{B}$ as it is usually written. The relation between $\vec{E}$ and $\vec{D}$, and $\vec{B}$ and $\vec{H}$, can be seen in Equations (2.2a) and (2.2b).

$$\vec{B} = \mu\vec{H} \tag{2.2a}$$

$$\vec{D} = \epsilon\vec{E} \tag{2.2b}$$

By using the relation Equation (2.2a), Maxwell's equations can be represented as in Equations (2.3a) to (2.3d), where the left side is Maxwell's equation expressed in differential form, and the right side is Maxwell's equation expressed in integral form.

$$\vec{\nabla}\cdot\vec{E} = \frac{\rho}{\epsilon} \qquad \oiint_S \vec{E}\cdot\mathrm{d}\vec{s} = \frac{Q_{\text{free,included}}}{\epsilon} \tag{2.3a}$$

$$\vec{\nabla}\cdot\vec{H} = 0 \qquad \oiint_S \vec{H}\cdot\mathrm{d}\vec{s} = 0 \tag{2.3b}$$

$$\vec{\nabla}\times\vec{E} = -\mu\frac{\partial\vec{H}}{\partial t} - \vec{M} \qquad \oint_C \vec{E}\cdot\mathrm{d}\vec{l} = -\mu\frac{\partial}{\partial t}\int_S \vec{H}\cdot\mathrm{d}\vec{s} - \int_S \vec{M}\cdot\mathrm{d}\vec{s} \tag{2.3c}$$

$$\vec{\nabla}\times\vec{H} = \epsilon\frac{\partial\vec{E}}{\partial t} + \vec{J} \qquad \oint_C \vec{H}\cdot\mathrm{d}\vec{l} = \epsilon\frac{\partial}{\partial t}\int_S \vec{E}\cdot\mathrm{d}\vec{s} + I_{\text{free,included}} \tag{2.3d}$$

If we assume an $e^{\mathrm{j}\omega t}$ time dependence, where t is the time, $\omega = 2\pi f$ is the angular frequency and f is the frequency, we can rewrite the differential Maxwell's equation, by taking the differentiation, as shown in Equations (2.4a) to (2.4d). This is the differential Maxwell's equation expressed in phasor form.

$$\nabla\times\vec{E} = -\mathrm{j}\omega\mu\vec{H} - \vec{M} \tag{2.4a}$$

$$\nabla\times\vec{H} = \mathrm{j}\omega\epsilon\vec{E} + \vec{J} \tag{2.4b}$$

$$\nabla\cdot\vec{E} = \frac{\rho}{\epsilon} \tag{2.4c}$$

$$\nabla\cdot\vec{H} = 0 \tag{2.4d}$$

By considering a region without sources, which is linear, isotropic, and homogeneous, where $\vec{M}$, $\vec{J}$, and ρ are all equal to zero, then Equations (2.4a) and (2.4b) simplify to [see 1, chapter 1 page 15-16],

$$\nabla\times\vec{E} = -\mathrm{j}\omega\mu\vec{H}, \tag{2.5a}$$

$$\nabla\times\vec{H} = \mathrm{j}\omega\epsilon\vec{E}. \tag{2.5b}$$

Equation (2.5) can be used for solving for either $\vec{E}$ or $\vec{H}$, by taking the curl of Equation (2.5a) and then using the expression in Equation (2.5b) (or vice verse which would lead to Equation (2.8) instead of Equation (2.7)), we get

$$\nabla \times \nabla \times \vec{E} = -\mathrm{j}\omega\mu\nabla \times \vec{H} = \omega^2\mu\epsilon\vec{E}. \tag{2.6}$$

The Left Hand Side (LHS) of Equation (2.6) can be simplified by using the vector identity Equation (A.3d). With the assumption of a source free region, we have that $\nabla \cdot \vec{E} = 0$, then Equation (2.6) simplifies to

$$\nabla^2\vec{E} + \omega^2\mu\epsilon\vec{E} = 0. \tag{2.7}$$

This is the Helmholtz equation, for $\vec{E}$. The identical equation can be derived for $\vec{H}$, with the same reasoning, which is

$$\nabla^2\vec{H} + \omega^2\mu\epsilon\vec{H} = 0 \tag{2.8}$$

The wave number $k = \omega\sqrt{\mu\epsilon}$ is defined, with the unit $[\mathrm{m}^{-1}]$, and is used to describe the spatial variations of the wave [see 1, chapter 1 page 15-16]. The Helmholtz equation is the fundamental equation used in wave theory.

2.2.1 Plane wave

Equation (2.7) can be furthered simplified by using the vector identity
Equation (A.3a) and in free-space (that is, $k = \omega\sqrt{\mu_0\epsilon_0} = k_0$), Equation (2.7) becomes [see 1, chapter 1 page 20-23],

$$\nabla^2\vec{E} + k_0^2\vec{E} = \frac{\partial^2\vec{E}}{\partial x^2} + \frac{\partial^2\vec{E}}{\partial y^2} + \frac{\partial^2\vec{E}}{\partial z^2} + k_0^2\vec{E} = 0 \tag{2.9}$$

This differential equation can be solved by using the method of separation of variables, where it is assumed that each component of the electric field $\vec{E}$ can be expressed as a product of functions, each dependent on only one of the coordinates, for example,

$$E_x(x, y, z) = a(x)b(y)c(z) \tag{2.10}$$

By using this in Equation (2.9) and then dividing it with the product of the new functions abc, then because of the single coordinate dependence of $a(x)$,$b(y)$,$c(z)$, Equation (2.10) becomes for E_x

$$\frac{1}{a}\frac{\partial^2 a}{\partial x^2} + \frac{1}{b}\frac{\partial^2 b}{\partial y^2} + \frac{1}{c}\frac{\partial^2 c}{\partial z^2} + k_0^2 = 0. \tag{2.11}$$

Examining Equation (2.11), it can be assumed that each component must be a constant for the equality to hold. This leads to three new equations

$$\frac{\partial^2 a}{\partial x^2} + ak_x^2 = 0; \quad \frac{\partial^2 b}{\partial y^2} + bk_y^2 = 0; \quad \frac{\partial^2 c}{\partial z^2} + ck_z^2 = 0; \tag{2.12}$$

Then Equations (2.11) and (2.12) give,

$$k_x^2 + k_y^2 + k_z^2 = k_0^2. \tag{2.13}$$

The separated ordinary differential equations in Equation (2.12) have the solution of the form $e^{\pm \mathrm{j}k_i i}$, where $i = x, y, z$. Even if both solutions are possible, only the wave propagating in the positive direction will be considered,

$$E_x(x, y, z) = Ae^{-\mathrm{j}(k_x x + k_y y + k_z z)}, \tag{2.14}$$

where, A is an arbitrary amplitude constant. The exponent can be rewritten as $-\mathrm{j}(k_x x + k_y y + k_z z) = -\mathrm{j}\vec{k} \cdot \vec{r}$, giving the compact form of Equation (2.14) as,

$$E_x(x, y, z) = Ae^{-\mathrm{j}\vec{k}\cdot\vec{r}}. \tag{2.15}$$

The solutions for E_y and E_z are in similar forms,

$$E_y(x, y, z) = Be^{-\mathrm{j}\vec{k}\cdot\vec{r}}, \tag{2.16a}$$

$$E_z(x, y, z) = Ce^{-\mathrm{j}\vec{k}\cdot\vec{r}}, \tag{2.16b}$$

but with different amplitudes, B and C. Because of the earlier assumption of source free region $\nabla \cdot \vec{E} = 0$, it implies that if

$$\vec{E}_0 = A\hat{\mathrm{x}} + B\hat{\mathrm{y}} + C\hat{z} \Longrightarrow \vec{E} = \vec{E}_0 e^{-\mathrm{j}\vec{k}\cdot\vec{r}}, \tag{2.17}$$

which leads to

$$\nabla \cdot \vec{E} = \nabla \cdot \vec{E}_0 e^{-\mathrm{j}\vec{k}\cdot\vec{r}} = \vec{E}_0 \cdot \nabla e^{-\mathrm{j}\vec{k}\cdot\vec{r}} + e^{-\mathrm{j}\vec{k}\cdot\vec{r}} \nabla \cdot \vec{E}_0 = -\mathrm{j}\vec{k} \cdot \vec{E}_0 e^{-\mathrm{j}\vec{k}\cdot\vec{r}} = 0. \tag{2.18}$$

The last equality in Equation (2.18) can only be true if $\vec{k} \cdot \vec{E}_0 = 0$, that is, the amplitude of the electric field must be orthogonal to the propagation direction. Waves with this property are called plane waves [see 1, chapter 1 page 20-23]. If we have a plane wave propagating in z direction in a Cartesian coordinate system, the electric field can be described as

$$\vec{E}(x, y, z) = \vec{E}_t(x, y)\, e^{-\mathrm{j}k_z z} = (E_x\hat{\mathrm{x}} + E_y\hat{\mathrm{y}})\, e^{-\mathrm{j}k_z z}. \tag{2.19}$$

The magnetic field can be determined by taking the cross product of the unit vector $\hat{z}$ and the electric field in Equation (2.19) and dividing it with the free-space impedance $\eta_0 = \sqrt{\frac{\mu_0}{\epsilon_0}}$ [see 4, chapter 2 page 23-24]. This gives

$$\vec{H}(x, y, z) = \frac{1}{\eta_0}\hat{\mathrm{z}} \times \vec{E}(x, y, z) = \frac{1}{\eta_0}(-E_y\hat{\mathrm{x}} + E_x\hat{\mathrm{y}})\, e^{-\mathrm{j}k_z z}, \tag{2.20}$$

where,

$$\eta_0 = 377\,\Omega \approx 120\pi\,\Omega, \quad k_0 = \frac{2\pi}{\lambda}, \quad \lambda = \frac{\mathrm{c}_0}{f},$$

where λ is the wavelength and c_0 is the speed of light in vacuum. The average power density can be calculated by taking the real part of the cross product of the electric field and the conjugate of the magnetic field divided by two [see 4, chapter 2 page 23-24],

$$\vec{S}_{avg} = \frac{1}{2}\mathfrak{Re}\left[\vec{E} \times \vec{H}^*\right] = \frac{1}{2\eta_0}\left(\vec{E} \times \hat{\mathrm{z}} \times \vec{E}^*\right) = \frac{1}{2\eta_0}\left(|E_x|^2 + |E_y|^2\right)\hat{\mathrm{z}}. \tag{2.21}$$

2.2.2 Spherical wave expansion

Equation (2.7) for an arbitrary scalar field in spherical coordinates becomes,

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial f}{\partial r}\right)+\frac{1}{r^2\sin(\theta)}\frac{\partial}{\partial\theta}\left(\sin(\theta)\frac{\partial f}{\partial\theta}\right)+\frac{1}{r^2\sin^2(\theta)}\frac{\partial^2 f}{\partial\varphi^2}+\omega^2\mu\epsilon f=0. \quad (2.22)$$

Stratton solves this by using separation of variables and assuming that the scalar field can be expressed as $f = a(r)b(\theta)c(\varphi)$ [see 6, chapter 7]. By doing this the following three equations can be found,

$$r^2\frac{d^2a}{dr^2}+2r\frac{da}{dr}+\left(k^2r^2-p^2\right)a=0, \quad (2.23a)$$

$$\frac{1}{\sin(\theta)}\frac{d}{d\theta}\left(\sin(\theta)\frac{db}{d\theta}\right)+\left(p^2-\frac{m^2}{\sin^2(\theta)}\right)b=0, \quad (2.23b)$$

$$\frac{d^2c}{d\varphi^2}+m^2c=0, \quad (2.23c)$$

where p and m are separation constants, and because of the periodicity of 2π of the function $c(\varphi)$, with solution $c(\varphi) = C_1\cos(m\varphi) + C_2\sin(m\varphi)$, where C_1 and C_2 are constants, it follows that to $m \in \mathbb{Z}$. By substituting $\xi = \cos(\theta)$ in Equation (2.23b), giving,

$$(1-\xi^2)\frac{d^2b}{d\xi^2}-2\xi\frac{db}{d\xi}+\left(p^2-\frac{m^2}{1-\xi^2}\right)b=0, \quad (2.24)$$

one can see that it is Legendre's differential equation when $q^2 = n(n+1), n \in \mathbb{N}$, with three singularities at $\xi = -1$, $\xi = 1$, $\xi = \infty$. When $m = 0$ the solution is the Legendre polynomials

$$P_n(\xi)=\frac{1}{2^n n!}\frac{d^n}{d\xi^n}(\xi^2-1)^n, \quad (2.25)$$

and when m is instead a positive integer the solution becomes

$$b(\xi)=P_n^m(\xi)=\frac{(-1)^m}{2^n n!}(1-\xi^2)^{m/2}\frac{d^{m+n}}{d\xi^{m+n}}(\xi^2-1)^n, \quad (2.26)$$

where m is limited by $0 \le m \le n$, otherwise $P_n^m(\xi) = 0$, [see 6, chapter 7]. The solution for Equation (2.23a) is a cylinder function [see 6, chapter 6.5],

$$a(r) = Z_n^{(c)}(kr), \quad (2.27)$$

where, [see 3, chapter 2]:

$$Z_n^{(1)} = j_n(kr) \quad \text{spherical Bessel function} \quad (2.28a)$$

$$Z_n^{(2)} = n_n(kr) \quad \text{spherical Neumann function} \quad (2.28b)$$

$$Z_n^{(3)} = h_n^{(1)}(kr) = j_n(kr) + \mathrm{j}n_n(kr) \quad \text{spherical Hankel function (1th)} \quad (2.28c)$$

$$Z_n^{(4)} = h_n^{(2)}(kr) = j_n(kr) - \mathrm{j}n_n(kr) \quad \text{spherical Hankel function (2th)} \quad (2.28d)$$

Notice that $c = 1$ and $c = 2$ are solutions for standing waves, $c = 3$ is a solution for an outwards propagating wave and $c = 4$ is a solution for an inwards propagating wave. Combining the results gives,

$$f_{nm}^{(c)}(r,\theta,\varphi) = Z_n^{(c)}(kr)P_n^m(\cos(\theta))_{\sin}^{\cos}(m\varphi). \quad (2.29)$$

Hansen then uses the vector functions $\vec{m}$ and $\vec{n}$ which are defined by

$$\vec{m}_{nm}^{(c)} = \nabla f_{nm}^{(c)} \times \vec{r}, \tag{2.30a}$$

$$\vec{n}_{nm}^{(c)} = \frac{\nabla}{k} \times \vec{m}_{nm}^{(c)}. \tag{2.30b}$$

These two vector functions are connected through curl operations, making them suitable for representing the electric and magnetic fields in a homogeneous medium. After derivation [see 3, chapter 2], these equations become

$$\vec{m}_{nm}^{(c)} = \pm\, z_n^{(c)}(kr)\frac{mP_n^m(\cos(\theta))}{\sin(\theta)}{}^{\cos}_{\sin}(m\varphi)\widehat{\theta} - z_n^{(c)}(kr)\frac{dP_n^m(\cos(\theta))}{d\theta}{}^{\cos}_{\sin}(m\varphi)\widehat{\varphi}, \tag{2.31a}$$

$$\begin{aligned}\vec{n}_{nm}^{(c)} =& \frac{n(n+1)}{kr} Z_n^{(c)}(kr) P_n^m(\cos(\theta)){}^{\cos}_{\sin}(m\varphi)\widehat{\mathrm{r}} + \\ & \frac{1}{kr}\frac{d}{d(kr)}\left[kr z_n^{(c)}(kr)\right]\frac{dP_n^m(\cos(\theta))}{d\theta}{}^{\cos}_{\sin}(m\varphi)\widehat{\theta} \pm \\ & \frac{1}{kr}\frac{d}{d(kr)}\left[kr z_n^{(c)}(kr)\right]\frac{mP_n^m(\cos(\theta))}{\sin(\theta)}{}^{\cos}_{\sin}(m\varphi)\widehat{\varphi}.\end{aligned} \tag{2.31b}$$

A more compact notation was introduced by, Jensen [7], $\vec{F}_{smn}^{(c)}$ where $s = 1$ denotes the $\vec{m}$ function and $s = 2$ the $\vec{n}$ function. In combination, Hansen used a function with some minor changes from Equation (2.29) [see 3, chapter 2],

$$f_{nm}^{(c)}(r,\theta,\varphi) = z_n^{(c)}(kr) P_n^{|m|}(\cos(\theta)) e^{\mathrm{j}m\varphi}, \tag{2.32}$$

where $n = 1, 2, 3, \ldots$ and $-n \leq m \leq n$. Also, the factor $e^{\mathrm{j}m\varphi}$ was used instead of ${}^{\cos}_{\sin}(m\varphi)$. However, unlike Jensen's approach, a power-normalized form will be used, where a single outgoing spherical wave will have an amplitude of 1 and a radiated power of $\frac{1}{2}$W. Combining this with $\bar{P}_n^{|m|}(\cos(\theta))$, which is the normalized associated Legendre function defined by Belousov [8], we get

$$\vec{F}_{mn}^{(c)}(r,\theta,\varphi) = \frac{1}{\sqrt{2\pi}}\frac{1}{\sqrt{n(n+1)}}\left(-\frac{m}{|m|}\right)^m z_n^{(c)}(kr)\,\bar{P}_n^{|m|}(\cos(\theta))\, e^{\mathrm{j}m\varphi}. \tag{2.33}$$

Then inserting Equation 2.33 in Equation (2.30) [see 3, chapter 2],

$$\vec{F}_{1mn}^{(c)}(r,\theta,\varphi) = \nabla \vec{F}_{mn}^{(c)}(r,\theta,\varphi) \times \vec{r} \tag{2.34a}$$

$$\vec{F}_{2mn}^{(c)}(r,\theta,\varphi) = \frac{\nabla}{k} \times \vec{F}_{1mn}^{(c)}(r,\theta,\varphi) \tag{2.34b}$$

The electric field can be written as,

$$\vec{E}(r,\theta,\varphi) = \frac{k}{\sqrt{\eta}} \sum_{csnm} Q_{smn}^{(c)} \vec{F}_{smn}^{(c)}(r,\theta,\varphi), \tag{2.35}$$

and the associated magnetic field can be expressed as,

$$\vec{H}(r,\theta,\varphi) = \frac{1}{\mathrm{j}\omega\mu} \times \vec{E}(r,\theta,\varphi). \tag{2.36}$$

We get the expression for outward radiated power [see 3, chapter 2],

$$P = \frac{1}{2}\sum_{snm} |Q_{snm}^{(3)}|^2 \text{ W}. \tag{2.37}$$

2.3 Time-harmonic electromagnetic fields

When electric or magnetic fields are generated by a current source which has a steady sinusoidal time variation, the time dependency has the form [4],

$$\cos(\omega t + \phi) = \mathfrak{Re}\left(e^{\mathrm{j}(\omega t+\phi)}\right), \tag{2.38}$$

where ϕ is the phase offset in radians. All fields obtained from this source will also be time-harmonic fields. Then, the vector functions for the electric and magnetic fields can be written as [4],

$$\vec{E}(x,y,z,t) = \mathfrak{Re}\left(\vec{E}(x,y,z)\, e^{\mathrm{j}(\omega t+\phi)}\right), \tag{2.39a}$$

$$\vec{H}(x,y,z,t) = \mathfrak{Re}\left(\vec{H}(x,y,z)\, e^{\mathrm{j}(\omega t+\phi)}\right), \tag{2.39b}$$

which corresponds with the assumption made in Section 2.2.

2.4 Antennas

An antenna is an object capable of receiving, transmitting, or both receiving and transmitting electromagnetic waves [9]. Antennas have the property to either enhance or attenuate the received signal, known as gain, and it depends on the direction of the antenna.

2.4.1 Probe pattern

A probe pattern describes the characteristics of an antenna. From a probe pattern one can tell the directivity of the antenna, and which gain in which direction it has. Probe patterns are often simulated using computer software, but could also be measured in a CATR. This is accomplished by placing the probe in the rotation center of the barbeque, that is, sweeping a sphere with a radius of zero. This will give a measured probe pattern. But the problem is if there are reflections and noise present in the CATR, the measurement will capture these and give an incorrect probe pattern. In Figure 2.4, an example of a probe pattern can be seen.

2.4.2 Polarization

The polarization of the electromagnetic waves seen by the antenna is described by the desired component and the undesired one of the electric field. These are called co-polar E_{co} and cross-polar E_{xp} respectively, and are pointing in the same direction as the unit vectors $\widehat{\mathrm{co}}$ and $\widehat{\mathrm{xp}}$ [4]. There are different type of polarizations like linearly polarized or circularly polarized. Note that the definitions given in Sections 2.4.2.1 and 2.4.2.3 assumes that it is a plane wave which is propagating in the positive z-axis.

2.4.2.1 Linear polarization

A y-polarized plane wave corresponds to the unit vectors $\widehat{\mathrm{co}} = \widehat{\mathrm{y}}$ and $\widehat{\mathrm{xp}} = \widehat{\mathrm{x}}$, and is a linearly polarized electric field. Kildal gives the definition,

$$\widehat{\mathrm{co}} = \cos(\xi)\,\widehat{\mathrm{x}} + \sin(\xi)\,\widehat{\mathrm{y}}, \quad E_{co} = \vec{E}_t \cdot \widehat{\mathrm{co}} \tag{2.40a}$$

$$\widehat{\mathrm{xp}} = \cos(\xi)\,\widehat{\mathrm{x}} - \sin(\xi)\,\widehat{\mathrm{y}}, \quad E_{xp} = \vec{E}_t \cdot \widehat{\mathrm{xp}} \tag{2.40b}$$

that describes an arbitrary linearly polarization, where ξ is an arbitrary angle to describe the linear polarization and $\vec{E}_t$ is the transverse electric field [4].

2.4.2.2 Elliptical polarization

Elliptical polarization is defined as [4],

$$\widehat{\mathrm{co}} = \left[\widehat{\mathrm{x}} + Ae^{\mathrm{j}\Delta\phi}\widehat{\mathrm{y}}\right]/\sqrt{1+A^2}, \qquad E_{co} = \vec{E}_t \cdot \widehat{\mathrm{co}} \tag{2.41a}$$

$$\widehat{\mathrm{xp}} = \left[-Ae^{-\mathrm{j}\Delta\phi}\widehat{\mathrm{x}} + \widehat{\mathrm{y}}\right]/\sqrt{1+A^2}, \qquad E_{xp} = \vec{E}_t \cdot \widehat{\mathrm{xp}}, \tag{2.41b}$$

where A and $\Delta\phi$ are real constants. Usually elliptical polarization is not of interest, so no further explanation will be given.

2.4.2.3 Circular polarization

A special case of elliptical polarization is circular polarization. We get that the right-hand circular polarization is defined as [4],

$$\widehat{\mathrm{co}} = \frac{(\widehat{\mathrm{x}} - \mathrm{j}\widehat{\mathrm{y}})}{\sqrt{2}}, \qquad E_{co} = \vec{E}_t \cdot \widehat{\mathrm{co}} \tag{2.42a}$$

$$\widehat{\mathrm{xp}} = \frac{(\widehat{\mathrm{x}} + \mathrm{j}\widehat{\mathrm{y}})}{\sqrt{2}}, \qquad E_{xp} = \vec{E}_t \cdot \widehat{\mathrm{xp}}, \tag{2.42b}$$

and the left-hand circular polarization is defined as [4],

$$\widehat{\mathrm{co}} = \frac{(\widehat{\mathrm{x}} + \mathrm{j}\widehat{\mathrm{y}})}{\sqrt{2}}, \qquad E_{co} = \vec{E}_t \cdot \widehat{\mathrm{co}} \tag{2.43a}$$

$$\widehat{\mathrm{xp}} = \frac{(\widehat{\mathrm{x}} - \mathrm{j}\widehat{\mathrm{y}})}{\sqrt{2}}, \qquad E_{xp} = \vec{E}_t \cdot \widehat{\mathrm{xp}}. \tag{2.43b}$$

2.4.3 Stokes Polarization Parameters

The polarization can be described with the Stokes polarization parameters [10],

$$S_0 = E_{0x}^2 + E_{0y}^2, \tag{2.44a}$$

$$S_1 = E_{0x}^2 - E_{0y}^2, \tag{2.44b}$$

$$S_2 = 2\mathfrak{Re}\left(E_{0x}^* E_{0y}\right), \tag{2.44c}$$

$$S_3 = 2\mathfrak{Im}\left(E_{0x}^* E_{0y}\right), \tag{2.44d}$$

where E_{0x} and E_{0y} are the electric field in $\hat{x}$ and $\hat{y}$ directions, respectively. By evaluating the expressions in Equations (2.44a) to (2.44d), one can determine which polarization the electric field has. The combination of S_0, S_1 and S_2 describes a linear polarization as long as $S_3 = 0$. S_3 describes how elliptical polarized the wave is [10].

2.5 Anechoic chamber

Anechoic chamber are used to test antennas, especially measure radiation diagrams to better understand the performance of the antennas. To be able to do this, the walls of the room are covered with shielding, called absorbents. These absorbents are often carbon loaded foam pyramids, which absorbs the RF-signals, hence, there are next to zero reflections inside the room. With this, one can accurately measure the radiation diagrams from antennas without interference from multi-path or other sources of errors [11].

2.5.1 Compact Antenna Test Range (CATR)

CATRs are a variant of anechoic chambers. The main purpose of a CATR is to accurately measure the far-field of antennas in a "rather" compact room. CATRs have a reflector which the feeder antenna radiates a spherical wave onto, and the reflected wave collimates into a planar wave at the DUT [12]. This volume, where we have a plane wave, is called the QZ, and here it is possible to do accurate plane wave measurements of the AUT.

2.5.1.1 Error sources

Some of the primary error sources for antenna measurements in the CATR is [13]:

- Alignment of AUT, (geometrical)
- Multi-path, (geometrical)
- Test zone quality, (geometrical)
- RF path dynamic/static variation, (geometrical)
- RF system linearity, (RF system fidelity)
- RF system dynamic range, (RF system fidelity)
- Leakage and cross talk, (RF system fidelity)
- Channel imbalance, (RF system fidelity)
- System drift, (environmental)
- Random errors, (environmental)

These errors are hard to eliminate completely, but geometrical errors can be suppressed by careful calibration and precise placement of the reflector, AUT, et cetera. For the RF system, there is not much to do except to use top of the line equipment, like Vector Network Analyzer (VNA), cables, feeder antennas, et cetera.

2.5.1.2 Quiet Zone (QZ)

The Quiet Zone (QZ) has been mentioned in earlier sections in different contexts, but what does the QZ actually mean? The QZ is the volume where a DUT is preferably placed. Here, the spherical wave, radiated from the feeder antenna, collimates into a plane wave, that is, a "simulated" far-field by converting spherical waves to plane waves using the reflector [14]. Hence, the volume of the QZ is defined as the region where the plane wave exists and where its phase and amplitude ripples remain below a specified threshold [15].

2.6 Field Probe Scanner (FPS)

Field Probe Scanner (FPS) is traditionally used to determine the QZ. They are especially used when installing the CATR, to validate the installation, but are often used from time to time to test the CATR performance and looking for anomalies [16]. The FPS has a linear mounted probe, where the probe can be rotated for the different polarizations, and a slider, which purpose is to mode the probe linearly around the axis. An illustration of this can be seen in Figure 2.2.

The caveat with using a FPS is that it is an expensive piece of equipment, it is very cumbersome, and it takes a lot of time to do the necessary measurements. Hence, a new, more efficient method, is preferred.

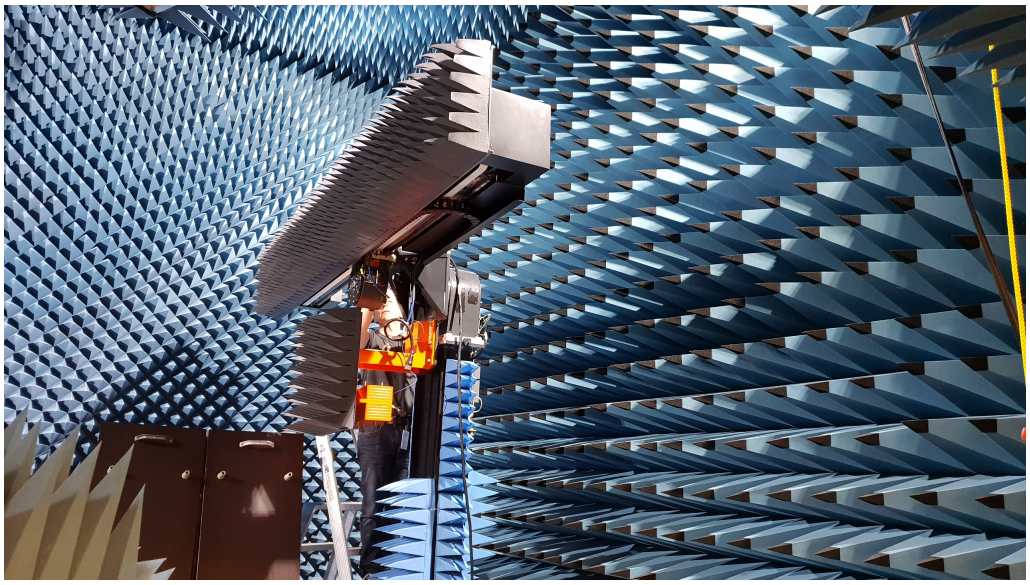


Figure 2.2: Picture of Ericsson's FPS. Courtesy of Ericsson AB.

2.7 Spherical probe scanning

In Spherical Probe Scanning the AUT in the CATR are oriented so the measured surface becomes a sphere instead of a plane as in FPS, see Section 2.6. In Figure 2.3 a frame from a simulated SPS in MATLAB is shown. The figure to the left illustrates the device's position described in the CATR coordinate system, and the figure to the right shows the measured points in the AUT coordinate system.

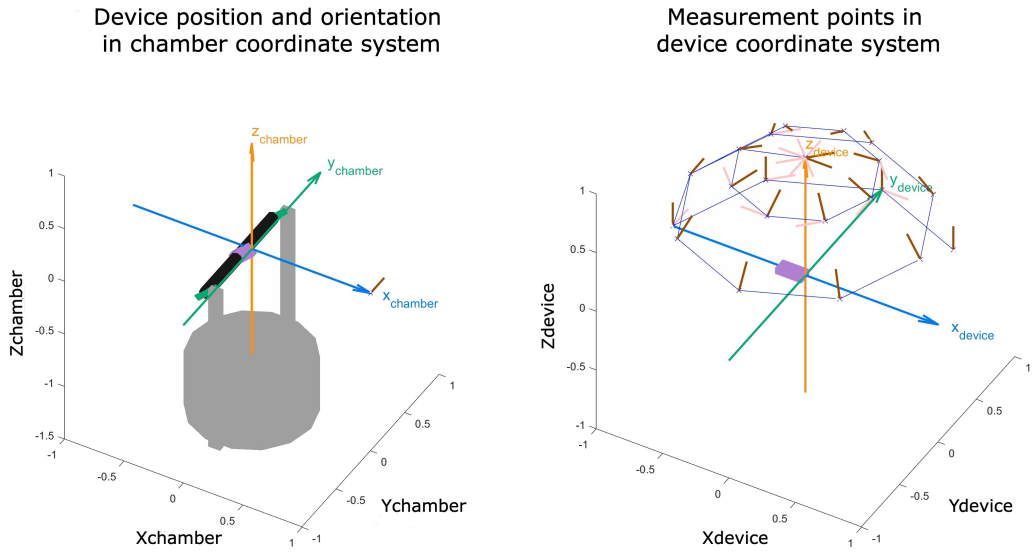


Figure 2.3: Frame from a simulated SPS in MATLAB, generated with scripts from Ericsson's MATLAB library.

2.8 Probe compensation

This section will present the different approaches to probe compensation depending on if the measurement was taken made in the near field or far field of the antenna.

2.8.1 Rotation symmetric probes

When measuring a spherical near field, also known as spherical waves, probe compensation becomes quite complex. Hansen describes an algorithm to compensate for a rotational symmetric probe, which uses the properties of spherical mode coefficients [3],

$$w\left(\chi_0, \theta_0, \varphi_0\right) = \frac{1}{2} \sum_{s\mu mn} R^p_{s\mu n} e^{\mathrm{j}m\varphi} d^n_{\theta_0} e^{\mathrm{j}\mu\chi} Q_{smn}. \tag{2.45}$$

Here w is the transmission formula described in [3], R is the spherical mode coefficient for the probe used, Q is the spherical mode coefficient for the QZ and $e^{\mathrm{j}m\varphi} d^n_{\mu m}\left(\theta_0\right) e^{\mathrm{j}u\chi}$ describes the Euler rotation, with the rotation $\chi_0, \theta_0, \varphi_0$. These rotations are made for each mode to align the AUT with the probe. The index smn describes the SWE for the AUT and $s\mu n$ describe the SWE for the probe.

By following the steps provided in the article *Qualitative Quiet Zone Comparison Using Spherical Near-Field Scanning* written by M. Dirix and D. Heberling [17], Equation (2.45) can be reduced to:

$$w^n_{1m}() = \frac{1}{2}\left(Q_{1mn} R^p_{11n} + Q_{2mn} R^p_{21n}\right) \tag{2.46a}$$

$$w^n_{-1m}() = \frac{1}{2}\left(Q_{1mn} R^p_{1-1n} + Q_{2mn} R^p_{2-1n}\right) \tag{2.46b}$$

In this thesis, all measurement was made in the antenna's far field, where the measured waves are plane and not spherical. This should lead to a more simple approach for the probe compensation, which will be explained in Section 2.8.2. Therefore, there is no need to use this transmission formula in Equation (2.45) in this thesis, and no further explanation about it is provided.

2.8.2 Far field

For an ideal measurement in free-space, where the measurement is in the far field region instead of the near field region, the operation for probe compensation should not be so complex as described above. This is because the gain pattern of the probe directly correlates to the received signal. Hence, just by dividing the received signal, at each point, with the respective gain in that point, we remove the probes effect and only get the incoming wave left. Here, we assume that there is only one incoming wave, with a known propagation direction and power. But it could be applied to multiple incoming wave, if the propagation direction and power is known for the other wave as well.

2.8.3 Alternative approach

If the measurement is not ideal, we get a superposition of both polarizations, resulting in a system of equations. Then the approach described in Section 2.8.2 will be too big of an approximation. The more general expression explained in this section was given from private communication [18].

A measurement in a direction $\widehat{\mathrm{r}}(\theta, \varphi)$, in a CATR, corresponds to

$$v(r\widehat{r}) = \left[\mathbf{R} \cdot \vec{E}_n^p(\mathbf{R}^{-1} \cdot \widehat{\mathrm{x}})\right] \cdot \vec{E}(r\widehat{\mathrm{r}}),$$

modulo a multiplicative constant. Here,

- $r\widehat{\mathrm{r}}$ is the measurement point on the sphere
- $\mathbf{R} = \mathbf{R}(\widehat{\mathrm{z}}, \varphi) \cdot \mathbf{R}(\widehat{\mathrm{y}}, \theta)$ is a rotation of $\widehat{\mathrm{z}}$ to $\widehat{\mathrm{r}}$ representing the movement of the turntable,
- $\vec{E}_n^p$ is the probe pattern for the two positions (un-rotated and rotated $\pi/2$), and
- $\vec{E}$ is the field to measure.

Performing the measurement with a single-polarized probe corresponds to one measurement with an un-rotated probe and the second with an rotated probe (rotated $\pi/2$). Hence,

$$\begin{cases} v_1(r\widehat{\mathrm{r}}) = \left[\mathbf{R} \cdot \vec{E}_1^p(\mathbf{R}^{-1} \cdot \widehat{\mathrm{x}})\right] \cdot \vec{E}(r\widehat{\mathrm{r}}), \\ v_2(r\widehat{\mathrm{r}}) = \left[\mathbf{R} \cdot \vec{E}_2^p(\mathbf{R}^{-1} \cdot \widehat{\mathrm{x}})\right] \cdot \vec{E}(r\widehat{\mathrm{r}}), \end{cases} \tag{2.47}$$

uses spherical coordinates r, θ, φ in a fixed chamber coordinate system. The unit dyadic $\mathbf{I} = \widehat{\mathrm{r}}\widehat{\mathrm{r}} + \widehat{\theta}\widehat{\theta} + \widehat{\varphi}\widehat{\varphi}$ can squeeze into Equation (2.47) to get

$$\begin{cases} v_1(r\widehat{\mathrm{r}}) = \left[\mathbf{R} \cdot \vec{E}_1^p(\mathbf{R}^{-1} \cdot \widehat{\mathrm{x}})\right] \cdot \mathbf{I} \cdot \vec{E}(r\widehat{\mathrm{r}}) = P_{1\theta}E_\theta + P_{1\varphi}E_\varphi, \\ v_2(r\widehat{\mathrm{r}}) = \left[\mathbf{R} \cdot \vec{E}_2^p(\mathbf{R}^{-1} \cdot \widehat{\mathrm{x}})\right] \cdot \mathbf{I} \cdot \vec{E}(r\widehat{\mathrm{r}}) = P_{2\theta}E_\theta + P_{2\varphi}E_\varphi, \end{cases} \tag{2.48}$$

where

$$\begin{cases} P_{n\theta} = \left[\mathbf{R} \cdot \vec{E}_n^p(\mathbf{R}^{-1} \cdot \hat{\mathrm{x}})\right] \cdot \widehat{\theta}, \\ P_{n\varphi} = \left[\mathbf{R} \cdot \vec{E}_n^p(\mathbf{R}^{-1} \cdot \hat{\mathrm{x}})\right] \cdot \widehat{\varphi}. \end{cases} \tag{2.49}$$

2.9 Source free and scatter free region

If we consider a source free and scatter free region, which means that there is nothing there that can affect an incoming wave, then the incoming wave will propagate freely inside this region. The outgoing wave will be the incoming wave propagating out of the region and the relation for the incoming wave and the outgoing wave is described by Equations (2.50c) to (2.50f), which was given from private communication [19].

$$\vec{E} = \mathfrak{Re}[\vec{E}] + \mathrm{j}\mathfrak{Im}[\vec{E}] \tag{2.50a}$$

$$\vec{E}(\theta, \varphi) = E_\theta(\theta, \varphi)\widehat{\theta}(\theta, \varphi) + E_\varphi(\theta, \varphi)\widehat{\varphi}(\theta, \varphi) \tag{2.50b}$$

$$\begin{aligned} \mathfrak{Re}[\vec{E}(\pi - \theta, \varphi + \pi) \cdot \underbrace{\widehat{\theta}(\pi - \theta, \varphi + \pi)}_{\widehat{\theta}(\theta,\varphi)}] &= \mathfrak{Re}[E_\theta(\pi - \theta, \varphi + \pi)] \\ &= \mathfrak{Re}[E_\theta(\theta, \varphi)] \end{aligned} \tag{2.50c}$$

$$\begin{aligned} \mathfrak{Re}[\vec{E}(\pi - \theta, \varphi + \pi) \cdot \underbrace{\widehat{\varphi}(\pi - \theta, \varphi + \pi)}_{-\widehat{\varphi}(\theta,\varphi)}] &= -\mathfrak{Re}[E_\varphi(\pi - \theta, \varphi + \pi)] \\ &= \mathfrak{Re}[E_\varphi(\theta, \varphi)] \end{aligned} \tag{2.50d}$$

$$\begin{aligned} \mathfrak{Im}[\vec{E}(\pi - \theta, \varphi + \pi) \cdot \underbrace{\widehat{\theta}(\pi - \theta, \varphi + \pi)}_{\widehat{\theta}(\theta,\varphi)}] &= \mathrm{j}\mathfrak{Im}[E_\theta(\pi - \theta, \varphi + \pi)] \\ &= -\mathrm{j}\mathfrak{Im}[E_\theta(\theta, \phi)] \end{aligned} \tag{2.50e}$$

$$\begin{aligned} \mathfrak{Im}[\vec{E}(\pi - \theta, \varphi + \pi)\, \underbrace{\widehat{\varphi}(\pi - \theta, \varphi + \pi)}_{-\widehat{\varphi}(\theta,\varphi)}] &= -\mathrm{j}\mathfrak{Im}[E_\varphi(\pi - \theta, \varphi + \pi)] \\ &= \mathrm{j}\mathfrak{Im}[E_\varphi(\theta, \phi)] \end{aligned} \tag{2.50f}$$

2.10 Figures

In this section, the different figures will be explained to make it easier to comprehend the results. Note that these figures are from other sections in this report and will be explained in their relevant sections.

2.10.1 Far field

The following figures, which will be explained, are in the far field domain.

2.10.1.1 Amplitude/gain

Figure 2.4 is an example of a probe pattern describing how the gain changes over different directions for a full sphere, where, the top and bottom part in the figure represents the components E_θ and E_φ, respectively. The x- and y- axis describe how the gain varies over φ and θ, respectively. Lastly, the colorbar explains how the gain varies with the different colors. Note that both the angles φ and θ are in degrees [°] and not in radians, while the gain is in decibel [dB]. There are some figures with a radial component, where this component is presented under the E_φ. Moreover, if we have a measured signal instead, then the principle is the same, only that we have an amplitude instead of a gain.

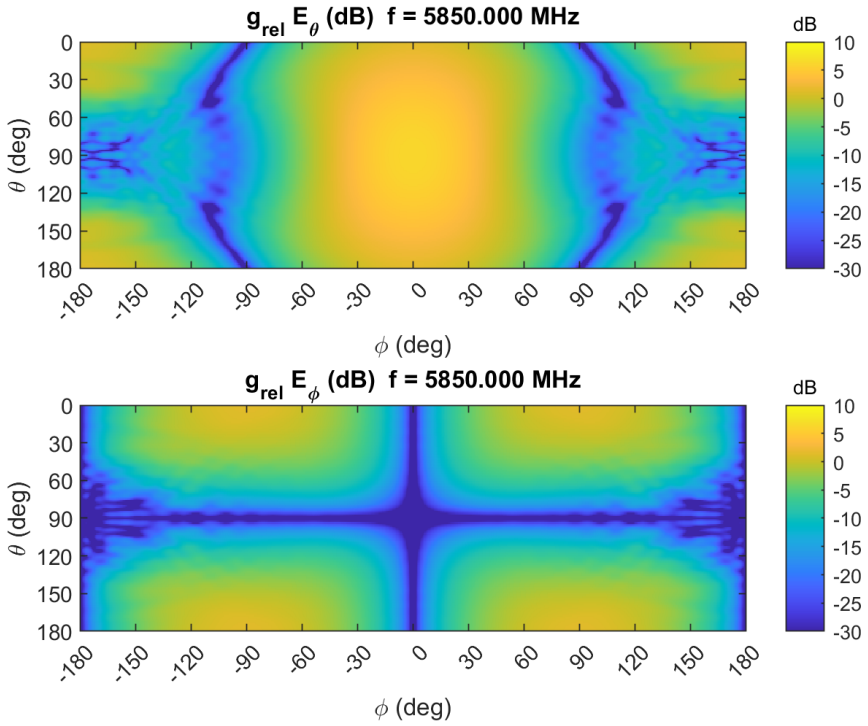


Figure 2.4: Example of a probe pattern showing the amplitude/gain in different directions, for both components E_θ and E_φ, respectively. Simulated probe pattern of OEW585 with boresight in x-direction.

Sometimes the full sphere plots are difficult to understand, and it is easier to study the cuts instead, that is, letting either θ or φ be constant and the other one vary. In Figure 2.5 we can see an example of a cut, where φ is constant, θ varies, also the x- and y- axis represent which angle [°] and amplitude [dB], respectively. The different electric components are shown in different colors.

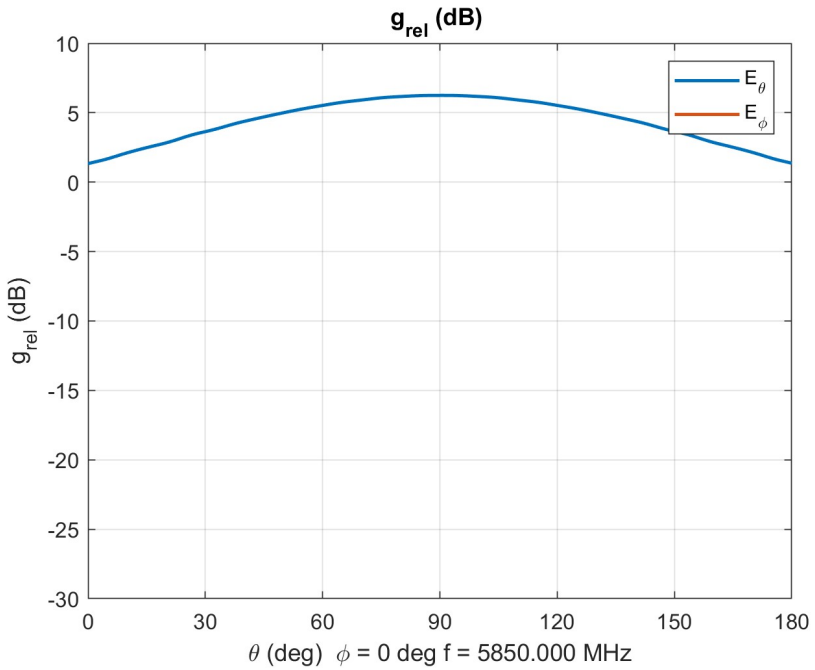


Figure 2.5: Example of cut of a full sphere, where $\varphi = 0°$ and θ varies. This cut is of the simulated probe pattern of OEW585 with boresight in x-direction.

2.10.1.2 Polarization ellipse

To be able to see how the polarization vary over the sphere, one can add polarization ellipses on the sphere, where the red, blue, and black colors describes if the polarization is right-handed elliptical, left-handed elliptical or linearly polarized.

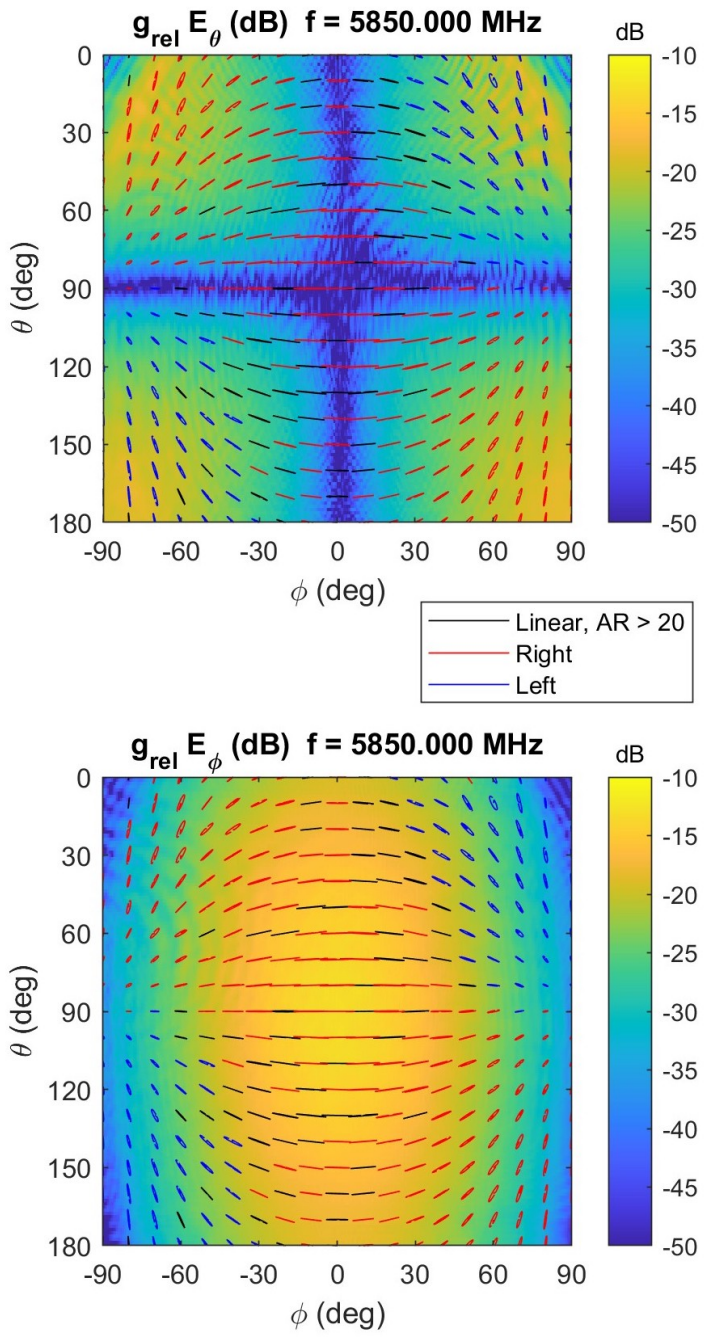


Figure 2.6: Example of polarization ellipses, describing the polarization along the probe pattern. Measured signal in CATR with OEW585.

2.10.2 Phase

Instead of analyzing the amplitude/gain, the phase can be observed. Figure 2.7 is an example of a probe pattern describing how the phase changes over different directions for a full sphere. Everything is the same as in Figure 2.4, the only thing different is that the colorbar now explains how the phase vary with the different colors.

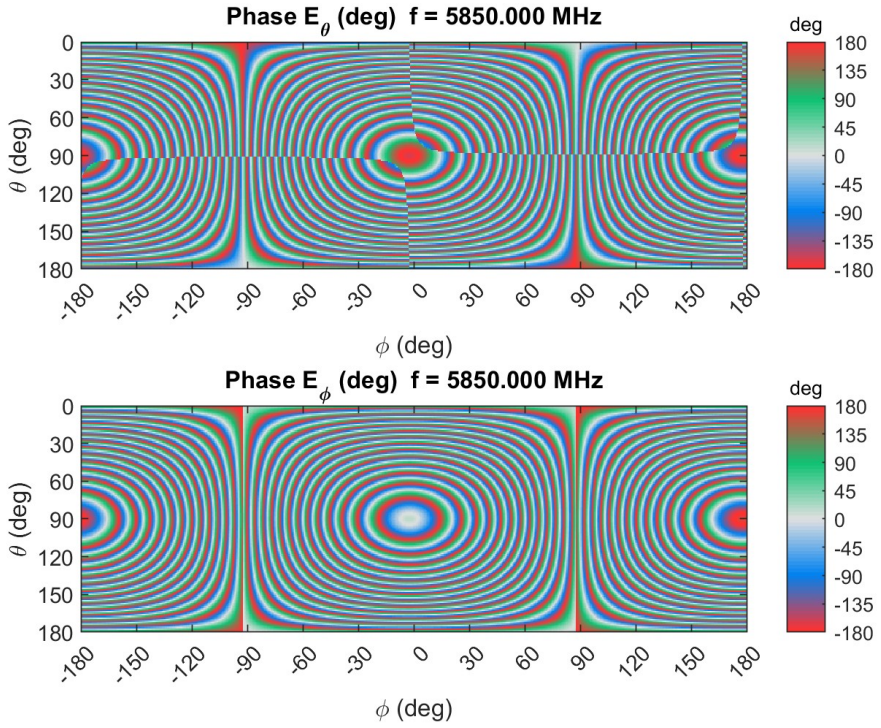
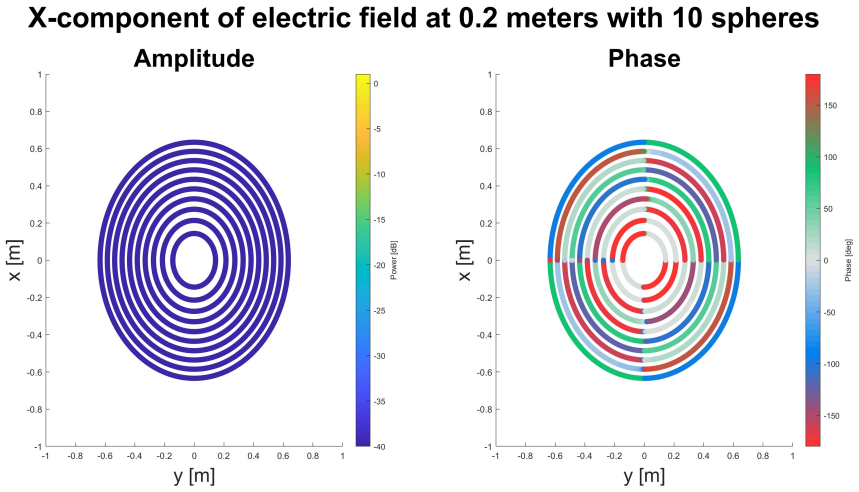


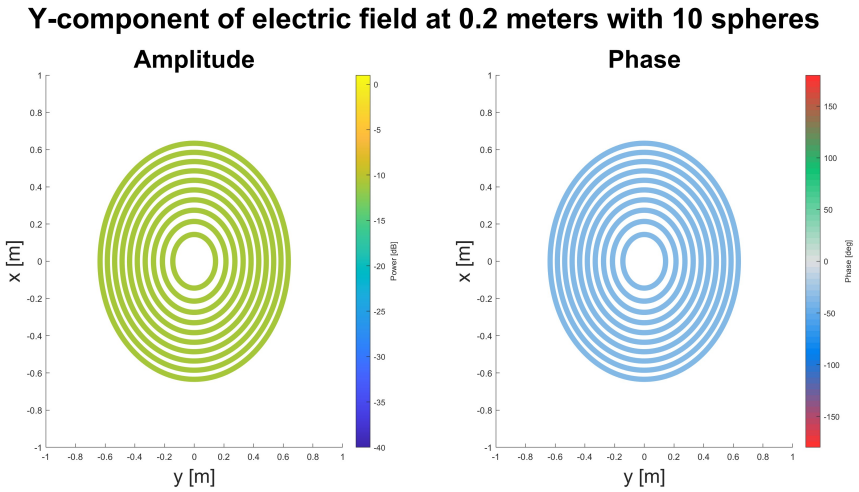
Figure 2.7: Example of a probe pattern showing the phase in different directions, for both components E_θ and E_φ, respectively. Simulated probe pattern of OEW585 with boresight in x-direction.

2.10.3 Cross section of sphere

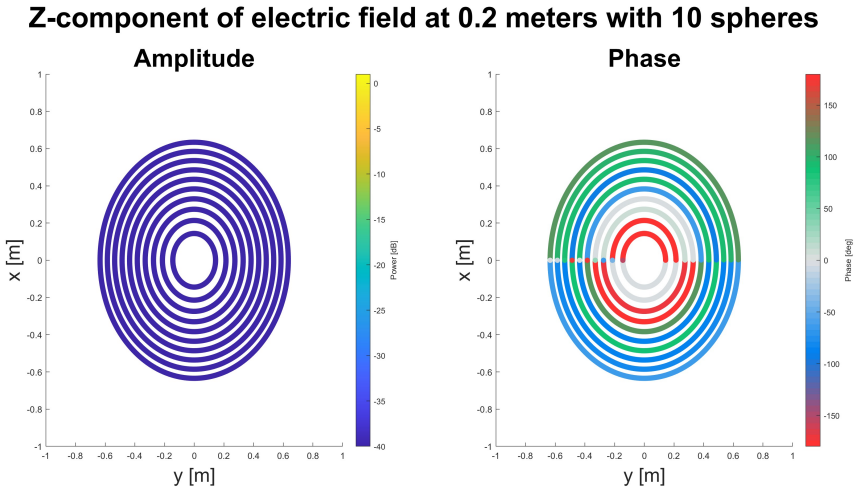
In Figure 2.8 the cross section of ten full spheres can be seen, where the axis represent where in space we are, where Figure 2.8 shows the x-, y- and z-components, respectively. Note that these figures corresponds with the xy-plane at $z = 0.2\,\mathrm{m}$, and the colorbars is describing the same things as mention before.



(a) x-component



(b) y-component



(c) z-component

Figure 2.8: Example of x-, y- and z-component of a cross section at $d = 0.2\,\mathrm{m}$ of ten reconstructed electric fields, from the field in Figure 4.1, along the surface of a sphere with different radius.

2.11 Optimatizion

During the optimization process, the function *fmincon* was trying to minimize a cost function f, in Figure 2.9 each iteration of this process can be seen. The y-axis shown the value of the cost function, the x-axis show which iteration and the purple rhombus indicate the value for each iteration.

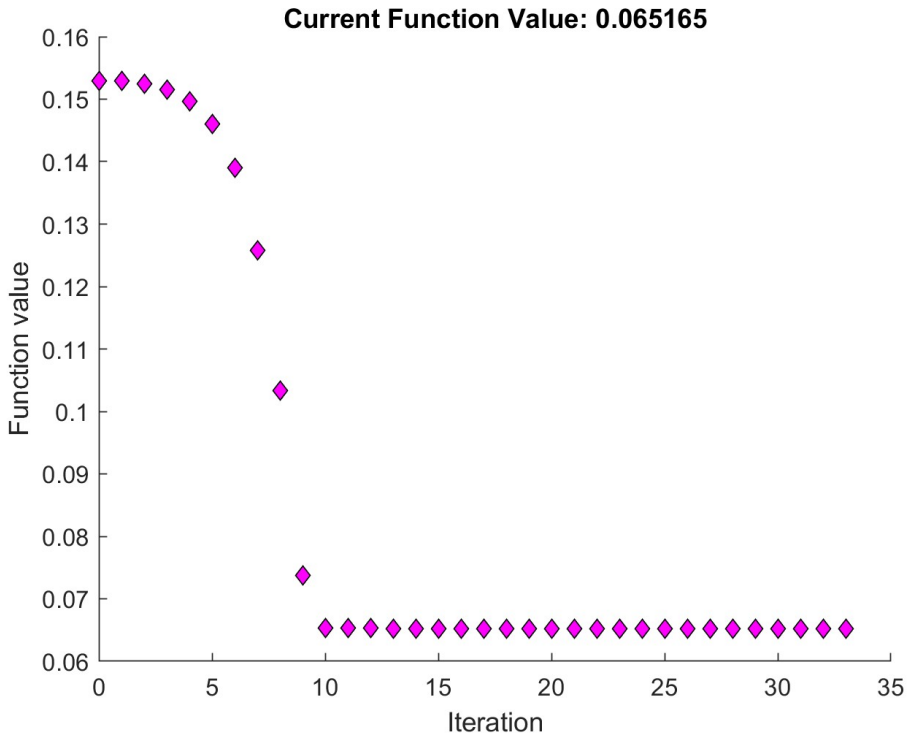


Figure 2.9: Example of polarization ellipses optimization graph.

2.12 Spherical Wave Expansion (SWE)

The far field can be converted to a SWE, an example of the SWE can be seen in Figure 2.10. As explained in Section 2.2.2, the expansion is expressed in the modes m and n, this notation is used by Hansen. Note that in the figures $n = l$ and that TEr and TMr corresponds to $s = 1$ and $s = 2$, respectively, where the x- and y-axis shows which n and m mode each point refer to. Now the colorbar is describing how much power in [dBW] are in each mode.

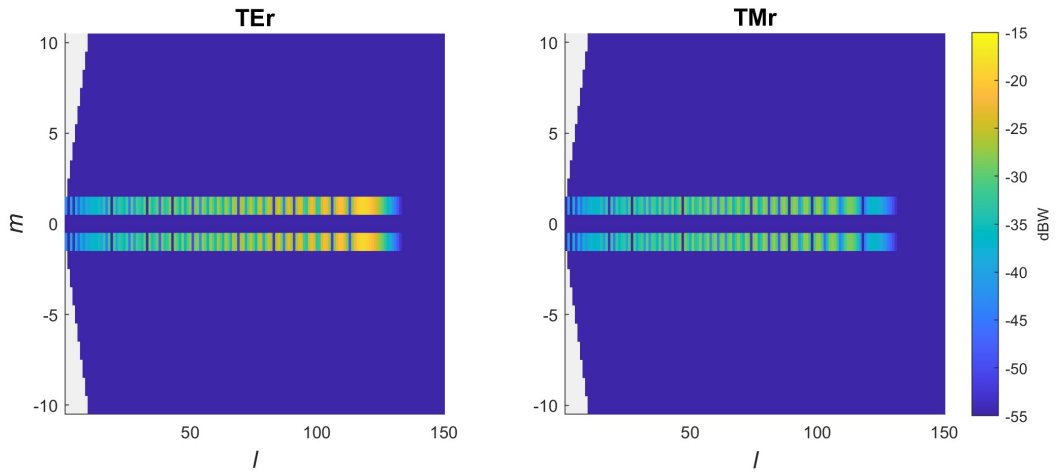


Figure 2.10: Example of SWE of a perfect aligned y-polarized plane wave propagating along the z-direction.

3

Method

This chapter will describe the measurement setup used in the CATR. Moreover, the data processing of the measurements and the method of analyzing the QZ will be presented and explained. Note, some MATLAB functions that are used will not be described since they are Ericsson's property, for the scripts that were written by us see Appendix C.

3.1 Measurement setup

The measurements were made in Ericsson's CATR, an illustration over the chamber can be seen in Figure 3.2. The AUT used when doing measurements was an Open Ended Waveguide (OEW), shown in Figure 3.1[3], which is a linear polarized probe.

Figure 3.1: Open Ended Waveguide used during measurements (OEW585)[3].

The measurements were done at a frequency of 5.85 GHz, and the feeder, which was a feed antenna LGF-11-400-WB-DL from MVG, was switching between horizontal and vertical polarization in each point (so both polarizations were measured in each point). A full sphere was measured, that is, θ was swept from 0 to π and φ was swept from $-\pi$ to π, with the probe at an offset from the rotation center (which is the radius of the sphere).

[3]Courtesy of MVG [20].

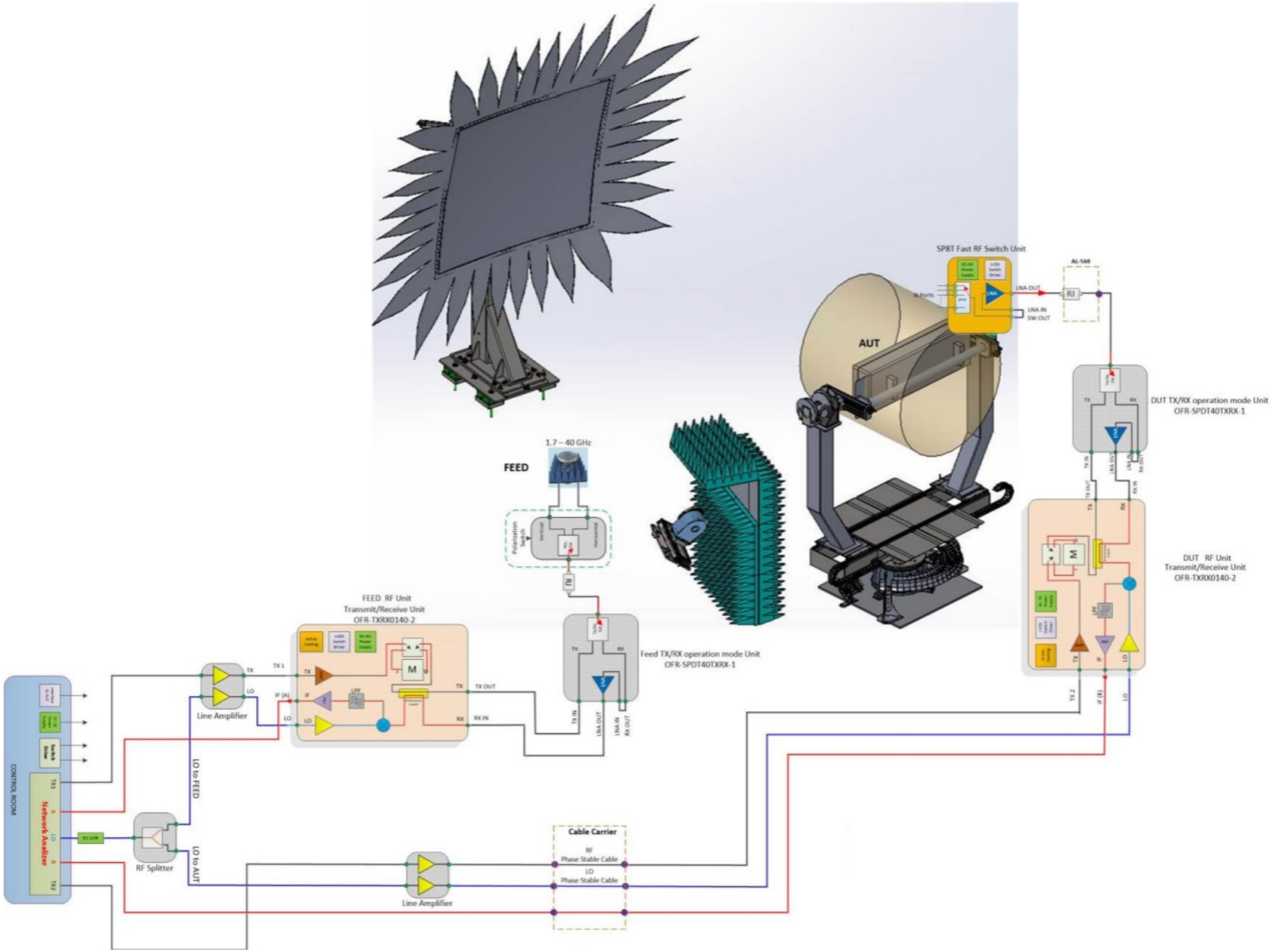


Figure 3.2: Illustration of the main system of the test setup (CATR) used to collect data. Here, the different colored lines represent the flow of different frequencies, where the red line is the intermediate frequency, the blue one is for the local oscillator and the black one is the radio frequency. Courtesy of Ericsson AB.

3.2 Algorithm

The algorithm consists of two parts, the first part is processing of the measured signal, which is, probe compensation and general optimization/cleanup of the measured signal. The second part consists of analyzing the quiet zone, that is, extracting a plane inside the sphere, or removing the plane wave.

3.2.1 Load measurement data

The data acquired from the measurements were in the form of MiDAS data files, which are ASCII files. Since an OEW was used as probe, described in Section 3.1, we need to load two measurements, so both polarizations have been measured. This is needed to be able to "stitch" together the complete signal. By rotating the probe 90° for the second measurement, we have measured both polarizations. The complete signal E_θ component comes from the first measurement E_θ, and the complete signal E_φ component comes from the second measurement E_φ, that is,

$$\vec{E}_{\text{complete}} = \left(\vec{E}_1 \cdot \widehat{i}\right) \cdot \widehat{\theta} + \left(\vec{E}_2 \cdot \widehat{i}\right) \cdot \widehat{\varphi}, \quad i = \theta, \varphi \tag{3.1}$$

where $\vec{E}_1$ is the first measurement and $\vec{E}_2$ is the second measurement. Since only one complete signal is needed, it is enough to use $i = \theta$.

For the *alternative approach* described in Section 2.8.3, then there is no need to "stitch" together the signals, we would just use the two measurements for the probe in the two orientations.

3.2.2 Probe compensation

The following section will present the methods described in Section 2.8 for probe compensation.

3.2.2.1 Far field approach

As described in Section 2.8.2, each measurement point needs to be divided with the respective probe gain in that same point. So, the gain pattern of the probe had to be loaded, see Figure 3.3[4]. To mitigate singularities and null depths, the signal, and pattern, was converted to circular polarization.. This was done to not create "infinity" or "not a number" points (that was, when dividing with zero or zero divided by zero), see Appendix C.4 for MATLAB code.

[4]Courtesy of MVG, which provided us with a simulated gain pattern.

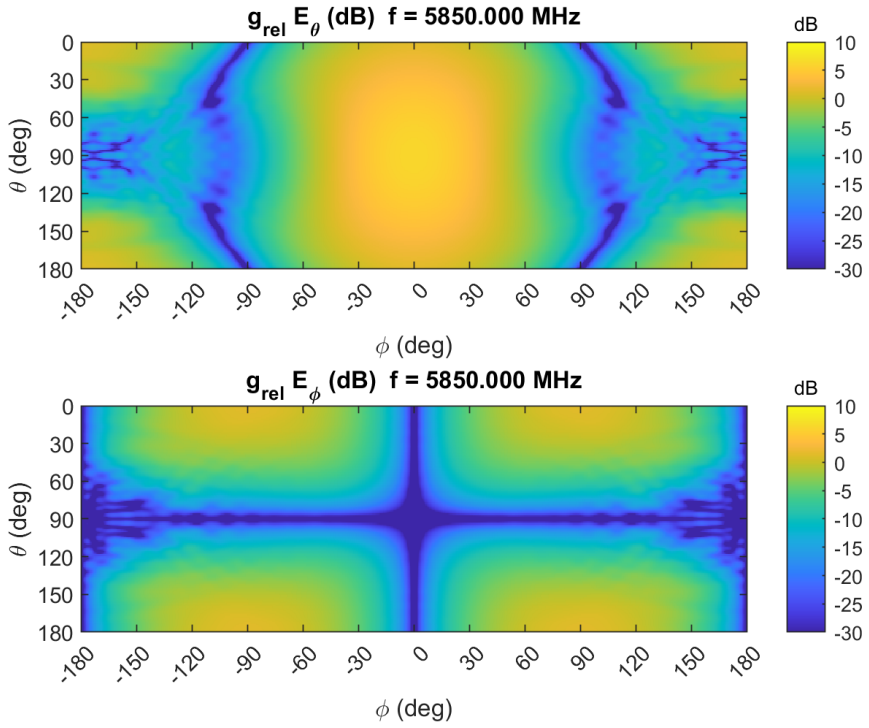


Figure 3.3: Simulated probe pattern of OEW585 with boresight in x-direction.

3.2.2.2 Alternative approach

The alternative approach to the probe compensation is described in Section 2.8.3. It works like this:

1. Back rotate $\widehat{x}$ to z-axis one measurement point in the measured signal (that is, one θ and φ value in the measured signal).
2. Convert the result in step 1 to spherical coordinates, $(x, y, z) \mapsto (r, \theta, \varphi)$.
3. Interpolate the probe pattern (which in this case has its boresight in z-axis) so it contains the θ and φ value calculated in the step 2 and extract this value and respective $\vec{E}$-field in that point.
4. Convert this point to Cartesian coordinates, $(\theta, \varphi, E_\theta, E_\varphi) \mapsto (x, y, z, E_x, E_y, E_z)$.
5. Rotate the result in step 4 back to the respective measurement point in step 1 and convert it back to spherical coordinates, $(x, y, z, E_x, E_y, E_z) \mapsto (\theta, \varphi, E_\theta, E_\varphi)$.
6. Repeat step 1 to 5 for all points in the measured signal.

From this algorithm, we have basically simulated a measurement in the CATR, which results in $P_{n\theta}$ and $P_{n\varphi}$ in Equation (2.49). The next step is to probe compensate the signal, which can be done by solving for E_θ and E_φ in the equation system in Equation (2.48), see Appendix C.5 for MATLAB code.

3.2.2.3 Measured vs simulated probe pattern

In the beginning of the thesis, the measured probe pattern from the CATR was used. However, upon analyzing the results, it became apparent that this pattern included numerous inaccuracies. As a solution, a simulated pattern was provided by MVG. Both azimuth and elevation cuts for the measured and simulated data are shown in Figures 3.4 and 3.5. From observing the probe pattern in Figure 3.3, the assumption that E_θ represents the co-polar component, and E_φ represents the cross-polar, which is also noticed in both the azimuth and elevation cuts. By comparing the measured and the simulated probe pattern, it is clear that they differ from each other. These difference likely originating from inaccuracies and error sources within the chamber itself. To mitigate the impact of these inaccuracies, the simulated probe pattern was used.

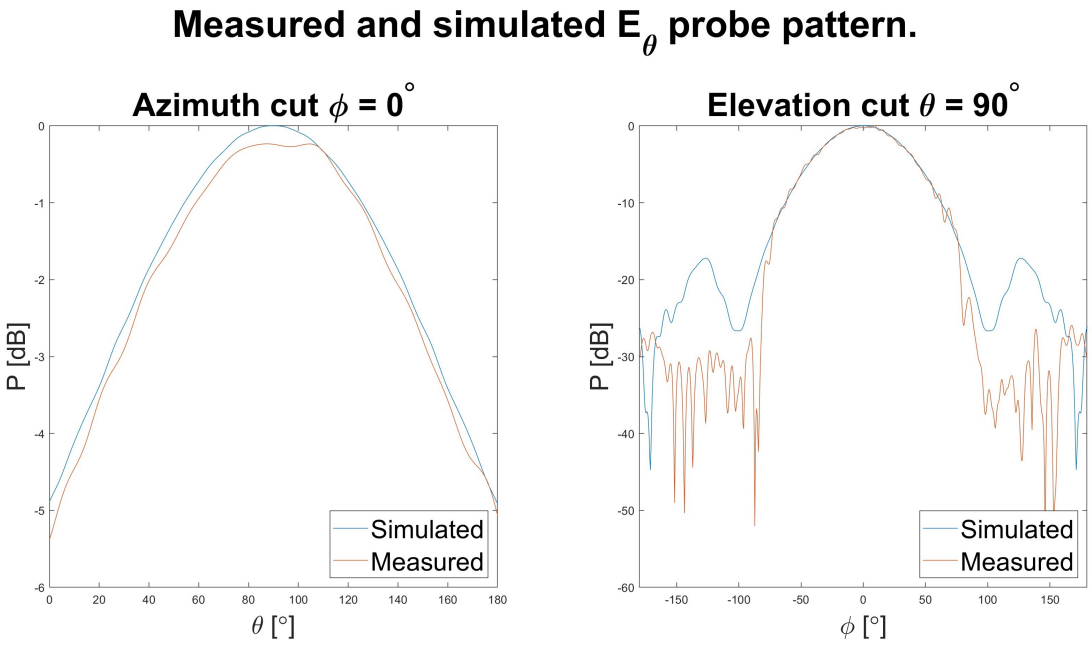


Figure 3.4: Measured and simulated azimuth and elevation cut of E_θ probe pattern of OEW585.

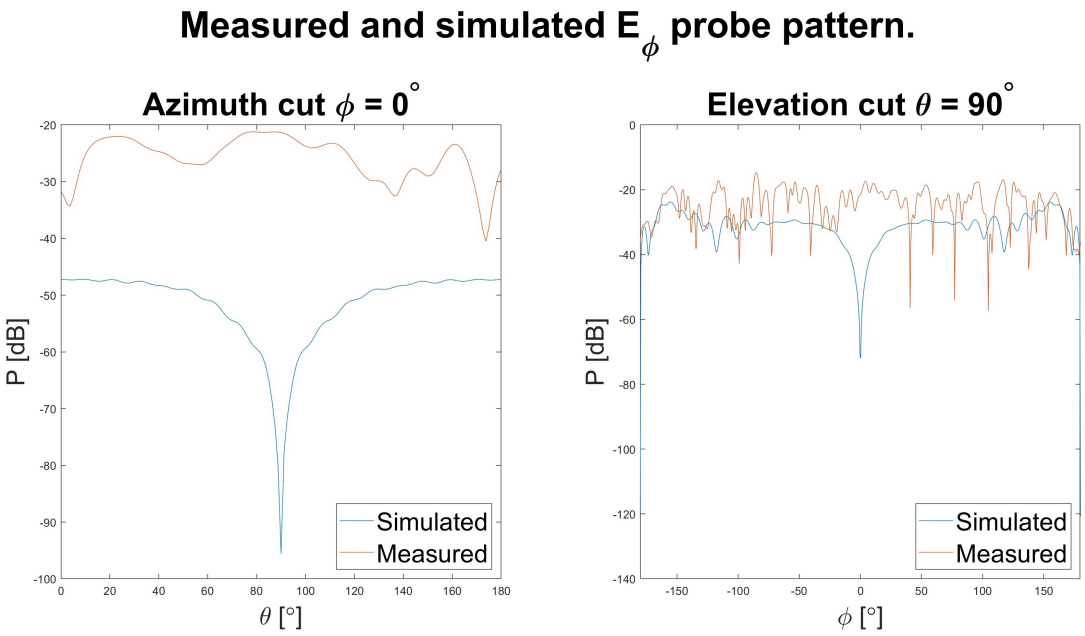


Figure 3.5: Measured and simulated azimuth and elevation cut of E_φ probe pattern of OEW585.

Based on the azimuth and elevation cuts showed in Figures 3.4 and 3.5, along with the probe pattern shown in Figure 3.3, it was evident that the majority of the obtained information from the measured signal was expected to be within the front hemisphere. Since only the front hemisphere contained "useful" information, the back hemisphere was neglected and extrapolated from the front hemisphere. That is, the front half sphere was an incoming wave and the back half sphere was an outgoing wave extrapolated from the incoming wave. Now, theoretically, the signal should be a plane wave after the probe compensation.

3.2.3 Extension of field to back hemisphere

As explained in the above paragraph, only the front hemisphere contained "useful" information. The method used needs data over a full sphere and, hence, the back hemisphere had to be extrapolated. This is done by assuming that we have a source free and scatter free region. The equations that describe how the front hemisphere relates to the back hemisphere are described in Section 2.9 and the implementation of this can be seen in Appendix C.4.

3.2.4 Analyzing QZ

After the data was probe compensated and extended to a full sphere, as explained in Section 3.2.3, the data was converted to a SWE. Now, when the spherical mode coefficient was known, Equation (2.35) could be used to reconstruct the electric field along the surface of a sphere with desired radius. This was done by a MATLAB script, shown in Appendix C.1, which calculated all necessary spheres needed to obtain an arbitrary plane which lied inside the original measured sphere. This plane represents the cross-section of the sphere which is perpendicular to the propagation direction, see Figure 3.6. As long as the measured sphere is inside the QZ, both the amplitude and phase should be constant along the constructed plane.

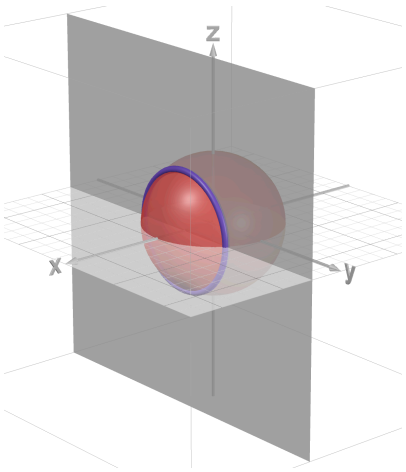


Figure 3.6: Example of cross section of a sphere.

3.3 Optimizing

The method functioned well with the theoretically generated plane wave. However, numerous issues arose with the measured data acquired in the CATR. These problems could have stemmed from misalignment in the measurement setup, affecting both the probe and the feed antenna. Additionally, there might have been misalignment between the simulated probe pattern and the measured signal. To address these issues, various optimization methods were implemented, with the optimization function *fmincon* in MATLAB. This function uses a cost function as an input to minimize while it going through different values for the variables looking for a minimum.

Since the OEW is linearly polarized, which was also confirmed by the simulated pattern, the signal should also be linearly polarized across the entire pattern. Therefore, the initial optimization step involved compensating for the phase mismatch between the two feeders. The goal was to align the measured signal as closely as possible to linear polarization, and the cost function used was the absolute value of the third stokes parameter squared,

$$f_1 = \sum\sum \left(2\mathfrak{Im}\left(E_\theta^*\right) E_\varphi e^{\mathrm{j}x}\right)^2 d\theta d\varphi, \tag{3.2}$$

where x is the optimization variable used to minimize Equation (3.2). Note that this step is made after the probe compensation for the alternative approach, because of the superposition of the polarizations.

Next, the simulated probe pattern needed to be aligned with the measured signal to improve the *far field approach* probe compensation, note that the *alternative approach* skips this step and goes directly to the next one. This alignment was performed simultaneously within the probe compensation function in MATLAB,

see Appendix C.3. In Equation (3.3a) the cost function used for this optimization is displayed, and in Equations (3.3b) and (3.3c) the relation between the variables and electric field is described. The optimization function optimized the variables x_1 to x_4 to minimize Equation (3.3a). In Equation (3.3b), $\mathbf{R}$ is a rotation matrix that rotates x_i around respective unit vector and in Equation (3.3c), the pattern is offset with the phase x_4 radians.

$$f_2 = \max\left(|E_\theta|\right) - \max\left(\left|E_{\theta_{\text{pattern}}}\right|\right) \tag{3.3a}$$

$$\vec{E}_{\text{pattern}} = \mathbf{R}\left(\hat{\mathrm{x}}, x_1\right) \cdot \mathbf{R}\left(\hat{\mathrm{y}}, x_2\right) \cdot \mathbf{R}\left(\hat{\mathrm{z}}, x_3\right) \cdot \vec{E}_{\text{pattern}} \tag{3.3b}$$

$$\vec{E}_{\text{pattern}} = \vec{E}_{\text{pattern}} e^{\mathrm{j}x_4} \tag{3.3c}$$

The final optimization involved subtracting the theoretical plane wave from the probe-compensated data. The objective was to minimize the residual error. The cost function used in this optimization step is shown in Equation (3.4a), where $\Delta E_\theta = E_\theta - E_{\theta_{\text{plane wave}}}$ and $\Delta E_\varphi = E_\varphi - E_{\varphi_{\text{plane wave}}}$. This step is quite similar to the last one if we compare the variables used to minimize the cost function, but here the optimization function is also allowed to change the amplitude of the electric field of the plane wave. The cost function is changed to calculate the remaining power after subtracting the best fitted plane wave from the measured probe compensated signal.

$$f_3 = \sum\sum\left(|\Delta E_\theta|^2 + |\Delta E_\varphi|^2\right)|\sin(\theta)|\, d\theta d\varphi, \tag{3.4a}$$

$$\vec{E}_{\text{plane wave}} = \mathbf{R}\left(\hat{\mathrm{x}}, x_1\right) \cdot \mathbf{R}\left(\hat{\mathrm{y}}, x_2\right) \cdot \mathbf{R}\left(\hat{\mathrm{z}}, x_3\right) \cdot \vec{E}_{\text{plane wave}} \tag{3.4b}$$

$$\vec{E}_{\text{plane wave}} = x_4 \vec{E}_{\text{plane wave}} e^{\mathrm{j}x_5} \tag{3.4c}$$

3.4 Validation

The best way of validating the method would be by comparing against the FPS measurements. By extracting the same plane from the spheres as where the FPS measurement was made in, we should have the same properties, that is, phase and amplitude.

Another way to validate the method is to subtract the best fitted plane wave from the probe compensated measurement. And then analyze the remaining residue, which should only contain the information about the errors present in the QZ.

4

Results

This chapter will present the results of the method for both the theoretical and measured data. Additionally, it will cover some findings discovered during the post processing of the data.

4.1 Theoretical

To get an understanding of the method, a theoretical plane wave (generated in MATLAB) was first applied to the method. The generation of the plane wave is described in Section 2.2.1. The following plane wave is y-polarized and is propagating along the z-axis. It has a max amplitude of 1 (0 dB) and a frequency of 5.85 GHz. The signal can be seen in Figure 4.1.

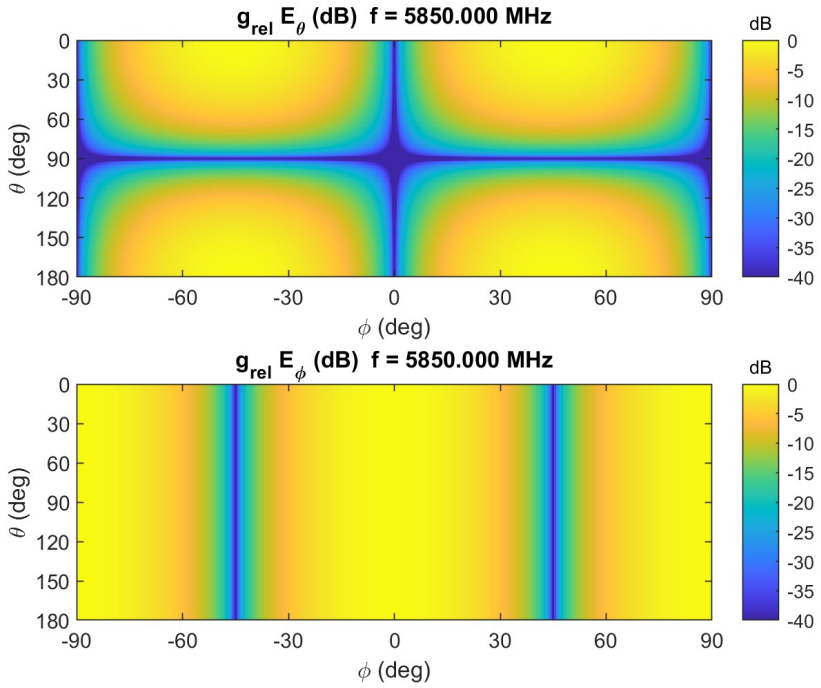
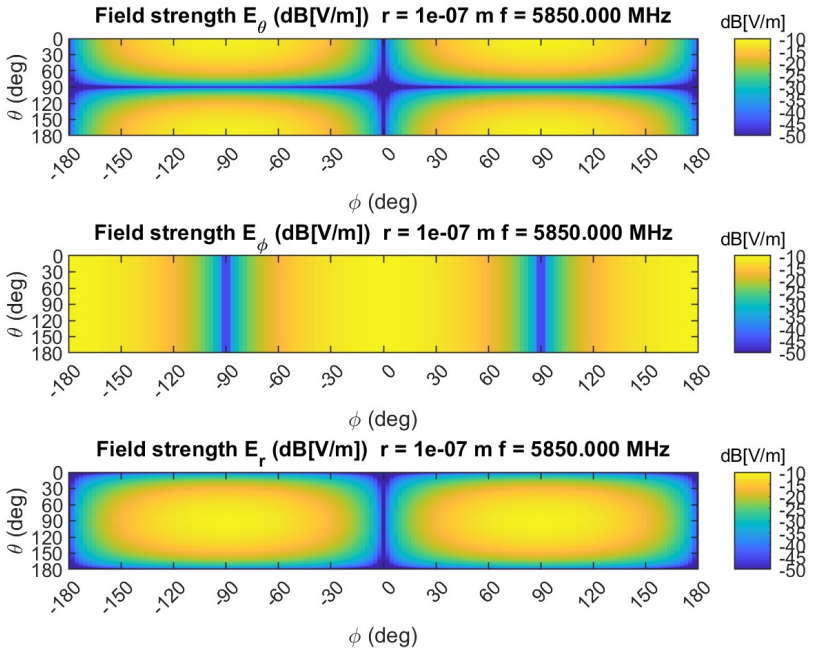
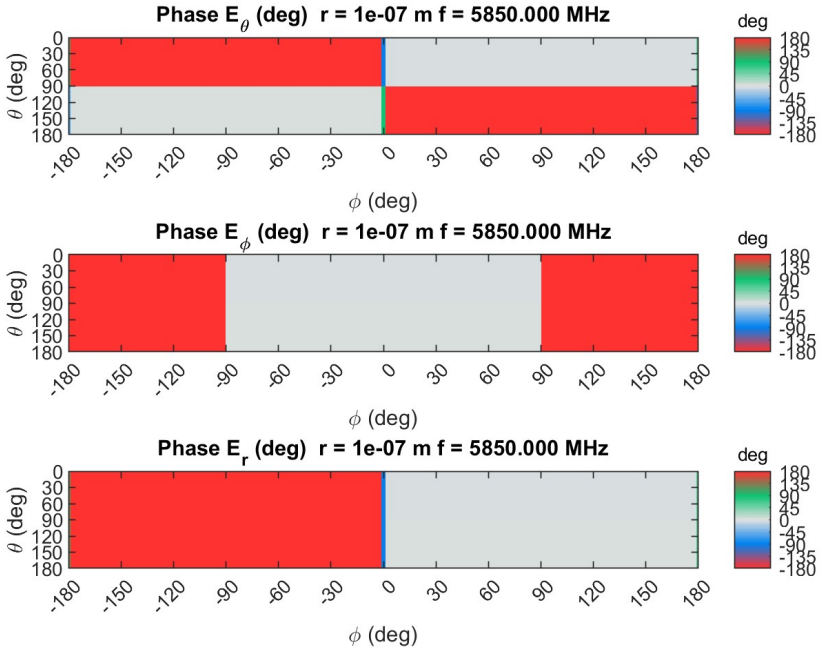


Figure 4.1: Plane wave, y-polarized, propagating along the z-axis, projected on a sphere.

By reconstructing a sphere at a radius of 0 m (approximately), we get Figure 4.2. We can see that the amplitude seems to have the same properties as the original theoretical sphere, but the phase is constant, which is what we expect.



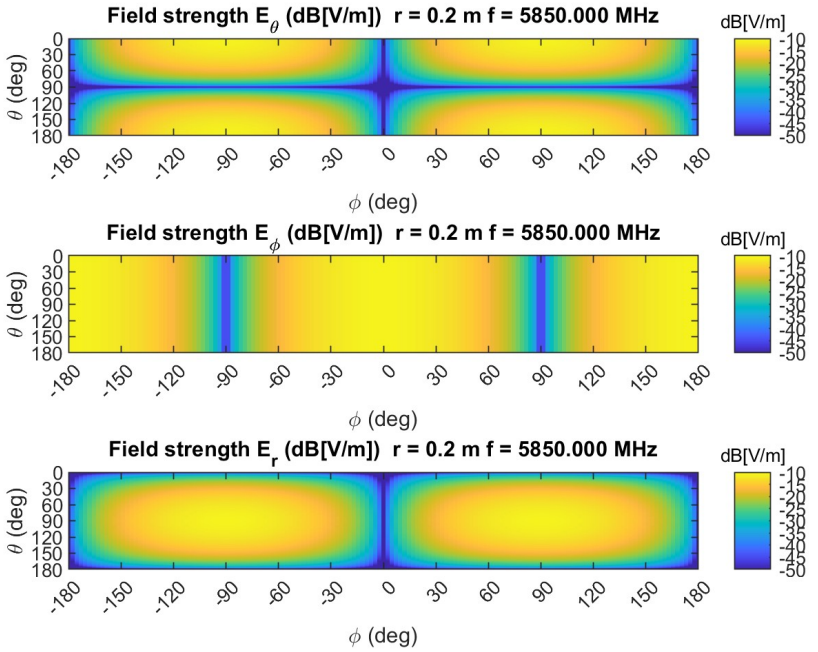
(a) Amplitude



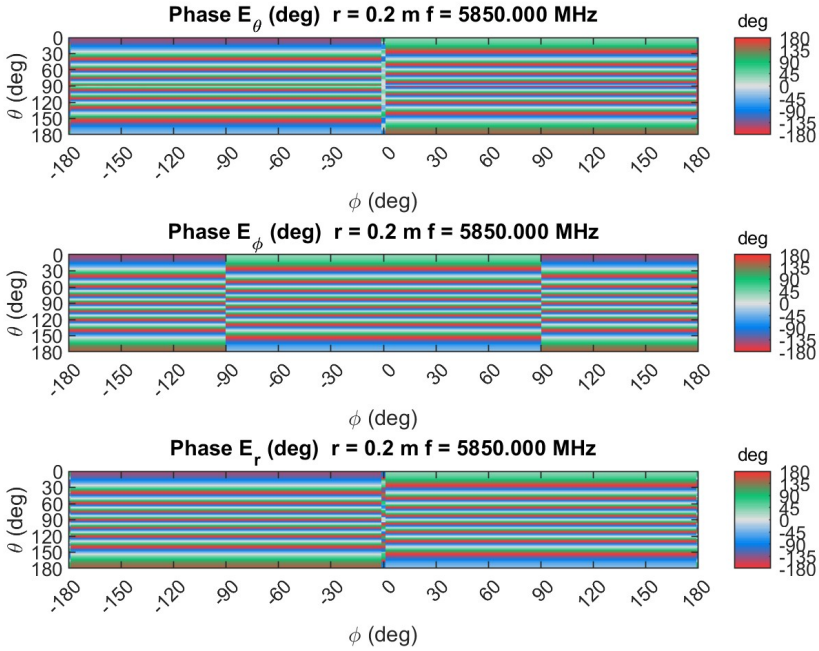
(b) Phase

Figure 4.2: Theoretical wave in Figure 4.1 reconstructed at a radius of $r \approx 0$ m.

The same thing can be done for a sphere with radius of $r = 0.2\,\mathrm{m}$, which can be seen in Figure 4.3. Here, we can see that the phase varies over the sphere, which is expected.



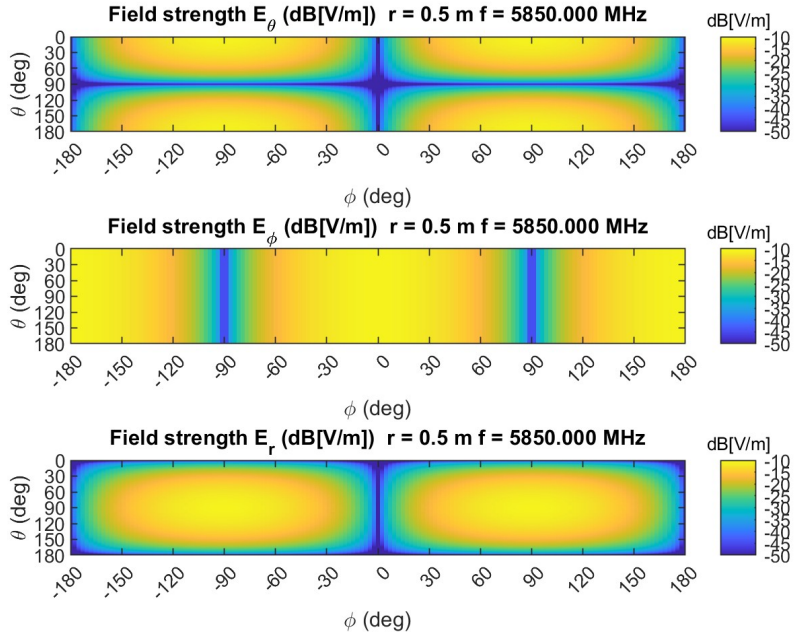
(a) Amplitude



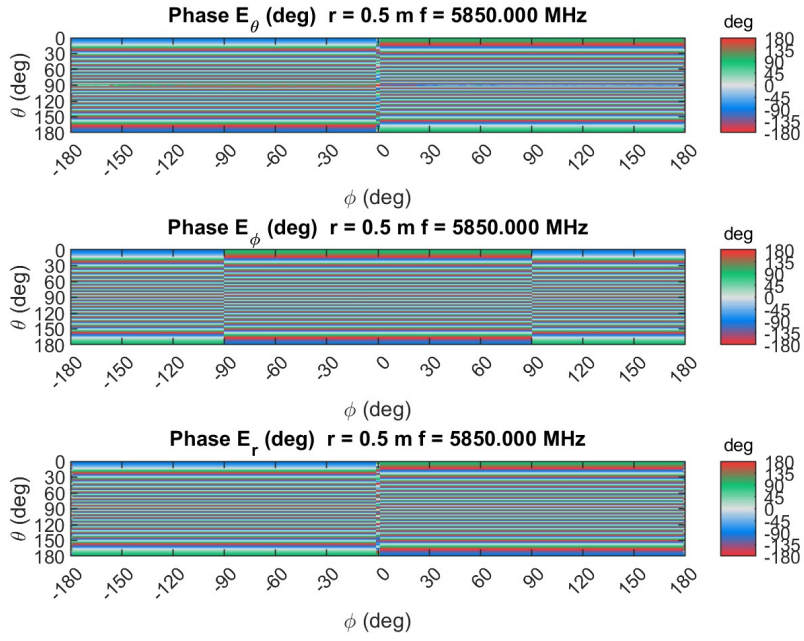
(b) Phase

Figure 4.3: Theoretical wave in Figure 4.1 reconstructed at a radius of $r = 0.2\,\mathrm{m}$.

The same thing can be done for a sphere with radius of $r = 0.5\,\mathrm{m}$, which can be seen in Figure 4.4. Here, the phase varies more than the last sphere. This reconstructed sphere is also identical to the starting sphere, which is expected since they are at the same radius.



(a) Amplitude



(b) Phase

Figure 4.4: Theoretical wave in Figure 4.1 reconstructed at a radius of $r = 0.5\,\mathrm{m}$.

From multiple reconstructed spheres, in this case ten spheres, at different symmetrical separated radius, we get the plane at $d = 0.2\,\mathrm{m}$, shown in Figure 4.5 for the E_y-component. The phase and amplitude is constant in the plane, which is what we expect from the y-polarized plane wave.

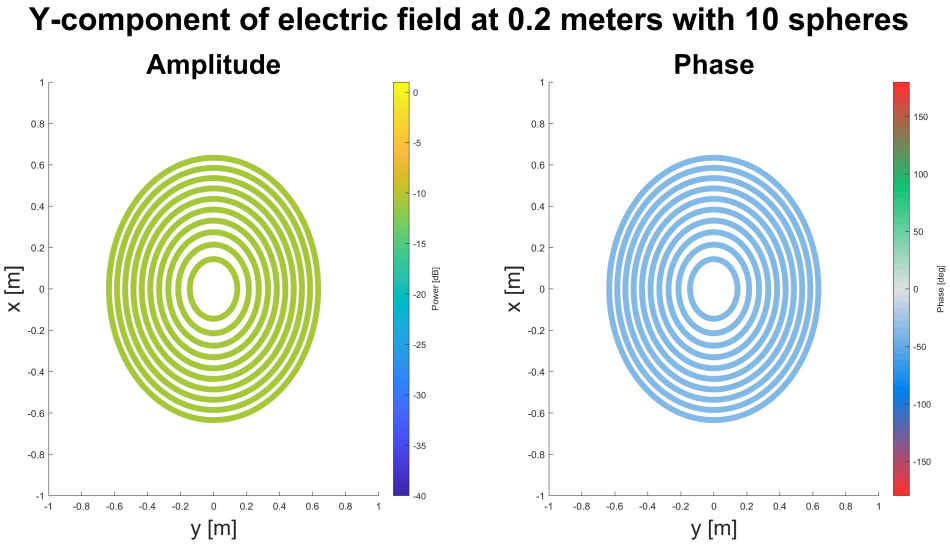
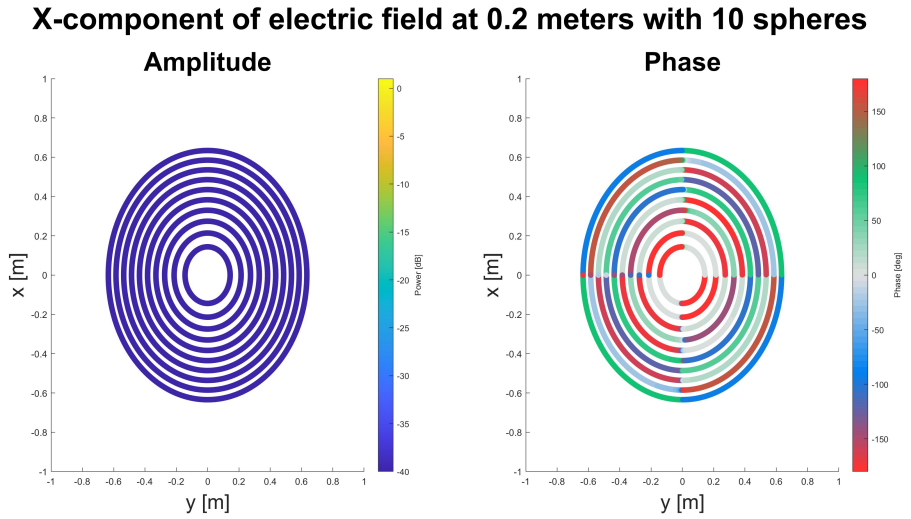
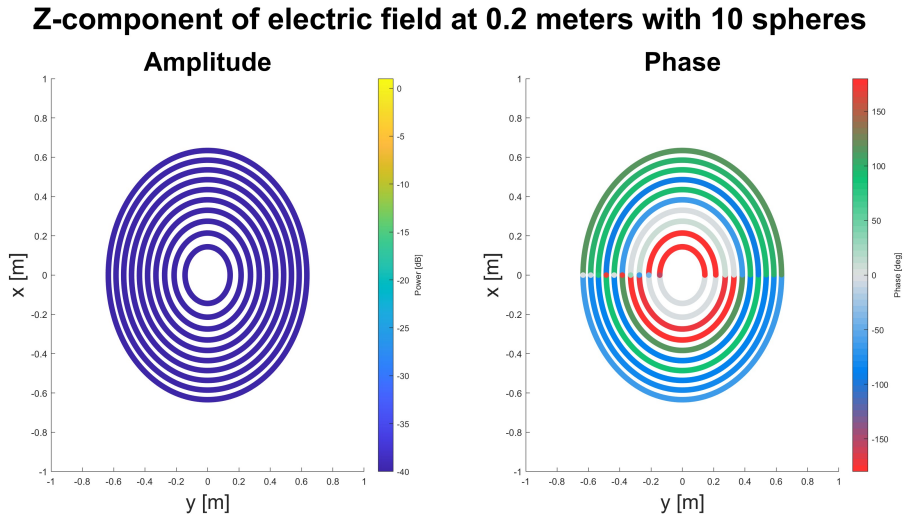


Figure 4.5: y-component of cross section at $d = 0.2\,\mathrm{m}$ of ten reconstructed electric fields, from the field in Figure 4.1, along the surface of a sphere with different radius.

If we also look at the E_x- and E_z-component, as shown in Figure 4.6, we see that the amplitude is zero, which is what we expect since we do not have any x- or z-polarization in the plane wave. The phase was numerical unstable due to the amplitude being almost zero.



(a) x-component

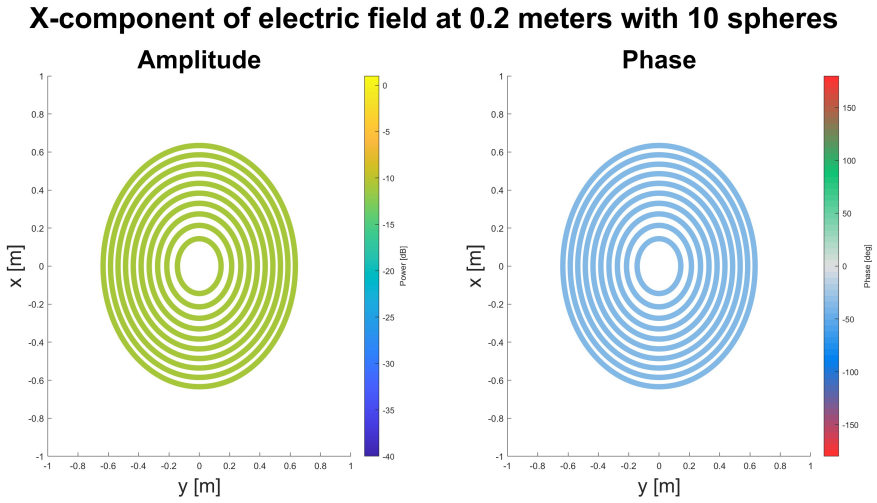


(b) z-component

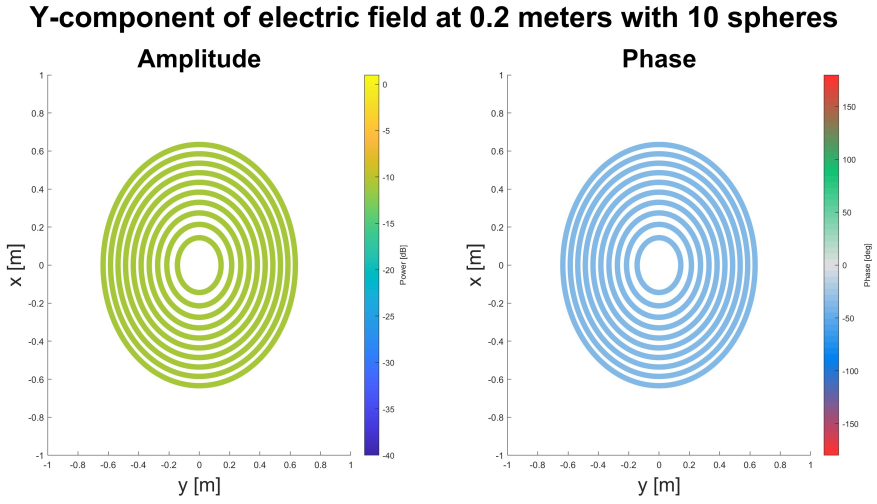
Figure 4.6: x- and z-component of cross section at $d = 0.2\,\mathrm{m}$ of ten reconstructed electric fields, from the field in Figure 4.1, along the surface of a sphere with different radius.

From this theoretical result, the measurement result should yield a similar result, that is, a constant amplitude and phase in the plane of choice.

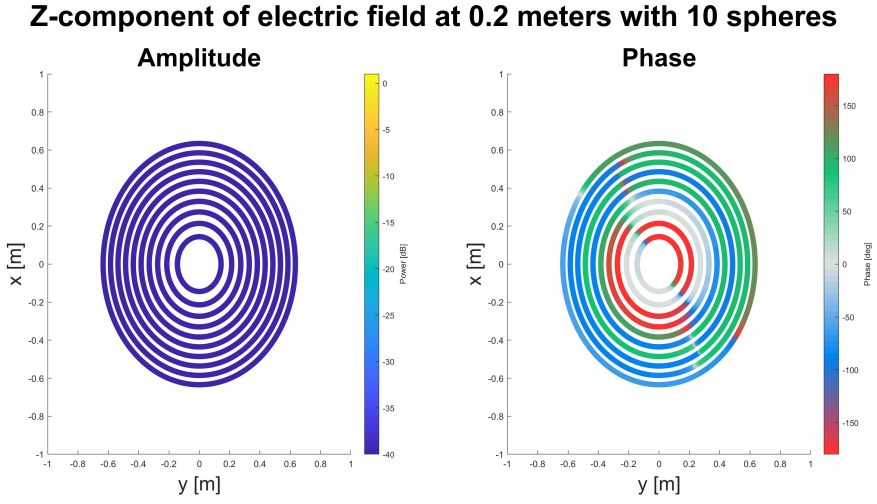
The method also worked to reconstruct and distinguish a superposition of two theoretical plane waves with x- and y -polarization, which can be seen in Figure 4.7.



(a) x-component



(b) y-component



(c) z-component

Figure 4.7: x-, y- and z-component of cross section at $d = 0.2\,\mathrm{m}$ of ten reconstructed electric fields, from an x- and y-polarized plane wave propagating along the z-axis, along the surface of a sphere with different radius.

4.2 Measurement

The following section will present the results from measurements made, the first part will present the initial approach, that is, the method described in Section 3.2. The second part will present the achieved result we got from the alternative probe compensation described in Section 2.8.3.

4.2.1 Far field approach

If we look at Figure 4.8, we can clearly see that the phase between E_θ and E_φ are mismatched (there are ellipses present in the figure).

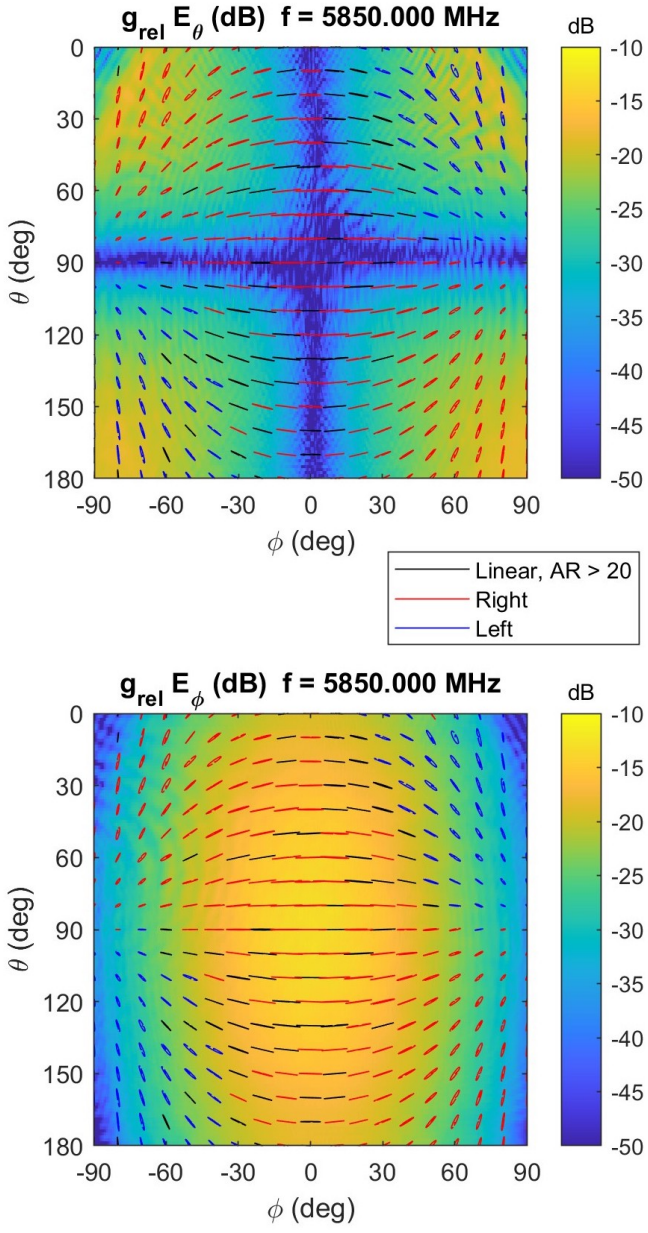


Figure 4.8: Polarization ellipses before optimizing phase.

By optimizing the phase mismatch for the two polarizations in Figure 4.8, which is explained in Section 3.3, we get the optimization that can be seen in Figure 4.9. It is clearly that the cost function is decreased.

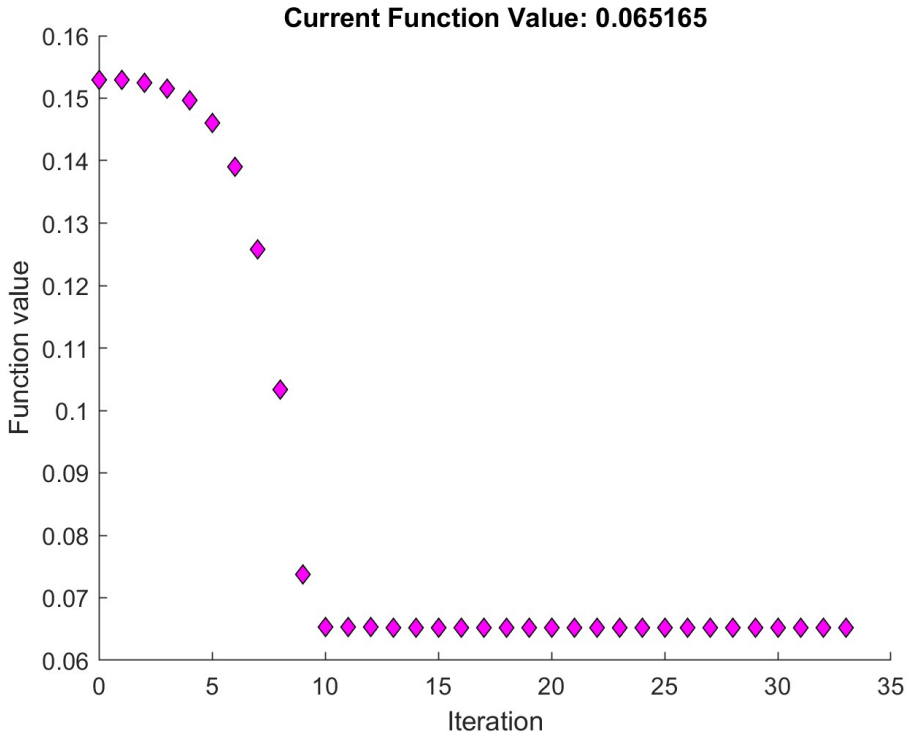


Figure 4.9: Optimization graph for phase.

The optimization gave a phase shift of $\Delta\phi = 10.6127°$ which results in the following polarization ellipses, after phase compensation, shown in Figure 4.10.

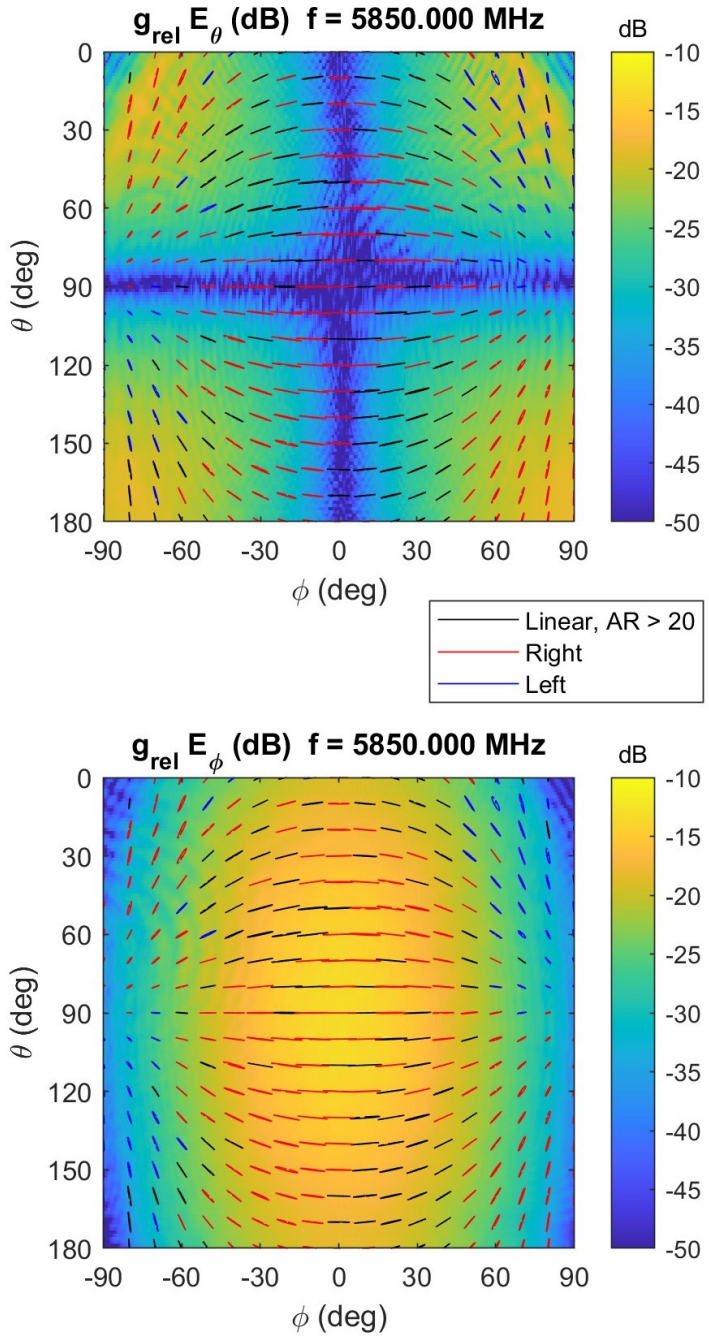


Figure 4.10: Polarization ellipses after optimizing phase.

Since the probe pattern did not necessarily match up with the signal perfectly, we optimize the probe compensation accordingly to Section 3.3. The optimization can be seen in Figure 4.11.

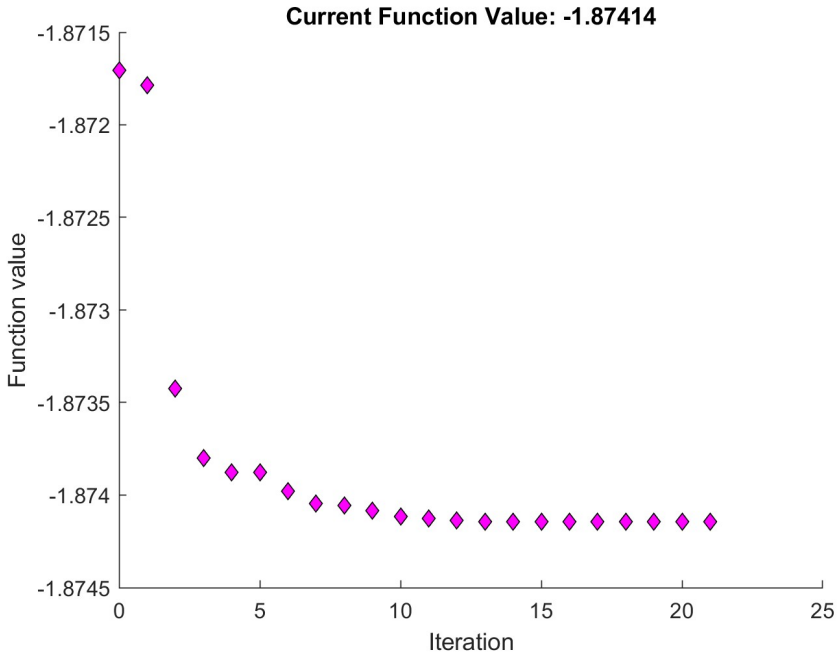


Figure 4.11: Optimization of probe compensation.

Once again, the cost function is decreased, and we get that the probe pattern needs to be rotated $[2.6684°, 0.1663°, -0.3720°]$ around respective $[x, y, z]$ axis. This results in Figure 4.12.

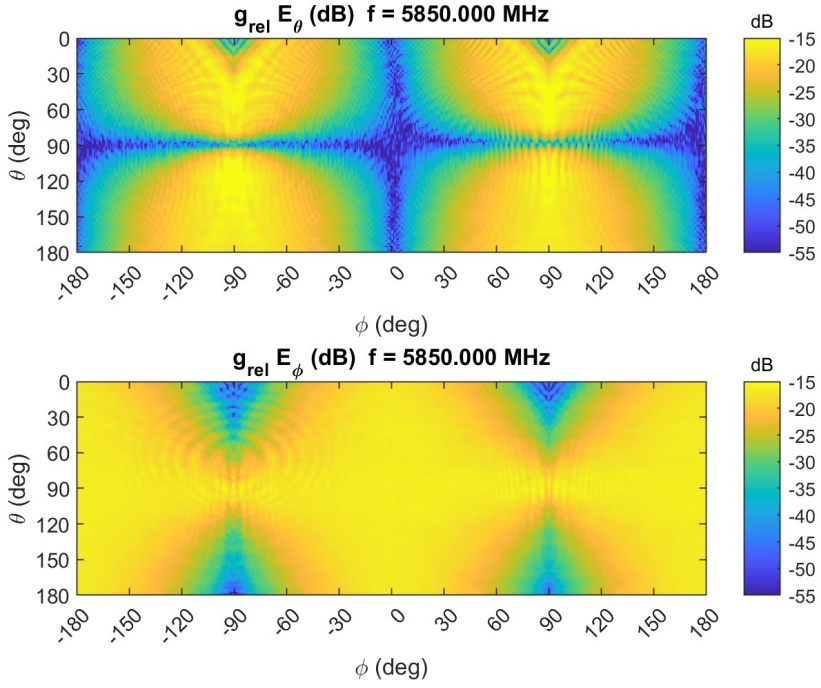


Figure 4.12: Probe compensation after optimization.

Another optimization was performed to align the theoretically generated plane wave with the measured probe compensated signal before removing the theoretical plane wave, as described in Section 3.3. This optimization process can be seen in Figure 4.13.

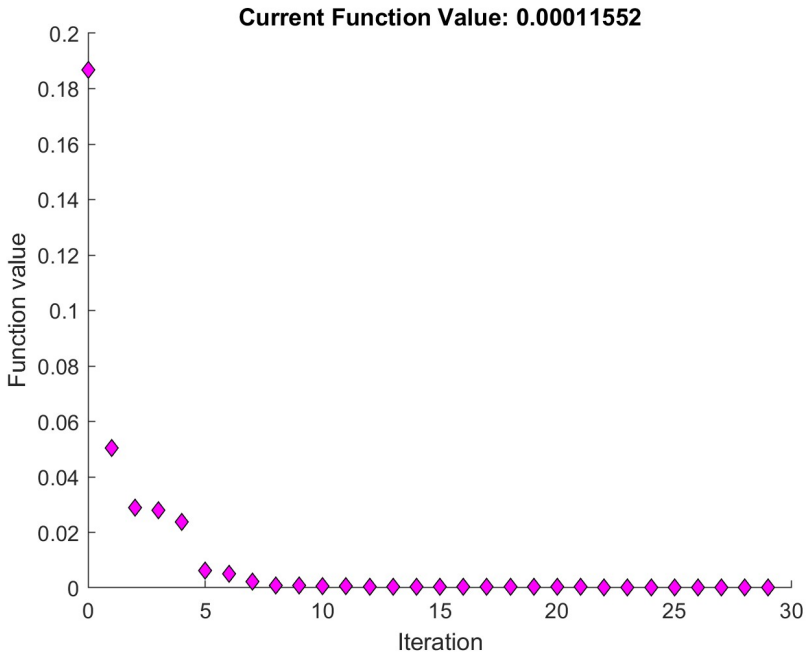


Figure 4.13: Optimization of plane wave removal.

From the optimization results, which are clearly decreasing, we get that the plane wave needs to be rotated $[0.5064°, -0.0754°, -2.1248°]$ around respective $[x, y, z]$ axis, with an amplitude of $A = 0.1550$, a radius of $r = 0.669\,\mathrm{m}$ and a phase offset of $\Delta\phi = 176.0718°$, which results in the plane wave in Figure 4.14.

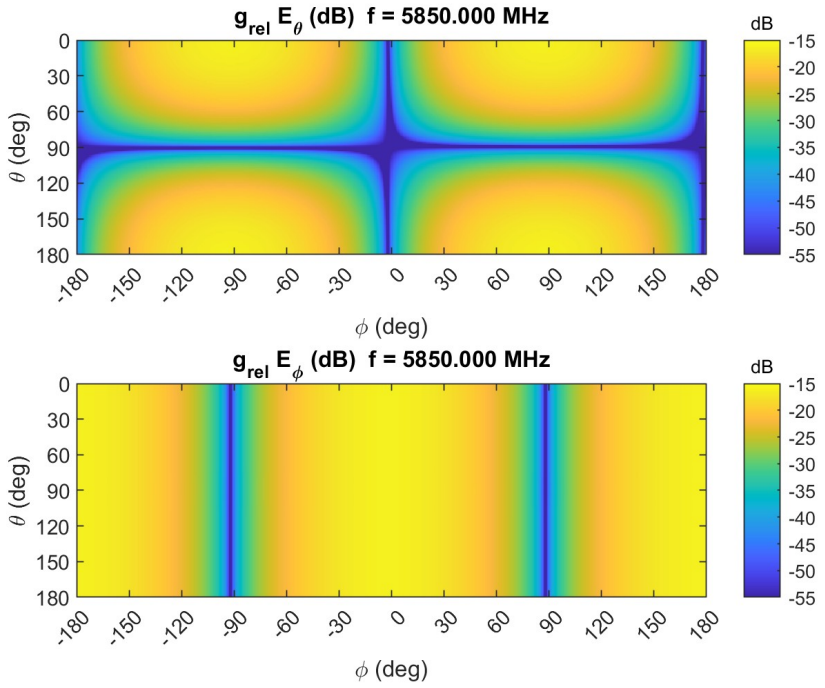


Figure 4.14: Theoretical plane wave after optimization.

The remaining residue of the measured data, when the plane wave is removed, can be observed in Figure 4.15.

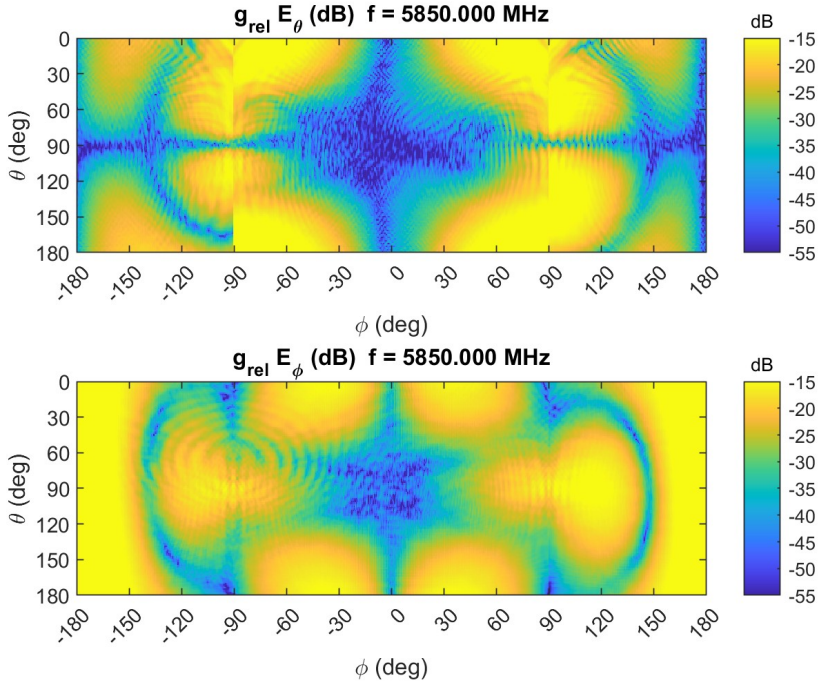


Figure 4.15: Residue left after subtracting the optimized theoretical plane wave from the measured probe compensating signal.

4.2.2 Alternative approach to probe compensation

The method described in Section 3.2.2.2 yields Figure 4.16, for the probe in the vertical orientation.

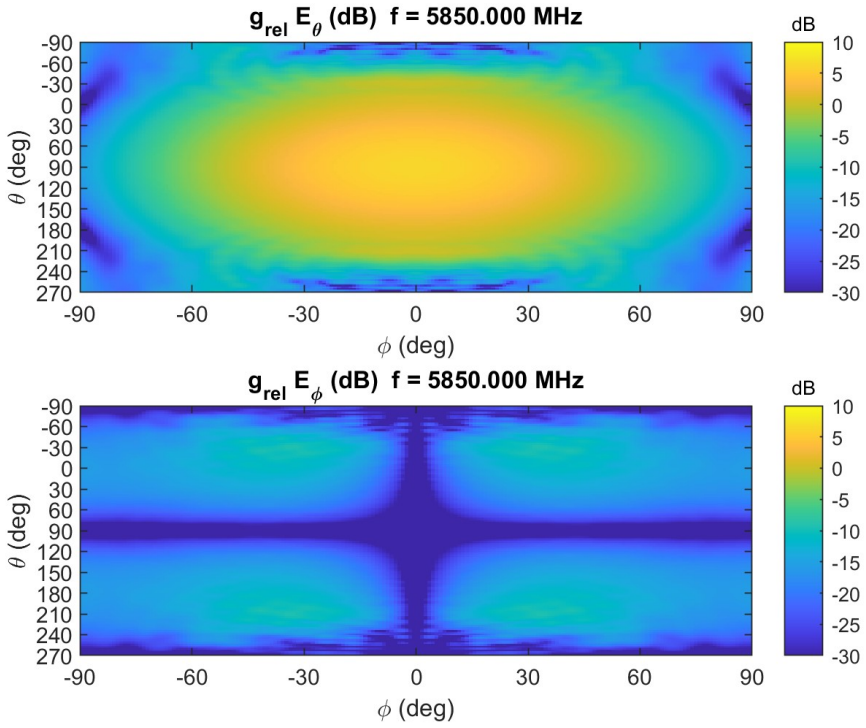


Figure 4.16: Calculated probe pattern in vertical orientation.

When the probe is rotated 90° around its z-axis, then we get Figure 4.17, that is, the probe in the horizontal orientation.

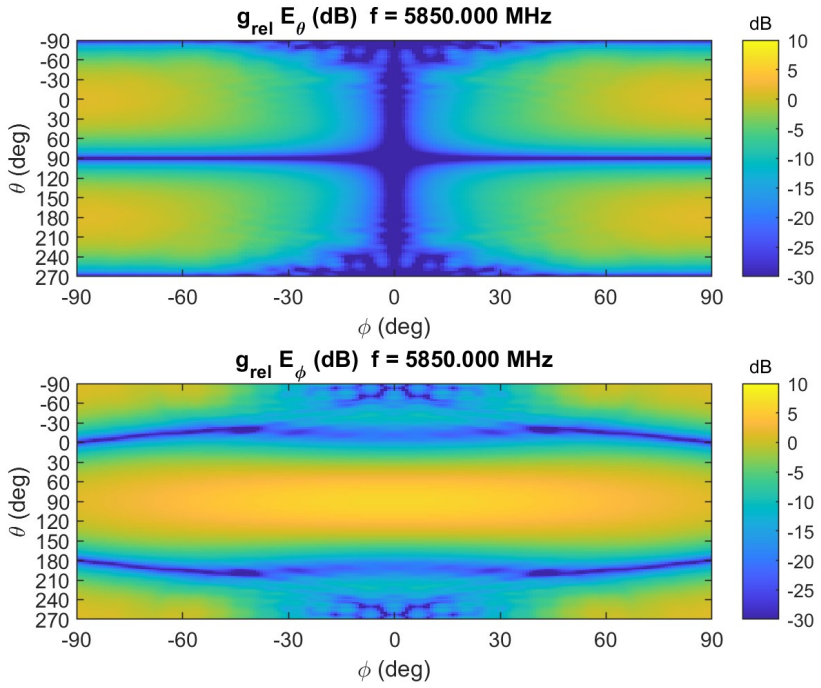


Figure 4.17: Calculated probe pattern in horizontal orientation.

The probe compensation was obtained from solving the equations system in Equation (2.48) and can be seen in Figure 4.18. Note that only a half sphere was used to solve the equations system.

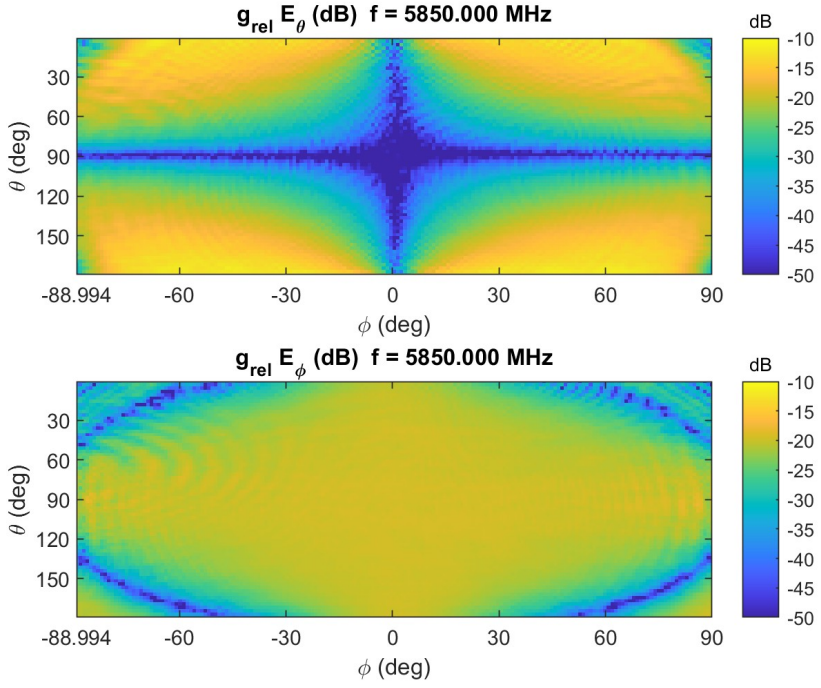


Figure 4.18: Probe compensation from solving the system in Equation (2.48).

Now, to get linear polarization, the optimization was done as described in Section 3.3, but for the probe compensated data, where the process can be seen in Figure 4.19. The cost function was clearly decreased, which yielded a phase offset of the E_φ component off $\Delta\phi = -7.8921°$.

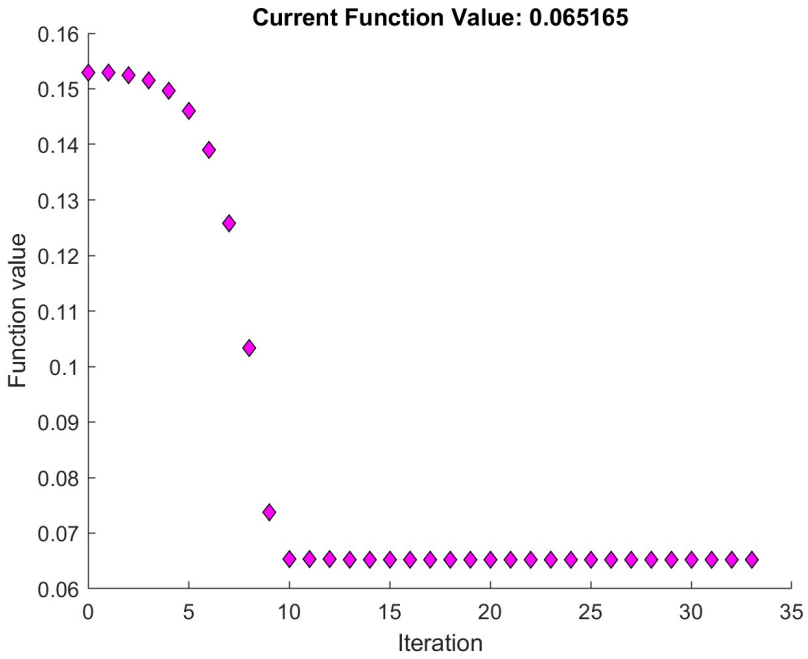


Figure 4.19: Optimization graph for phase.

The linear polarization optimization results can be observed in Figure 4.20

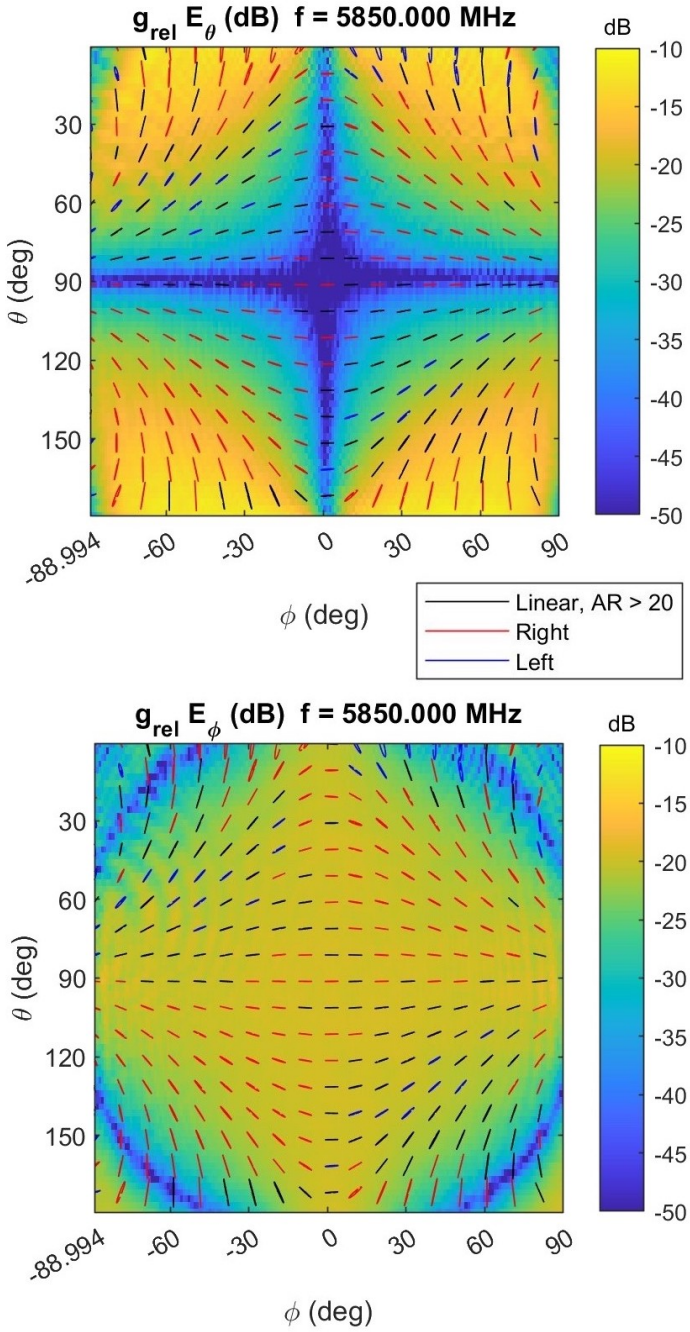


Figure 4.20: Polarization ellipses after optimizing phase.

The plane wave removal optimization was done as described in Section 3.3. The cost function was once again decreased, which can be seen in Figure 4.21.

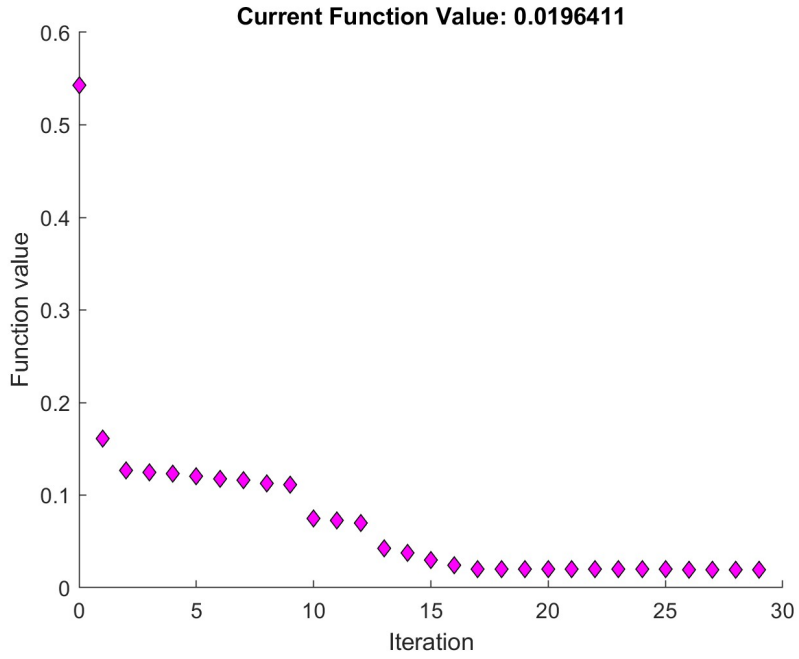


Figure 4.21: Optimization of plane wave removal.

From the optimization process, we get that the plane wave needs to be rotated $[0.5833°, -0.0864°, -2.1724°]$ around respective $[x, y, z]$ axis, have an amplitude of $A = 0.1861$, a radius of $r = 0.6664\,\text{m}$ and a phase offset of $\Delta\phi = 168.8236°$, which results in the plane wave in Figure 4.22.

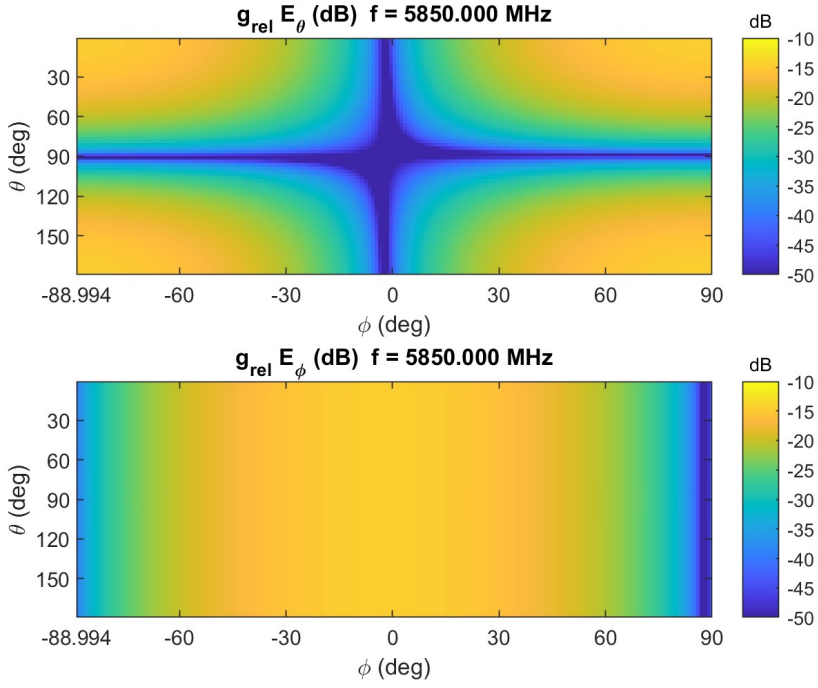


Figure 4.22: Theoretical plane wave after optimization.

The remaining residue of the measured data, when the plane wave is removed, can be observed in Figure 4.15.

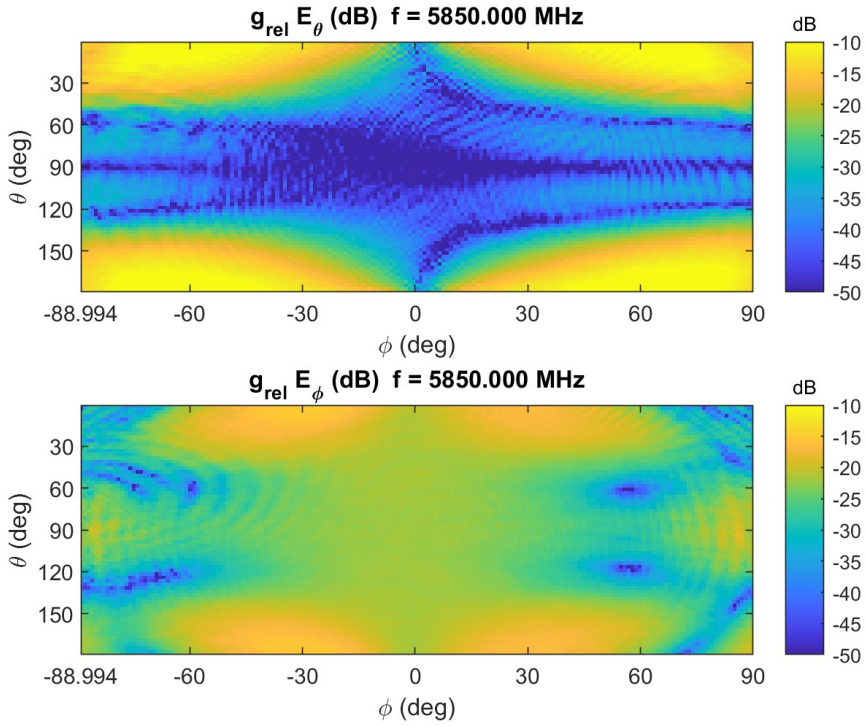


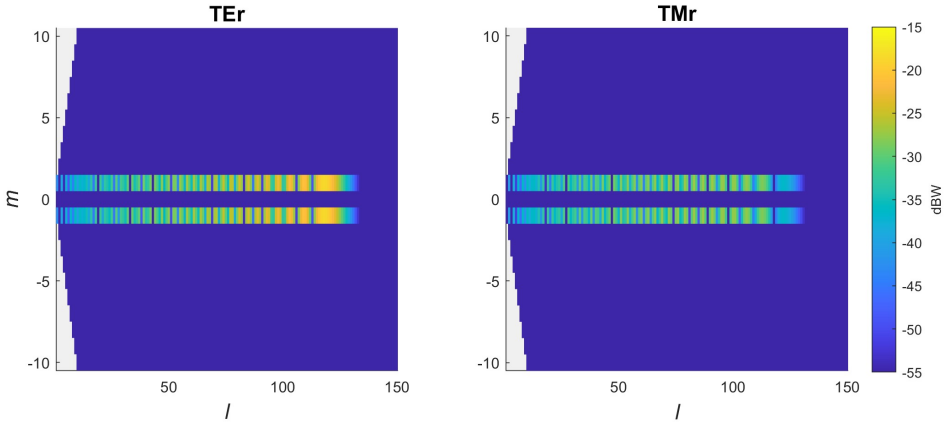
Figure 4.23: Residue left after subtracting the optimized theoretical plane wave from the measured probe compensating signal.

4.3 Findings

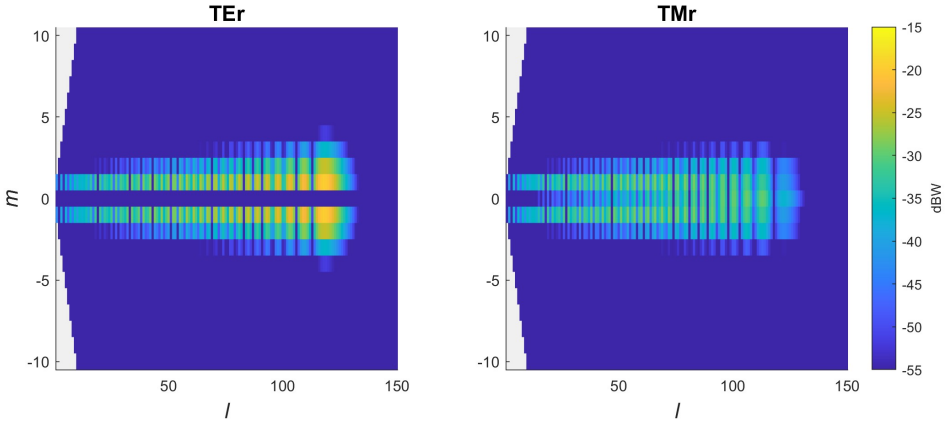
During the post processing of the data, some interesting behaviors were noticed, which will be presented below.

4.3.1 Alignment

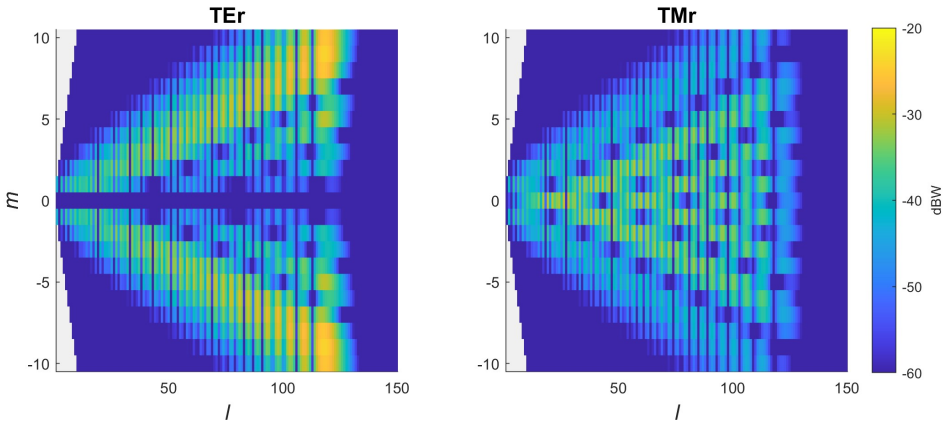
Before optimizing to best align the measured probe compensated data, it was noticed that the SWE conversion was sensitive to small alignment errors. In Figure 4.24, it can be seen that even a half-degree offset in the y-axis had a significant impact on the power distribution of the SWE modes, that is, the power has been smeared out on the modes.



(a) No rotation



(b) 0.5° rotation



(c) 5° rotation

Figure 4.24: SWE of a y-polarized plane propagating (in the beginning) along the z-axis with different rotations around the y-axis.

4.3.2 Rotation

To align the measured data with the theoretical or simulated data, all of these can be rotated. When the measured data from the vertical feeder were rotated, small artifacts emerged, which can be seen in Figures 4.25 and 4.26. Here, the measured data was rotated 90° around the x-axis and then rotated back to the original alignment.

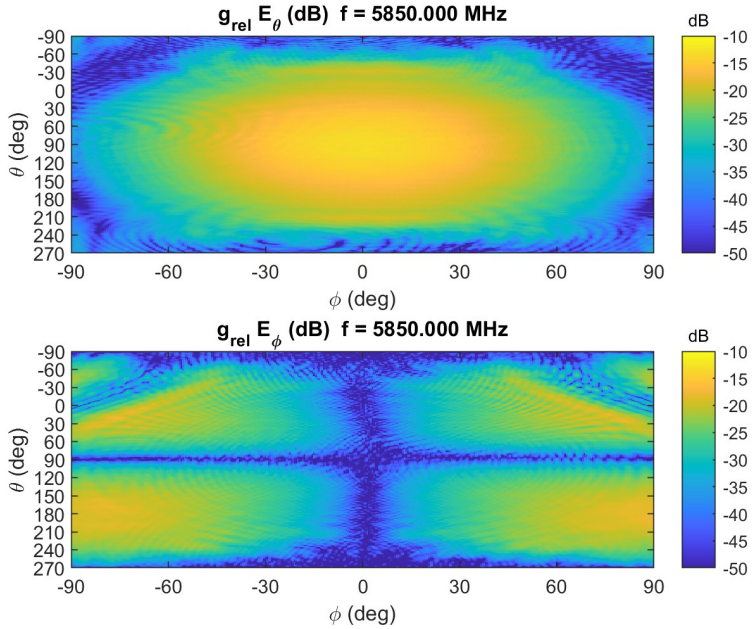


Figure 4.25: A full sphere measurement for the vertical feeder in the CATR without any transformations or rotations.

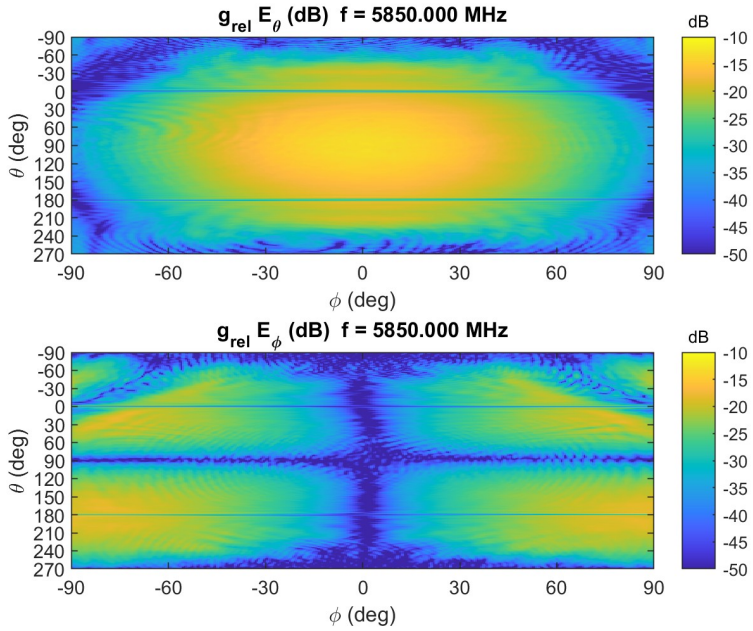


Figure 4.26: A full sphere measurement was conducted for the vertical feeder in the CATR involving two rotations around the x-axis. The first rotation was 90° around the x-axis, followed by a second rotation with the same angle but in the opposite direction, effectively undoing the initial rotation

If the simulated probe pattern from MVG was rotated instead, no artifacts were found, as seen in Figures 4.27 and 4.28. These discoveries led to rotating the measured data as little as possible and rotate the theoretical data instead.

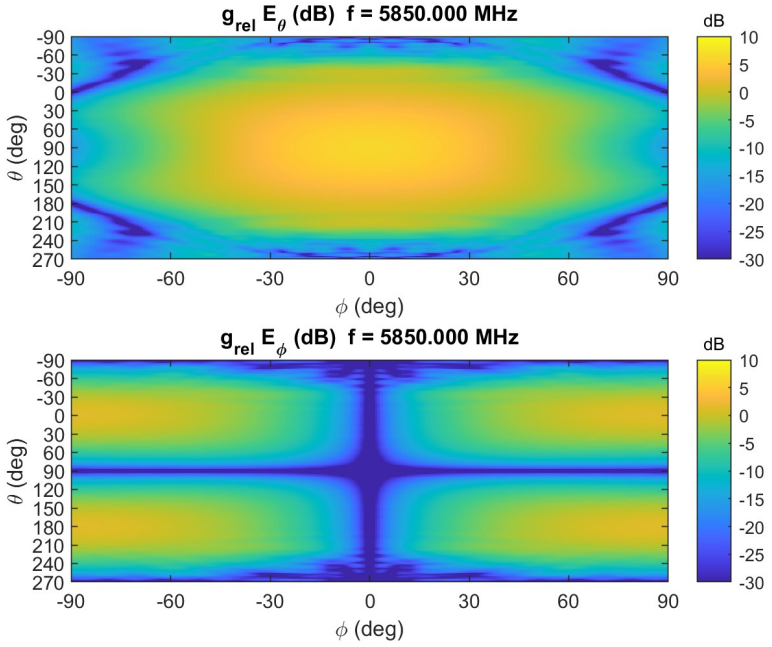


Figure 4.27: A full sphere measurement simulation of the probe pattern from MVG in vertical position without any transformations or rotations.

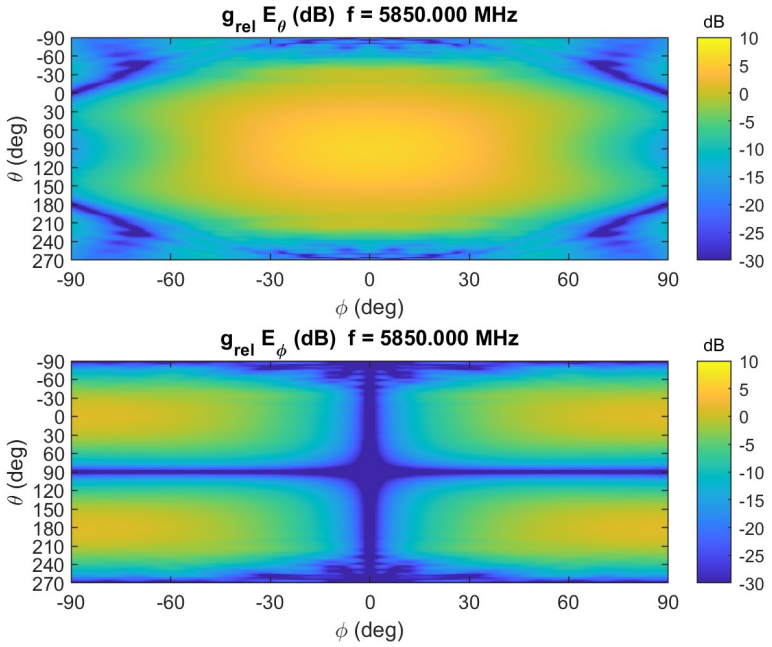


Figure 4.28: A full sphere measurement simulation of the probe pattern from MVG in vertical position with two rotations. The first rotation was 90° around the x-axis, followed by a second rotation with the same angle but in the opposite direction, effectively undoing the initial rotation

4.3.3 Interpolation

After the other discoveries, it was tested if the interpolation applied on the simulated probe pattern from MVG affected the data. In Figures 4.29 and 4.30 the azimuth cut is shown for both the data without any interpolations and in the data with multiple interpolations. After both upscaling and downscaling and then returning to the original amount off points, the interpolations did not have a relevant impact on the data.

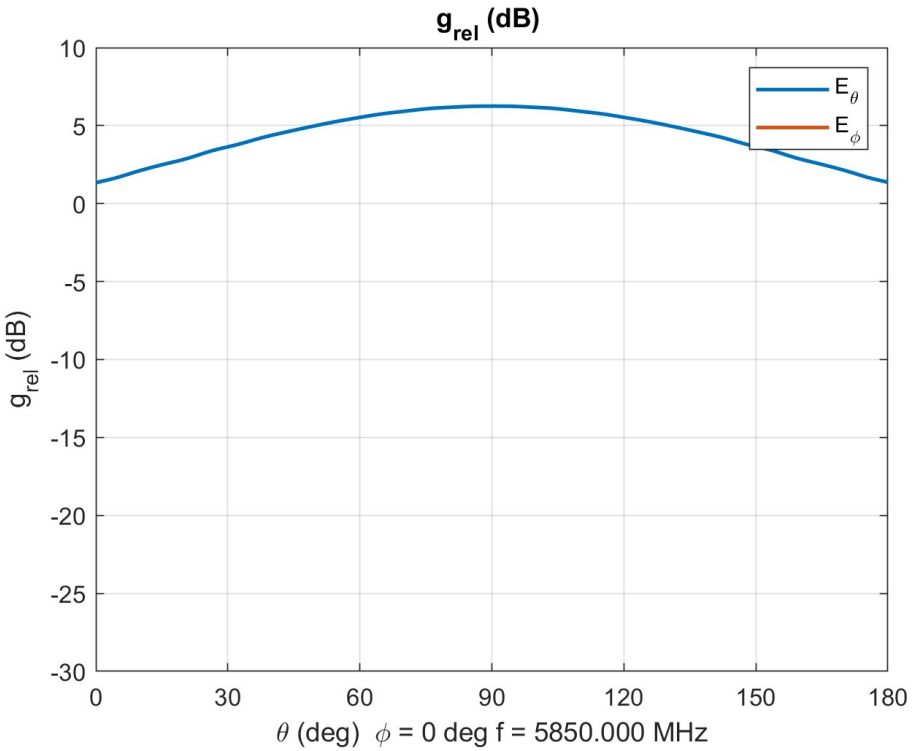


Figure 4.29: Azimuth cut of the simulated probe pattern from MVG, without any interpolation.

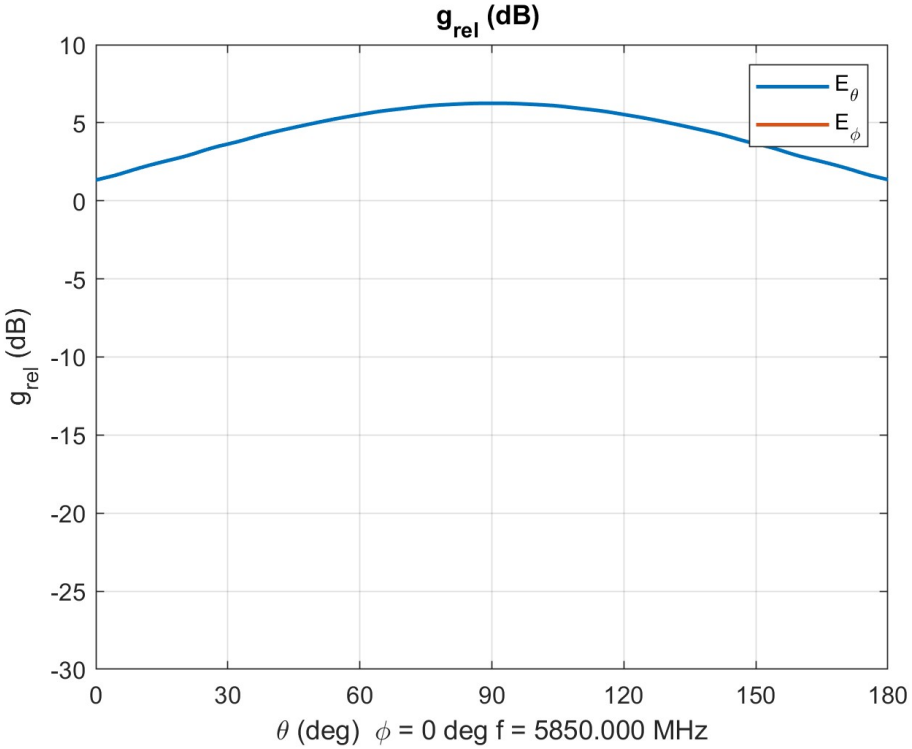


Figure 4.30: An azimuth cut of the simulated probe pattern from MVG was performed, involving three interpolations. The first interpolation was done to upscale the data, followed by a down scaling interpolation, and finally returning to the original amount of points.

4.3.4 Polarization used in probe compensation

One problem with the probe compensation was that it were numerical unstable. This is a consequence of the zero depth of the simulated probe pattern from MVG, because the measured signal is divided by zero, as can be seen in Figure 4.31. To mitigate this, the polarization of the field was transformed to circular polarization during the process instead of theta, phi and then back to theta, phi polarization. Yielding the result in Figure 4.32.

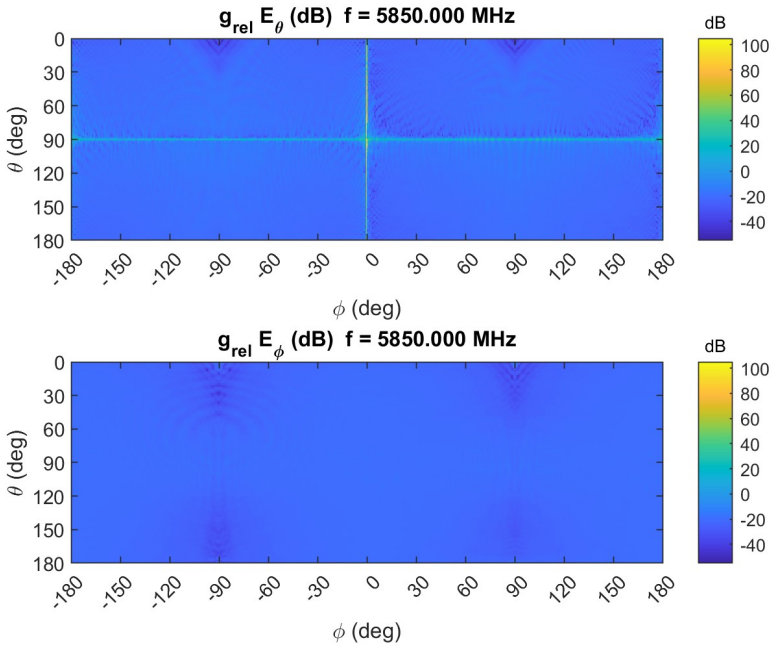


Figure 4.31: Probe compensation without handling the numerical instability which emerges from dividing with zero.

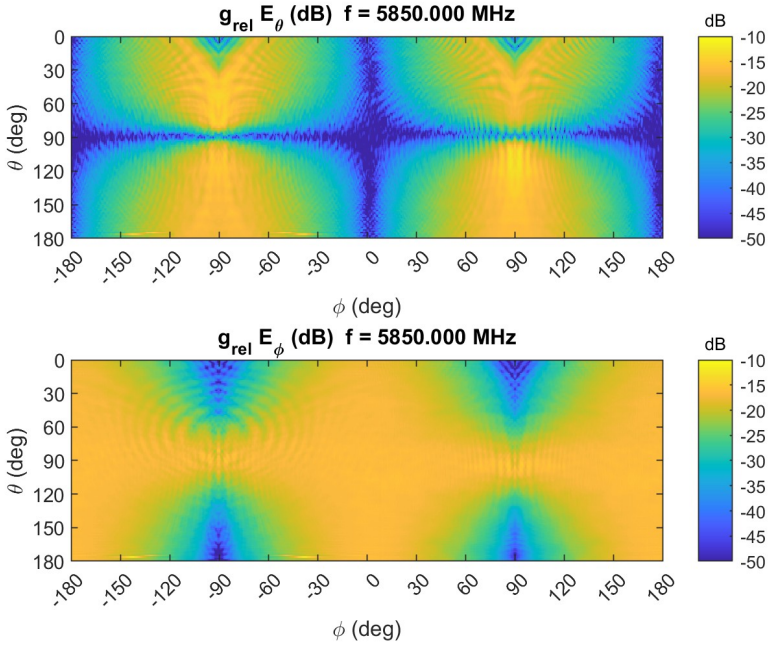


Figure 4.32: Probe compensation with circular polarization trick to compensate for zero depth of the simulated probe pattern from MVG.

4.3.5 Phase drift

After trying to remove the plane wave with the optimization, which can be seen in Figure 4.15, we can see that we only removed the signal in the middle. A similar thing can be observed in Figure 4.33, which used the same cost function as described in Section 3.3 but looking at a different area on the sphere for the remaining power. This yielded that only the outer part was now removed and not the middle part as in Figure 4.15.

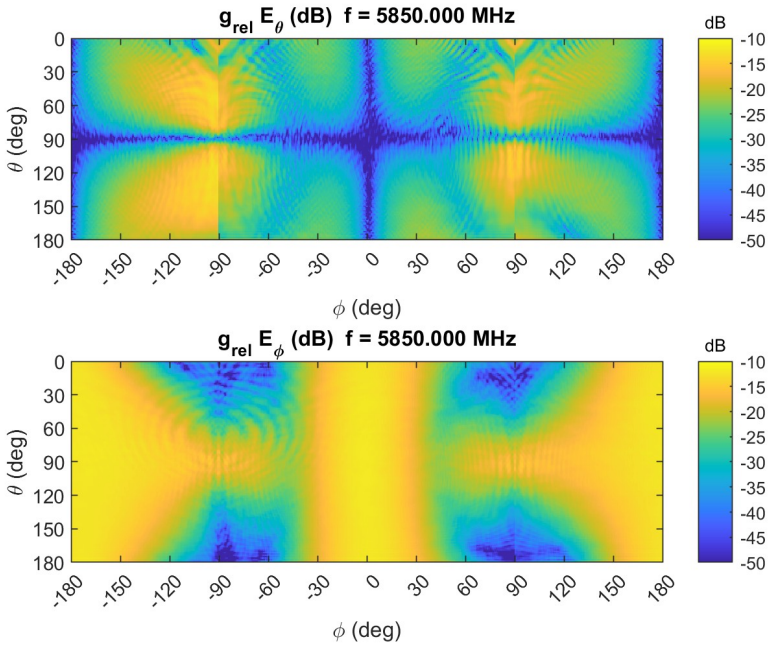
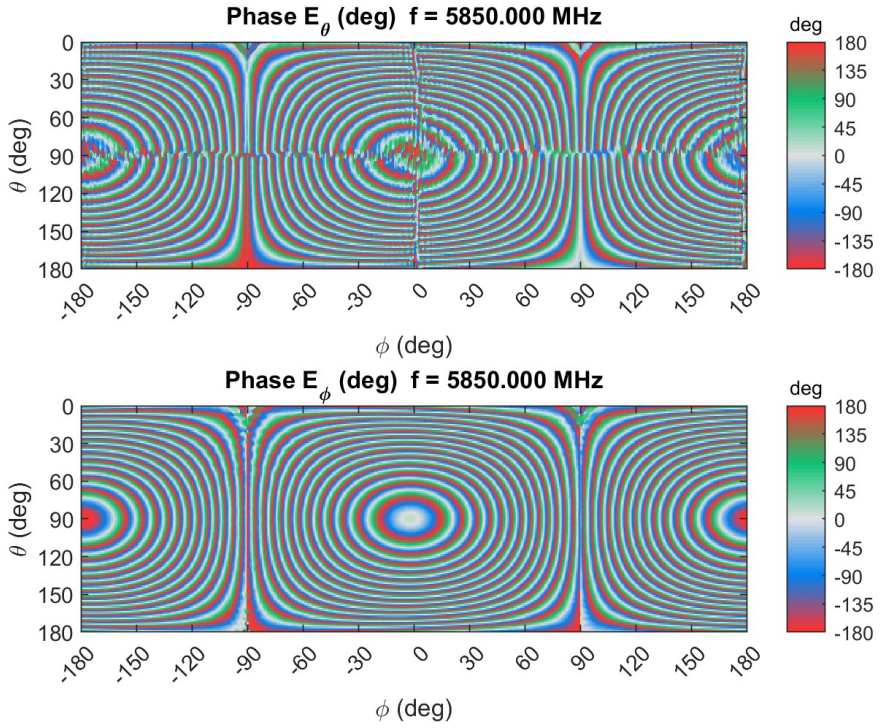
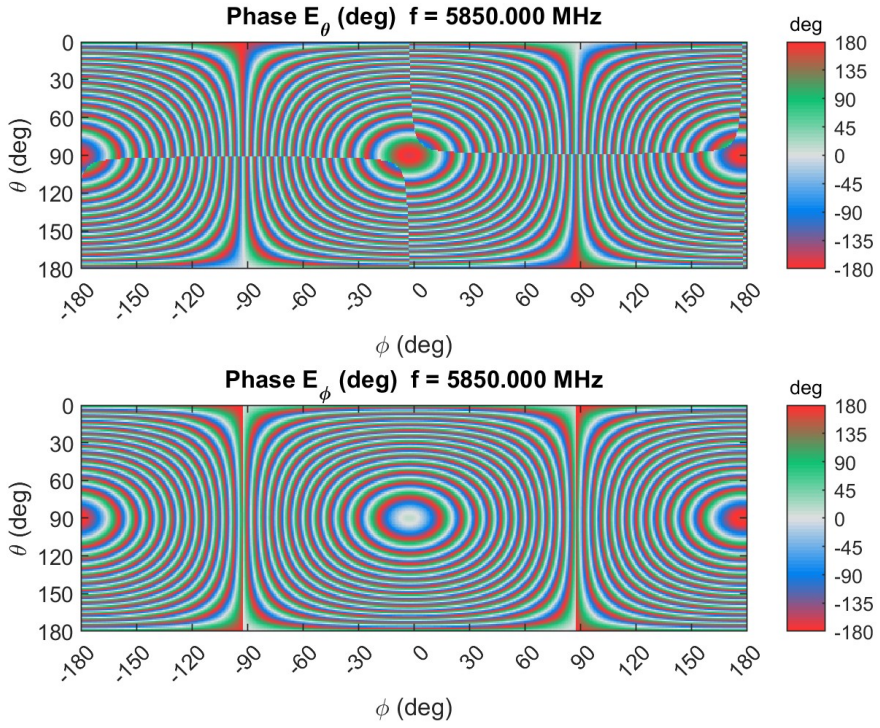


Figure 4.33: Removal of plane wave with another cost function.

This is especially apparent if we look at the phase of the probe compensated signal, and the "optimized plane wave" shown in Figure 4.34. The phase matches up in the middle but starts to deviate from each other at the edges (that is $\varphi = \pm 90°$).



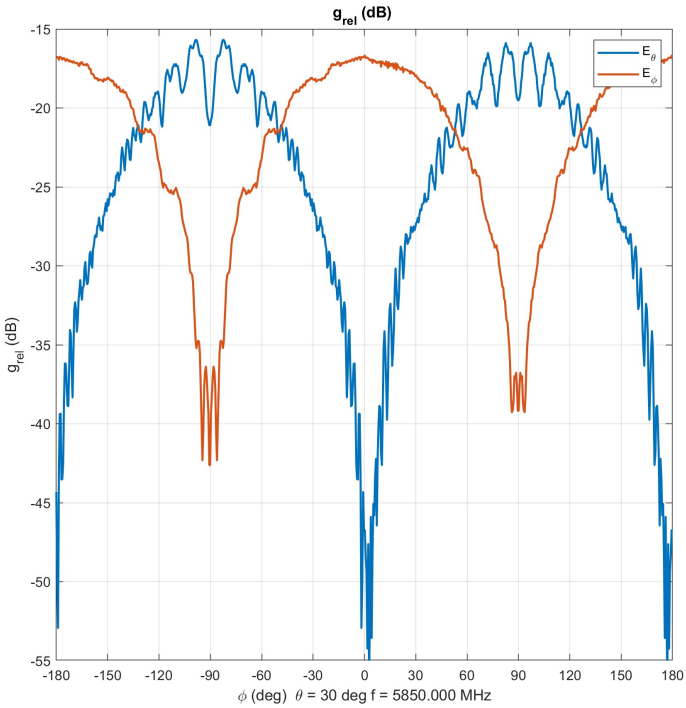
(a) Probe compensated signal



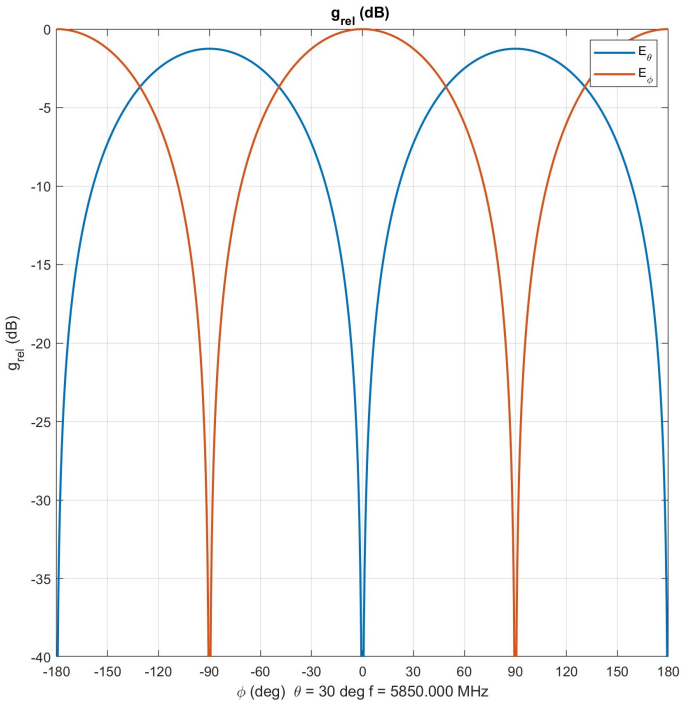
(b) Plane wave

Figure 4.34: Phase comparison of probe compensated signal and plane wave.

If we look at a cross section for the probe compensated signal and a plane wave, Figure 4.35, it seems like the plane wave should be able to remove the compensated cross section. But this is not the case.



(a) Probe compensated signal



(b) Plane wave

Figure 4.35: Cross section at $\theta = 30°$.

5

Conclusion

In this chapter, the result will be discussed and potential errors and inaccuracies of the method will be explored. Ideas of future work will also be stated to improve the method.

5.1 Discussion

The overall results seem to indicate that the method works, at least when we look at the theoretical result, see Section 4.1. But we do not get a good result from real measurement, and the following error sources could be the culprit.

5.1.1 Error sources

There are multiple error sources, both for the measured data and for the post processing. The following will try to explain what may have affect the method negatively.

5.1.1.1 Numerical instability

Because all the measurements were in the far field region, as described in Section 2.8, the probe compensation could be performed by dividing the measured data with the simulated probe pattern from MVG. From observing the simulated probe pattern from MVG in Figure 3.3, it can be seen several places that the pattern approaches zero, leading to numerical instability due to division be zero (or close to zero). Further, this yielded unreasonable results, where the most amount of power is near the lows of the measured signal. To mitigate this, we changed to circular polarization, as described by Section 2.8.2, but that itself created other issues. For example, if there was any unbalance between the polarizations, the conversion to circular polarization will not be accurate. Hence, by solving one problem, we get another problem (or constraint) with the signal.

5.1.1.2 Measurement setup

There were a lot of error sources during the measurement setup, some have been explained in Section 2.5.1.1. The error sources that affected the method the most were:

- Alignment of AUT

- Multi-path
- System Drift

When it came to the alignment of AUT, this affected the result a lot. If the setup is not perfectly aligned, then the electric field will not enter parallel to the AUT. During this report, it was noticed that the SWE conversions from far field were very sensitive to small alignment errors, see Section 4.3.1. This could, in theory, be compensated for when processing the data. But the caveat is numerical instability of the measured data. By rotating the measured signal, the data gets destroyed, see Section 5.1.1.1. The workaround is to do all operations without interpolation or rotation of the signal, but at the end, to analyze the quiet zone plane, we would have to rotate the data to the z-axis which may destroy the data (as described in Section 4.3.2).

For the multi-path, we assumed that the reflected signals had such low power compared to the original signal. If that was the case, then we should only have seen slight variations at the points where the multi-path signal was captured, which should not have affected the analysis that much. But, if the reflected signals had higher powers, that were more comparable with the original signal, then there would have been problems. Then, the solution would be to probe compensate at each point where a signal is incoming, and also remove the signal, that is the "plane wave", in this direction. But as said, we assumed that these multi-path signals had such low power compared to the original signal that they did not affect the result too much.

System drift, which is because of temperature fluctuations, phase drift, mechanical drift, et cetera, are a big part which is hard to compensate for. This would mean that we need more insight in the system than what we have from the measurements. Temperature fluctuations would change the wavelength of the signal (since the air and cables would have different impedances), and to be able to adjust for temperature fluctuations, we need access to the inside temperature of the chamber. For the phase drift, we would need a different measurement which shows how the oscillator drifts during long measurements (since the measurements we used took more than 24 h to take). Then, with this data, we could adjust each point with this drift. For the mechanical drift, which could be a slight drift in the step motors when they move, could be adjusted for if we had a measured double sphere. Hence, we could compare each point of the two spheres (but then the phase drift needs to be adjusted to give meaningful information). As it stands, we neglected all these error sources stated here, since we assumed they are small and since we did not have enough data to analyze this. Moreover, if there was more time, one could also analyze these and try to compensate for them to get better data.

5.1.2 Method

The method we apply in this thesis has never been used. As stated by both Hansens book [3] and M. Dirix and D. Heberling article [17], if we were to use a rotationally symmetric probe then we would only have ± 1 probe coefficient modes (or at least an approximate to these modes). This would have simplified the probe compensation, since we could have used the theory described in Section 2.8.1. In this thesis, we

used an OEW, which is not a rotation symmetric probe. At first glance, this did not seem like a problem since we applied the theory stated in Section 2.8.2. But if we look at the result, when we try to remove the plane wave, as shown in Figure 4.15, something seems to be wrong. As shown in Section 4.3.5, the phase does not match up. Whether this was because of phase drift or probe compensation is unclear since Figure 4.35 indicates that the *far field approach* to the probe compensation, described in Section 2.8.2, yields a good result.

It seems that the assumptions made for both probe compensation methods were too gross. But this is quite hard to evaluate because no one has done the probe compensation this way.

Note that the probe compensated data has been inverted, that is, $E_\theta = E_\varphi$ and vice versa. This was done to match the plane wave, and most likely, emerged when the measured data was loaded. It could also be due to a missing rotation or a typo in the code. Because of time constrains, this was never found out. The assumption here is that it has no impact on the results. See Appendices B.2.1 and B.3 to observe that the old results suggested that we needed to invert the probe compensated data.

5.2 Summary

The method seems promising, but we could not get the result that we wished for. If that depends on probe compensation or the errors stated above is unclear at the moment. Further research needs to be done. But since the method works for a theoretical signal (that is an already probe compensated signal), it would indicate that the method works. The hard part is the probe compensation.

5.3 Future work

The setup we used was not accordingly to Hansens book. The main difference is how the measured data was probe compensated and processed. But a good exercise would be to do it accordingly to Hansens algorithm, and then test the proposed method in this report. If the method works with Hansens algorithm, then the method works. But the way we measured and processed the data is incorrect.

The use of the *alternative method* for the probe compensation, described in Section 2.8.3, is much more rigorous than the method we used. If done correctly, then this would leave no questions or doubts if the method works or not. But one need to ensure that the assumptions made is correct and does not simplify the problem too much.

As stated in Section 5.1.1, some of these error sources could be compensated for. Since we did not have the needed data available to compensate for these error sources, nor the time, it could be worth looking into them and trying to compensate for them.

Bibliography

[1] D. M. Pozar, *Microwave engineering; 4th ed.* Hoboken, NJ: Wiley, 2012, ISBN: 978-0-470-63155-3.

[2] Navipedia, *Military applications.* https://gssc.esa.int/navipedia/index.php/Military_Applications, [Online; accessed 18 Jan-2024].

[3] J. E. Hansen, Ed., *Spherical Near-field Antenna Measurements* (Electromagnetic Waves). Institution of Engineering and Technology, 1988. DOI: 10.1049/PBEW026E.

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

[5] Ag2gaeh, *Unit vectors in spherical coordinates*, https://en.wikipedia.org/wiki/Spherical_coordinate_system#/media/File:Kugelkoord-lokb-e.svg, [Online; accessed 17 Apr-2024].

[6] J. A. Stratton, *Electromagnetic Theory.* McGraw-Hill Companies (New York), 1941, ISBN: 0470131535.

[7] F. Jensen, "Electromagnetic near-field - far-field correlations," 1970.

[8] S. L. Belousov, Ed., *Tables of Normalized Associated Legendre Polynomials.* Pergamon Press, Oxford, 1962.

[9] Catherine G. Manning, *Field probes*, https://www.nasa.gov/general/what-is-an-antenna/, [Online; accessed 14 may-2024].

[10] Edward Collett, *Field guide to polarization*, https://spie.org/publications/spie-publication-resources/optipedia-free-optics-information/fg05_p12-14_stokes_polarization_parameters, [Online; accessed 16 may-2024].

[11] Antenna Test Lab, *What is an anechoic chamber?* https://antennatestlab.com/antenna-education-tutorials/what-is-an-anechoic-chamber, [Online; accessed 23 Jan-2024].

[12] Microwave Vision Group, *Compact antenna test range*, https://www.mvg-world.com/en/products/antenna-measurement/compact-ranges/compact-antenna-test-range, [Online; accessed 29 May-2024].

[13] J. McCormick, J. Boyce, J. Sayers, and J. Murray, "The impact on measurement accuracy of specifying a compact antenna test range for high power testing," *2nd European Conference on Antennas and Propagation (EuCAP 2007)*, Dec. 2007. DOI: 10.1049/ic.2007.1243.

[14] Institute of High Frequency Technology, *Compact range*, https://www.ihf.rwth-aachen.de/en/research/research-topics/antenna-measurement/compact-range, [Online; accessed 2 Feb-2024].
[15] Microwave Vision Group, *Large or small catr - quiet zone quality is critical*, https://www.mvg-world.com/en/news/large-or-small-catr-quiet-zone-quality-is-critical, [Online; accessed 2 Feb-2024].
[16] Microwave Vision Group, *Field probes*, https://www.mvg-world.com/media/992/download/reference, [Online; accessed 2 Feb-2024].
[17] M. Dirix and D. Heberling, "Qualitative quiet zone comparison using spherical near-field scanning," Nov. 2012, pp. 1–4, ISBN: 978-1-4673-2218-8. DOI: 10.1109/LAPC.2012.6402969.
[18] Jonas Fridén, *Probe compensation for QZ spherical scanning*, unpublished, [Given; may-2024].
[19] Jonas Fridén, *Mirror field*, unpublished, [Given; may-2024].
[20] Microwave Vision Group, *Open-ended waveguides*, https://www.mvg-world.com/en/products/antennas/measurement-probes-and-feeds/open-ended-waveguides, [Online; accessed 13 Mar-2024].

A

Vector identities

Scalar product

$$\begin{aligned} \boldsymbol{A}\cdot\boldsymbol{B} &= |\boldsymbol{A}|\cdot|\boldsymbol{B}|\cos(\boldsymbol{A},\boldsymbol{B}) = A_xB_x + A_yB_y + A_zB_z \\ |\boldsymbol{A}| &= \sqrt{A_x^2 + A_y^2 + A_z^2} \end{aligned} \tag{A.1}$$

Vector product

$$|\boldsymbol{A}\times\boldsymbol{B}| = |\boldsymbol{A}||\boldsymbol{B}|\sin(\boldsymbol{A},\boldsymbol{B}) \tag{A.2a}$$

$$\boldsymbol{A}\times\boldsymbol{B} = -\boldsymbol{B}\times\boldsymbol{A} = \begin{vmatrix} \hat{\boldsymbol{x}} & \hat{\boldsymbol{y}} & \hat{z} \\ A_x & A_y & A_z \\ B_x & B_y & B_z \end{vmatrix} =$$

$$= (A_yB_z - A_zB_y)\,\hat{\boldsymbol{x}} + (A_zB_x - A_xB_z)\,\hat{\boldsymbol{y}} + (A_xB_y - A_yB_x)\,\hat{z} \tag{A.2b}$$

$$\boldsymbol{A}\cdot(\boldsymbol{B}\times\boldsymbol{C}) = \boldsymbol{B}\cdot(\boldsymbol{C}\times\boldsymbol{A}) = \boldsymbol{C}\cdot(\boldsymbol{A}\times\boldsymbol{B}) \tag{A.2c}$$

$$(\boldsymbol{A}\times\boldsymbol{B})\cdot(\boldsymbol{C}\times\boldsymbol{D}) = (\boldsymbol{A}\cdot\boldsymbol{C})(\boldsymbol{B}\cdot\boldsymbol{D}) - (\boldsymbol{A}\cdot\boldsymbol{D})(\boldsymbol{B}\cdot\boldsymbol{C}) \tag{A.2d}$$

$$(\boldsymbol{A}\times\boldsymbol{B})\times\boldsymbol{C} = \boldsymbol{B}(\boldsymbol{A}\cdot\boldsymbol{C}) - \boldsymbol{A}(\boldsymbol{B}\cdot\boldsymbol{C}) \tag{A.2e}$$

$$\boldsymbol{A}\times(\boldsymbol{B}\times\boldsymbol{C}) = \boldsymbol{B}(\boldsymbol{A}\cdot\boldsymbol{C}) - \boldsymbol{C}(\boldsymbol{A}\cdot\boldsymbol{B}) \tag{A.2f}$$

Miscellaneous formulae involving $\nabla = \hat{x}\frac{\partial}{\partial x} + \hat{y}\frac{\partial}{\partial y} + \hat{z}\frac{\partial}{\partial z}$

$$\Delta f = \nabla^2 f = \nabla\cdot(\nabla f) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} \quad \text{(Laplace operator)} \tag{A.3a}$$

$$\nabla\times(\nabla f) = \boldsymbol{0} \tag{A.3b}$$

$$\nabla\cdot(\nabla\times\boldsymbol{A}) = 0 \tag{A.3c}$$

$$\nabla\times(\nabla\times\boldsymbol{A}) = \nabla(\nabla\cdot\boldsymbol{A}) - \nabla^2\boldsymbol{A} \tag{A.3d}$$

$$\nabla\cdot(f\boldsymbol{A}) = (\nabla f)\cdot\boldsymbol{A} + f(\nabla\cdot\boldsymbol{A}) \tag{A.3e}$$

$$\nabla\times(f\boldsymbol{A}) = f(\nabla\times\boldsymbol{A}) + (\nabla f)\times\boldsymbol{A} \tag{A.3f}$$

$$\nabla\cdot(\boldsymbol{A}\times\boldsymbol{B}) = \boldsymbol{B}\cdot(\nabla\times\boldsymbol{A}) - \boldsymbol{A}\cdot(\nabla\times\boldsymbol{B}) \tag{A.3g}$$

$$\nabla\times(\boldsymbol{A}\times\boldsymbol{B}) = (\boldsymbol{B}\cdot\nabla)\boldsymbol{A} - (\boldsymbol{A}\cdot\nabla)\boldsymbol{B} + \boldsymbol{A}(\nabla\cdot\boldsymbol{B}) - \boldsymbol{B}(\nabla\cdot\boldsymbol{A}) \tag{A.3h}$$

$$\nabla(\boldsymbol{A}\cdot\boldsymbol{B}) = (\boldsymbol{A}\cdot\nabla)\boldsymbol{B} + (\boldsymbol{B}\cdot\nabla)\boldsymbol{A} + \boldsymbol{A}\times(\nabla\times\boldsymbol{B}) + \boldsymbol{B}\times(\nabla\times\boldsymbol{A}) \tag{A.3i}$$

$$\operatorname{grad} f = \nabla f \quad \operatorname{div}\boldsymbol{A} = \nabla\cdot\boldsymbol{A} = \frac{\partial A_x}{\partial x} + \frac{\partial A_y}{\partial y} + \frac{\partial A_z}{\partial z} \tag{A.3j}$$

$$\operatorname{curl}\boldsymbol{A} = \operatorname{rot}\boldsymbol{A} = \nabla\times\boldsymbol{A} \tag{A.3k}$$

B

Old results

Here, old results used to motivate some choice made in this Master's thesis will be presented

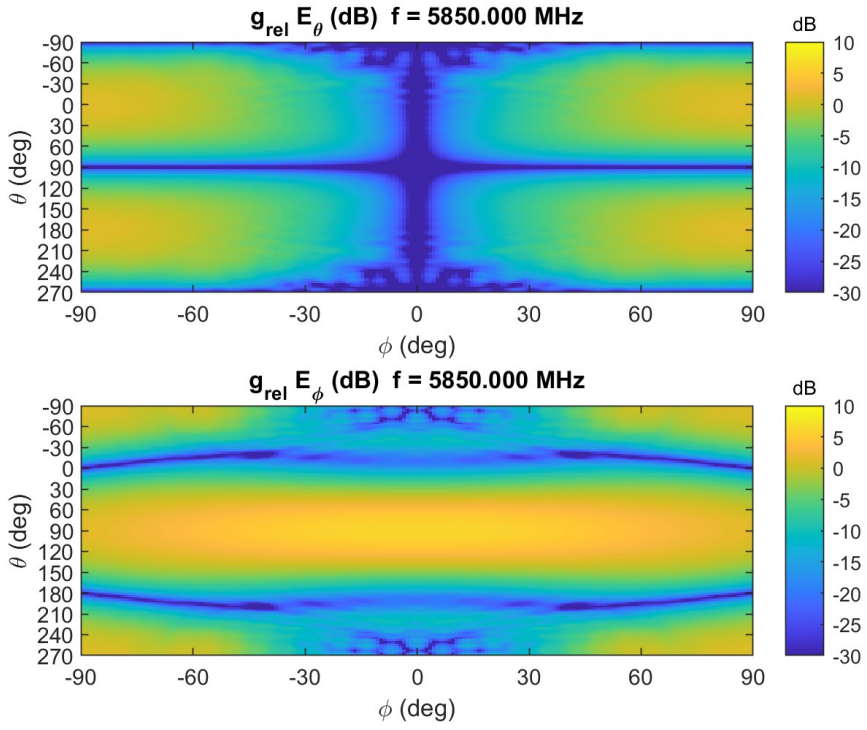
B.1 Alternative approach

Old results relevant to the alternative approach can be found here.

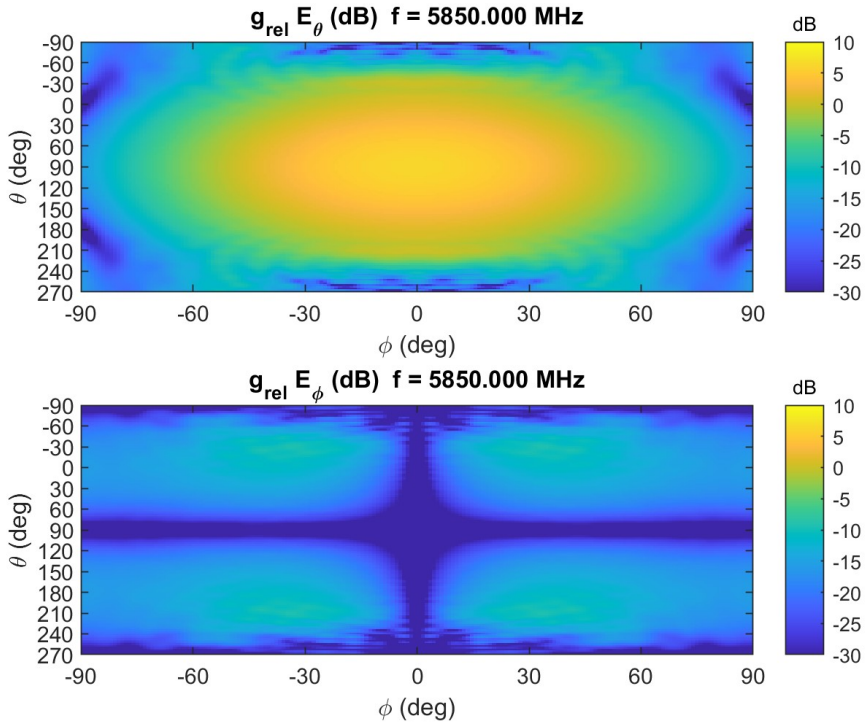
B.2 Sign of E_θ and E_φ

In the section, the old result that was used to see what sign the different components of the probe pattern should have for the alternative approach described in Section 2.8.3, will be presented.

In, Figure B.1 we can see the results of the rotation of the simulated probe pattern for the alternative approach when both E_θ and E_φ are positive. While in Figure B.1 presents, the results of the same rotation bit when E_θ are positive and E_φ are negative. Lastly, in Figure B.1 the results if we swap the signs between E_θ and E_φ, that is, E_θ are negative and E_φ are positive. From these figures one can assume that both E_θ and E_φ sohuld be positive.

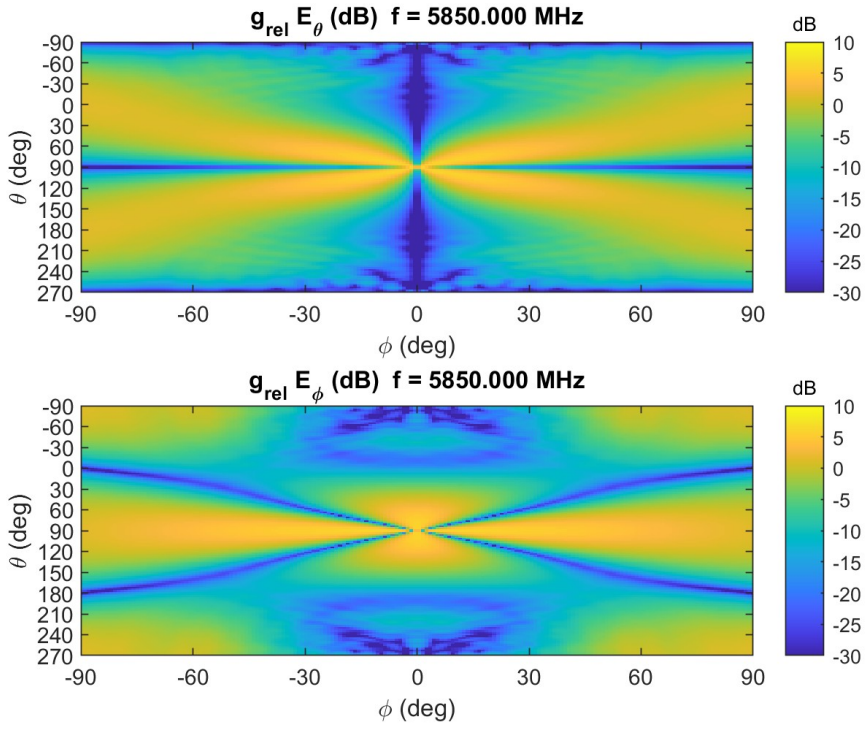


(a) H-polarization

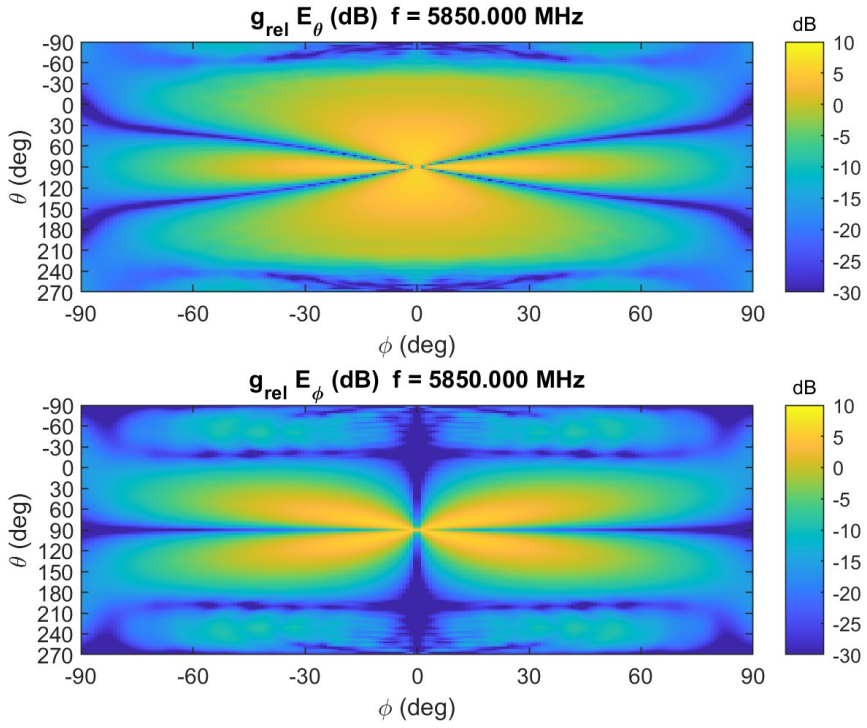


(b) V-polarization

Figure B.1: H- and V-polarization of probe pattern for alternative approach. (Both E_θ, E_φ are positive and in the probe coordinates system)

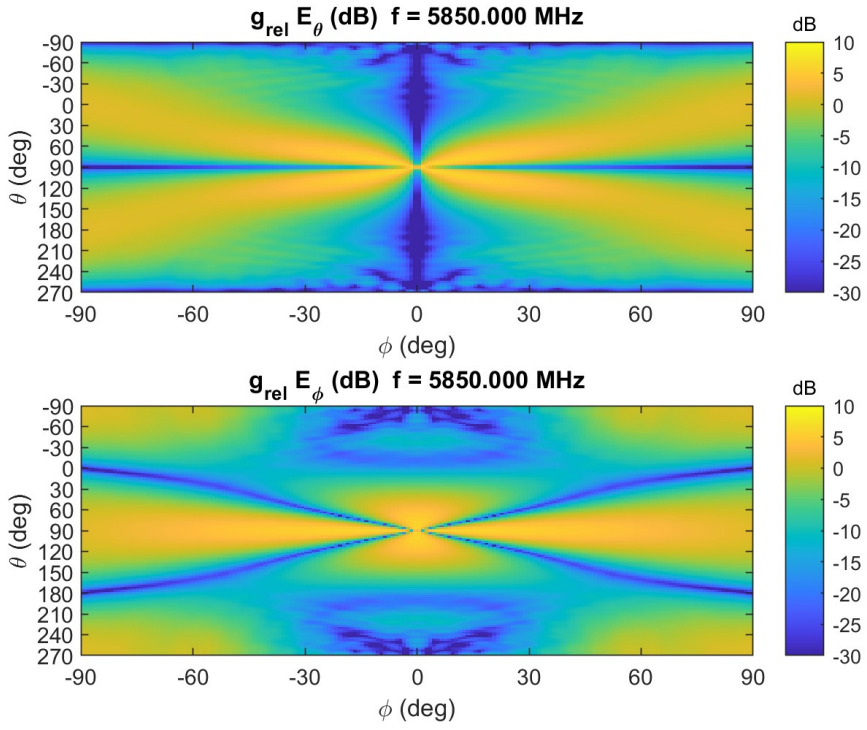


(a) H-polarization

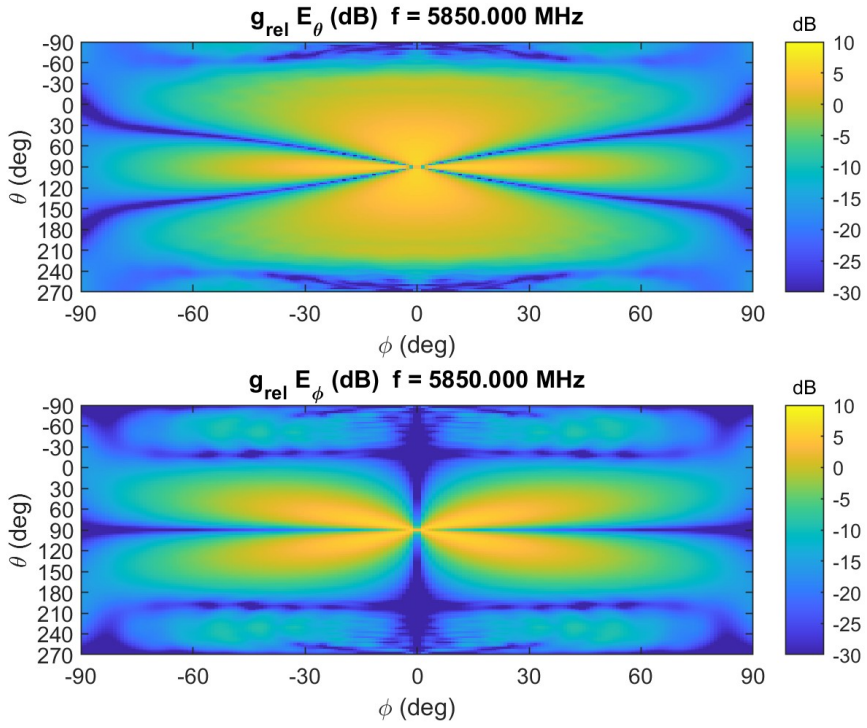


(b) V-polarization

Figure B.2: H- and V-polarization of probe pattern for alternative approach. (Here E_θ are positive, E_φ are negative and in the probe coordinates system)



(a) H-polarization



(b) V-polarization

Figure B.3: H- and V-polarization of probe pattern for alternative approach. (Here E_θ are negative, E_φ are positive and in the probe coordinates system)

B.2.1 Probe compensation

From comparing the Figures B.4 and B.5 one can see that we have to swap the components E_θ and E_φ of the probe compensated data to be able to get similar characteristics as the plane wave.

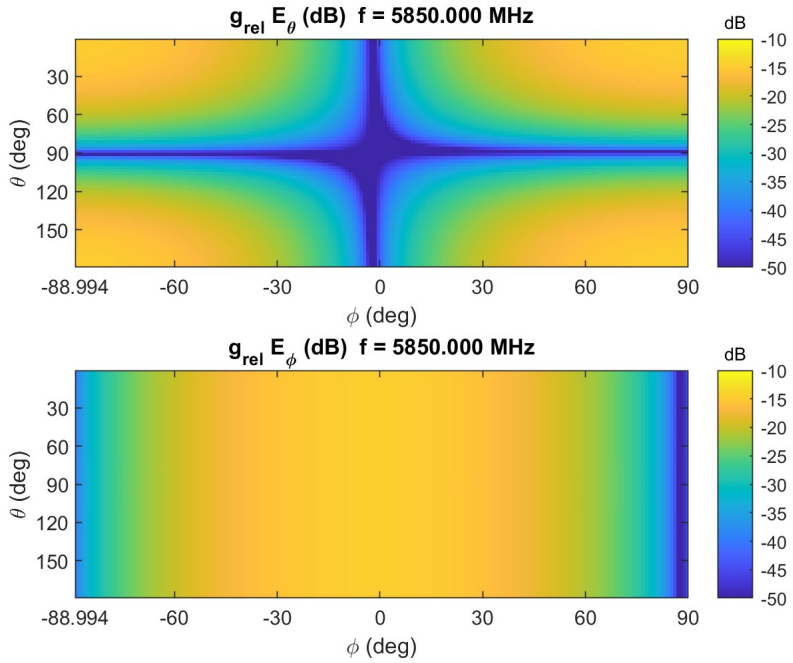


Figure B.4: Theoretical generated y-polarized plane wave propagating in z direction.

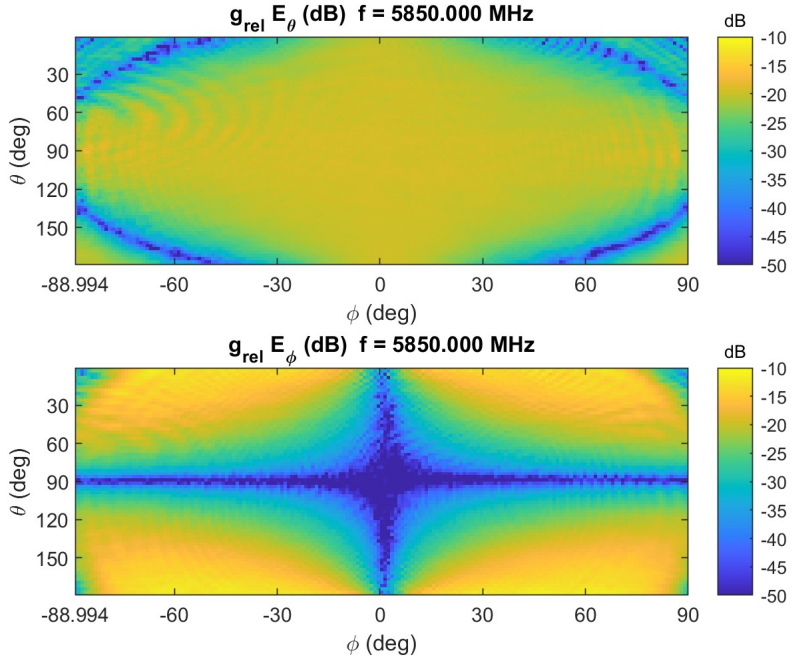


Figure B.5: Probe compensated data.

B.3 Far field approach

From comparing the Figures B.6 and B.7 one can see that we have to swap the components E_θ and E_φ of the probe compensated data to be able to get similar characteristics as the plane wave.

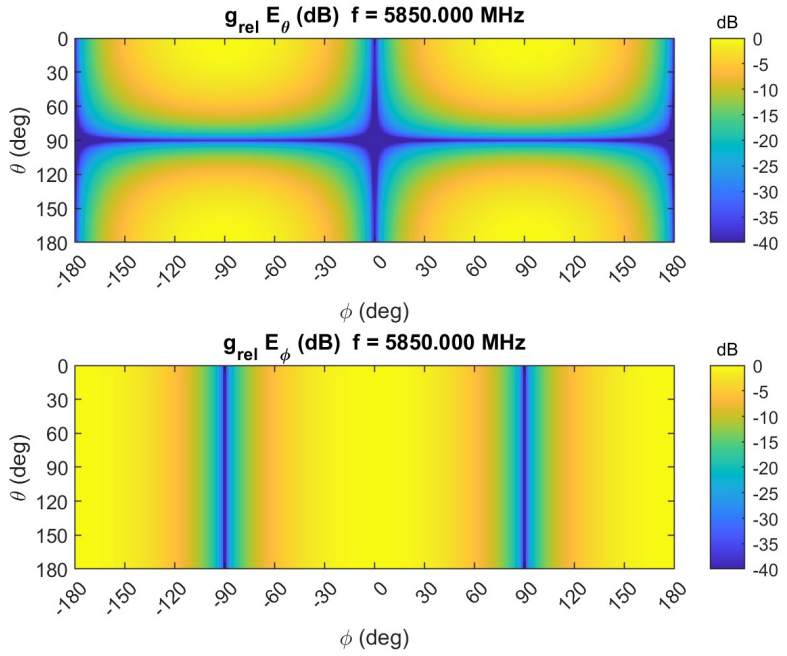


Figure B.6: Theoretical generated y-polarized plane wave propagating in z direction.

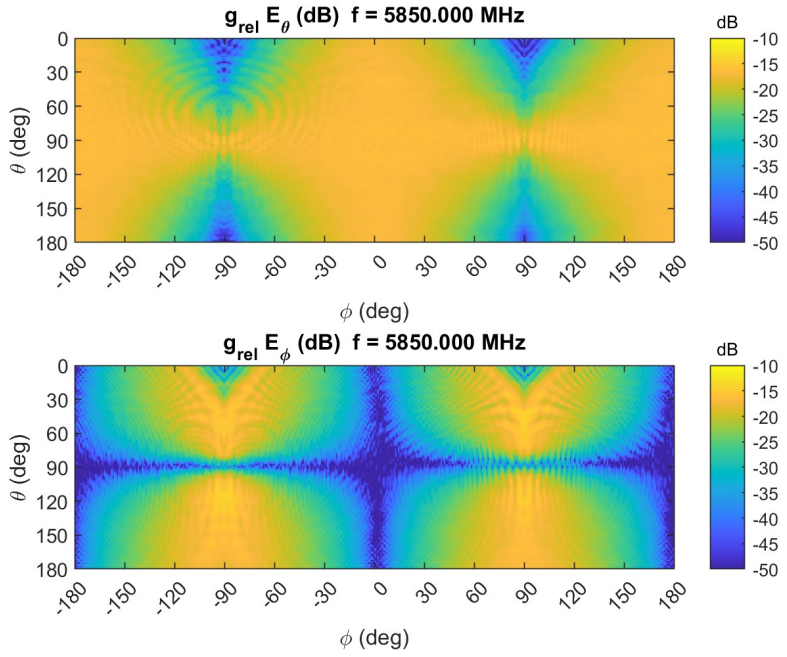


Figure B.7: Probe compensated data.

C

MATLAB code

Note that only the MATLAB codes written by us will be included here, all other code from Ericsson MATLAB library will not be displayed or explained as they are a property of Ericsson AB. Note that all functions with the name "eamlib_..." are functions from the Ericsson MATLAB library.

C.1 Get QZ plane

```
function getQZplane(field, r0, varargin)

% getQZplane(field, propagation_direction, radius)
%
% Calculates the cross section of spheres
%
% This plane is useful for evaluating the quiet zone by
% reconstructing spehres with different radius from a measurement
% and plots the amplitude and phase in this plane
%
% INPUT
%           field       Eamlib struct of input spherical measured
   field (farfield)
%                       Which is probe compensated
%           r0          Radius of measurement sphere
%           varargin    options like distance, prop_dir, noSphere and
%                       dbLimit. See description below
%
% OUTPUT
%           Plot of amplitude and phase at desired plane
%
% VARAGIN
%           distance    Which plane should be calculated, origin at
   center
%                       of sphere
%           prop_dir    Direction of propagation of plane wave
%           noSpheres   Number of spheres to be calculated
%           dblimit     Limit of colorbar in plot
```

```
% See also, probeCompensate_farfield, reconstruct_field
%
%% Example A
% field.type = 'farfield';
% field.f = 5.85e9;
% field.theta = linspace(0,pi,301)';
% field.phi =  linspace(-pi,pi,151)';
% [theta,phi] = ndgrid(field.theta,field.phi);
% r0 = 0.5; k = 2.*pi.*field.f./c_0;
% field.E_phi = (cos(phi)).*exp(-1i*k*cos(theta)*r0);
% field.E_theta = (cos(theta).*sin(phi)).*exp(-1i*k*cos(theta)*r0);
% field = eamlib_flatten_comp(field);
% getQZplane(field, r0)
%%
% Original code by ealdfil & ecarrik
% Supervised by Jonas Fridén and Anders Jernberg
% Copyright Ericsson AB 2024

%% Validate input variables
if strcmp(field.type,{'farfield'})

    if  isreal(r0) && r0 > 0

        default.distance = 0;
        validate.distance = @(x) isreal(x) & r0 > abs(x);

        default.prop_dir = 'z';
        validate.prop_dir = @(x) any(strcmp(x,{'x', 'y', 'z'}));

        default.noSpheres = 5;
        validate.noSpheres = @(x) isint(x) && x > 0;

        default.dbLimit = 40;
        validate.dbLimit = @(x) isreal(x) && x > 0;

        [distance,prop_dir,noSpheres,dbLimit] = ...
            eamlib_parseInput(varargin, default, validate);
    else
        error('eamlib:invalidInput',[' r0 need to be real positve
           number'])
    end
else
    error('eamlib:invalidInput',[ mfilename ' operates only on
       eamstructs of type farfield'])
end
%%
```

```
71
72  % Creates figures
73  E_x = figure; E_y = figure; E_z = figure;
74
75  % Switch case for the different propagation directions of the plane
        wave
76  switch prop_dir
77      case 'x'
78          % TODO: Fix this
79          error('eamlib:doesNoteExist','The option does not exist at
                the moment, please use eamlib_rotate so the field
                propagtes in z-direction')
80      case 'y'
81          % TODO: Fix this
82          error('eamlib:doesNoteExist','The option does not exist at
                the moment, please use eamlib_rotate so the field
                propagtes in z-direction')
83      case 'z'
84
85          % Calculate the SWE of the input field
86          swe_field = sft2swe(farfield2sft(field));
87
88          % TODO: Change this
89          % Truncataion of modes
90          swe_field = TruncModes(swe_field,150, 20);
91
92          % Calculate the separation of spheres
93          separation = (r0 - abs(distance)) / (noSpheres);
94
95          % Calculate the spheres
96          for i = 1:noSpheres
97
98              % Radius of the sphere
99              radiusNewSphere = abs(distance) + separation * i;
100
101             % Needed theta for the desired plane
102             theta = acos(distance / radiusNewSphere);
103
104             % Check if theta exists
105             if ~isempty(find(field.theta==theta))
106
107                 % Reconstruct sphere at radius
108                 sphere = reconstruct_field(swe_field,radiusNewSphere,
                        r0,field.theta,field.phi);
109             else
110
```

```
111                 % Find closest value to theta
112                 [~,closestIndex] = min(abs(field.theta-theta));
113
114                 % Check if desired theta value should be to right or
                        left in
115                 % the vector of the closest value
116                 if field.theta(closestIndex)-theta < 0
117                     ltheta = [field.theta(1:closestIndex); theta;
                            field.theta(closestIndex+1:end)];
118                 else
119                     ltheta = [field.theta(1:closestIndex-1); theta;
                            field.theta(closestIndex:end)];
120                 end
121
122                 % Reconstruct sphere at radius
123                 sphere = reconstruct_field(swe_field,radiusNewSphere,
                        r0,ltheta,field.phi);
124             end
125
126             % Extract the data points for plane at initial sphere
127             plane = circle(sphere, radiusNewSphere, distance,
                    prop_dir);
128
129             % Calculate the circle where points are at
130             r_circle = radiusNewSphere*sin(plane.theta);
131             x = r_circle * cos(plane.phi);
132             y = r_circle * sin(plane.phi);
133
134             % Plot amplitude and phase of E_x component
135             plotComp(E_x,plane.E_x,x,y)
136
137             % Plot amplitude and phase of E_y component
138             plotComp(E_y,plane.E_y,x,y)
139
140             % Plot amplitude and phase of E_z component
141             plotComp(E_z,plane.E_z,x,y)
142
143         end
144
145 % Add title to plot E_x
146 sgtitle(E_x, ['X-component of electric field at ',num2str(distance),
        ' meters with ',num2str(noSpheres) ' spheres'])
147 % Add plot settings for E_x
148 addPlotSettings(E_x,dbLimit,prop_dir)
149
150 % Add title to plot E_y
```

```
151 sgtitle(E_y, ['Y-component of electric field at ',num2str(distance),
        ' meters with ',num2str(noSpheres) ' spheres'])
152 % Add plot settings for E_y
153 addPlotSettings(E_y,dbLimit,prop_dir)
154
155 % Add title to plot E_z
156 sgtitle(E_z, ['Z-component of electric field at ',num2str(distance),
        ' meters with ',num2str(noSpheres) ' spheres'])
157 % Add plot settings for E_z
158 addPlotSettings(E_z,dbLimit,prop_dir)
159
160 end
161 end
162
163 %% Extract points in desired plane
164 function circle = circle(field, radius, distance, prop_dir)
165
166 % circle = circle(field, radius, distance, prop_dir)
167 %
168 % Extract the electric field components in a circle, which lies in
        the
169 % desired plane from the spherical field
170 %
171 %
172 % INPUT
173 %           field       Eamlib struct of type spherical farfield
174 %           radius      Radius of spherical field
175 %           distance    Which plane should be calculated, origin at
        center
176 %                       of sphere
177 %           prop_dir    Direction of propagation of plane wave
178 %
179 % OUTPUT
180 %           circle      Circle from the spherical field in the
        desired
181 %                       plane
182 %
183 % The output components are in cartesian coordinates, which were
        converted
184 % from spherical coordiantes. Notice that the input field is a
        spherical
185 % field, which implies that the E_r component is important.
186 %
187 % Original code by ealdfil & ecarrik
188 % Supervised by Jonas Fridén and Anders Jernberg
189 % Copyright Ericsson AB 2024
```

```
190 %%
191
192 % Unflatten the input field to make the matrix multiplication work
193 field = eamlib_unflatten_comp(field,'true');
194
195 % Save input field variables for easier handeling
196 % Theta values
197 theta = field.theta;
198 % Theta index
199 index_theta = 1:length(theta);
200 % Phi values
201 phi = field.phi;
202 % Phi index
203 index_phi = 1:length(phi);
204 % Radial component of electric field
205 E_r = field.E_r;
206 % Theta component of electric field
207 E_theta = field.E_theta;
208 % Phi component of electric field
209 E_phi = field.E_phi;
210
211 % Calculate which angle for the desired plane
212 angle = acos(distance/radius);
213
214 % Initiate theta, phi
215 circle.phi = field.phi;
216 circle.theta = angle;
217
218 switch prop_dir
219     case 'x'
220         % Find which points (i.e. what index of phi) lies in the
              plane
221         % TODO: Fix this
222         error('eamlib:doesNoteExist','The option does not exist at
              the moment, please use eamlib_rotate so the field
              propagtes in z-direction')
223     case 'y'
224         % Find which points (i.e. what index of phi) lies in the
              plane
225         % TODO: Fix this
226         error('eamlib:doesNoteExist','The option does not exist at
              the moment, please use eamlib_rotate so the field
              propagtes in z-direction')
227     case 'z'
228         % Find which points (i.e. what index of theta) lies in the
              plane
```

```
229         index_theta = find(field.theta==angle);
230         theta = theta(index_theta);
231 end
232
233 % Create theta, phi grid
234 [theta, phi] = ndgrid(theta,phi);
235
236 % Transform from spherical to cartesian field, using
237 % x = sin(th)*cos(phi)*r_comp + cos(th).*cos(phi)*th_comp - sin(phi)*
238 % phi_comp
239 % y = sin(th)*sin(phi)*r_comp + cos(th).*sin(phi)*th_comp + cos(phi)*
240 % phi_comp
241 % z = cos(theta)*r_comp - sin(theta)*th_comp;
242 % th = theta
243
244 % Calculations for E_x
245 circle.E_x =  E_r(index_theta,index_phi).*sin(theta).*cos(phi) + ...
246     E_theta(index_theta,index_phi).*cos(theta).*cos(phi) - ...
247     E_phi(index_theta,index_phi).*sin(phi);
248
249 % Calculations for E_y
250 circle.E_y = E_r(index_theta,index_phi).*sin(theta).*sin(phi) + ...
251     E_theta(index_theta,index_phi).*cos(theta).*sin(phi) + ...
252     E_phi(index_theta,index_phi).*cos(phi);
253
254 % Calculations for E_z
255 circle.E_z = E_r(index_theta,index_phi).*cos(theta) - ...
256     E_theta(index_theta,index_phi).*sin(theta);
257
258 end
259
260 %%
261 function addPlotSettings(fig,dbLimit,prop_dir)
262 % addPlotSettings(fig,dbLimit)
263 %
264 % Help function which sets the plot settings
265 %
266 % INPUT
267 %           fig         Name of matlab figure you want to plot on
268 %           dbLimit     Limit of colorbar in plot
269 %           prop_dir    Direction of propagation of plane wave
270 %
271 % This function plots sets the plot settings, like axis1, axis2 label
    ,
272 % set colorbar limits, subplot titles and set eamlib phase color
273 %
```

```
% Original code by ealdfil & ecarrik
% Supervised by Jonas Fridén and Anders Jernberg
% Copyright Ericsson AB 2024
%%

set(0,'CurrentFigure',fig);

% Add title and axis of amplitude plot
subplot(1,2,1);
c1 = colorbar;
clim([-dbLimit 1])
c1.Label.String = 'Power [dB]';
title('Amplitude');
xlabel('y [m]');
switch prop_dir
    case 'x'
        % TODO: Fix this
        error('eamlib:doesNoteExist','The option does not exist at
            the moment, please use eamlib_rotate so the field
            propagtes in z-direction')
    case 'y'
        % TODO: Fix this
        error('eamlib:doesNoteExist','The option does not exist at
            the moment, please use eamlib_rotate so the field
            propagtes in z-direction')
    case 'z'
        ylabel('x [m]');
end
% Add title and axis of phase plot
subplot(1,2,2);
c2 = colorbar;
colormap(subplot(1,2,2),eamlib_colormap('phase'))
clim([-180 180])
c2.Label.String = 'Phase [deg]';
title('Phase');
xlabel('y [m]');
switch prop_dir
    case 'x'
        % TODO: Fix this
        error('eamlib:doesNoteExist','The option does not exist at
            the moment, please use eamlib_rotate so the field
            propagtes in z-direction')
    case 'y'
        % TODO: Fix this
        error('eamlib:doesNoteExist','The option does not exist at
            the moment, please use eamlib_rotate so the field
```

```
            propagtes in z-direction')
313     case 'z'
314         ylabel('x [m]');
315 end
316 
317 end
318 %%
319 function plotComp(fig,comp,axis1, axis2)
320 
321 % plotComp(fig,comp,x,y)
322 %
323 % Help function which plots phase and amplitude of electric field
        component
324 %
325 % INPUT
326 %           fig         Name of matlab figure you want to plot on
327 %           comp        Electric field component
328 %           axis1       axis1-coordinates for the plot
329 %           axis2       axis1-coordinates for the plot
330 %
331 % This function plots the amplitude of the electric field component
        and
332 % phase on fig
333 %
334 % Original code by ealdfil & ecarrik
335 % Supervised by Jonas Fridén and Anders Jernberg
336 % Copyright Ericsson AB 2024
337 %%
338 
339 % Set the right figure for electric field component
340 set(0,'CurrentFigure',fig);
341 
342 % Plot amplitude of the electric field component
343 subplot(1,2,1);
344 scatter(axis1,axis2,[],20*log10(abs(comp)),'filled');
345 hold on;
346 
347 % Plot phase of the electric field component
348 subplot(1,2,2);
349 scatter(axis1,axis2,[],angle(comp)*180/pi,'filled');
350 hold on;
351 
352 end
```

C.2 Reconstruct field

```
function newfield = reconstruct_field(swe,r,r0,theta,phi)
% newfield = reconstruct_field(swe,r,r0,ltheta,lphi)
%
% Reconstruct spherical field from given spherical wave expansion at desired
% radius using Vector Spherical harmonics and Spherical radial wave functions
%
% Input
%
%   swe         Spherical wave expansion (eamlib struct)
%   r           Radius of new sphere [m]
%   r0          Radius of old sphere [m]
%   theta       Theta values for the new spherical field
%   phi         Phi values for the new spherical field
%
% Output
%
%   newfield    New spherical field (eamlib struct)
%
%   The output is calculated using the functions VSH and local SRWF
%   function. Where the hankel function has been changed to bessel
%   function of first kind. Because we are using a standing wave finite
%   at orgin, see Hansen (Apendix 1 page 312)
%% Example A
% field.type = 'farfield';
% field.f = 5.85e9;
% field.theta = linspace(0,pi,301)';
% field.phi =  linspace(-pi,pi,151)';
% [theta,phi] = ndgrid(field.theta,field.phi);
% r0 = 0.5; k = 2.*pi.*field.f./c_0;
% field.E_phi = (cos(phi)).*exp(-1i*k*cos(theta)*r0);
% field.E_theta = (cos(theta).*sin(phi)).*exp(-1i*k*cos(theta)*r0);
% field = eamlib_flatten_comp(field);
% swe = sft2swe(farfield2sft(field,0));
% spherical_field = reconstruct_field(swe,r0,r0,field.theta,field.phi)
%%
% Original code by ealdfil & ecarrik
% Supervised by Jonas Fridén and Anders Jernberg
% Copyright Ericsson AB 2024
%% Check Input
if strcmp(swe.type,{'swe'})
```

```
41      if r0 < r
42          warning('eamlib:input', 'This method might not work to
                reconstruct a field outside the orginal sphere')
43      end
44  else
45      error('eamlib:invalidInput',[ mfilename ' operates only on
            eamstructs of type farfield'])
46  end
47
48  %%
49  % Get L and M from swe
50  L=a.l(end);
51  M=swe.m(end);
52
53  % Wave number
54  k = 2*pi*swe.f/c_0;
55
56  % Setting up the new field
57  newfield.type = 'spherical_field';
58  newfield.f = swe.f;
59  newfield.r = r;
60
61  % Setting up theta and phi values and grid
62  newfield.theta = theta;
63  newfield.phi = phi;
64
65  % Initiate matrix (vectors) Et_theta, Et_phi and Er_r
66  Et_theta = zeros(length(theta),length(phi));
67  Et_theta = reshape(Et_theta,[],1);
68
69  Et_phi = zeros(length(theta),length(phi));
70  Et_phi = reshape(Et_phi,[],1);
71
72  Er_r = zeros(length(theta),length(phi));
73  Er_r = reshape(Er_r,[],1);
74
75  for l=1:L
76
77      % Sets global index for TE and TM
78      global_index=swe.l==l;
79
80      % Caclulates radial functions
81      [z1,z2,z3]=SRWF(l,k*r);
82      [z0_1,z0_2,z0_3]=SRWF(l,k*r0);
83
84      % Compensate for old radius
```

```
85      z1 = z1/z0_1;
86      z2 = z2/z0_2;
87      z3 = z3/z0_2;
88
89      % TE Transverse part
90      [Ar,Atheta,Aphi,m,~,~] = VSH(l,1,theta,phi,M);
91
92      % Reshape
93      Atheta = reshape(Atheta,length(m(:,1,1)),[]);
94      Aphi = reshape(Aphi,length(m(:,1,1)),[]);
95
96      % Sum E_theta and E_phi for TE transverse part
97      Et_theta = Et_theta + shiftdim(sum(swe.TE_r(global_index,:).*z1.*
            Atheta));
98      Et_phi = Et_phi + shiftdim(sum(swe.TE_r(global_index,:).*z1.*Aphi
            ));
99
100     % TM Transverse part
101     [Ar,Atheta,Aphi,m,~,~] = VSH(l,2,theta,phi,M);
102
103     % Reshape
104     Atheta = reshape(Atheta,length(m(:,1,1)),[]);
105     Aphi = reshape(Aphi,length(m(:,1,1)),[]);
106
107     % Sum E_theta and E_phi for TM transverse part
108     Et_theta = Et_theta + shiftdim(sum(swe.TM_r(global_index,:).*z2.*
            Atheta));
109     Et_phi = Et_phi + shiftdim(sum(swe.TM_r(global_index,:).*z2.*Aphi
            ));
110
111     % Longitudinal (radial) part
112     [Ar,Atheta,Aphi,m,~,~] = VSH(l,3,theta,phi,M);
113
114     % Reshape
115     Ar = reshape(Ar,length(m(:,1,1)),[]);
116
117     % Sum E_r TM Longitudinal (radial) part
118     Er_r = Er_r  + shiftdim(sum(swe.TM_r(global_index,:).*z3.*Ar));
119 end
120
121 % Set the new fields
122 newfield.E_theta = Et_theta;
123 newfield.E_phi = Et_phi;
124 newfield.E_r = Er_r;
125
126 % Flatten the new field
```

```
127 newfield = eamlib_flatten_comp(newfield);
128 end
129
130 %%
131 function [z1,z2,z3]=SRWF(l,kr)
132 % Removed, property of Ericsson AB
133 end
```

C.3 Far field approach

```
clear matlab
clc, clear all, close all

%% Load and set up the fields

% Get username of user
username = getenv('username');
path2files = append('C:\Users\',username,'\Ericsson\Spherical Probing
    of Quiet Zone - General\Overleaf\test matlab\mätdata\r665mm\');

% config for midas
config.feed.probeV = 'theta';
config.feed.probeH = 'phi';
config.pos.name = 'BBQ90';

% Loading signal and pattern in to matlab
signalV = eamlib_read_midas_data(append(path2files, '
   OEW585V_r665mm_LGF-11-400_IFBW1kHz_RX_2FS.txt'),config);
signalH = eamlib_read_midas_data(append(path2files, '
   OEW585H_r665mm_LGF-11-400_IFBW1kHz_RX_FS.txt'),config);

signalV = signalV.field;
signalH = signalH.field;

% Change sign on phi comp ty coordinate system?
signalH.E_phi = -signalH.E_phi;
signalV.E_phi  = -signalV.E_phi;

% Select frequency 5.85Ghz
signalV = eamlib_select_subset(signalV,'f_ind',1);
signalH = eamlib_select_subset(signalH,'f_ind',1);

% Flip theta so it goes the right way
signalH = eamlib_flipdim(signalH,'theta');
signalH = eamlib_flipdim(signalH,'phi');

signalV = eamlib_flipdim(signalV,'phi');
signalV = eamlib_flipdim(signalV,'theta');

% Select subset for half sphere
signalV = eamlib_select_subset(signalV, 'phi_lim',[-pi*1.001 pi
   *1.001]);
signalV = eamlib_select_subset(signalV,'theta_lim',[-0.0001 pi
   *1.0001]);
```

```
signalH = eamlib_select_subset(signalH, 'phi_lim',[-pi*1.001 pi*1.001]);
signalH = eamlib_select_subset(signalH,'theta_lim',[-0.0001 pi*1.0001]);

signalV_complete = eamlib_select_subset(signalV, 'phi_lim',[-pi*1.001/2 pi*1.001/2]);
signalV_complete = eamlib_select_subset(signalV_complete,'theta_lim',[-0.0001 pi*1.0001]);
signalH_complete = eamlib_select_subset(signalH, 'phi_lim',[-pi*1.001/2 pi*1.001/2]);
signalH_complete = eamlib_select_subset(signalH_complete,'theta_lim',[-0.0001 pi*1.0001]);

%% Signal V load and correct phase (polarisation)

% Combine components from the different measurements
signalV_complete.E_phi = signalH_complete.E_theta;

% Calculate and adjust for phase using stokes parameters
minimum = @(a) sum(abs(2*imag(conj(signalV_complete.E_phi).*signalV_complete.E_theta*exp(1i*a))).^2);
options = optimset('PlotFcns',@optimplotfval,'TolFun',1e-20);
[x, fval, exitflag] = fminsearch(minimum, 0, options);
signalV_complete.E_theta = signalV_complete.E_theta*exp(1i*x);

figure; plot_eamstruct(signalV_complete)

%% Load pattern

% Initiate probe pattern
theo_pattern = grasp2eamstruct(append('C:\Users\',username,'\Ericsson\Spherical Probing of Quiet Zone - General\Measurements\OEW585\grasp\CSTGraspFeed_FFapprox_on_(farfield (f=5.85) [1]_L3-CoCx).cut'));
theo_pattern = Ludwig32thetaphi(theo_pattern,1); %Tried -1, didn't work

%% Signal V probe pattern

% Rotate to SignalV
theo_patternV = theo_pattern;
theo_patternV = eamlib_rotate(theo_patternV,[0,1,0,pi/2]);
theo_patternV = eamlib_rotate(theo_patternV,[1,0,0,pi/2]);

% Select same subset as measured signal
```

```
theo_patternV = eamlib_interp(theo_patternV,signalV.theta,signalV.phi
    );

figure; plot_eamstruct(theo_patternV)

%% Set optimisation variables for probe compensation

lb = [-3*pi/180, -3*pi/180, -3*pi/180, -pi];
ub = [3*pi/180, 3*pi/180, 3*pi/180, pi];
x0 = [0, 0, 0, 0];
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];
options = optimset('PlotFcns',@optimplotfval,'TolFun',1e-15,'
    Algorithm','sqp');

%% SignalV probe compensation

[x1, fval, exitflag] = fmincon(@(a)probe_comp_farfield_opt(
    signalV_complete, theo_patternV, a),x0,A,b,Aeq,beq,lb,ub,nonlcon,
    options)

% Adust accordinly to optimization
theo_patternV.E_theta = theo_patternV.E_theta .*exp(1i .*  x(4));
theo_patternV.E_phi = theo_patternV.E_phi .*exp(1i .*  x(4));
temp_swe = sft2swe(farfield2sft(theo_patternV,0));
temp = eamlib_rotate(temp_swe,[0,1,0,x(2)]);
temp = eamlib_rotate(temp,[1,0,0,x(1)]);
temp = eamlib_rotate(temp,[0,0,1,x(3)]);
theo_patternV = sft2farfield(swe2sft(temp),theo_patternV.theta,
    theo_patternV.phi);
theo_patternV = eamlib_interp(theo_patternV,signalV_complete.theta,
    signalV_complete.phi);

figure; plot_eamstruct(theo_patternV)

% Calculate the optimum probe compensation
compensated_signalV = probeCompensate_farfield(signalV_complete,
    theo_patternV,'x');

% Inversion error, need to flip after probe comp to match up with
    plane wave
temp = compensated_signalV.E_theta;
compensated_signalV.E_theta = -compensated_signalV.E_phi;
```

```
compensated_signalV.E_phi = temp;

figure; plot_eamstruct(compensated_signalV)

%% Signal V remove plane wave

% Set optimisation variables
lb = [-3*pi/180, -3*pi/180, -3*pi/180, 0.623, max(abs(
    signalV_complete.E_theta))-0.1, 0, 5.84];
ub = [3*pi/180, 3*pi/180, 3*pi/180, 0.69, max(abs(signalV_complete.
    E_theta))+0.1, 2*pi, 5.86];
x0 = [0, 0, 0, 0.6638, 1, 0, 5.85];
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];

options = optimset('PlotFcns',@optimplotfval,'TolFun',1e-20,'
    Algorithm','sqp');
[x, fval, exitflag] = fmincon(@(a)remove_best_plane_wave(
    compensated_signalV,a),x0,A,b,Aeq,beq,lb,ub,nonlcon,options)

% Initiate the plane wave
plane_wave.type = 'farfield';
plane_wave.f = x(7)*1e9;
plane_wave.theta = linspace(0, pi, 181)';
plane_wave.phi =  linspace(-pi,pi,361)';
[theta,phi] = ndgrid(plane_wave.theta,plane_wave.phi);
r0 = x(4); k = 2.*pi.*plane_wave.f./c_0;
x_var=r0*sin(theta).*cos(phi);
plane_wave.E_phi = x(5).*(cos(phi)).*exp(1i.*k.*x_var+1i*x(6));
plane_wave.E_theta = x(5).*(cos(theta).*sin(phi)).*exp(1i.*k.*x_var+1
    i*x(6));
plane_wave = eamlib_flatten_comp(plane_wave);

% Adust accordinly to optimization
plane_wave_swe = sft2swe(farfield2sft(plane_wave,0));
plane_wave_swe = eamlib_rotate(plane_wave_swe,[0,1,0,x(2)]);
plane_wave_swe = eamlib_rotate(plane_wave_swe,[1,0,0,x(1)]);
plane_wave_swe = eamlib_rotate(plane_wave_swe,[0,0,1,x(3)]);
plane_wave = sft2farfield(swe2sft(plane_wave_swe),compensated_signalV
    .theta,compensated_signalV.phi);

figure; plot_eamstruct(plane_wave)

```

```
153 % Remove best fitted plane wave
154 compensated_signalV.E_theta = compensated_signalV.E_theta -
        plane_wave.E_theta;
155 compensated_signalV.E_phi = compensated_signalV.E_phi - plane_wave.
        E_phi;
156
157 figure; plot_eamstruct(compensated_signalV)
158
159 %% Function to find the best plane wave to remove
160 function result = remove_best_plane_wave(field,a)
161
162 % Create the plane wave
163 plane_wave.type = 'farfield';
164 plane_wave.f = 5.85e9;
165 plane_wave.theta = field.theta;
166 plane_wave.phi =  field.phi;
167 [theta,phi] = ndgrid(plane_wave.theta,plane_wave.phi);
168 r0 = a(4); k = 2.*pi.*plane_wave.f./c_0;
169 x=r0*sin(theta).*cos(phi);
170 plane_wave.E_phi = a(5).*(cos(phi)).*exp(1i.*k.*x+1i*a(6));
171 plane_wave.E_theta = a(5).*(cos(theta).*sin(phi)).*exp(1i.*k.*x+1i*a
        (6));
172 plane_wave = eamlib_flatten_comp(plane_wave);
173
174 % Rotate the plane wave
175 temp_swe = sft2swe(farfield2sft(plane_wave,0));
176 temp = eamlib_rotate(temp_swe,[0,1,0,a(2)]);
177 temp = eamlib_rotate(temp,[1,0,0,a(1)]);
178 temp = eamlib_rotate(temp,[0,0,1,a(3)]);
179 plane_wave = sft2farfield(swe2sft(temp),field.theta,field.phi);
180
181 % Remove the plane wave from the field
182 field.E_theta = field.E_theta - plane_wave.E_theta;
183 field.E_phi = field.E_phi - plane_wave.E_phi;
184
185 % Calculate the residue power in area
186 field = eamlib_select_subset(field,'theta_lim',[59.9*pi/180, 120.1*pi
        /180]);
187 field = eamlib_select_subset(field,'phi_lim',[-30.1*pi/180, 30.1*pi
        /180]);
188 result = eamlib_dOmega(field,[59.9*pi/180, 120.1*pi/180,-30.1*pi/180,
         30.1*pi/180])*eamlib_relative_gain(field);
189
190 end
191 %% Calculate the optimum probe compensation
192 function result = probe_comp_farfield_opt(field,pattern,a)
```

```
193
194 % Phase offset the probe pattern
195 pattern.E_theta = pattern.E_theta.*exp(1i .*  a(4));
196 pattern.E_phi = pattern.E_phi.*exp(1i .*  a(4));
197
198 % Rotate the probe pattern
199 temp_swe = sft2swe(farfield2sft(pattern,0));
200 temp = eamlib_rotate(temp_swe,[0,1,0,a(2)]);
201 temp = eamlib_rotate(temp,[1,0,0,a(1)]);
202 temp = eamlib_rotate(temp,[0,0,1,a(3)]);
203 pattern = sft2farfield(swe2sft(temp),pattern.theta,pattern.phi);
204 pattern = eamlib_interp(pattern,field.theta,field.phi);
205
206 % Probe compensate the signal and evaluate
207 comp_sig = probeCompensate_farfield(field, pattern,'x');
208 result = (max(max(abs(comp_sig.E_theta))))-(max(max(abs(pattern.
    E_theta))));
209 end
```

C.4 Probe compensate far field

```
function compensated_field = probeCompensate_farfield(field, pattern,
    prop_dir)
% compensated_field = probeCompensate_farfield(field, pattern)
%
% Compensate the field witeh probe used in the measurment
%
% INPUT
%           field       Eamlib struct of type farfield
%           Pattern     Probe pattern for the measurement
%           prop_dir    Direction of propagation of plane wave
%
% OUTPUT
%          compensated_field  Probe compensated farfield
%
% NOTE: This only works because we are using farfield patterns.
%
% Original code by ealdfil & ecarrik
% Supervised by Jonas Fridén and Anders Jernberg
% Copyright Ericsson AB 2024
%% Validate input variables

% Check if both field and patttern is struct of type farfield
if strcmp(field.type,{'farfield'}) && strcmp(pattern.type,{'farfield'
   })

else
    error('eamlib:invalidInput',[ mfilename ' operates only on
       eamstructs of type farfield'])
end

%% Intiate setting

% switch prop_dir
%     case 'x'
%         % Setting theta and phi values for a x propagating wave
%         theta = linspace(0*pi/180,180*pi/180,301);
%         phi = linspace(-pi/2,pi/2,75);
%     case 'y'
%         % TODO: Fix this
%         error('eamlib:doesNoteExist','The option does not exist at
   the moment, please use eamlib_rotate so the field propagtes in z-
   direction')
%     case 'z'
%         % Setting theta and phi values for a z propagating wave
```

```
%         theta = linspace(0,pi/2,301);
%         phi = linspace(-pi,pi,75);
% end
% theta = field.theta;
% phi = field.phi;

% Interpolate field and probe pattern
% field = eamlib_interp(field, theta, phi);
% pattern = eamlib_interp(pattern, theta, phi);

% Changing polarization to theta phi, so calcultions will work
% field = pol2thetaphi(field);
% pattern = pol2thetaphi(pattern);


%% Probe compensation

% pattern.E_phi = -pattern.E_phi;
% Change polarization to avoid singularites
field = thetaphi2circular(field);
pattern = thetaphi2circular(pattern);

% Intiate compensated field
compensated_field = field;

% Find comp
comp = find_comp(field);

% Making the probe compenstion, this works because we are in farfield
notToLowIdx_comp1 = (20*log10(abs(field.(comp{1})))  > -400);
notToLowIdx_comp2 = (20*log10(abs(field.(comp{2})))  > -400);
% field.(comp{1})(~notToLowIdx_comp1) = 10^(-100/20);
% field.(comp{2})(~notToLowIdx_comp2) = 10^(-100/20);
compensated_field.(comp{1})(notToLowIdx_comp1) = field.(comp{1})(
    notToLowIdx_comp1)./abs(pattern.(comp{1})(notToLowIdx_comp1)); %
    ha med fasen rimligt?
compensated_field.(comp{2})(notToLowIdx_comp2) = field.(comp{2})(
    notToLowIdx_comp2)./abs(pattern.(comp{2})(notToLowIdx_comp2));

% Change back polarization to theta phi
compensated_field = pol2thetaphi(compensated_field);
pattern = pol2thetaphi(pattern);

% Find comp
comp = find_comp(compensated_field);

```

```
% Set all values over 5 db = 0, (higher values are only nonsens
    because
% we are dividing with very small numbers)
limit = 0;
compensated_field.(comp{1})(20*log10(abs(compensated_field.(comp{1}))
    ) > limit) = 10^(limit/20).*exp(1i.*angle(compensated_field.(comp
    {1})(20*log10(abs(compensated_field.(comp{1}))) > limit)));
compensated_field.(comp{2})(20*log10(abs(compensated_field.(comp{2}))
    ) > limit) = 10^(limit/20).*exp(1i.*angle(compensated_field.(comp
    {2})(20*log10(abs(compensated_field.(comp{2}))) > limit)));

%% extend field to full sphere

% Unflatten field so we can use eamlib_shadow
compensated_field = eamlib_unflatten_comp(compensated_field, 'true');
field_shadow = eamlib_shadow(compensated_field);

switch prop_dir
    case 'x'
        % Merge field together for x propagating wave
        % compensated_field.(comp{1}) = [flip(field_shadow.(comp{1})
            (:,1:(length(field.phi)-1)/2),2) compensated_field.(comp
            {1}) flip(field_shadow.(comp{1})(:,(length(field.phi)+1)
            /2:end),2)];
        % compensated_field.(comp{2}) = [flip(field_shadow.(comp{2})
            (:,1:(length(field.phi)-1)/2),2) compensated_field.(comp
            {2}) flip(field_shadow.(comp{2})(:,(length(field.phi)+1)
            /2:end),2)];
        % compensated_field.phi = linspace(-pi,pi,length(field.phi)
            *2)';
        compensated_field.(comp{1}) = [flip(real(compensated_field.(
            comp{1})(:,1:(length(field.phi)-1)/2))-1i*imag(
            compensated_field.(comp{1})(:,1:(length(field.phi)-1)/2))
            ,2) compensated_field.(comp{1}) flip(real(
            compensated_field.(comp{1})(:,(length(field.phi)+1)/2:end)
            )-1i*imag(compensated_field.(comp{1})(:,(length(field.phi)
            +1)/2:end)),2)];
        compensated_field.(comp{2}) = [flip(-real(compensated_field.(
            comp{2})(:,1:(length(field.phi)-1)/2))+1i*imag(
            compensated_field.(comp{2})(:,1:(length(field.phi)-1)/2))
            ,2) compensated_field.(comp{2}) flip(-real(
            compensated_field.(comp{2})(:,(length(field.phi)+1)/2:end)
            )+1i*imag(compensated_field.(comp{2})(:,(length(field.phi)
            +1)/2:end)),2)];
        compensated_field.phi = linspace(-pi,pi,length(field.phi)*2)
            ';
```

```
104     case 'y'
105         % TODO: Fix this
106         error('eamlib:doesNoteExist','The option does not exist at
                the moment, please use eamlib_rotate so the field
                propagtes in z-direction')
107     case 'z'
108         % Merge field together for z propagating wave
109         compensated_field.(comp{1}) = [compensated_field.(comp{1});
                flip(flip(field_shadow.(comp{1}),1),2)];
110         compensated_field.(comp{2}) = [compensated_field.(comp{2});
                flip(flip(field_shadow.(comp{2}),1),2)];
111         compensated_field.theta = linspace(0,pi,length(field.theta)
                *2)';
112 end
113
114 % Flatten field again
115 compensated_field = eamlib_flatten_comp(compensated_field);
116
117 end
```

C.5 Alternative approach

```
function [probe_rot_H, probe_rot_V, feeder2, feeder1, C] =
    new_probe_compensation_test(method,plotOpt)
% [probe_rot_H, probe_rot_V, feeder2, feeder1, C] =
    new_probe_compensation_test(method,plotOpt)
%
% Test code for the alternative approach discused in the Master's
    thesis
% Spherical Probing of Quiet Zone. NOTE: The old method is used in
    the
% report.
%
% INPUT
%       method        string should be new or old for which method
    used
%       plotOpt       string should true or false, true will plot
    everything
%
% OUTPUT
%
%       probe_rot_H   rotate probe pattern horizontal orientation
%       probe_rot_V   rotate probe pattern vertical orientation
%       feeder2         horizontal polarization
%       feeder1         vertical polarization
%       C             cond matrix of probe
%
%% Check input

if ~strcmp(method,'new') && ~strcmp(method,'old')
    error('eamlib:invalidInput','method need to be new or old')
end
if  ~strcmp(plotOpt,'true') && ~strcmp(plotOpt,'false')
    error('eamlib:invalidInput','plotOpt need to be a either true or
        false')
end

%% Load and set up the fields

% get username of user
username = getenv('username');
path2files = append('C:\Users\',username,'\Ericsson\Spherical Probing
    of Quiet Zone - General\Overleaf\test matlab\mätdata\r665mm\');

% config for midas
config.feed.probeV = 'theta';
```

```
config.feed.probeH = 'phi';
config.pos.name = 'BBQ90';


% loading signal and pattern in to matlab
signalV = eamlib_read_midas_data(append(path2files, '\
    OEW585V_r665mm_LGF-11-400_IFBW1kHz_RX_2FS.txt'),config);
signalH = eamlib_read_midas_data(append(path2files, '
    OEW585H_r665mm_LGF-11-400_IFBW1kHz_RX_FS.txt'),config);

signalV = signalV.field;
signalH = signalH.field;

% Change sign on second component (phi component)
comp = find_comp(signalV);
signalH.(comp{2})  = -signalH.(comp{2});
signalV.(comp{2})  = -signalV.(comp{2});


% select frequency 5.85Ghz
signalV = eamlib_select_subset(signalV,'f_ind',1);
signalH = eamlib_select_subset(signalH,'f_ind',1);

% flip theta so it goes the right way
signalH = eamlib_flipdim(signalH,'theta');
signalH = eamlib_flipdim(signalH,'phi');

signalV = eamlib_flipdim(signalV,'phi');
signalV = eamlib_flipdim(signalV,'theta');

% select subset
originalV = eamlib_select_subset(signalV, 'phi_lim',[-pi/2*1.001 pi
    /2*1.001]);
originalV = eamlib_interp(originalV,linspace(-pi/2,3/2.*pi,floor(
    length(originalV.theta)/2))',linspace(-pi/2,pi/2,floor(length(
    originalV.phi)/2))');
originalH = eamlib_interp(signalH,originalV.theta,originalV.phi);

%% Original signal V
% Linear optimization
% minimum = @(a) sum(abs(2*imag(conj(originalV.E_theta).*originalV.
    E_phi*exp(1i*a))).^2);
% options = optimset('TolFun',1e-20);
% [x, fval, exitflag] = fminsearch(minimum, 0, options);
% originalV.E_phi = originalV.E_phi*exp(1i*x);

```

```
77 %% Original signal H
78 % Linear optimization
79 % minimum = @(a) sum(abs(2*imag(conj(originalH.E_phi).*originalH.
       E_theta*exp(1i*a))).^2);
80 % options = optimset('TolFun',1e-20);
81 % [x, fval, exitflag] = fminsearch(minimum, 0, options);
82 % originalH.E_theta = originalH.E_theta*exp(1i*x);
83 %% Load pattern
84 theo_pattern = grasp2eamstruct(append('C:\Users\',username,'\Ericsson
       \Spherical Probing of Quiet Zone - General\Measurements\OEW585\
       grasp\CSTGraspFeed_FFapprox_on_(farfield (f=5.85) [1]_L3-CoCx).cut
       '));
85 theo_patternH = Ludwig32thetaphi(theo_pattern,1); %Tried -1, didn't
       work
86 theo_patternV = eamlib_rotate(theo_patternH,[0,0,1,pi/2]);
87
88 %% Probe rotation NOTE old method
89 if strcmp(method,'old')
90     % Desired point to rotate to
91     theta = originalV.theta; phi = originalV.phi;
92
93     % Change to sft for faster computation
94     theo_patternV_sft = farfield2sft(theo_patternV,0);
95     theo_patternH_sft = farfield2sft(theo_patternH,0);
96
97     % Vector to matrix
98     ogV = eamlib_unflatten_comp(originalV,'true');
99     ogH = eamlib_unflatten_comp(originalH,'true');
100
101    % Matrix size
102    n = length(theta); m = length(phi);
103    % Initiate matrix
104    P_theta_H = zeros(n,m);
105    P_phi_H = zeros(n,m);
106    P_theta_V = zeros(n,m);
107    P_phi_V = zeros(n,m);
108    C = zeros(n,m);
109    E_th_V = zeros(n,m);
110    E_ph_V = zeros(n,m);
111    E_th_H = zeros(n,m);
112    E_ph_H = zeros(n,m);
113
114    for n = 1:length(theta)
115        for m=1:length(phi)
116            % Backrotation from measured point
```

```
117             inv_Rot_1 =  rot_mat([0, 1, 0],-theta(n)) * rot_mat([0,
                    0, 1],-phi(m)) * [1; 0; 0];
118
119             % Converting the cartesian coordinates to spherical ones
120             [th,ph] = cart2sphere(inv_Rot_1(1),inv_Rot_1(2),inv_Rot_1
                    (3));
121
122             % Check if theta and phi exist in probe pattern
123             % NOTE: slow and old way
124             % if false
125             % if ~ismember(th,theta) && ~ismember(ph,phi)
126             %     % Calculate theta
127             %     [val,idx]=min(abs(theta-th));
128             %     minVal=theta(idx);
129             %
130             %     if minVal<th
131             %         new_theta = [theta(1:idx); th; theta(idx+1:end)
                    ];
132             %     else
133             %         new_theta = [theta(1:idx-1); th; theta(idx:end)
                    ];
134             %     end
135             %
136             %     % Calculate phi
137             %     [val,idx]=min(abs(phi-ph));
138             %     minVal=phi(idx);
139             %
140             %     if minVal<th
141             %         new_phi = [phi(1:idx); ph; phi(idx+1:end)];
142             %     else
143             %         new_phi = [phi(1:idx-1); ph; phi(idx:end)];
144             %     end
145             %
146             %     % Getting the new patterns that have the desired
                    points
147             %     patternV = eamlib_interp(theo_patternV, new_theta,
                    new_phi);
148             %     patternV = eamlib_select_subset(patternV,'theta',th
                    ,'phi',ph);
149             %     patternH = eamlib_interp(theo_patternH, new_theta,
                    new_phi);
150             %     patternH = eamlib_select_subset(patternH,'theta',th
                    ,'phi',ph);
151             %
152             % elseif ~ismember(th,theta)
153             %     % Calculate theta
```

```
154             %     [val,idx]=min(abs(theta-th));
155             %     minVal=theta(idx);
156             %
157             %     if minVal<th
158             %         new_theta = [theta(1:idx); th; theta(idx+1:end)
                    ];
159             %     else
160             %         new_theta = [theta(1:idx-1); th; theta(idx:end)
                    ];
161             %     end
162             %     % Getting the new patterns that have the desired
                    points
163             %     patternV = eamlib_interp(theo_patternV, new_theta,
                    phi);
164             %     patternV = eamlib_select_subset(patternV,'theta',th
                    ,'phi',ph);
165             %     patternH = eamlib_interp(theo_patternH, new_theta,
                    phi);
166             %     patternH = eamlib_select_subset(patternH,'theta',th
                    ,'phi',ph);
167             %
168             % elseif ~ismember(ph,phi)
169             %     % Calculate phi
170             %     [val,idx]=min(abs(phi-ph));
171             %     minVal=phi(idx);
172             %
173             %     if minVal<ph
174             %         new_phi = [phi(1:idx); ph; phi(idx+1:end)];
175             %     else
176             %         new_phi = [phi(1:idx-1); ph; phi(idx:end)];
177             %     end
178             %     % Getting the new patterns that have the desired
                    points
179             %     patternV = eamlib_interp(theo_patternV, theta,
                    new_phi);
180             %     patternV = eamlib_select_subset(patternV,'theta',th
                    ,'phi',ph);
181             %     patternH = eamlib_interp(theo_patternH, theta,
                    new_phi);
182             %     patternH = eamlib_select_subset(patternH,'theta',th
                    ,'phi',ph);
183             %
184             % else
185             %     % Getting the new patterns that have the desired
                    points
```

```
186             %     patternV = eamlib_select_subset(theo_patternV,'
                    theta',th,'phi',ph);
187             %     patternH = eamlib_select_subset(theo_patternH,'
                    theta',th,'phi',ph);
188             % end
189             % end
190
191             % Getting the new patterns that have the desired points
192             patternH = sft2farfield(theo_patternH_sft,th,ph);
193             patternV = sft2farfield(theo_patternV_sft,th,ph);
194
195             % Pattern H
196             % change coordinates to cartesian
197             [x,y,z,E_x,E_y,E_z] = sphere2cart(patternH);
198
199             % rotate the desired point
200             Rot_1 = rot_mat([0, 0, 1],phi(m)) * rot_mat([0, 1, 0],
                    theta(n));
201             cart = Rot_1 * [x; y; z];
202             E_cart = Rot_1 * [E_x; E_y; E_z];
203             [~,~,E_theta_H,E_phi_H] = cart2sphere(cart(1),cart(2),
                    cart(3),E_cart(1),E_cart(2),E_cart(3));
204
205             % Pattern V
206             % change coordinates to cartesian
207             [x,y,z,E_x,E_y,E_z] = sphere2cart(patternV);
208
209             % rotate the desired point
210             Rot_1 = rot_mat([0, 0, 1],phi(m)) * rot_mat([0, 1, 0],
                    theta(n));
211             cart = Rot_1 * [x; y; z];
212             E_cart = Rot_1 * [E_x; E_y; E_z];
213             [~,~,E_theta_V,E_phi_V] = cart2sphere(cart(1),cart(2),
                    cart(3),E_cart(1),E_cart(2),E_cart(3));
214
215             % saves the rotated point
216             %pattern H
217             P_theta_H(n,m) = E_theta_H;
218             P_phi_H(n,m) = E_phi_H;
219
220             %pattern V
221             P_theta_V(n,m) = E_theta_V;
222             P_phi_V(n,m) = E_phi_V;
223
224             % P-matrix
225             P = [E_theta_V, E_phi_V;E_theta_H, E_phi_H];
```

```
226
227             % Calculate the cond of the probe matrix
228             C(n,m) = cond(P);
229
230             % Probe compensation feeder1
231             E = P\[ogV.E_theta(n,m); ogH.E_theta(n,m)];
232             E_th_V(n,m) = E(1);
233             E_ph_V(n,m) = E(2);
234
235             % Probe compensation feeder2
236             E = P\[ogV.E_phi(n,m); ogH.E_phi(n,m)];
237             E_th_H(n,m) = E(1);
238             E_ph_H(n,m) = E(2);
239         end
240
241     end
242
243     % Save the rotated probe pattern
244     % Pattern H
245     probe_rot_H = originalV;
246     probe_rot_H.E_theta = P_theta_H;
247     probe_rot_H.E_phi = P_phi_H;
248     % Matrix to vector
249     probe_rot_H = eamlib_flatten_comp(probe_rot_H);
250
251     % Pattern V
252     probe_rot_V = originalV;
253     probe_rot_V.E_theta = P_theta_V;
254     probe_rot_V.E_phi = P_phi_V;
255     % Matrix to vector
256     probe_rot_V = eamlib_flatten_comp(probe_rot_V);
257
258     % Save the probe compensated pattern
259     % feeder1
260     feeder1 = originalV;
261     feeder1.E_theta = E_th_V;
262     feeder1.E_phi = E_ph_V;
263     % Matrix to vector
264     feeder1 = eamlib_flatten_comp(feeder1);
265
266     % feeder2
267     feeder2 = originalV;
268     feeder2.E_theta = E_th_H;
269     feeder2.E_phi = E_ph_H;
270     % Matrix to vector
271     feeder2 = eamlib_flatten_comp(feeder2);
```

```
272
273     if strcmp(plotOpt,'true')
274         figure('name', 'Probe pattern V'); plot_eamstruct(probe_rot_V
                )
275         figure('name', 'Probe pattern H'); plot_eamstruct(probe_rot_H
                )
276         figure('name', 'feeder1 probe compensated'); plot_eamstruct(
                eamlib_select_subset(feeder1,'theta_lim', [-0.001 pi
                *1.001], 'phi_lim', [-89.1*pi/180 90.1*pi/180]))
277         figure('name', 'feeder2 probe compensated'); plot_eamstruct(
                eamlib_select_subset(feeder2,'theta_lim', [-0.001 pi
                *1.001], 'phi_lim', [-89.1*pi/180 90.1*pi/180]))
278     end
279
280     %% Solve equation systems to obtain probe compensated field NOTE
            old method
281     % Seperate the signal in to two polarization, v1,v2 feeder1 and
            v3,v4 feeder2
282     % if false
283     % NOTE: slow and old way
284     % temp = eamlib_select_subset(originalV,'theta_lim', [-0.001 pi
            *1.001], 'phi_lim', [-89.1*pi/180 90.1*pi/180]);
285     % v1 = temp.E_theta; % feeder1 theta
286     % v3 = temp.E_phi; % feeder1 theta
287     %
288     % temp = eamlib_select_subset(originalH,'theta_lim', [-0.001 pi
            *1.001], 'phi_lim', [-89.1*pi/180 90.1*pi/180]);
289     % v2 = temp.E_theta; % feeder1 phi
290     % v4 = temp.E_phi; % feeder2 phi
291     %
292     % % Probe pattern V
293     % temp = eamlib_select_subset(rot_probe_V,'theta_lim', [-0.001 pi
            *1.001], 'phi_lim', [-89.1*pi/180 90.1*pi/180]);
294     % A = temp.E_theta;
295     % B = temp.E_phi;
296     %
297     % % Probe pattern H
298     % temp = eamlib_select_subset(rot_probe_H,'theta_lim', [-0.001 pi
            *1.001], 'phi_lim', [-89.1*pi/180 90.1*pi/180]);
299     % C = temp.E_theta;
300     % D = temp.E_phi;
301     % syms E_theta E_phi
302     % E_th_V = zeros(size(v1));
303     % E_ph_V = zeros(size(v1));
304     % E_th_H = zeros(size(v1));
305     % E_ph_V = zeros(size(v1));
```

```
306     %
307     % % Vertical polarzation
308     % for n = 1: length(v1)
309     %
310     % eqn1 = A(n) .* E_theta + B(n) .* E_phi == v1(n);
311     % eqn2 = C(n) .* E_theta + D(n) .* E_phi == v2(n);
312     %
313     % X = solve([eqn1,eqn2], [E_theta E_phi],'ReturnConditions',true)
            ;
314     %
315     % E_th_V(n) = X.E_theta;
316     % E_ph_V(n) = X.E_phi;
317     %
318     % end
319     % % Horizontal polarzation
320     % syms E_theta E_phi
321     % for n = 1: length(v1)
322     %
323     % eqn1 = A(n) .* E_theta + B(n) .* E_phi == v3(n);
324     % eqn2 = C(n) .* E_theta + D(n) .* E_phi == v4(n);
325     %
326     % X = solve([eqn1,eqn2], [E_theta E_phi],'ReturnConditions',true)
            ;
327     %
328     % E_th_H(n) = X.E_theta;
329     % E_ph_H(n) = X.E_phi;
330     %
331     % end
332     %
333     % % Save the probe compensated pattern
334     % % feeder1
335     % feeder1 = eamlib_select_subset(originalV,'theta_lim', [-0.001
            pi*1.001], 'phi_lim',[-89*pi/180 90.1*pi/180]);
336     % feeder1.E_theta = E_th_V;
337     % feeder1.E_phi = E_ph_V;
338     % % Matrix to vector
339     % feeder1 = eamlib_flatten_comp(feeder1);
340     %
341     % % feeder2
342     % feeder2 = eamlib_select_subset(originalV,'theta_lim', [-0.001
            pi*1.001], 'phi_lim',[-89*pi/180 90.1*pi/180]);
343     % feeder2.E_theta = E_th_H;
344     % feeder2.E_phi = E_ph_H;
345     % % Matrix to vector
346     % feeder2 = eamlib_flatten_comp(feeder2);
347     %
```

```
348     % if strcmp(plotOpt,'true')
349     %     figure('name', 'feeder1 probe compensated'); plot_eamstruct
            (feeder1)
350     %     figure('name', 'feeder2 probe compensated'); plot_eamstruct
            (feeder2)
351     % end
352     %
353     % end
354 end
355 %% Probe rotation and probe compensation
356 if strcmp(method,'new')
357     % Desired point to rotate to
358     theta = originalV.theta; phi = originalV.phi;
359     % Change to sft for faster computation
360     theo_patternV_sft = farfield2sft(theo_patternV,0);
361     theo_patternH_sft = farfield2sft(theo_patternH,0);
362     % Vector to matrix
363     ogV = eamlib_unflatten_comp(originalV,'true');
364     ogH = eamlib_unflatten_comp(originalH,'true');
365     % signalV_interp = eamlib_interp(originalV,);
366     % TODO: fix this so cover sphere uniform type 1
367
368     % Matrix size
369     n = length(theta); m = length(phi);
370     % Initiate matrix
371     P_theta_H = zeros(n,m);
372     P_phi_H = zeros(n,m);
373     P_theta_V = zeros(n,m);
374     P_phi_V = zeros(n,m);
375     C = zeros(n,m);
376     E_th_V = zeros(n,m);
377     E_ph_V = zeros(n,m);
378     E_th_H = zeros(n,m);
379     E_ph_H = zeros(n,m);
380
381     for n = 1:length(theta)
382         for m=1:length(phi)
383
384             % Backrotation from measured point
385             inv_Rot_2 =   rot_mat([0, 1, 0],-theta(n)) * rot_mat([0,
                    0, 1],-phi(m)) * [1; 0; 0];
386
387             % Converting the cartesian coordinates to spherical ones
388             [th,ph] = cart2sphere(inv_Rot_2(1),inv_Rot_2(2),inv_Rot_2
                    (3));
389
```

```
390             % Getting the new patterns that have the desired points
391             patternH = sft2farfield(theo_patternH_sft,th,ph);
392             patternV = sft2farfield(theo_patternV_sft,th,ph);
393
394
395             % Pattern V
396             % change coordinates to cartesian in the probe coordinate
                    system
397             [~,~,~,E_x,E_y,E_z] = sphere2cart(patternV);
398             % Calculate the probe gain in the desired direction
399             P(1,1) = cos(theta(n))*cos(phi(m))*E_x + cos(theta(n))*
                   sin(phi(m))*E_y - sin(theta(n))*E_z;
400             P(1,2) = -sin(phi(m))*E_x + cos(phi(m))*E_y;
401
402             % Pattern H
403             % change coordinates to cartesian in the probe coordinate
                    system
404             [~,~,~,E_x,E_y,E_z] = sphere2cart(patternH);
405             % Calculate the probe gain in the desired direction
406             P(2,1) = cos(theta(n))*cos(phi(m))*E_x + cos(theta(n))*
                   sin(phi(m))*E_y - sin(theta(n))*E_z;
407             P(2,2) = -sin(phi(m))*E_x + cos(phi(m))*E_y;
408
409             % Calculate the cond of the probe matrix
410             C(n,m) = cond(P);
411
412             % Saves pattern to be able to illustrate the pattern
413             P_theta_V(n,m) = P(1,1);
414             P_theta_H(n,m) = P(2,1);
415             P_phi_V(n,m) = P(1,2);
416             P_phi_H(n,m) = P(2,2);
417
418             % Probe compensation feeder1
419             E = P\[ogV.E_theta(n,m); ogH.E_theta(n,m)];
420             E_th_V(n,m) = E(1);
421             E_ph_V(n,m) = E(2);
422
423             % Probe compensation feeder2
424             E = P\[ogV.E_phi(n,m); ogH.E_phi(n,m)];
425             E_th_H(n,m) = E(1);
426             E_ph_H(n,m) = E(2);
427         end
428
429     end
430
431     % Save the rotated probe pattern
```

```
    % Pattern V
    probe_rot_V = originalV;
    probe_rot_V.E_theta = P_theta_V;
    probe_rot_V.E_phi = P_phi_V;
    % Matrix to vector
    probe_rot_V = eamlib_flatten_comp(probe_rot_V);

    % Pattern H
    probe_rot_H = originalV;
    probe_rot_H.E_theta = P_theta_H;
    probe_rot_H.E_phi = P_phi_H;
    % Matrix to vector
    probe_rot_H = eamlib_flatten_comp(probe_rot_H);

    % Save the probe compensated pattern
    % feeder1
    feeder1 = originalV;
    feeder1.E_theta = E_th_V;
    feeder1.E_phi = E_ph_V;
    % Matrix to vector
    feeder1 = eamlib_flatten_comp(feeder1);

    % feeder2
    feeder2 = originalV;
    feeder2.E_theta = E_th_H;
    feeder2.E_phi = E_ph_H;
    % Matrix to vector
    feeder2 = eamlib_flatten_comp(feeder2);

    if strcmp(plotOpt,'true')
        figure('name', 'Probe pattern V'); plot_eamstruct(probe_rot_V
            )
        figure('name', 'Probe pattern H'); plot_eamstruct(probe_rot_H
            )
        figure('name', 'feeder1 probe compensated'); plot_eamstruct(
            feeder1)
        figure('name', 'feeder2 probe compensated'); plot_eamstruct(
            feeder2)
    end

end

%% Flip components
feeder1 = eamlib_select_subset(feeder1,'theta_lim', [-0.001 pi
    *1.001], 'phi_lim', [-89.1*pi/180 90.1*pi/180]);
temp = feeder1.E_theta;
```

```
feeder1.E_theta = feeder1.E_phi;
feeder1.E_phi = temp;

%% Linear optimization
minimum = @(a) sum(abs(2*imag(conj(feeder1.E_theta).*feeder1.E_phi*
    exp(1i*a))).^2);
options = optimset('TolFun',1e-20);
[x, fval, exitflag] = fminsearch(minimum, 0, options);
feeder1.E_phi = feeder1.E_phi*exp(1i*x);

%% Signal V remove plane wave

% Set optimisation variables
lb = [-2.5*pi/180, -2.5*pi/180, -2.5*pi/180, 0.623, max(abs(feeder1.
    E_theta))-0.1, 0, 5.84];
ub = [2.5*pi/180, 2.5*pi/180, 2.5*pi/180, 0.69, max(abs(feeder1.
    E_theta))+0.1, 2*pi, 5.86];
x0 = [0, 0, 0, 0.6638, 1, 0, 5.85];
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];

options = optimset('PlotFcns',@optimplotfval,'TolFun',1e-20,'
    Algorithm','sqp');
[x, fval, exitflag] = fmincon(@(a)remove_best_plane_wave(feeder1,a),
    x0,A,b,Aeq,beq,lb,ub,nonlcon,options)

% Initiate the plane wave
plane_wave.type = 'farfield';
plane_wave.f = x(7)*1e9;
plane_wave.theta = linspace(0, pi, 181)';
plane_wave.phi =  linspace(-pi,pi,361)';
[theta,phi] = ndgrid(plane_wave.theta,plane_wave.phi);
r0 = x(4); k = 2.*pi.*plane_wave.f./c_0;
x_var=r0*sin(theta).*cos(phi);
plane_wave.E_phi = x(5).*(cos(phi)).*exp(1i.*k.*x_var+1i*x(6));
plane_wave.E_theta = x(5).*(cos(theta).*sin(phi)).*exp(1i.*k.*x_var+1
    i*x(6));
plane_wave = eamlib_flatten_comp(plane_wave);

% Adust accordinly to optimization
plane_wave_swe = sft2swe(farfield2sft(plane_wave,0));
plane_wave_swe = eamlib_rotate(plane_wave_swe,[0,1,0,x(2)]);
plane_wave_swe = eamlib_rotate(plane_wave_swe,[1,0,0,x(1)]);
```

```
513 plane_wave_swe = eamlib_rotate(plane_wave_swe,[0,0,1,x(3)]);
514 plane_wave = sft2farfield(swe2sft(plane_wave_swe),feeder1.theta,
        feeder1.phi);
515
516 % Remove best fitted plane wave
517 feeder1_residue = feeder1;
518 feeder1_residue.E_theta = feeder1.E_theta - plane_wave.E_theta;
519 feeder1_residue.E_phi = feeder1.E_phi - plane_wave.E_phi;
520
521 if strcmp(plotOpt,'true')
522     figure('name', 'optimized plane wave'); plot_eamstruct(plane_wave
            )
523     figure('name', 'feeder1 after plane wave removal');
            plot_eamstruct(feeder1_residue)
524 end
525
526
527 end
528 %% Function to find the best plane wave to remove
529 function result = remove_best_plane_wave(field,a)
530
531 % Create the plane wave
532 plane_wave.type = 'farfield';
533 plane_wave.f = 5.85e9;
534 plane_wave.theta = linspace(0, pi, 181)';
535 plane_wave.phi =  linspace(-pi,pi,361)';
536 [theta,phi] = ndgrid(plane_wave.theta,plane_wave.phi);
537
538 r0 = a(4); k = 2.*pi.*plane_wave.f./c_0;
539 x=r0*sin(theta).*cos(phi);
540
541 plane_wave.E_phi = a(5).*(cos(phi)).*exp(1i.*k.*x+1i*a(6));
542 plane_wave.E_theta = a(5).*(cos(theta).*sin(phi)).*exp(1i.*k.*x+1i*a
        (6));
543 plane_wave = eamlib_flatten_comp(plane_wave);
544
545 % Rotate the plane wave
546 temp_swe = sft2swe(farfield2sft(plane_wave,0));
547 temp = eamlib_rotate(temp_swe,[0,1,0,a(2)]);
548 temp = eamlib_rotate(temp,[1,0,0,a(1)]);
549 temp = eamlib_rotate(temp,[0,0,1,a(3)]);
550 plane_wave = sft2farfield(swe2sft(temp),field.theta,field.phi);
551
552 % Remove the plane wave from the field
553 field.E_theta = field.E_theta - plane_wave.E_theta;
554 field.E_phi = field.E_phi - plane_wave.E_phi;
```

```
555
556 % Calculate the residue power in area
557 field = eamlib_select_subset(field,'theta_lim',[29.1*pi/180, 150.1*pi
       /180]);
558 field = eamlib_select_subset(field,'phi_lim',[-60.1*pi/180, 60.1*pi
       /180]);
559 result = eamlib_dOmega(field,[29.1*pi/180, 150.1*pi/180,-60.1*pi/180,
        60.1*pi/180])*eamlib_relative_gain(field);
560
561 end
```