Politecnico di Torino

DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATIONS
Master's Degree in Electronic Engineering

Master's Degree Thesis

Field Manipulating Surfaces

Supervisors
Prof. Paola Pirinoli
Prof. Giuseppe Vecchi
Dott. Michele Beccaria

Author
Letizia Bricco

July, 2025

Chi è normale non ha troppa fantasia.
Edoardo Bennato, *Venderò* (1976)

Contents

List of Tables

List of Figures

List of Definitions and Theorems

Acronyms

3GPP 3rd Generation Partnership Project

ABC Ambient Backscatter Communication

AI Artificial Intelligence

AM Additive Manufacturing

BLL Back Lobe Level

BS Broadside

CHDM 1,4-CycloHexaneDiMethanol

DoF Degree of Freedom

EA Evolutionary Algorithm

EG Ethylene Glycol

EM ElectroMagnetism, ElectroMagnetic

EMA Effective Medium Approximation

EMT Effective Medium Theory

FNBW First Null Beam Width

FPGA Field Programmable Gate Array

FSS Frequency Selective Surface

HPBW Half Power Beam Width

GA Genetic Algorithm

I2I communication Indoor-to-Indoor communication

IAB Integrated Access and Backhauling

IoE/IoT Internet of Everything/Things

LoS Line of Sight

LTTL Linearly-Tapered Transmission Line

MEMS Micro Electro-Mechanical System

MCU Micro-Controller Unit

MG Maxwell Garnett

MIMO Multiple-Input and Multiple-Output

MW MicroWave

OA Optimization Algorithm

O2I communication Outdoor-to-Indoor communication

O2O communication Outdoor-to-Outdoor communication

OTO-EMS Optically-Transparent Opportunistic ElectroMagnetic Skin

PCB Printed Circuit Board

QoS Quality of Service

PETG Polyethylene Terephthalate Glycol

PO Physical Optics

PTA Purified Terephthalic Acid

RA ReflectArray

RF Radio Frequency

RIS Reconfigurable Intelligent Surface

SEE Smart Electromagnetic Environment

SES Smart Electromagnetic Skin

SLL Side Lobe Level

SRE Smart Radio Environment

TA TransmitArray

TE Transverse Electric

TEM Transverse ElectroMagnetic

TL Transmission Line

TM Transverse Magnetic

TSES Transmissive Smart Electromagnetic Skin

UC Unit Cell

Abstract

5G and 6G communication networks aim to ensure fast data rates, wide bandwidth, improved coverage and minimal latency; operating in mm-wave and sub-THz frequency bands, however, presents remarkable challenges including free-space loss, building penetration loss and strong interactions with obstacles, possibly leading to coverage gaps.

Integrating active and passive devices into the environment has been widely proposed as a means of enhancing coverage without expanding the number of base stations. Among the others, Smart Electromagnetic Skins (SESs) offer a passive, thin-surface solution for redirecting incident fields in desired directions; depending on specific applications, SESs may serve as either reflective or transmitting devices, contributing to the creation of a Smart Radio Environment (SRE). In particular, a passive transmitting SES integrated into windows presents an opportunity to strengthen the connection between base stations and indoor terminals. However, due to its specific placement, the design must balance performance with minimal interference in visibility and light passage.

Traditional structures consist of meshed opaque conductors on transparent substrates but often neglect window integration. Alternative designs have introduced multi-layer structures directly embedded in the window, printing a thin conductive layer between two glass panes. Since metallization affects both transparency and radiation efficiency, recent research efforts have explored the potential of a fully dielectric transparent smart skin that is directly integrated with the window. This is the solution investigated in this Thesis: in particular, an innovative Unit Cell (UC) model is introduced, focusing on the analytical design of multi-layer, purely dielectric configurations. This design method is presented and validated in a variety of test cases, in order to provide a tool that can be exploited at design time without the need to resort to CAD simulations, which are both time-consuming and computationally heavy. The strengths of the proposed scheme are represented not only by its speed and its more-than-satisfactory level of accuracy, but also by its suitability for the characterization of complex multi-layer cells. Additionally, the method's adaptability extends to tapered structures, modeled as non-uniform transmission lines, further broadening the potential applications.

In order to enhance the performance of the UC while fulfilling the phase constraint required when the cell is embedded in a transmitting surface, an optimization strategy based on the Genetic Algorithm is developed, with emphasis on reducing back radiation and control of the thickness of the entire structure, including the glass layer. Preliminary findings, presented for a number of geometric configurations, show that, accepting a small phase error, the performance of the cell can be significantly improved, obtaining satisfactory values for both reflection and transmission coefficient in most angle configurations. The approach used for the UC is then extended to Transmitarray and SES design and validated through full-wave electromagnetic simulations, demonstrating the model's effectiveness for both feed and plane-wave incidence.

Ultimately, this analysis offers a faster alternative to typical simulation-based design while significantly reducing computational costs, paving the way for broader exploration of novel configurations in future research.

Chapter 1

Introduction

1.1 Smart Radio Environments

Smart Radio Environments (SREs) represent a transformative paradigm in wireless communication systems, aiming to optimize the propagation of electromagnetic (EM) waves through the strategic deployment of intelligent nodes whose response to incident fields can be dynamically tuned to enhance connectivity, data rates, and coverage [10].

In fact, next-generation communication systems will require handling an enormous amount of data, which will have to be exchanged with the highest possible accuracy and the lowest possible latency, thus dramatically increasing the Quality of Service (QoS) compared to state-of-the-art systems [11]. This holds not only for mainstream online communication and Internet of Things/Everything (IoT/IoE) applications, but also for emerging fields such as autonomous vehicles, remote healthcare, industrial automation and systems driven by Artificial Intelligence (AI) and Machine Learning (ML) algorithms, where reliable and efficient connectivity is a crucial aspect. Current 5G and 6G networks might not be the smartest solution to face these challenges because, the more frequency increases, the more likely it is to introduce losses and connectivity gaps.

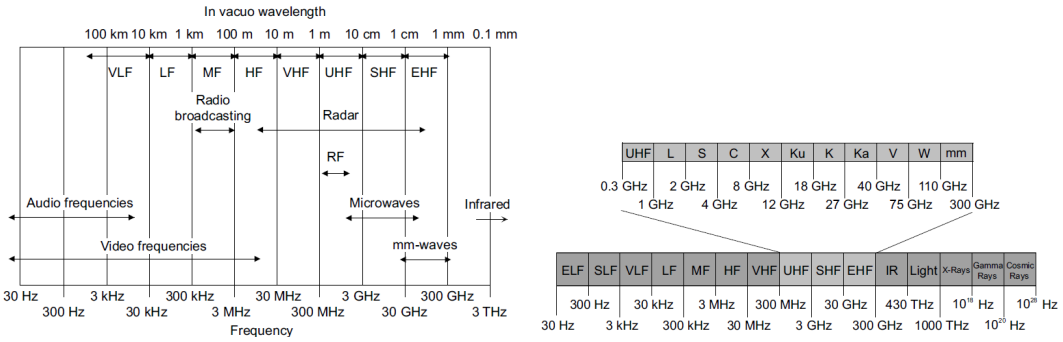
To better understand this aspect, let us quickly recall the typical operating frequency ranges of these communication systems, which all belong to the microwave- or millimeter-wave bands depicted in Figure 1.1. According to 3GPP[1], both 5G and 6G can operate across different frequency ranges, summarized in Table 1.1.

Table 1.1: 5G and 6G networks frequency bands (adapted from [6] and [27]).

System	Range	Frequency Band	Description
5G	FR1 (Sub-6 GHz)	$410 \div 7125$ MHz	good coverage, moderate speed
5G	FR2 (mmWave)	$24.25 \div 52.6$ GHz	limited coverage, high speed
6G	FR3 (Upper Midband)	$7 \div 24$ GHz	trade-off coverage/speed
6G	Sub-TeraHertz	$90 \div 300$ GHz	short range, high speed
6G	TeraHertz	$0.3 \div 3$ THz	limited range, ultra-high speed

Though mm-waves and sub-THz waves allow for larger bandwidths and higher data rates, the increase in frequency imposed by the evolution of technology poses several challenges and

[1]The 3rd Generation Partnership Project (3GPP) is a global collaboration between telecommunications standards organizations. It was established in 1998 to develop protocols for mobile networks, starting with 3G and evolving through 4G (LTE), 5G, and now working toward 6G.



(a) Frequency ranges **(b)** Frequency bands

Figure 1.1: Frequency ranges and frequency bands according to the IEEE convention [17].

technical problems:

1. The short wavelength ($f > 30\,\text{GHz} \Rightarrow \lambda < 1\,\text{cm}$) negatively affects the **Line of Sight** (LoS) propagation, transforming even small objects into potential obstacles for successful data exchange [20].

2. When working in sub-THz and THz bands, data transfer over long distances is difficult because of **high atmospheric attenuation** and **propagation loss**; whilst this allows for frequency reuse and spatial diversity, it also poses challenges for achieving ubiquitous connectivity, making it difficult to ensure 100% coverage even in densely deployed network environments [11].

3. The propagation properties of high-frequency fields strongly depend on the atmospheric conditions, which influence absortive and dispersive effects. The high unpredictability of climatic conditions increases the **complexity of channel modeling** of this band [11].

4. Heterogeneous hardware constraints are present because a huge number of different communication systems and protocols need to coexist in the same environment and should be integrated into a single platform [11].

5. To maximize resource utilization and QoS, efficient spectrum management is crucial and **spectrum-sharing strategies** must be implemented, i.e., it is necessary to develop heterogeneous networks where different systems synchronize transmission on the same frequency. It is therefore important to investigate new strategies for standard, parallel and successive interference cancellation [11].

6. Beam management becomes challenging in high-frequency bands and, despite massive MIMO systems[2] being promising, it also introduces new challenges because of the unpredictable variability of the physical model [11].

The challenges outlined above are precisely why researchers are actively exploring the concept of a **Smart Electromagnetic Environment** (SEE) or **Smart Radio Environment** (SRE), which aim to address these limitations by intelligently shaping signal propagation.

Rather than increasing the number of base stations and repeaters, which would probably solve the coverage gap issue but would also result in increasing costs and power consumption,

[2]In RADAR and radio systems, MIMO (Multiple-Input and Multiple-Output) is a method for multiplying the capacity of a radio link using multiple transmission and receiving antennas to exploit multipath propagation.

Figure 1.2: Different kinds of smart nodes typically embedded in a SRE [10].

this approach focuses on the design of **intelligent metasurface antennas**, whose radiated field can be manipulated and adapted to the changing operating conditions.

In materials physics, an **electromagnetic metasurface** is an artificial sheet-like material with sub-wavelength characteristic dimensions and/or periodicity; it is composed of elementary blocks called **unit cells** (UCs) that actively modulate the behavior of incident electromagnetic fields. Depending on its specific design, a metasurface can bend, focus, or filter waves with exceptional precision. These properties enable a wide range of applications: beyond next-generation wireless systems, metasurfaces are used in the development of flat lenses, holograms, and stealth technologies.

To control and tune the properties of a metasurface, different physical mechanisms have been used up to now:

- Use of electronic circuits, embedding *pin* switches or varactor diodes and typically controlled by Micro-Controller Units (MCUs) or Field-Programmable Gate Arrays (FPGAs) that load the elements of the surface in different ways depending on the specific application;
- Use of liquid crystals, graphene or other controllable materials;
- Use of phase-change materials, whose phase transition is controlled by physical parameters, e.g., temperature or light intensity.

The unprecedented development of such technologies over recent years enables designing and implementing four kinds of SRE metasurface nodes, as Figure 1.2 shows:

Smart Electromagnetic Skin (SES). Passive planar or curved metasurfaces designed to manipulate electromagnetic waves without any active electronic components. SESs can control reflection, absorption, or transmission properties through their geometric and material configuration, enabling functionalities such as beam steering or polarization conversion with minimal power consumption.

They can operate in either reflection or transmission mode, generating a fixed reflected or transmitted beam that ensures good coverage in regions characterized by a no-LoS link with the base station. In particular, reflective SESs are typically aimed at enhancing Outdoor-to-Outdoor (O2O) and Indoor-to-Indoor (I2I) communication, while transmissive surfaces are designed to improve primarily Outdoor-to-Indoor (O2I) links.

Since these structures do not amplify the incident signal, they need no power supply and it is not necessary for them to embed conductive layers, which makes them particularly low-cost and efficient.

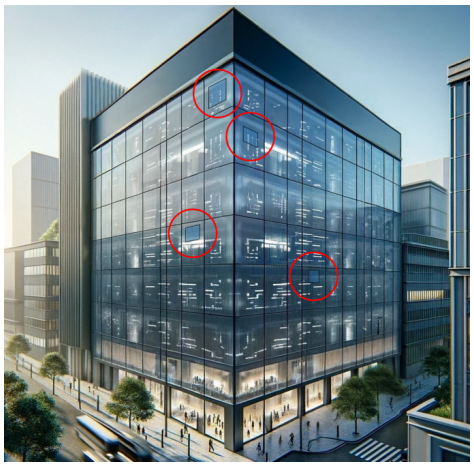


Figure 1.3: Transmitting metasurfaces for O2I communication.

In particular, this Thesis work focuses on the design of **innovative dielectric-only Smart Electromagnetic Skins** operating in **transmitting mode**, which should be integrated into window panes in order to enhance the quality of wireless communication between the base station and indoor users, as shown in Figure 1.3. More details on SES design and state-of-the-art technologies can be found in Section 1.2.2.

Reconfigurable Intelligent Surface (RIS). Passive metasurfaces characterized by anomalous routing and oftern relying on external control to change their properties in time and, possibly, serve different mobile terminals. By adjusting the phase, amplitude, and polarization of incident waves, RIS can achieve precise control over signal propagation.

They do not provide signal boosting and only consume a limited amount of power to control the different states of the metasurface unit cells to generate beams in the desired directions.

Smart Repeater. Active smart antennas that manipulate the EM field enabling amplification and redirection of the field radiated by the base station.

Integrated Access and Backhauling (IAB) node. Active devices that can decode and amplify the EM field: in practice, they behave as smaller and simpler secondary base stations, allowing not only signal boosting, but also signal regeneration.

From this short introduction, it appears clear that the concept of SREs is rooted in the integration of **advanced electromagnetic theory** with **cutting-edge communication technologies**. Unlike traditional wireless systems that rely on active transmission and reception, SREs leverage passive elements to control the environment itself, effectively turning it into a smart medium for signal transmission. This approach not only reduces power consumption but also mitigates interference and improves spectral efficiency.

Apart from the four categories of metasurfaces introduced above, one of the key innovations in this field is the use of **Ambient Backscatter Communication** (ABC) [45], which allows devices to communicate by reflecting existing signals rather than generating new ones. This technology is particularly beneficial for low-power and IoT applications, as it significantly reduces energy requirements.

Research in ABC has focused on improving the efficiency and reliability of backscatter systems. For example, advanced modulation techniques have been developed to enhance data rates and

minimize signal degradation. Furthermore, the integration of ABC with RIS has been investigated to achieve even greater control over signal propagation and energy utilization.

Another notable development is the integration of **Artificial Intelligence** (AI) with RIS [12], facilitating adaptive learning and optimization of signal paths. AI-driven algorithms can predict and adjust to environmental changes, ensuring consistent performance and reliability.

The potential applications of SREs are vast, ranging from urban communication networks to remote sensing and beyond. In urban settings, SREs can enhance connectivity in dense environments by dynamically redirecting signals around obstacles. In remote areas, they can provide cost-effective solutions for extending network coverage. Machine learning techniques, such as reinforcement learning and neural networks, have been applied to SRE systems to enable predictive modeling and dynamic adjustments. For instance, AI can predict the impact of obstacles on signal propagation and adjust RIS configurations accordingly. Additionally, AI can facilitate the coordination of multiple RIS within a network, optimizing overall system performance.

Looking ahead, the future of SREs lies in the continued convergence of AI, machine learning, and advanced meta-materials: as these technologies evolve, they promise to unlock new possibilities for wireless communication, paving the way for smarter, more efficient networks.

1.2 State of the Art

1.2.1 SRE Challenges and Opportunities

Recent research has focused on optimizing the design and implementation of RISs and SESs to maximize their efficiency and adaptability. For instance, advanced meta-materials with tunable properties have been developed to enable real-time adjustments to signal conditions. Additionally, the integration of RIS with existing communication infrastructure has been explored to enhance network performance without requiring significant hardware upgrades.

Despite the promising advancements in SRE technologies, several challenges remain to be addressed. One of the primary challenges is the **complexity** of integrating RIS with existing communication infrastructure. Ensuring compatibility and seamless operation requires significant research and development efforts.

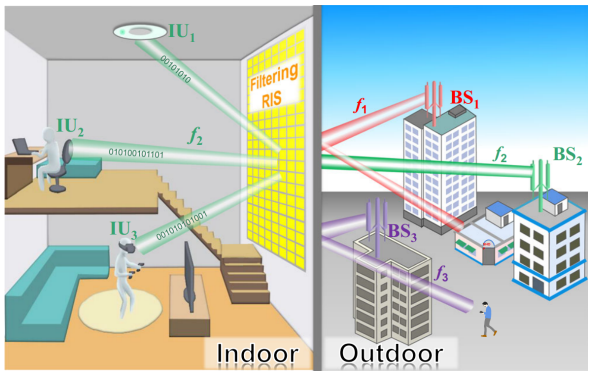
Another challenge is the **scalability** of SRE systems. As the number of devices and users within a network increases, maintaining optimal performance becomes increasingly difficult. Advanced algorithms and efficient hardware designs are needed to address this issue.

On the other hand, the opportunities presented by SREs are vast. The ability to **control electromagnetic wave propagation** opens up new possibilities for applications such as wireless power transfer, secure communication, and environmental sensing. Additionally, the integration of SREs with emerging technologies, such as 5G and beyond, promises to revolutionize the wireless communication landscape.

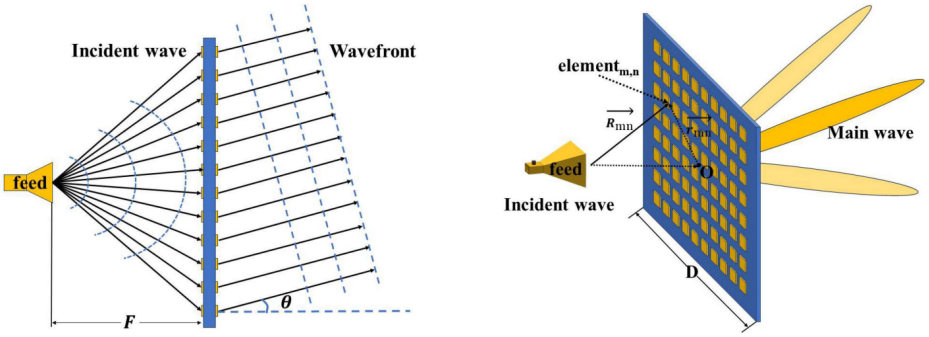
1.2.2 Overview on SES and TA

As anticipated in the previous paragraph, **Smart Electromagnetic Skins** (SESs) are planar or curved passive intelligent surfaces, typically conformal to the surface they are attached to, aiming at manipulating the impinging electromagnetic waves and redirecting them into a desired direction. As Figure 1.4a shows, SESs operating in transmitting mode are designed to enhance connectivity in O2I communication links without increasing the number of base stations and without any need for signal amplification.

To some extent, smart skins operating in transmitting mode (**Transmissive SESs** or **TSESs**) can be seen as the evolution of **Transmitarray Antennas** (TA), that are thin transmitting



(a) SES [22]



(b) TA [49]

Figure 1.4: Typical application scenario of a transmitting SES (a) and scheme of a TA (b).

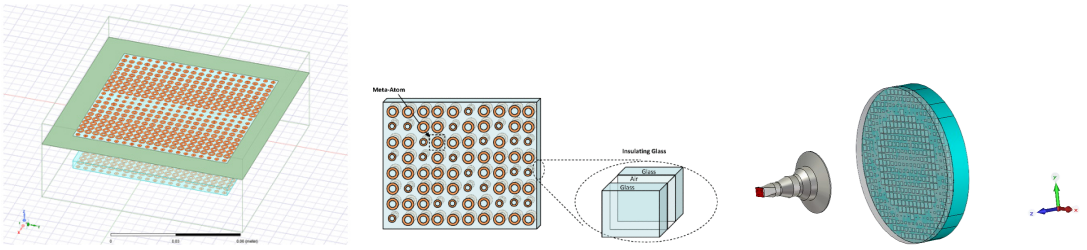
surface illuminated by a feed source located on an equivalent focal point (Figure 1.4b). On the surface there is a quasi-periodic array of elements ([1], [32]) whose transmission coefficients are individually designed in order to

- Transform the phase front of the incident field generated by the feed into a planar phase front; in this way, a focused beam can be achieved with high gain ($> 20\,\text{dBi}$) and efficiency;
- Implement a *phase compensation mechanism*, i.e., modify the phase of the incident wave in order to steer the impinging beam in the desired direction.

Unlike other antenna technologies, Transmitarrays are typically low-cost due to the absence of active components and their straightforward implementation using standard microstrip techniques on Printed Circuit Boards (PCBs). However, PCB-based design is not always feasible for the realization of a SRE, where the smart surface must be integrated within the surrounding environment with the minimum visual impact.

In recent years, several configurations have been studied to improve the design of TAs and TSESs characterized by optical transparency:

- In [24], the design of an optically transparent TA is carried out, without taking into account the effects of the window.
- In [39], an Optically-Transparent Opportunistic ElectroMagnetic Skin (OTO-EMS) is proposed: in this case the glass is used as a multi-layer structure, separating two thin film layers, where the conductive pattern is printed (Figures 1.5a and 1.5b).



(a) OTO-EMS [39] **(b)** OTO-EMS, detail [39] **(c)** Dielectric-only TA [25]

Figure 1.5: State-of-the art TAs and SESs available in literature.

- In [25], the potential of an alternative configuration, consisting of a dielectric-only transparent smart skin that is integrated with the window, is explored and preliminary results are presented (Figure 1.5c).

1.3 Electromagnetic Design

To better understand the scope of this work, it is important to emphasize that, regardless of the final application, establishing the electromagnetic model of a system — by solving Maxwell's equations within the domain of interest — has always been one of the most challenging aspects of modern electronics, especially when compared to the principles underlying analog and digital design.

On the one hand, **low-frequency analog electronics** deals with continuous signals and exploits the governing principles of circuit theory, signal processing and control systems to develop active circuits embedding either amplifiers, filters, oscillators and mixers. The CAD tools used by designers to validate their models are SPICE-like simulators, e.g, LTSpice, PSpice and Cadence Spectre: these programs analyze circuits by creating a netlist, i.e., a text description of the network, and using Kirchoff's laws to find the values of voltage and current at each node; depending on the complexity of the network, the mathematical model of the circuit can have variable complexity, but a solution is always found with a reasonably low computational cost.

On the other hand, **digital designers** aim at synthesizing elements like microprocessors, logic gates and memory units using hardware description languages, e.g., VHDL and Verilog, synthesis tools and FPGA/ASIC development platforms, exploiting the principles of Boolean logic, finite state machines and synchronous timing. Thanks to the high level of predictability of this kind of circuits, nowadays the design process is highly automatized and advanced tools exist to automatically simulate and synthesize the system, obtaining detailed reports about timing, area occupation and power consumption.

When dealing with **high-frequency design** and **electromagnetic modeling**, instead, the aim is typically to evaluate a field distribution, impedance matching, wave propagation through a material or a radiation pattern exploiting the Maxwell's equations and the other principles of electromagnetic wave theory, e.g., the equivalence theorem.

In order to do this, there are no standardized methodologies as the difficulty of the problem from the mathematical and computational point of view strongly depends on the complexity of the system and on the specific boundary conditions. Since, in general, it is very difficult to solve the problem in closed form, the most common approach consists in resorting to full-wave simulations with EM solvers like CST MW Studio® [46], HFSS or FEKO.

All these programs are high-performance 3D EM analysis software packages that provide tools to design and assess the performance of any electromagnetic systems. Despite their versatility

and high-performance computing capabilities, the computational cost of the simulations can be very high, especially when dealing with physically large structures. Furthermore, despite the results being precise and reliable, the accuracy of the proposed solution strongly depends on the dimension of the mesh elements, which can become very small when operating at high frequency, i.e., small wavelength.

As far as metasurfaces are concerned, the design approach exploited in most research work relies almost completely on full-wave simulations, which are exploited both to find the best **unit cell configuration** and to **assess the performance** of the antenna, resorting to an anlytical model only to evaluate the phase distribution of the structure to get the desired radiation pattern.

Despite the accuracy of the calculations performed by the CAD program, this approach has also many drawbacks: above all, the huge computational cost of a design based only on simulations prevents a thorough exploration of potential unit cell and antenna configurations, which limits confidence in the optimality of the final design.

Moreover, as simulation outcomes can vary significantly depending on the setup and may even conflict with each other, this methodology can only be deemed fully reliable if the simulation framework has been previously validated against empirical results from analogous designs — a condition that is not consistently met in metasurface-based applications, which remain an active area of research and development, with scarce experimental data available.

1.4 Motivation

In light of all the considerations made in this Chapter, the aim of this work is to develop and validate an analytical model that provides antenna engineers an efficient tool to conduct a preliminary design and performance assessment of **innovative multi-layer dielectric-only metasurfaces for O2I communication**.

This model is intended to become a fast and reliable alternative to full-wave simulations, enabling efficient exploration of design parameters and offering physical insight into the underlying electromagnetic phenomena; it is founded upon the following core principles:

Advanced electromagnetics and materials physics. Exploited to evaluate the electric and magnetic properties of each metasurface element; in particular, the Clausius-Mossotti relation [5] and the Maxwell Garnett model [16] are used to find the effective permittivity of the elementary cells in terms of geometry and dielectric constants of the constituent materials;

Transmission-line theory and matrix description of two-ports . Used to model signal propagation across the antenna by calculating the reflection and transmission coefficients of each metasurface element;

Advanced techniques for mathematical optimization. Used to obtain the best possible values of the scattering coefficients of the unit cells, thus enhancing the overall performance of the antenna;

Antenna array theory. Used to evaluate the radiation pattern of the smart surface as the superposition of the contributions of the single unit cells.

By bridging the gap between purely simulation-based and experimental approaches, the proposed methodology seeks to streamline the metasurface design process and facilitate the integration of Smart Electromagnetic Skins into next-generation wireless networks.

In fact, once the analytical model is successfully validated against empirical results, full-wave simulations become necessary only as a final check of performance prior to the realization of the prototype, significantly reducing the overall computational effort and accelerating the development cycle.

Keeping into account the specific application considered in this Thesis, an unconventional approach has been adopted. As already said, since the transmitting surface is to be integrated into the glass pane of a window, traditional PCB-based transmitting antennas are not appropriate; therefore, a purely dielectric, multi-layered configuration has been introduced, exploiting the promising results presented in [25]. This innovative solution offers two significant advantages:

- The **window's glass** itself becomes an integral part of the smart structure; consequently, its physical characteristics — such as thickness, dielectric permittivity and number of glass layers and air gaps — must be incorporated into the design methodology. This integration perfectly exemplifies the concept of *smart radio environment*: the materials and structures that are present in the antenna environment actively contribute to the manipulation of the electromagnetic field, rather than serving merely as passive obstacles to signal propagation.
- The smart skin exhibits minimal visual impact, provided that it is fabricated using **transparent materials** and manufactured through **appropriate techniques**. This aspect is particularly relevant in applications where aesthetics and visual discretion are crucial design constraints, and therefore introduces an additional set of engineering challenges that must be carefully addressed, finding a reasonable trade-off between performance and transparency.

1.5 Thesis Outline

In this Section, the organization of the Thesis work is briefly summarized.

Chapter 1. Introduces the concept of Smart Radio Environments (SREs), discusses the motivation behind the research, and outlines the main challenges and opportunities in the field. The chapter also presents the modeling and analysis of Smart Electromagnetic Skins (SESs) and Transmitarrays (TA), with a particular focus on dielectric-only transmitting metasurfaces for Outdoor-to-Indoor (O2I) communication.

Chapter 2. Provides an overview of advanced optimization techniques for electromagnetic design, with a particular focus on Genetic Algorithms (GA), which is the basis of the analytical model presented in the following chapters.

The chapter discusses the theoretical foundations of optimization in the context of metasurface and antenna design, reviews common approaches such as gradient-based and stochastic methods, and highlights the advantages of using GAs for complex, multi-parameter problems. Practical implementation details and application examples relevant to the Thesis are also presented.

Chapter 3. Focuses on the design and optimization of the unit cell (UC), which is the fundamental building block of the metasurface. The chapter details the analytical modeling of the unit cell, discusses the selection of materials and geometries, and presents the optimization strategies employed to achieve the desired electromagnetic response. Both theoretical and simulation-based results are compared to validate the proposed design methodology.

Chapter 4. Describes the synthesis and analysis of a complete Transmitarray antenna starting from the UC model developed in Chapter 3.

First of all, an analytical model based on the principles of Physical Optics (PO) is developed and implemented in MATLAB® [35] to predict the far-field radiation pattern of the antenna.

Then, this model is successfully validated against full-wave simulations, that also allow for a more realistic evaluation of the antenna's overall performance in terms of side-lobe level, back-lobe level, beamwidth and efficiency. Results are provided for broadside, non-broadside and mechanically-steered TAs; it is also shown that, properly tuning the mathematical optimization technique for the UCs, the quality of the radiation pattern of the structure can be significantly enhanced.

Chapter 5. Adapts the design model validated for Transmitarrays to planar SESs illuminated by a linearly polarized plane wave, which plays the role of the field generated by a base station. The adopted methodology is the same as Transmitarrays.

Chapter 6. Summarizes the main findings of the Thesis, discusses the implications of the developed analytical model, and outlines potential directions for future research in the field of smart electromagnetic environments and metasurface-based antennas.

Chapter 2

Advanced Techniques for Optimization

2.1 Terminology and Definitions

In mathematics, **optimization** is the selection of a best element with regard to some criteria from some set of available alternatives. In the simplest case, an optimization problem consists of minimizing (or maximizing) a real function (the **objective**) by systematically choosing input values for design variables within an allowed set while satisfying all constraints [51].

The basic components of an optimization problem are listed below.

Definition 2.1.1 (Design/optimization variables)**.** *Denoted with* $\mathbf{x}$*, it is a set of* $n \in \mathbb{N} \setminus \{0\}$ *independent unknowns or variables, whose domain of definition* $\mathbb{X} \subseteq \mathbb{R}^n$ *is a linear space referred to as decision set:*

$$\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{X} \subseteq \mathbb{R}^n. \tag{2.1}$$

The optimization variables can either be continuous, discrete or variable depending on the nature of the problem; if $n = 1$ *the optimization problem is said univariate, if* $n > 1$ *it is said multivariate.*

Definition 2.1.2 (Objective/cost/fitness function)**.** *Denoted with* f*, it is a real function of the design variables* $\mathbf{x}$*, which expresses the main aim of the model and is either to be maximized or minimized:*

$$\mathbb{R}^n \ni \mathbf{x} \mapsto f(\mathbf{x}) \in \mathbb{R}. \tag{2.2}$$

Definition 2.1.3 (Feasible set)**.** *Denoted with* $\mathcal{F} \subseteq \mathbb{X}$*, it is a set of* $m \in \mathbb{N}$ *equality or (strict) inequality constraints that, by means of an equal number of gauge functions* $g_i : \mathbb{X} \to \mathbb{R}$*,* $i = 1, \dots, m$*, allow the design variables to take on certain values but exclude others:*

$$\mathcal{F} := \{\, \mathbf{x} \in \mathbb{X} \;:\; g_i(\mathbf{x}) = / \leq / < 0,\ i = 1, \dots, m \,\} \subseteq \mathbb{X}. \tag{2.3}$$

If $m = 0$*, the problem is said unconstrained, if* $m > 0$ *it is said constrained. Even though, in principle, constraints are not essential, it has been argued that almost all real-world problems do have constraints, i.e., restrictions that must be satisfied to produce an acceptable design. Constraints can be broadly classified as*

- ***Behavioral*** *or* ***functional constraints****: they represent limitations on behavior and performance of the system;*
- ***Geometric*** *or* ***side constraints****: they represent physical limitations on design variables, such as availability, fabricability and transportability.*

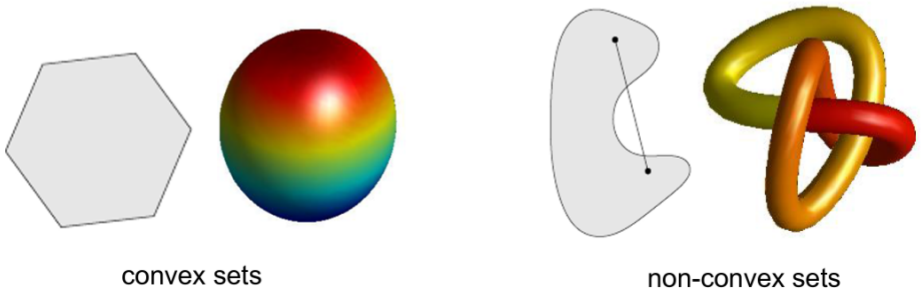


Figure 2.1: Examples of convex and non-convex sets [51].

Thanks to these definitions, the most general formulation of an optimization model is very simple and elegant:

$$\widetilde{\mathbf{x}} = \min_{\mathbf{x}\in\mathcal{F}} \{f(\mathbf{x})\} \quad \text{or} \quad \widetilde{\mathbf{x}} = \max_{\mathbf{x}\in\mathcal{F}} \{f(\mathbf{x})\}, \tag{2.4}$$

depending whether the objective function f must be minimized or maximized. Let us point out that every optimization model can be written as a *minimization problem* even in the case when f should be maximized; in fact:

$$\max_{\mathbf{x}\in\mathcal{F}} \{f(\mathbf{x})\} \equiv \min_{\mathbf{x}\in\mathcal{F}} \{-f(\mathbf{x})\}. \tag{2.5}$$

Some additional concepts and useful results are now presented.

Definition 2.1.4 (Global and local minimum)**.** *A point $\mathbf{x}^\star \in \mathcal{F} \subseteq \mathbb{X}$ is*

- *a **global minimum** if*
$$f(\mathbf{x}^\star) \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{F} \ : \ \mathbf{x} \neq \mathbf{x}^\star. \tag{2.6}$$
- *a **local minimum** if there exists an open neighborhood $\mathcal{N} \subset \mathcal{F}$ such that*
$$f(\mathbf{x}^\star) \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{N} \ : \ \mathbf{x} \neq \mathbf{x}^\star. \tag{2.7}$$
- *a **strict global minimum** if*
$$f(\mathbf{x}^\star) < f(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{F} \ : \ \mathbf{x} \neq \mathbf{x}^\star. \tag{2.8}$$
- *a **strict local minimum** if there exists an open neighborhood $\mathcal{N} \subset \mathcal{F}$ such that*
$$f(\mathbf{x}^\star) < f(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{N} \ : \ \mathbf{x} \neq \mathbf{x}^\star. \tag{2.9}$$

Definition 2.1.5 (Convex set)**.** *A set $\mathcal{S} \subseteq \mathbb{X}$ is convex if*

$$\forall\, \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{S}, \alpha \in (0,1) \implies \alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2 \in \mathcal{S}. \tag{2.10}$$

In practice, the line segment connecting any two points in $\mathcal{S}$ completely belongs to $\mathcal{S}$.

Definition 2.1.6 (Convex function)**.** *A function $f : \mathcal{S} \to \mathbb{R}$ is convex if*

- *$\mathcal{S}$ is convex;*
- $\forall\, \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{S}, \alpha \in (0,1) \implies f(\alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1-\alpha) f(\mathbf{x}_2)$.

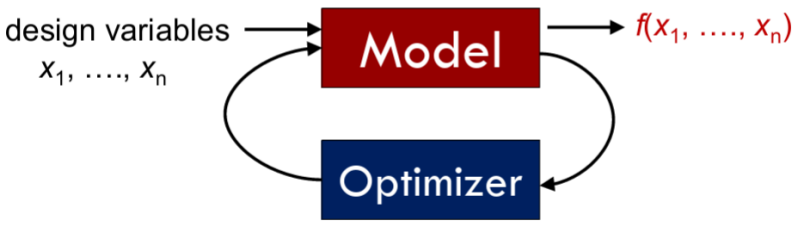


Figure 2.2: Iterative optimization algorithms [51].

The definition of strict convexity immediately follows replacing $\leq$ with $<$.

Theorem 2.1.1 (Conditions for convexity)**.** *Let $f : \mathcal{S} \to \mathbb{R}$ be a continuous function. The following facts hold:*

(a) If f is derivable on $\text{int}(\mathcal{S})$, then f is convex if and only if $f' : \text{int}(\mathcal{S}) \to \mathbb{R}$ is monotonically increasing.

(b) If f can be derived two times on $\text{int}(\mathcal{S})$, then f is convex if and only if $f''(\mathbf{x}) \geq 0 \; \forall \mathbf{x} \in \text{int}(\mathcal{S})$.

Theorem 2.1.2 (Minima of convex functions)**.** *If a function $f : \mathcal{S} \to \mathbb{R}$ is convex, then every local minimum of f is a global minimum. Moreover, if f is strictly convex, then the global minimum is unique.*

An immediate and important consequence of Theorem 2.1.2 is that, when optimizing a convex function, it is impossible to get stuck in local minima.

2.2 Iterative Optimization

Definition 2.2.1 (Optimization Algorithm)**.** *Finite step-by-step procedure that aims at finding the optimal solution of an optimization problem [52]. The most general classification of Optimization Algorithms (OAs) is the following:*

Exact OA. *Guarantees the optimality of the returned solution at the end of the procedure.*

Heuristic OA. *Does not give any guarantee on the optimality of the solution.*

Metaheuristic/approximation OA. *Specific heuristics that guarantees at least a maximum distance from the optimality in terms of the objective function.*

Since most optimization problems do not have any analytical solution, it is often necessary to resort to iterative OAs [51], whose operating principle is schematically shown in Figure 2.2.

Mathematically speaking, iterative methods optimize $f : \mathbb{R}^n \supseteq \mathbb{X} \to \mathbb{R}$ according to the this iteration procedure:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k, \quad \text{where} \quad \begin{cases} k \in \mathbb{N} \text{ is the iteration index,} \\ \mathbf{x}_k \in \mathbb{X} \text{ is the estimate of minimizer,} \\ \alpha_k \in [0, +\infty) \text{ is the step size,} \\ \mathbf{d}_k \in \mathbb{X} \text{ is the search direction,} \\ \mathbf{x}_0 \in \mathbb{X} \text{ is the initial condition.} \end{cases} \tag{2.11}$$

Definition 2.2.2 (Convergence)**.** *An iterative algorithm with initial condition $\mathbf{x}_0 \in \mathbb{X}$:*

(a) Globally converges to $\mathbf{x}^\star \in \mathbb{X}$ *if*

$$\lim_{k\to+\infty} \mathbf{x}_k = \mathbf{x}^\star \quad \forall \mathbf{x}_0 \in \mathbb{X}. \tag{2.12}$$

(b) Locally converges to $\mathbf{x}^\star \in \mathbb{X}$ *if*

$$\lim_{k\to+\infty} \mathbf{x}_k = \mathbf{x}^\star \quad \forall \mathbf{x}_0 \in \mathcal{N}(\mathbf{x}^\star), \tag{2.13}$$

where $\mathcal{N}(\mathbf{x}^\star) \subset \mathbb{X}$ *is an open set such that* $\mathbf{x}^\star \in \mathcal{N}(\mathbf{x}^\star)$ *and* $\mathbf{x}^\star = \min \mathcal{N}(\mathbf{x}^\star)$.

Definition 2.2.3 (Error)**.** *Let* $\{\mathbf{x}_0, \mathbf{x}_1, \dots \mathbf{x}_k, \dots\} \subset \mathbb{X}$ *be a sequence that converges to* $\mathbf{x}^\star \in \mathbb{X}$*; the error* $\mathbf{e}_k$ *at iteration* $k \in \mathbb{N}$ *is the quantity such that*

$$\mathbf{e}_k := \mathbf{x}_k - \mathbf{x}^\star, \quad \lim_{k\to+\infty} \mathbf{e}_k = \mathbf{0}. \tag{2.14}$$

Definition 2.2.4 (Rate)**.** *The sequence* $\{\mathbf{x}_0, \mathbf{x}_1, \dots \mathbf{x}_k, \dots\} \subset \mathbb{X}$ *converges to* $\mathbf{x}^\star \in \mathbb{X}$ *with rate* $r \in \mathbb{R}^+$ *if*

$$\lim_{k\to+\infty} \frac{\|\mathbf{e}_{k+1}\|}{\|\mathbf{e}_k\|^r} \in \mathbb{R}^+, \tag{2.15}$$

where $\|\bullet\|$ *is the* $\mathbb{R}^n$*-norm operator.*

Depending on the rate, the speed of convergence of an optimization algorithm is

- **Linear**: $r = 1$ and $\exists c \in (0, 1)$ such that, for k large enough:

$$\frac{\|\mathbf{e}_{k+1}\|}{\|\mathbf{e}_k\|} \leq c. \tag{2.16}$$

- **Superlinear**: $r = 1$ and

$$\lim_{k\to+\infty} \frac{\|\mathbf{e}_{k+1}\|}{\|\mathbf{e}_k\|} = 0. \tag{2.17}$$

- **Squared**: $r = 2$ and $\exists c \in [0, +\infty)$ such that, for k large enough:

$$\frac{\|\mathbf{e}_{k+1}\|}{\|\mathbf{e}_k\|^2} \leq c. \tag{2.18}$$

In practice, to implement an iterative optimization method, the following steps are necessary starting from $k = 0$ and $\mathbf{x}_k = \mathbf{x}_0$:

1. Computation of a search direction $\mathbf{d}_k$, which is parallel to $\nabla f(\mathbf{x}_k)$, i.e., it indicates the direction of maximum descendent of $f(\mathbf{x}_k)$:

$$\mathbf{d}_k = -\nabla f(\mathbf{x}_k). \tag{2.19}$$

2. Computation of a step size α_k with the aim of minimizing f, that is

$$f(\mathbf{x}_k + \alpha_k \mathbf{d}_k) < f(\mathbf{x}_k). \tag{2.20}$$

3. Update of $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$.
4. Check for convergence, i.e., $\nabla f(\mathbf{x}_{k+1}) = \mathbf{0}$.

2.3 Classification of Optimization Techniques

Depending on the information utilized, OAs can be classified into three categories [51]:

Zero-Order Methods. OAs that utilize only function ($f(\mathbf{x})$) and constraint ($g_i(\mathbf{x})$).

Typical examples are Random Search, Golden Section Search and several population based heuristic, such as Particle Swarm Optimization and Genetic Algorithms.

First-Order Methods. OAs that, apart from function ($f(\mathbf{x})$) and constraint ($g_i(\mathbf{x})$), utilize also partial derivatives or gradients of function ($\partial f(\mathbf{x})$) and constraint ($\partial g_i(\mathbf{x})$).

Typical examples are Gradient Descent and Stochastic Gradient Descent.

Second-Order Methods. OAs that, apart from function ($f(\mathbf{x})$) and constraint ($g_i(\mathbf{x})$), utilize also hessian and gradients information.

Typical examples are Newton Method and BFGS Method.

Optimization algorithms can be implemented quite easily in most high-frequency and electromagnetic CAD/simulation tools: for instance, CST MW Studio® provides a built-in optimization module, which allows users to define objective functions, constraints, and design variables directly within the simulation environment, enabling automated parameter sweeps and optimization runs.

The choice of optimization technique often depends on the problem's complexity, the availability of derivative information, and computational resources; nevertheless, the available options nare the following [41]:

- Covariance Matrix Adaptation Evolutionary Strategy:
- Trust Region Framework (TRF): fast and accurate local optimizer that robustly find the optimum within the constraints using a low number of evaluations;
- Genetic Algorithm (GA): global optimizer that generates points in the feasible set and then refines them through multiple generations;
- Particle Swarm Optimization: global optimizer that treats points in the parameter space as moving particles whose position changes at each iteration;
- Nelder Mead Simplex Algorithm: derivative-free local optimizer that uses the concept of a simplex (a polytope of $n + 1$ vertices in n dimensions) to search for the minimum by reflecting, expanding, and contracting the simplex in the parameter space; it is less dependent on the starting point than most local optimizers;
- Interpolated Quasi Newton: fast local optimizer, that works well as search algorithm for expensive problems; the feasible space is sampled in each variable direction and interpolation is used to estimate the gradient of the cost function;
- Classic Powell: local optimizer that robustly find the optimum within the constraints, sometimes needing many iterations as the optimum is being approached;
- Decap Optimization: specialized optimizer for PCB design, which calculates the most effective placement of decoupling capacitors with the Pareto front method.

Despite the potentialities of these algorithms, the optimization of the geometric parameters of a metasurface poses distinct challenges which makes difficult the direct exploitation of OAs inside simulation tools.

In fact, the design space is typically high-dimensional and strongly non-convex, riddled with numerous local minima due to complex electromagnetic interactions between individual elements. Additionally, each evaluation of the objective function often requires a full-wave electromagnetic simulation, which can be computationally prohibitive.

To address these issues, global optimization algorithms — particularly those that do not rely on gradient information, such as Genetic Algorithms or Particle Swarm Optimization — are frequently favored in metasurface design. These methods efficiently explore the design space and are less prone to becoming trapped in local minima, albeit at the cost of a greater number of function evaluations.

Therefore, selecting an appropriate optimization technique involves balancing the need for global search capabilities with computational resource constraints. To mitigate these limitations, the approach adopted in this thesis involves conducting the optimization analytically prior to performing full-wave simulations. The chosen optimization algorithm is the **Genetic Algorithm** (GA), described in detail in Section 2.4.

2.4 Genetic Algorithm

Among all the metaheuristic optimizers, **Evolutionary Algorithms** (EAs) in the last years definitively proved their effectiveness; usually, EAs are defined as generic population based metaheuristic optimization algorithms exploiting mechanisms inspired by biological evolution [51].

The core principles of an Evolutionary Algorithm are the following:

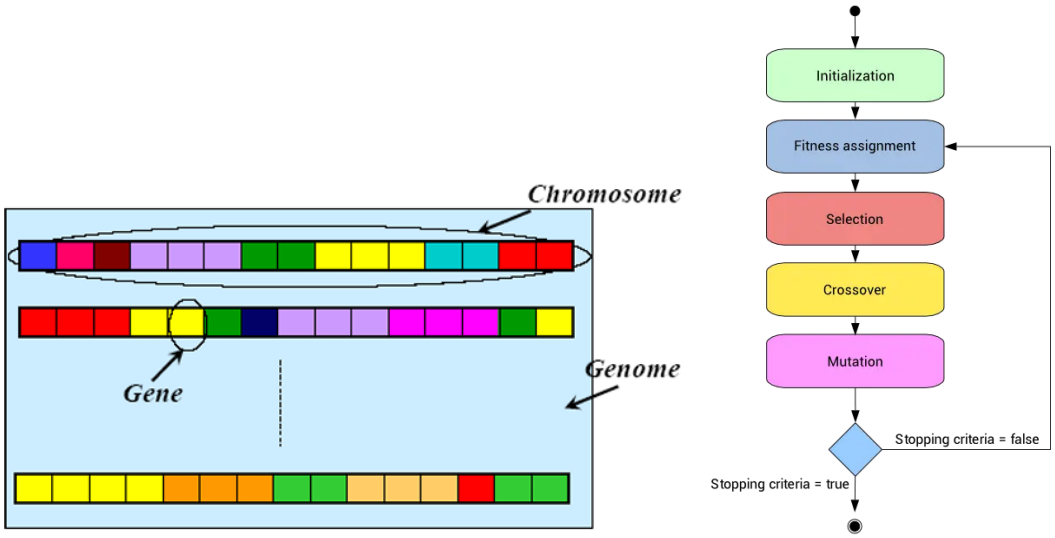
1. Candidate solutions to the optimization problem play the role of individuals in a population.
2. The objective function determines the environment within which the solutions "live".
3. Population evolves in *generations*: new individuals (offspring) are created by combining features of current individuals; in thisn process, a key role is played by randomness.
4. Individuals evolve using variation operators (e.g., *mutation*, *recombination*) acting directly on their solution representations.
5. The next generation consists of a mix of offspring and parents according to the *survivor selection* strategy.
6. The population size is (almost always) fixed.
7. Selection may involve multiple copies of a given parent individual.
8. The best individual is (almost always) carried over to the next generation.

Among others, the most common EA is the **Genetic Algorithm** (GA), developed by John Holland at University of Michigan in 1970s and now widely used in economics, engineering and machine learning applications.

The key concepts of this optimization method are the following; for a visual representation, see Figure 2.3a:

Individual. Any possible solution of the optimization problem.

Chromosome or Genotype. Proposed solution of the optimization problem.



(a) Chromosomes, genes and genomes [51] **(b)** Evolutionary cycle [18]

Figure 2.3: Genetic Algorithm.

Search Space. All possible solutions to the problem, i.e., the feasible set $\mathcal{F}$.

Population. Subset of solutions in the current generation.

Genome. Complete set of chromosomes.

Trait. Property of an individual.

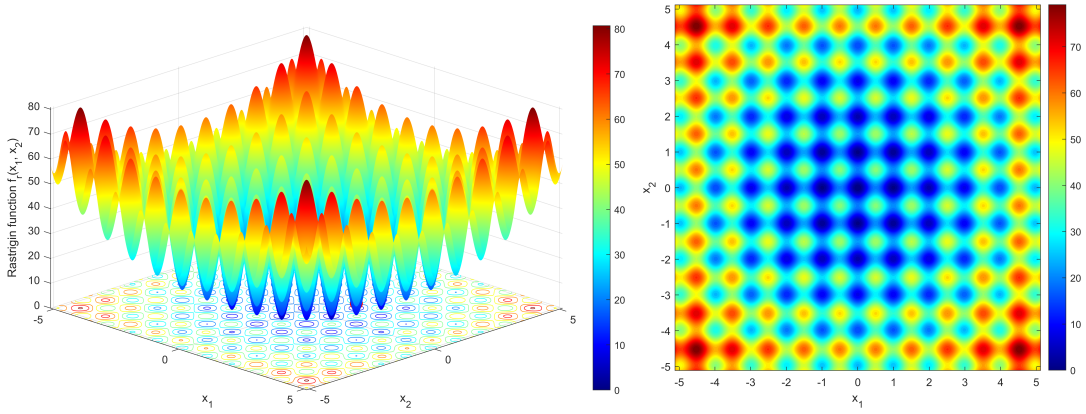
Gene. Property of a chromosome.

Locus. Position of a gene on the chromosome.

Evolutionary Cycle. Core of the OAs, it is a sequence of six steps, shown in Figure 2.3b, that allows finding the optimal solution:

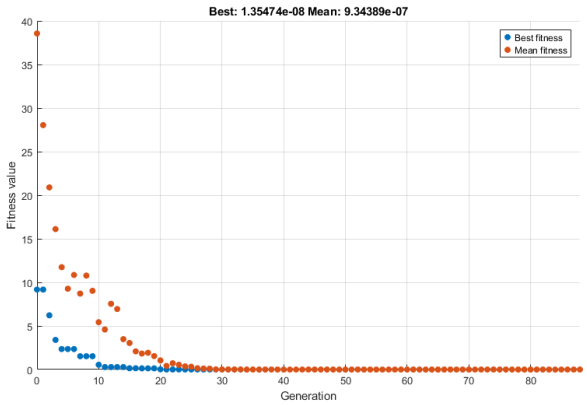
1. **Initialization**: randomly generate of a population of candidate solutions, often represented as arrays of bits.
2. **Fitness assignment**: assess of the fitness function of each solution based on defined criteria.
3. **Selection**: choose the most fit solutions and discard the weaker ones.
4. **Crossover**: combine selected *parent* solutions to create *children*, ideally inheriting optimal traits.
5. **Mutation**: introduce small random changes to maintain diversity and avoid stagnation.
6. **Repetition**: repeat the process across generations until a satisfactory solution is found or the population overall reaches an acceptable fitness level.

To test the functionalities of the GA, let us introduce the **Rastrigin function** of order $N \in \mathbb{N} \setminus \{0\}$, a non-convex function typically used as benchmark for many optimization algorithms



(a) Rastrigin function (3D)

(b) Rastrigin function (contours)



(c) Fitness improvement

Figure 2.4: Example of implementation of the GA algorithm with Rastrigin function.

[44]:

$$[5.12, 5.12]^n \ni \mathbf{x} \mapsto f_r(\mathbf{x}) := 10N + \sum_{i=1}^{N} \left[x_i^2 - 10\cos(2\pi x_i) \right] \in \mathbb{R}. \tag{2.21}$$

In the particular case $N = 2$, graphically represented in Figures 2.4a and 2.4b, Equation (2.21) reduces to

$$f_r(x_1, x_2) = 20 + x_1^2 + x_2^2 - 10\left[\cos(2\pi x_1) + \cos(2\pi x_2)\right], \quad (x_1, x_2) \in [5.12, 5.12]^2. \tag{2.22}$$

As the 3D and contour plots show, f_r displays a large number of local minima surrounding the global minimum

$$(\widetilde{x}_1, \widetilde{x}_2) = (0, 0) \implies f_r(\widetilde{x}_1, \widetilde{x}_2) = f_r(0, 0) = 0. \tag{2.23}$$

Due to its tophology, the Rastrigin function is a great stress-test for the Genetic Algorithm because the probability of falling inside a local minima during the optimization process is high: indeed, running the algorithm with the MATLAB® function `ga()` (see Section 3.4.2 for a detailed description), the optimal solution sometimes does not always fall in the right position, as the listings below clearly show.

```
Single objective optimization:
2 Variables

Options:
CreationFcn:       @gacreationuniform
CrossoverFcn:      @crossoverscattered
SelectionFcn:      @selectionstochunif
MutationFcn:       @mutationadaptfeasible

                                 Best           Mean      Stall
Generation      Func-count        f(x)           f(x)    Generations
    1              100           8.771          28.53        0
    2              147           3.542          21.19        0
    3              194           3.542          15.86        1
    4              241           1.556          14.04        0
    5              288           1.556           9.55        1
    6              335           1.556          9.539        2
    7              382           1.556           8.16        3
    8              429           1.556          6.156        4
    9              476           0.232          3.661        0
   10              523           0.232          2.619        1
   11              570           0.232          2.002        2
   12              617           0.117          1.008        0
   13              664          0.1004         0.8555        0
   14              711          0.1004         0.9056        1
   15              758          0.0123         0.6737        0
   16              805          0.0123         0.7242        1
   17              852          0.0123         0.5491        2
   18              899          0.0123          0.327        3
   19              946          0.0123         0.2545        4
   20              993        0.002611         0.1366        0
   21             1040        3.11e-05         0.0659        0
   22             1087        3.11e-05         0.1091        1
   23             1134        3.11e-05        0.06947        2
   24             1181        3.11e-05        0.03539        3
   25             1228        3.11e-05        0.02328        4
   26             1275        3.11e-05       0.009218        5
   27             1322        3.11e-05       0.008497        6
   28             1369        3.11e-05       0.001168        7
   29             1416       4.921e-06      0.0007673        0
   30             1463       3.372e-06      0.0003479        0

                                 Best           Mean      Stall
Generation      Func-count        f(x)           f(x)    Generations
   31             1510       2.393e-06      0.0002053        0
   32             1557       2.393e-06      0.0002668        1
   33             1604       2.393e-06      0.0001693        2
   34             1651       2.393e-06       8.862e-05       3
   35             1698       1.828e-06       6.247e-05       0
   36             1745       1.828e-06       4.377e-05       1
   37             1792       1.828e-06       1.772e-05       2
   38             1839       1.352e-07       1.509e-05       0
   39             1886       1.352e-07       1.426e-05       1
   40             1933       1.352e-07       6.424e-06       2
   41             1980       1.352e-07       3.725e-06       3
   42             2027       1.352e-07       2.861e-06       4
   43             2074       1.352e-07       1.606e-06       5
   44             2121       1.352e-07       1.412e-06       6
   45             2168       1.352e-07       1.559e-06       7
   46             2215       1.352e-07       1.545e-06       8
   47             2262       1.352e-07        1.42e-06       9
   48             2309       1.352e-07       1.717e-06      10
   49             2356       1.352e-07       1.797e-06      11
   50             2403       1.352e-07       1.693e-06      12
   51             2450       1.352e-07       1.682e-06      13
   52             2497       1.352e-07       2.114e-06      14
   53             2544       1.352e-07       2.006e-06      15
   54             2591       1.352e-07       1.702e-06      16
```

```
    55              2638       1.352e-07       1.903e-06       17
    56              2685       1.352e-07       2.196e-06       18
    57              2732       1.352e-07       1.441e-06       19
    58              2779       1.352e-07        1.42e-06       20
    59              2826       1.352e-07       1.526e-06       21
    60              2873       1.352e-07       1.281e-06       22

                                   Best           Mean      Stall
Generation      Func-count         f(x)           f(x)    Generations
    61              2920       1.352e-07       1.611e-06       23
    62              2967       1.352e-07       1.895e-06       24
    63              3014       1.352e-07       2.118e-06       25
    64              3061       1.352e-07       1.699e-06       26
    65              3108       1.352e-07       1.515e-06       27
    66              3155       1.352e-07       1.451e-06       28
    67              3202       1.352e-07       1.478e-06       29
    68              3249       1.352e-07       1.793e-06       30
    69              3296       1.352e-07        1.34e-06       31
    70              3343       1.352e-07       1.722e-06       32
    71              3390       1.352e-07       2.029e-06       33
ga stopped because the average change in the fitness value is less than options.FunctionTolerance.
Optimal Solution: x = [0.000001, 0.000026]
Optimal Function Value: f(x) = 0.000000
```

```
Single objective optimization:
2 Variables

Options:
CreationFcn:       @gacreationuniform
CrossoverFcn:      @crossoverscattered
SelectionFcn:      @selectionstochunif
MutationFcn:       @mutationadaptfeasible

                                   Best           Mean      Stall
Generation      Func-count         f(x)           f(x)    Generations
     1              100           10.19           31.32        0
     2              147           5.308            23.2        0
     3              194           2.305           18.48        0
     4              241           2.305           13.81        1
     5              288           2.305           13.47        2
     6              335           2.305           12.63        3
     7              382           2.305           10.06        4
     8              429           2.305            7.13        5
     9              476           2.012           4.485        0
    10              523           2.012           3.974        1
    11              570           2.012           3.605        2
    12              617           2.012           2.782        3
    13              664           2.003           2.605        0
    14              711               2           2.518        0
    15              758               2           2.555        1
    16              805           1.991            2.11        0
    17              852           1.991           2.112        1
    18              899           1.991            2.07        2
    19              946           1.991           2.035        3
    20              993           1.991           2.019        4
    21             1040           1.991           2.011        5
    22             1087            1.99           2.003        0
    23             1134            1.99           1.991        0
    24             1181            1.99           1.991        0
    25             1228            1.99           1.994        1
    26             1275            1.99           1.993        2
    27             1322            1.99           1.991        3
    28             1369            1.99            1.99        4
    29             1416            1.99            1.99        0
    30             1463            1.99            1.99        1

                                   Best           Mean      Stall
```

```
Generation      Func-count        f(x)           f(x)    Generations
   31             1510            1.99           1.99        2
   32             1557            1.99           1.99        0
   33             1604            1.99           1.99        0
   34             1651            1.99           1.99        1
   35             1698            1.99           1.99        2
   36             1745            1.99           1.99        3
   37             1792            1.99           1.99        0
   38             1839            1.99           1.99        1
   39             1886            1.99           1.99        0
   40             1933            1.99           1.99        1
   41             1980            1.99           1.99        2
   42             2027            1.99           1.99        3
   43             2074            1.99           1.99        0
   44             2121            1.99           1.99        1
   45             2168            1.99           1.99        2
   46             2215            1.99           1.99        3
   47             2262            1.99           1.99        4
   48             2309            1.99           1.99        5
   49             2356            1.99           1.99        6
   50             2403            1.99           1.99        7
   51             2450            1.99           1.99        8
   52             2497            1.99           1.99        0
   53             2544            1.99           1.99        1
   54             2591            1.99           1.99        2
   55             2638            1.99           1.99        3
   56             2685            1.99           1.99        4
   57             2732            1.99           1.99        5
   58             2779            1.99           1.99        6
   59             2826            1.99           1.99        7
   60             2873            1.99           1.99        8

                                  Best           Mean      Stall
Generation      Func-count        f(x)           f(x)    Generations
   61             2920            1.99           1.99        9
   62             2967            1.99           1.99       10
   63             3014            1.99           1.99       11
   64             3061            1.99           1.99       12
   65             3108            1.99           1.99       13
   66             3155            1.99           1.99       14
   67             3202            1.99           1.99       15
   68             3249            1.99           1.99       16
   69             3296            1.99           1.99       17
   70             3343            1.99           1.99       18
   71             3390            1.99           1.99        0
   72             3437            1.99           1.99        1
   73             3484            1.99           1.99        2
   74             3531            1.99           1.99        3
ga stopped because the average change in the fitness value is less than options.FunctionTolerance.
Optimal Solution: x = [0.994959, 0.994978]
Optimal Function Value: f(x) = 1.989918
```

Chapter 3

Unit Cell Design and Optimization

As anticipated in the previous chapters, the fundamental principles of EM modeling, physical optics and mathematical optimization can be effectively leveraged to design Transmitarrays and Smart Electromagnetic Skins through a purely theoretical approach, relying on numerical simulations only during the pre-prototyping phase for structural validation. This methodology is not only more time-efficient than conventional simulation-driven design flows, but it also holds significant promise for future research. By reducing computational overhead, it enables the exploration of a broader design space and a wider range of configurations, as envisaged in [14].

In this chapter, an **innovative unit cell model** tailored for TAs has been developed starting from the equivalent circuit model presented in [30] for a perforated dielectric-only RA, suitably modified and extended to a generic multi-layer configuration (Section 3.1). Each layer has been modeled as an equivalent transmission line section, whose parameters depend on both the choice of the materials and the geometry of the hole.

Following validation against CST MW Studio® simulations (Section 3.2), the model is further extended to tapered structures (Section 3.3). A MATLAB® subroutine based on the **multi-objective genetic algorithm** (GA) is then implemented to optimize the geometric parameters of the UC (Section 3.4); the optimization targets include minimizing the reflection coefficient and maximizing the transmission coefficient for a specified set of incidence angles and a desired phase delay. Finally, the main challenges and limitations encountered during the modeling and optimization process are discussed in Section 3.5.

3.1 Unit Cell Design

Let f_0 be the **operating frequency** of the antenna and let Δf be the bandwidth of interest; since the TA will be designed to work effectively in the K_a-band[1], let us fix $f_0 = 30\,\text{GHz}$ and $\Delta f = 4\,\text{GHz}$; the operating frequency range will thus be

$$f \in \left[f_0 - \frac{\Delta f}{2}, f_0 + \frac{\Delta f}{2} \right] = [28, 32]\,\text{GHz}. \tag{3.1}$$

Let us also define the free-space operating **wavelength** λ_0 and **propagation constant** k_0:

$$\lambda_0 = \frac{c_0}{f_0} \quad \text{and} \quad k_0 = \frac{2\pi}{\lambda_0} \tag{3.2}$$

Since the UC will be embedded in a two-dimensional periodic structure, let us define a reference frame *Oxyz* characterized by

[1]The K_a-band is a portion of the MW part of the electromagnetic spectrum. Though there is no standard definition, *IEEE Standard letter designations for Radar Bands* defines it as the frequency range $27 \div 40\,\text{GHz}$.

- origin O at the center of the TA;
- xy plane coincident with the array plane;
- z axis orthogonal to the array, with positive direction towards the field source.

Let us also define the cartesian basis $\{\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}\}$ and the spherical basis $\left\{\hat{\mathbf{r}}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\varphi}}\right\}$. For a more detailed understanding of the adopted coordinate system, refer to the graphical representations shown in Chapter 4.

Finally, let

$$a = b = W \tag{3.3}$$

be the cell periodicities along the x and y direction, respectively. To keep the analysis as general as possible, for the time being it is not necessary to dive deep into further details about the geometry of the UC structure.

3.1.1 Maxwell Garnett Model

Once defined system of axes and the geometry of the structure, the first step to develop the equivalent circuit model of the UC is the determination of its **effective permittivity** using an accurate material science approach.

Coherently with the design flow adopted in [30], the analytical model developed by **Maxwell Garnett** in [16] was adopted.

This modeling, a.k.a. effective medium approximation (EMA) or effective medium theory (EMT), determines the properties of a composite material by averaging the multiple values of its constituents, using their relative fractions as weights.

The main limitation of this model is that the effective permittivity is an averaged dielectric characteristics of a microinhomogeneous medium and it is derived in quasi-static approximation, considering the electric field as homogeneous. As a consequence, this formulation cannot describe the particle size effect, though keeping into account these aspects is beyond the scope of this work.

For the sake of simplicity, let us concentrate on the case of a matrix embedding a single inclusion. As far as notation is concerned, index 1 refers to the inclusion material, while index 2 refers to the host material.

Let ϵ_1 and ϵ_2 be the complex permittivities of the two materials:

$$\begin{cases} \epsilon_1 = \epsilon_{r1}\,(1 - \mathrm{j}\tan\delta_1) \\ \epsilon_2 = \epsilon_{r2}\,(1 - \mathrm{j}\tan\delta_2) \end{cases} \tag{3.4}$$

Let us also define the *volume fraction* f_1 as the ratio between the inclusion volume V_1 and the total volume of the structure $V_1 + V_2$; since the unit cells of interest will be composed by a dielectric matrix (ϵ_2) and, possibly, an air inclusion (ϵ_2), let us introduce the following notation:

$$f_1 := \frac{V_1}{V_1 + V_2} = \frac{V_{\text{air}}}{V_{\text{tot}}} \tag{3.5}$$

The effective permittivity of the medium ϵ_{eff} must satisfy the following equation, derived from Lorentz local field correction theory and known as **Clausius-Mossotti formula** ([5], [28]):

$$\frac{\epsilon_{\text{eff}} - \epsilon_2}{\epsilon_{\text{eff}} + \epsilon_2} = \frac{4\pi}{3}\frac{\alpha_1}{\nu} \implies \epsilon_{\text{eff}} = \epsilon_2\,\frac{1 + \dfrac{8\pi}{3}\dfrac{\alpha_1}{\nu}}{1 - \dfrac{4\pi}{3}\dfrac{\alpha_1}{\nu}}, \tag{3.6}$$

where:

- α_1 is the polarizability of the inclusion material; modeling the inclusion particles as spheres with radius a, elementary electrostatics yields:

$$\alpha_1 = \left(\frac{\epsilon_1 - \epsilon_2}{\epsilon_1 + \epsilon_2}\right) a^3. \tag{3.7}$$

- ν is the specific volume per one inclusion molecule; if N is the total number of particles:

$$\nu := \frac{V_{\text{tot}}}{N} \equiv \frac{V_1 + V_2}{N}. \tag{3.8}$$

It is now possible to rewrite the volume fraction f_1 in the following way:

$$\begin{cases} V_1 = NV_{\text{particle}} \equiv N\,\dfrac{4\pi a^3}{3} \\ V_1 + V_2 = N\nu \end{cases} \implies f_1 = \frac{V_1}{V_1 + V_2} = \frac{4\pi}{3}\,\frac{a^3}{\nu}, \tag{3.9}$$

meaning that

$$\nu = \frac{4\pi a^3}{3f} \implies \frac{4\pi}{3}\,\frac{\alpha_1}{\nu} = f_1\,\frac{\epsilon_1 - \epsilon_2}{\epsilon_1 + \epsilon_2}. \tag{3.10}$$

Replacing Equation (3.10) into Equation (3.6), the expression of the effective permittivity as function of ϵ_1, ϵ_2 and f_1 is eventually obtained:

$$\epsilon_{\text{eff}} = \epsilon_2 \frac{2\epsilon_2 + \epsilon_1 + 2f_1\,(\epsilon_1 - \epsilon_2)}{2\epsilon_2 + \epsilon_1 - f_1\,(\epsilon_1 - \epsilon_2)}. \tag{3.11}$$

The MATLAB® routine to calculate the effective permittivity according to the MG model can be found in Appendix B.1.

3.1.2 Floquet analysis

At this point, the equivalent circuit of the UC can be derived modeling the structure according to the transmission line theory and considering it as a scatterer in a periodic array [30].

Each UC in the array behaves as a dynamic current source, typically referred to as **Floquet source** because the electromagnetic fields produced by the entire structure can be represented as the superposition of an infinite number of **Floquet modal functions** [8].

Depending on the frequency, some modes propagate (*propagating waves*) and some modes decay along the z-direction (*evanescent waves*). Both modes are indexed by m and n, respectively along the two x and y dimensions, according to the framework shown in Figure 3.1. In this sketch, "UC" represents a periodic unit cell boundary condition that enforces the two-dimensional periodicity along these dimensions.

Although the Transmitarray realized in this work are not perfectly periodical - the cells differ in terms of both cell height and hole size - for the time being it is a reasonable approximation.

In fact, the rationale for undertaking such infinite periodic array analysis is as follows [8]:

1. Elements in the central region of an electrically large array have similar features as that of an element in an infinite array;

2. The performance of a finite array can be predicted with a reasonable level of accuracy using infinite array results;

3. The infinite array results are useful to predict the mutual coupling between the elements in an array environment;

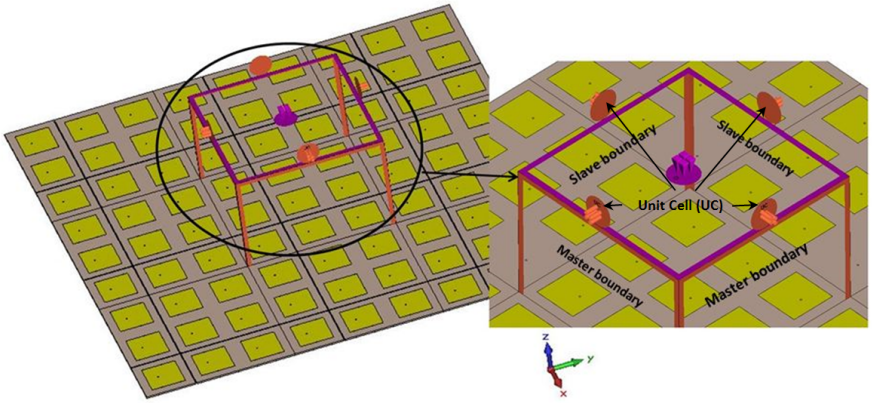


Figure 3.1: Typical Fouquet modal unit cell boundary condition.

4. The embedded element pattern that includes mutual coupling effects can be determined directly from the infinite array results, though this goes beyond the scope of this work.

The Floquet modal development naturally employs an S-matrix formulation for all of the modes, which in this work will be integrated with ABCD formalism to handle multi-layer structures.

The number of propagating modes depends on the frequency, scan angle, and unit cell size, but for a given array there ideally exists only one propagating mode pair corresponding to $m = n = 0$ ((TE_{00} and TM_{00})). This so-called **fundamental mode** represents a plane wave propagating in the scan direction when the array is operating in transmitting mode.

Let (θ, φ) be the scanning angle of the UC in spherical coordinates; the phase difference between adjacent sub-arrays situated along the x and y directions are

$$\begin{cases} \psi_x = k_0 W \sin\theta \cos\varphi \\ \psi_y = k_0 W \sin\theta \sin\varphi \end{cases} \tag{3.12}$$

It can be shown that the normalized modal electric field vectors, corresponding to the resulting Floquet modes, can be represented in the x and y directions through the following Floquet modal expansions ([30], [9]):

$$\begin{cases} \mathbf{e}_{\mathrm{TE}} = \dfrac{4\pi^2}{ab} \displaystyle\sum_{m=0}^{+\infty}\sum_{n=0}^{+\infty} \dfrac{k_{y,n}\,\hat{\mathbf{x}} - k_{x,m}\,\hat{\mathbf{y}}}{W\sqrt{k_{x,m}^2 + k_{y,n}^2}} \exp\left(-\mathrm{j}k_{x,m}x\right) \exp\left(-\mathrm{j}k_{y,n}y\right) \\ \mathbf{e}_{\mathrm{TM}} = \dfrac{4\pi^2}{ab} \displaystyle\sum_{m=0}^{+\infty}\sum_{n=0}^{+\infty} \dfrac{k_{y,n}\,\hat{\mathbf{x}} + k_{x,m}\,\hat{\mathbf{y}}}{W\sqrt{k_{x,m}^2 + k_{y,n}^2}} \exp\left(-\mathrm{j}k_{x,m}x\right) \exp\left(-\mathrm{j}k_{y,n}y\right) \end{cases} \tag{3.13}$$

where $k_{x,m}$ and $k_{y,n}$ are the Floquet wavenumbers:

$$\begin{cases} k_{x,m} = k_0 \sin\theta\cos\varphi + \dfrac{2\pi m}{a} \equiv k_0 \sin\theta\cos\varphi + \dfrac{2\pi m}{W}, & m \in \mathbb{N} \\ k_{y,n} = k_0 \sin\theta\sin\varphi + \dfrac{2\pi n}{b} \equiv k_0 \sin\theta\sin\varphi + \dfrac{2\pi n}{W}, & n \in \mathbb{N} \end{cases} \tag{3.14}$$

When dealing with a multi-layer structure, the modal wavenumber of each layer can be calculated in this way for both TE and TM modes:

$$k_{z,mn}^{\mathrm{layer}} = \sqrt{k_0^2 \epsilon_{\mathrm{layer}} - k_{x,m}^2 - k_{y,n}^2}, \tag{3.15}$$

where $\epsilon_{\text{layer}} \in \mathbb{C}$ is the effective permittivity of the layer under interest, calculated according to Equation (3.11).

Once $k_{z,mn}^{\text{layer}}$ is known, the modal admittances are defined as follows:

$$Y_{mn,\text{TE}}^{\text{layer}} = \frac{k_{z,mn}^{\text{layer}}}{\mu_0 \omega_0} \tag{3.16a}$$

$$Y_{mn,\text{TM}}^{\text{layer}} = \frac{\epsilon_0 \epsilon_{\text{layer}} \omega_0}{k_{z,mn}^{\text{layer}}} \tag{3.16b}$$

All the MATLAB® functions used to calculate these parameters are reported in Appendix B.2.1 (modal wavenumbers) and B.2.2 (modal admittance).

At this point, everything is available to develop an equivalent circuit model that allows for finding in a simple way the scattering parameters of each unit cell.

Before doing this, however, it is important to point out the following fact: although the number of propagating modes depends on frequency, scan angle and cell size, for an ideal array there ideally exists only **one propagating mode** pair (TE and TM) corresponding to $m = n = 0$. This so-called **fundamental mode** represents a plane wave propagating in the scan direction $(\theta, \varphi) \equiv (\theta_{00}, \varphi_{00})$ when the TA is operating in transmitting mode.

Therefore, from now on the subscript mn will be neglected in the symbols of all the physical quantities for the sake of simplicity.

3.1.3 Transmission-line Equivalent Circuit

Let us consider a UC composed by N dielectric layers with thickness $T^{(i)}$ and effective permittivity $\epsilon_{\text{eff}}^{(i)}$, $i = 1, \dots, N$.

Let us assume, without significant losses in terms of accuracy, that each section is characterized by a perfectly **isotropic behavior**: under this hypothesis, the sequence of layers can be represented as the cascade of N uniform transmission lines along the z direction.

The reference wavenumber and the reference impedance of the circuit, namely k_z^{ref} and $Z_{\text{TE|TM}}^{\text{ref}}$, are obtained applying Equations (3.15) and (3.16) to an air substrate, i.e., imposing $\epsilon_{\text{layer}} = 1$:

$$k_z^{\text{ref}} = \sqrt{k_0^2 - k_x^2 - k_y^2} \tag{3.17}$$

and

$$Z_{\text{TE|TM}}^{\text{ref}} = \left(Y_{\text{TE|TM}}^{\text{ref}}\right)^{-1} = \begin{cases} \dfrac{k_z^{\text{ref}}}{\mu_0 \omega_0} & \text{TE mode} \\ \dfrac{\epsilon_0 \omega_0}{k_z^{\text{ref}}} & \text{TM mode} \end{cases} \tag{3.18}$$

On the other hand, the propagation constant $k_z^{(i)}$ and the electrical lenght $\Theta^{(i)}$ for each layer $i = 1, \dots, N$ can be calculated as follows:

$$k_z^{(i)} = \sqrt{k_0^2 \epsilon_{\text{eff}}^{(i)} - k_x^2 - k_y^2} \implies \Theta^{(i)} = k_z^{(i)} T^{(i)}, \quad i = 1, \dots N, \tag{3.19}$$

Similarly, the characteristic impedance $Z_{\text{TE|TM}}^{\infty,(i)}$ is given by

$$Z_{\text{TE|TM}}^{\infty,(i)} = \left(Y_{\text{TE|TM}}^{(i)}\right)^{-1} = \begin{cases} \dfrac{k_z^{(i)}}{\mu_0 \omega_0} & \text{TE mode} \\ \dfrac{\epsilon_0 \epsilon_{\text{eff}}^{(i)} \omega_0}{k_z^{(i)}} & \text{TM mode} \end{cases} \tag{3.20}$$

Table 3.1: Material parameters.

	Resin	Glass, 4.3 mm	Glass, 6.08 mm
ϵ_{r2}	2.67	6.4	6.5
$\tan\delta_2$	0.0168	0.027	0.025

The circuit can now be easily solved with elementary TL theory; in particular, as pointed out in Appendix B.3, it is convenient to exploit ABCD formalism because it is particularly suitable for the analysis of cascaded networks.

Coherently with Equation (B.7), the **ABCD** matrix of the overall circuit, respectively for the TE and TM modes, can be calculated as follows:

$$\mathbf{ABCD}_{\mathrm{TE|TM}} = \prod_{i=1}^{N} \mathbf{ABCD}^{(i)}_{\mathrm{TE|TM}}, \tag{3.21}$$

Exploiting the conversion formulas listed in Appendix B.3.2, the S parameters are eventually obtained:

$$S_{11,\mathrm{TE|TM}} = \frac{A_{\mathrm{TE|TM}} Z^{\mathrm{ref}}_{\mathrm{TE|TM}} + B_{\mathrm{TE|TM}} - C_{\mathrm{TE|TM}} \left(Z^{\mathrm{ref}}_{\mathrm{TE|TM}}\right)^2 - D_{\mathrm{TE|TM}} Z^{\mathrm{ref}}_{\mathrm{TE|TM}}}{A_{\mathrm{TE|TM}} Z^{\mathrm{ref}}_{\mathrm{TE|TM}} + B_{\mathrm{TE|TM}} + C_{\mathrm{TE|TM}} \left(Z^{\mathrm{ref}}_{\mathrm{TE|TM}}\right)^2 + D_{\mathrm{TE|TM}} Z^{\mathrm{ref}}_{\mathrm{TE|TM}}} \tag{3.22a}$$

$$S_{21,\mathrm{TE|TM}} = \frac{2 Z^{\mathrm{ref}}_{\mathrm{TE|TM}}}{A_{\mathrm{TE|TM}} Z^{\mathrm{ref}}_{\mathrm{TE|TM}} + B_{\mathrm{TE|TM}} + C_{\mathrm{TE|TM}} \left(Z^{\mathrm{ref}}_{\mathrm{TE|TM}}\right)^2 + D_{\mathrm{TE|TM}} Z^{\mathrm{ref}}_{\mathrm{TE|TM}}} \tag{3.22b}$$

Contrary to the design flows presented in other works, e.g., [30], this ABCD-based approach has many advantages:

- It ensures modularity and scalability thanks to the exploitation of the ABCD formalism: in fact, adding a new layer to the structure simply implies performing an additional matrix multiplication before the conversion from ABCD to S parameters, which is not true altogether when using the "standard" approach based on scattering formalism;
- It can be used to analyze arbitrarily complex structures, because the impact of cell geometry and employed materials on the effective permittivity and TL parameters is modeled in a simple and elegant way by the MG formulas;
- It is not limited to the analysis of Transmitarrays: in fact, with few modifications, it can be easily adapted to the study of many different categories of smart skins, e.g, Reflectarrays and tapered structures.

3.1.4 Choice of the UC Materials

In view of the application of interest, the selected materials are now presented and analyzed; their parameters are listed in a compact form in Table 3.1.

3.1.4.1 PETG Resin Layers

All the analyzed configurations are realized with one or more layers of a commercial resin, namely **transparent PETG**, characterized by relative dielectric constant $\epsilon_r = 2.67$ and tangent loss $\tan\delta = 0.0168$.

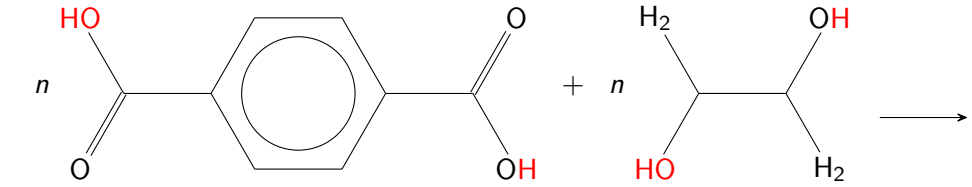
PETG (acronym for Polyethylene Terephthalate Glycol, formula $(C_{10}H_8O_4)n$), is a thermoplastic copolyester known for its durability, transparency and ease of processing [26].

It is widely used in packaging, medical applications, signage, and 3D printing due to its impact resistance and flexibility compared to standard PET.

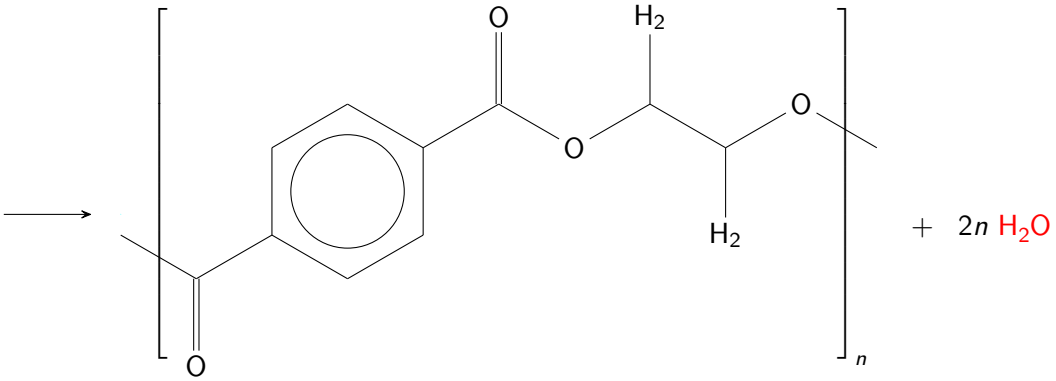
From the chemical point of view, PETG is a product of polycondensation of two monomers:

1. Purified Terephthalic Acid (a.k.a. PTA, $C_8H_6O_4$);
2. Ethylene Glycol (a.k.a. EG, $C_2H_6O_2$).

The polymerization reaction for a single molecule is shown here:

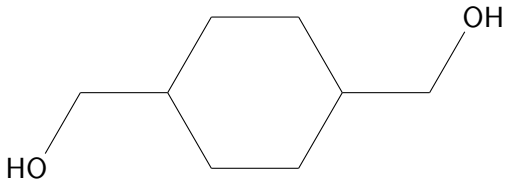


Purified Terephthalic Acid (PTA) Ethylene Glycol (EG)



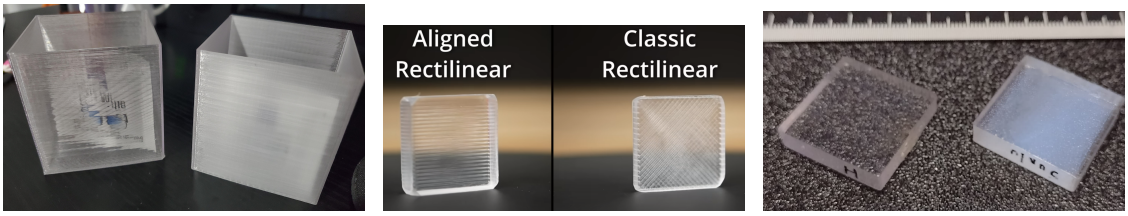
Polyethylene Terephthalate Glycol (PETG)

The addition of 1,4-Cyclohexanedimethanol (a.k.a. CHDM, $C_8H_6O_2$) disrupts the crystalline structure of the material, making it more flexible and impact-resistant while maintaining excellent optical clarity. This modification allows PETG to be easier to thermoform, extrude, and mold, making it ideal for complex designs and applications:



This material has been selected in view of fabricating a 3D-printed prototype of the Transmitarray for further validation of the model. The advantages and drawbacks of such a choice are listed below.

1. The best suited solution for the TA fabrication seems to be the use of **Additive Manufacturing** (AM) techniques, since technological limitations make the use of conventional machining approaches impracticable with high-frequency antennas, especially when small variable-size holes are present.



(a) Dry (left) vs wet (right) **(b)** Different printing techniques **(c)** Polished (left) vs raw (right)

Figure 3.2: Comparison between different printing techniques for clear PETG.

2. As far as costs are concerned, PETG is a very **cheap material** that can be easily bought online at no more than 25 €/kg. Its affordability allows for experimentation with different configurations and printing setup without any economic concerns.

3. The materials available for 3D-printing techniques are typically based on UV crosslinkable polymers, usually not optimized to provide functional **dielectric properties** [32].

   In particular, PETG is characterized by a quite small permittivity, which implies an increase of the total thickness since the effective wavelength depends on the dielectric constant and the filling factor f_1 (see Equation (3.5)).

   Furthermore, this material has non-negligible losses at the operating frequency, which affects the efficiency of the Transmitarray and could reduce its bandwidth.

4. The **limited resolution** of currently available 3D-printers ($> 10\,\mu m$) does not allow fabricating small features as those required at high frequencies. Furthermore, the fabrication process strongly limits the transparency of the material.

Since transparency is a key feature of the structures of interest, careful tuning of printing settings is required and, in most cases, it is necessary to resort to post-processing techniques. Here are some key rules that should be followed to enhance results:

- **Dry the filament**: PETG is a highly hygroscopic material, meaning it easily absorbs humidity from the air; as a consequence, the presence of moisture in the resin can cause bubbles and voids during the printing process, reducing transparency. Drying the filament for at least eight hours before printing improves clarity (see Figure 3.2a).

  Once printed, however, PETG is fairly resistant to water and does not degrade simply due to humidity. That said, long-term exposure to outdoor conditions - especially UV light - can cause the plastic to become brittle and degrade over time, which must be kept into account since the final aim is to mount the designed SES on a window pane.

- **Chose carefully infill geometry**: an aligned rectilinear infill pattern is better to remove internal voids, together with no top and bottom surfaces to reduce different print geometries that could create voids in the print (see Figure 3.2b).

- **Reduce speed**: a low printing speed ($\leq 20\,mm/s$) allows for the filament to flow more smoothly, reducing gaps and inconsistencies that can scatter light in an undesirable way.

- **Increase flow ratio**: a high extrusion multiplier ($> 100\%$) helps fill gaps and ensures complete layer adhesion, enhancing transparency.

- **Turn off cooling fans**: allowing the filament to cool naturally prevents uneven shrinkage, which can distort transparency.

- **Polish the surface**: post-processing techniques like acetone smoothing, dry sanding or clear coating can further improve transparency (see Figure 3.2c). It must be pointed out, however, that the effectiveness of this techniques may be reduced by the presence of very small-sized apertures.

In conclusion, PETG may be a suitable choice for the realization of a 3D-printed prototype to assess the performance of the designed structures - provided the printer settings are carefully tuned and the surface is properly post-processed - but its hygroscopic nature, susceptibility to UV degradation, and potential long-term wear may make it less ideal for mass production, especially in demanding environments.

3.1.4.2 Glass Layers

In some of the analyzed UC configurations, the presence of the window is taken into account and modeled as a glass layer, characterized by relative dielectric constant and tangent loss subject to small changes depending on the glass thickness T_g; in particular, the second and third row of Table 3.1 summarize the values of ϵ_{er} and $\tan\delta$ for two values of T_g, namely $T_g = 4.3\,\text{mm}$ and $T_g = 6.08\,\text{mm}$.

Since the aim of this work is primarily to build a simple and reliable model for the design and optimization of Smart Electromagnetic Skins, all the calculations and simulations are performed choosing the minimum glass thickness $T_g = 4.3\,\text{mm}$, as this choice slightly reduces the total volume of the structures and thus decreases the computational cost.

3.1.5 Choice of the UC Periodicity

Before analyzing in details the structures used for the development and validation of the model, let us fix the UC periodicity along x and y to

$$W = 0.3\lambda_0 = 3\,\text{mm}, \tag{3.23}$$

similarly to the choice made in [30] for the unit cell of a RA.

It is important to point out that the choice of the UC periodicity depends on several factors: a good starting point is ensuring a balance between phase coverage, transmission efficiency, and fabrication constraints.

Ideally, W should be a fraction of the operating wavelength, typically around $0.3\lambda_0 \div 0.5\lambda_0$, to maintain good electromagnetic performance while avoiding unwanted grating lobes. However, when dealing with high-frequency applications or advanced phase-control techniques, finer periodicities might be necessary for smoother phase gradients.

Let us spend some more words on the advantages and disadvantages of *increasing* or *decreasing* the UC periodicity:

Increasing the Periodicity. Larger unit cells can sometimes improve transmission efficiency and reduce fabrication complexity, especially if the design incorporates complex geometries.

On the other hand, f the periodicity exceeds $0.5\lambda_0$, there is the risk of introducing grating lobes, i.e., undesirable diffraction effects that can degrade beam focusing and efficiency. What is more, with larger periodicity, phase resolution may be reduced, making it harder to achieve smooth phase gradients for beam shaping.

Decreasing the Periodicity. Finer periodicity enhances phase control, allowing for better beam steering and sharper radiation patterns. It also minimizes unwanted scattering and improves the uniformity of wavefront transformation.

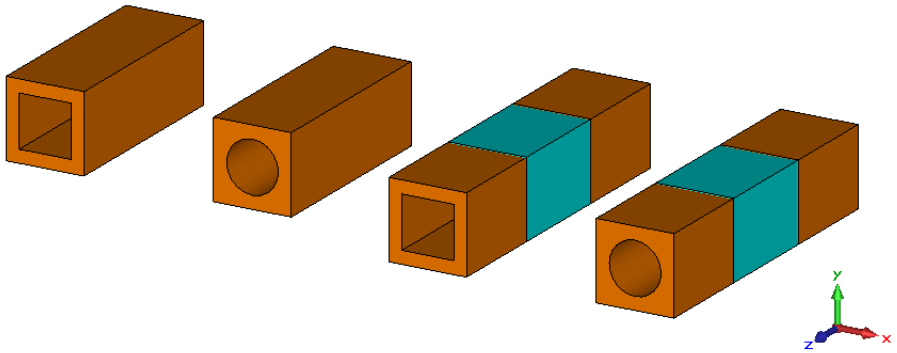


Figure 3.3: Overview of the UCs of interest; from left to right: UC-1, UC-2, UC-3, UC-4.

Fabrication, however, becomes more challenging, especially at high frequencies, since precise structuring of small elements can be difficult. Additionally, unit-cell coupling effects may become more significant, affecting overall performance.

Finally, while smoother phase gradients can improve directivity, excessive miniaturization may introduce losses due to material constraints and unwanted resonances.

3.1.6 Unit cells of Interest

Using the model and the materials introduced in the previous paragraphs, it is now possible to study some specific unit cell configurations and analyze how their scattering coefficients change by varying the geometrical parameters.

To develop and validate the theoretical model, **four UC structures** are initially considered.

1. **UC-1** (single-layer structure): single resin layer with a central square hole, characterized by size d and thickness T;
2. **UC-2** (single-layer structure): single resin layer with a central circular hole, characterized by diameter d and thickness T;
3. **UC-3** (three-layer structure): two external resin layers with a central square hole, characterized by size d and thickness $T/2$; to simulate the presence of the window, an internal glass layer with no holes and thickness T_g is added between the two dielectric sections;
4. **UC-4** (three-layer structure): two external resin layers with a central circular hole, characterized by diameter d and thickness $T/2$; to simulate the presence of the window, an internal glass layer with no holes and thickness T_g is added between the two dielectric sections.

From a practical standpoint, UC-3 and UC-4 are ideally obtained by "cutting" UC-1 and UC-2 in half and inserting the glass layer between the two resulting sections.

In all configurations, both d and T can vary in a pre-defined range in order to meet the optimization constraints.

A 3D sketch of the four unit cells, coherent with the coordinate system introduced at the beginning of Section 3.1, is shown in Figure 3.3.

On the other hand, Figures 3.4 and 3.5, represent the equivalent TL circuits of the single-layer and three-layer structures, respectively. The schemes must be interpreted as follows:

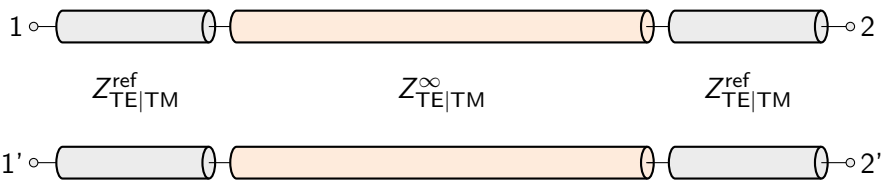


Figure 3.4: Single-layer UC equivalent model, holding for UC-1 and UC-2.

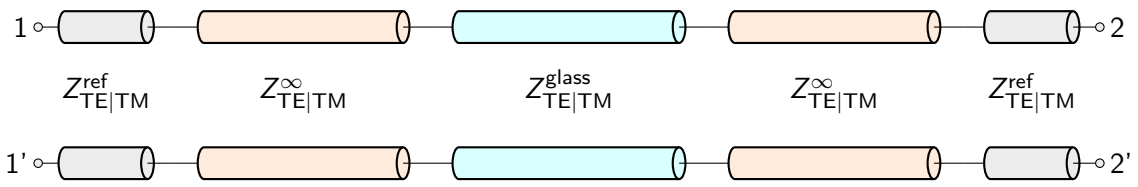


Figure 3.5: Three-layer UC equivalent model, holding for UC-3 and UC-4.

- Orange TL sections: resin layers with characteristic impedance $Z^{\infty}_{\text{TE|TM}}$, obtained with Equation (3.20) and affected by the hole size d;
- Light-blue TL sections: glass layers with characteristic impedance $Z^{\text{glass}}_{\text{TE|TM}}$, obtained again with Equation (3.20);
- Gray TL sections: free-space layers accounting for the presence of the Floquet ports; their presence does not affect the magnitude of the reflection and transmission coefficient, but it introduces a phase shift that must be taken into account when designing the entire TA to ensure a correct phase compensation, as pointed out in Chapter 4.

To begin with, let us set the cell height to

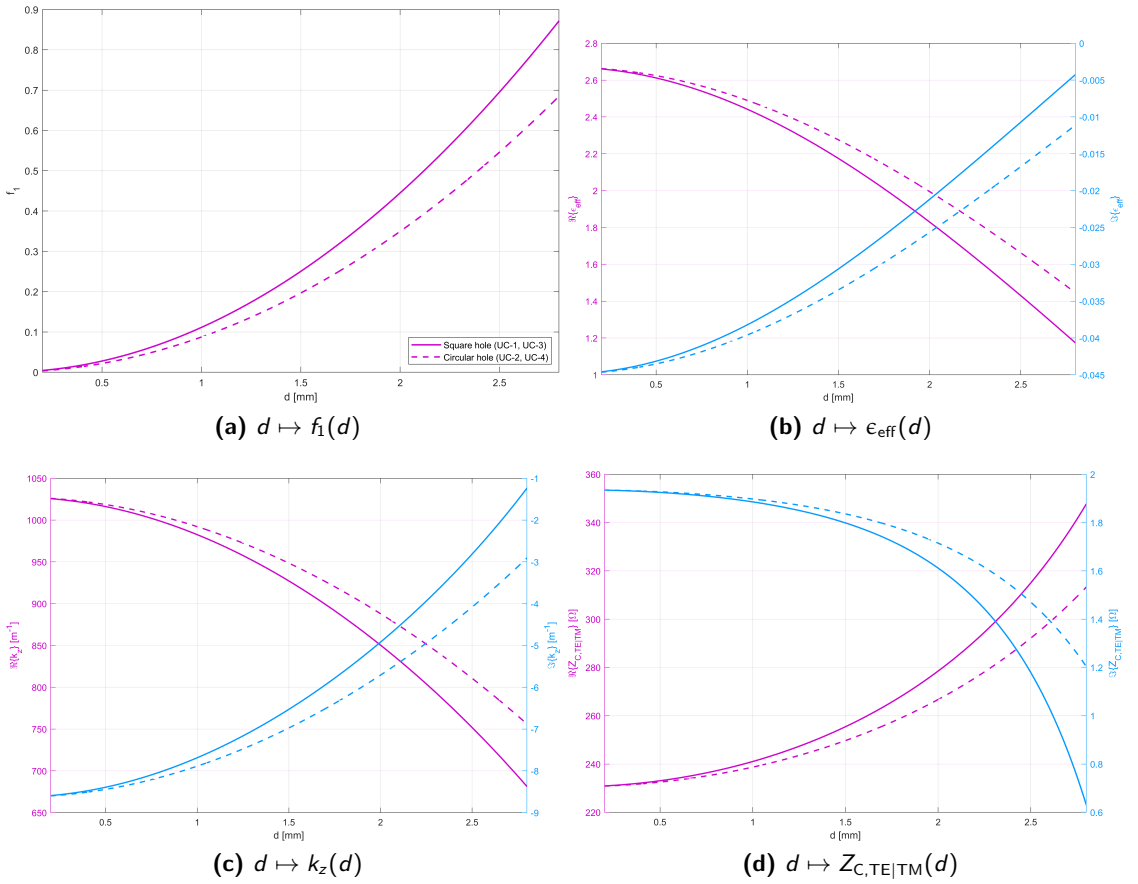
$$T = 0.8\lambda_0 = 8\,\text{mm}, \tag{3.24}$$

and let us assess the behavior of the quantities reported in Figure 3.6 as the hole size d changes in the range $0.2 \div 2.8\,\text{mm}$.

1. **Volume fraction**: as Figure 3.6a shows, $d \mapsto f_1(d)$ is a monotonically increasing function of the hole size. It is also possible to notice that, once fixed d, the UCs with circular hole (UC-3, UC-4) have a smaller volume fraction with respect to those with square hole (UC-1, UC-2), which can be easily understood with basic geometric considerations.
2. **Resin section parameters**:$d \mapsto \epsilon_{\text{eff}}(d)$ and $d \mapsto k_z(d)$ exhibit a similar behavior as function of the hole size:
   - $\Re\{k_z\}$ and $\Re\{Z_{\text{C}}\}$ are monotonically *decreasing* with d and the curves of UC-1 and UC-2 are lower than those of UC-3 and UC-4;
   - $\Im\{k_z\}$ and $\Im\{Z_{\text{C}}\}$ are both negative, they monotonically *increase* with d and the curves of UC-1 and UC-2 are characterized by *less negative* values than UC-3 and UC-4, meaning that losses have a higher impact when dealing with circular unit cells.

   On the other hand, $d \mapsto \Re\{Z_{\text{C,TE|TM}}(d)\}$ and $d \mapsto \Im\{Z_{\text{C,TE|TM}}(d)\}$ are, respectively, monotonically increasing and decreasing functions of d.

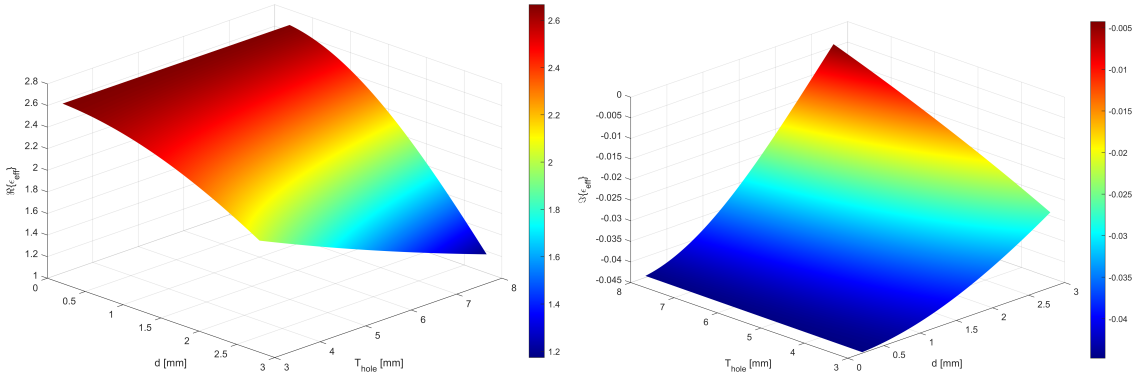
Let us point out the following facts:



(a) $d \mapsto f_1(d)$ **(b)** $d \mapsto \epsilon_{\text{eff}}(d)$

(c) $d \mapsto k_z(d)$ **(d)** $d \mapsto Z_{\text{C,TE|TM}}(d)$

Figure 3.6: Real (purple curves) and imaginary (light-blue curves) part of volume fraction (a), effective permittivity (b), modal wavenumber (c) and characteristic impedance (d) of the UC as function of the hole size. Continuous lines: square hole (UC-1, UC-3); dashed lines: circular hole (UC-2, UC-4).



(a) $(d, T_{\text{hole}}) \mapsto \Re\{\epsilon_{\text{eff}}\}(d, T_{\text{hole}})$ **(b)** $(d, T_{\text{hole}}) \mapsto \Im\{\epsilon_{\text{eff}}\}(d, T_{\text{hole}})$

Figure 3.7: Behavior of the effective permittivity if both d and T_{hole} change.

- The trend observed for ϵ_{eff}, k_z and Z_C can be easily justified by considering that, as d increases, the fraction of air composing the cell grows quadratically, causing these quantities to approach the characteristic values of free space:

$$\lim_{d/W\to 1} \epsilon_{\text{eff}}(d) = 1, \tag{3.25a}$$

$$\lim_{d/W\to 1} k_z(d) = k_0 = \frac{2\pi}{\lambda_0} \approx 628\,\text{m}^{-1}, \tag{3.25b}$$

$$\lim_{d/W\to 1} Z_C(d) = Z_0 = 120\pi\,\Omega. \tag{3.25c}$$

  Conversely, if d decreases, the cell increasingly behaves like a homogeneous block of resin:

$$\lim_{d/W\to 0} \epsilon_{\text{eff}}(d) = \epsilon_2 = \epsilon_{r2}\,(1 - \text{j}\tan\delta_2) \approx 2.67 - \text{j}0.045, \tag{3.26a}$$

$$\lim_{d/W\to 0} k_z(d) = k_{g2} = \frac{2\pi}{\lambda_{g2}} = \frac{2\pi\sqrt{\epsilon_2}}{\lambda_0} \approx (1027.43 - \text{j}8.63)\,\text{m}^{-1}. \tag{3.26b}$$

$$\lim_{d/W\to 0} Z_C(d) = \frac{Z_0}{\sqrt{\epsilon_2}} \approx (230.69 + \text{j}1.94)\,\Omega. \tag{3.26c}$$

- All the quantities analyzed above only depend on d or, to be more precise, on the ratio between the hole aperture ($\propto d^2$) and the cell periodicity ($\propto W^2$).

  This happens because the hole height coincides with the total cell height, namely $T_{\text{hole}} \equiv T$. For a more complete analysis, it is interesting to study how ϵ_{eff} changes when $T_{\text{hole}} \neq T$; the results of this two-parameter analysis are shown in Figure 3.7 and the considerations are similar to those made for the single-parameter case.

- In the cells with circular hole (UC-3, UC-4), the curves change more slowly with d, which results in worse performance because, even as the air fraction increases, losses remain non-negligible.

The MATLAB® routines to calculate the S parameters of the four proposed configurations, developed with the model presented in Section 3.1.3, are reported in Appendix B.4.1 (UC-1), B.4.2 (UC-2), B.4.3 (UC-3) and B.4.4 (UC-4).

The theoretical results obtained for all the structures (TE and TM modes) sweeping the values of θ and φ are shown in Figures 3.8-3.15.

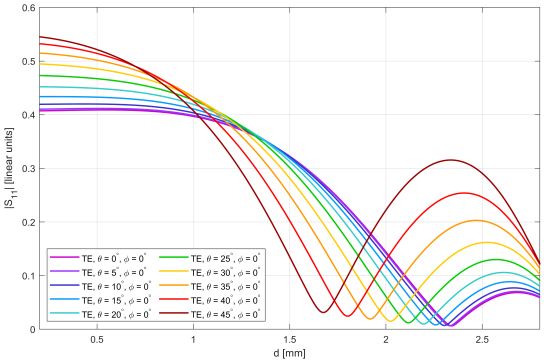
Observing these plots, the following considerations must be highlighted:

- The analysis is performed sweeping θ in the range $0 \div 45^\circ$ and φ in the range $0 \div 90^\circ$.

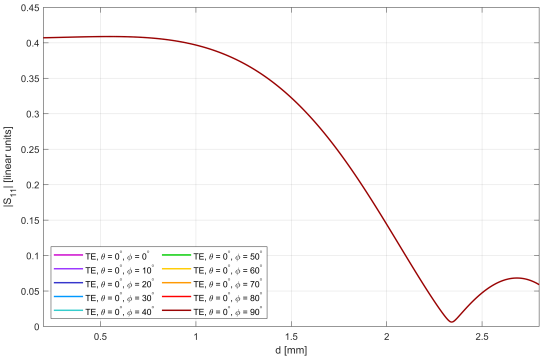
  In particular, as the analysis in Chapter 4 will point out, the *inclination angle* θ corresponds to the *scan angle* θ_f of the feed with respect to the Transmitarray; since θ_f is rarely bigger than 40°, it is not physically meaningful to analyze the behavior of the structure for $\theta > 45^\circ$.

  On the other hand, the *aximuth angle* φ can vary over a wider range because it is not subject to any geometrical constraint.

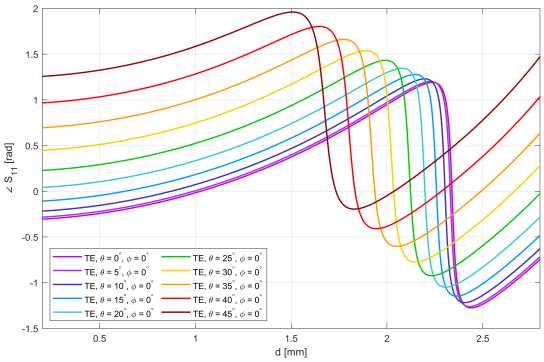
- It is immediately noticeable that, while the S parameters of the UC strongly depend on θ, they are completely transparent to the value of φ: this happens because the structure is symmetric with respect to geometrical rotations in the xy plane, meaning that its performance to be φ-invariant.



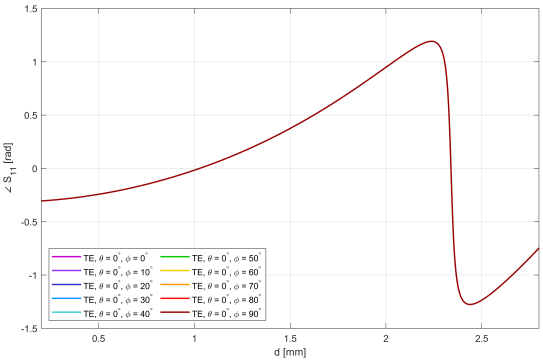
(a) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of θ)



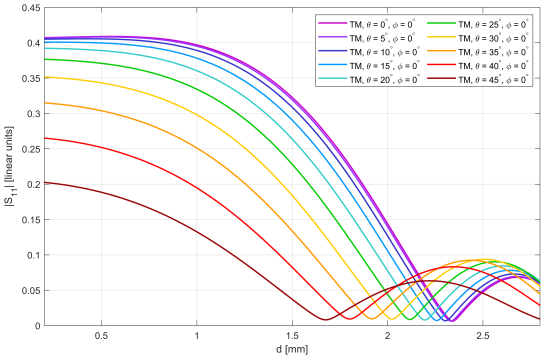
(b) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of φ)



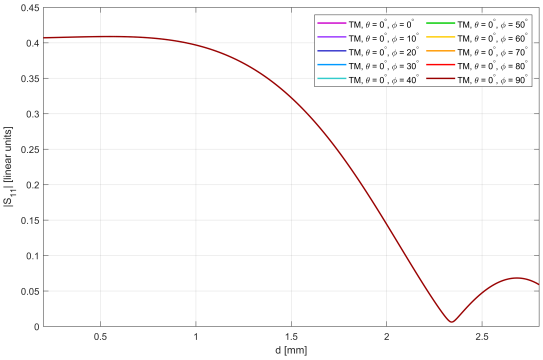
(c) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of θ)



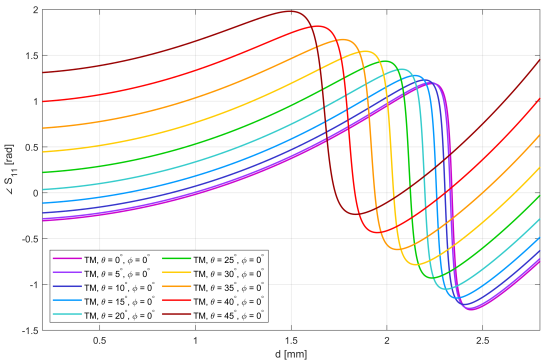
(d) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of φ)



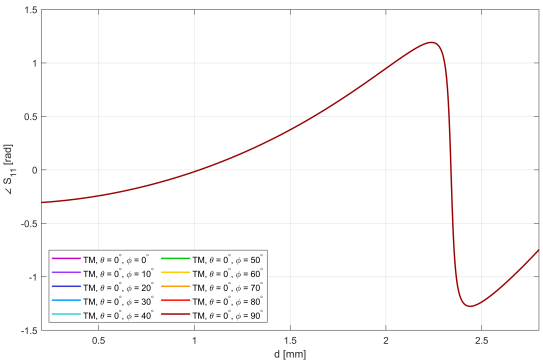
(e) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of θ)



(f) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of φ)

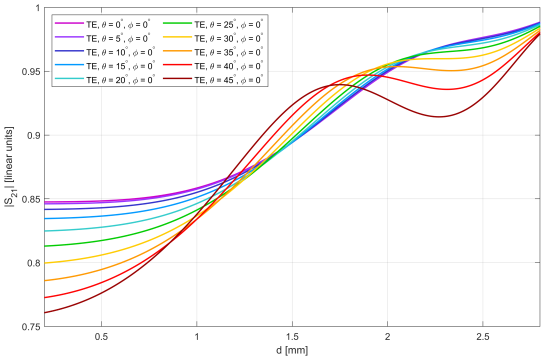


(g) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of θ)

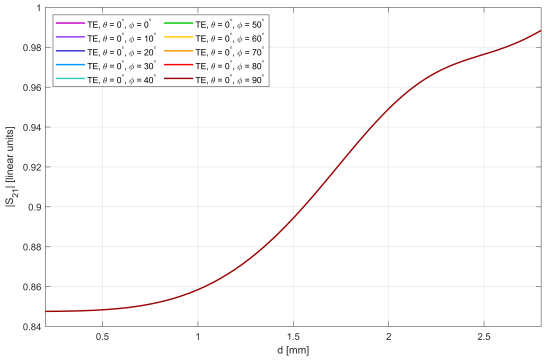


(h) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of φ)

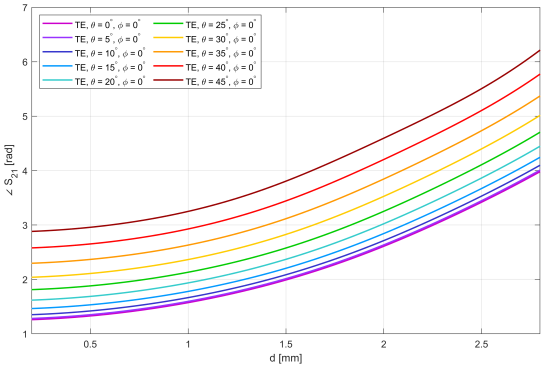
Figure 3.8: Reflection coefficient vs hole size d (UC-1, $T = 0.8\lambda_0$, TE & TM modes).



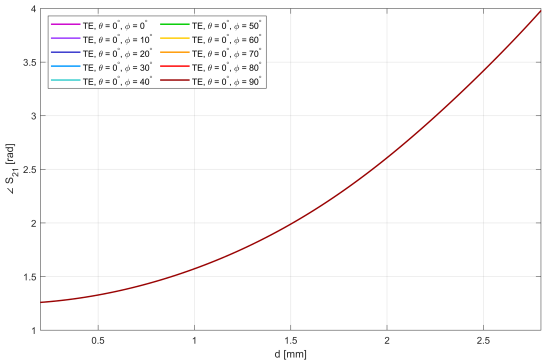
(a) $d \mapsto |S_{21}(d)|$ (TE mode, sweep of θ)



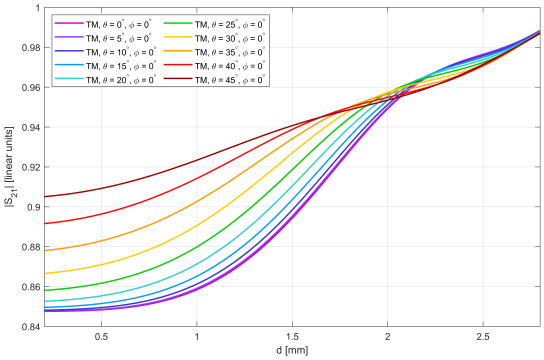
(b) $d \mapsto |S_{21}(d)|$ (TE mode, sweep of φ)



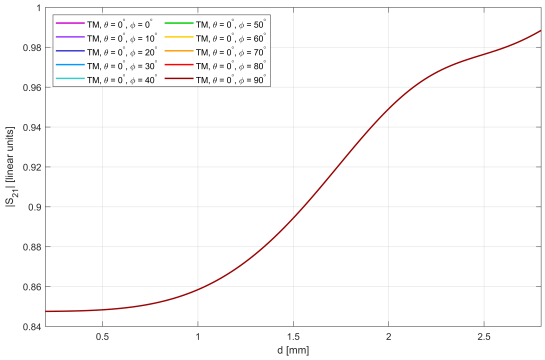
(c) $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of θ)



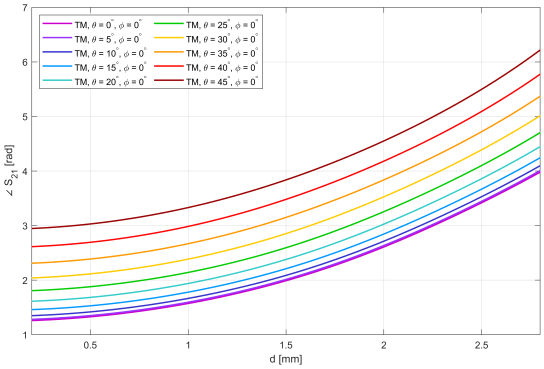
(d) $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of φ)



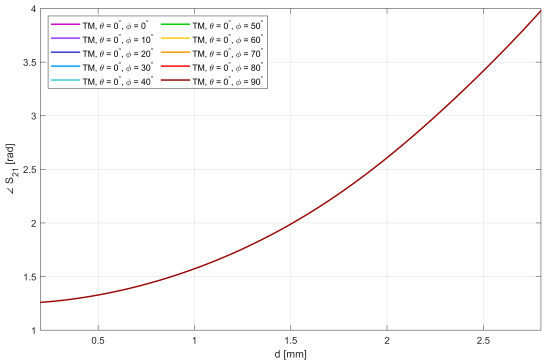
(e) $d \mapsto |S_{21}(d)|$ (TM mode, sweep of θ)



(f) $d \mapsto |S_{21}(d)|$ (TM mode, sweep of φ)

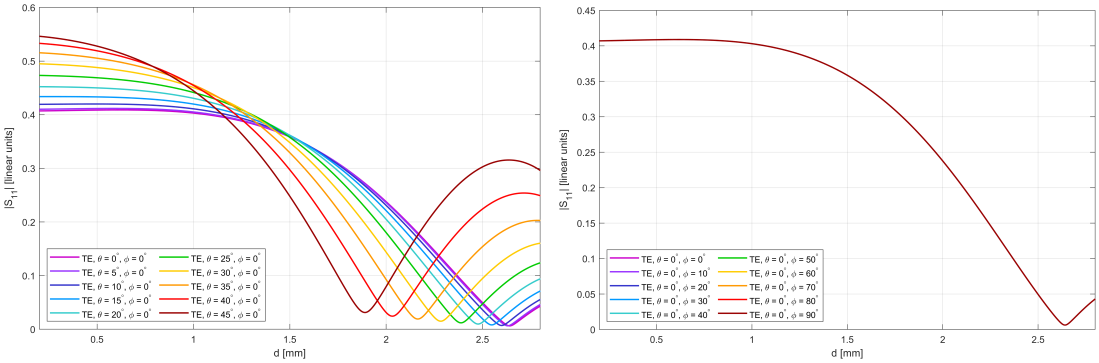


(g) $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of θ)



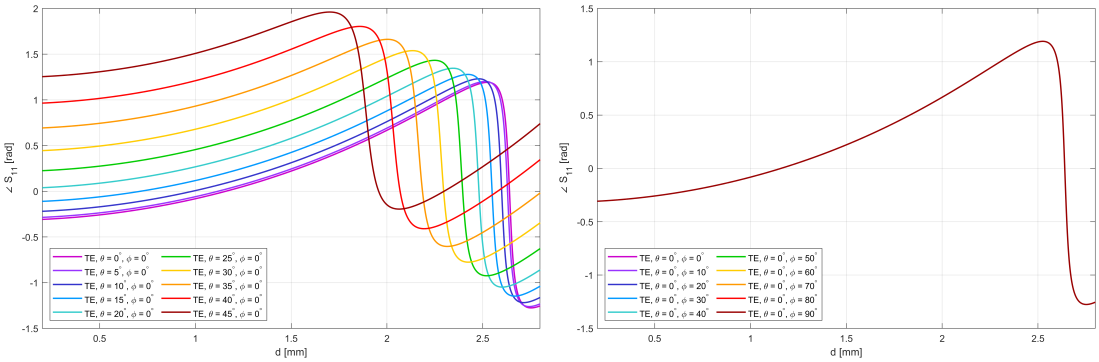
(h) $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of φ)

Figure 3.9: Transmission coefficient vs hole size d (UC-1, $T = 0.8\lambda_0$, TE & TM modes).



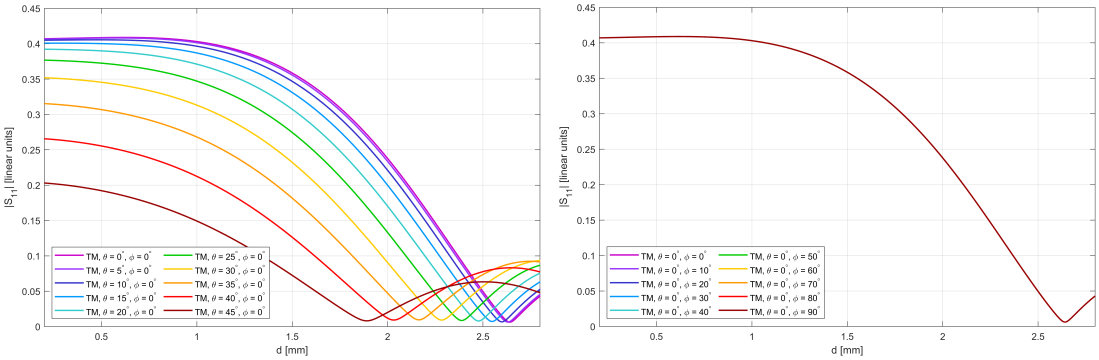
(a) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of θ)

(b) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of φ)



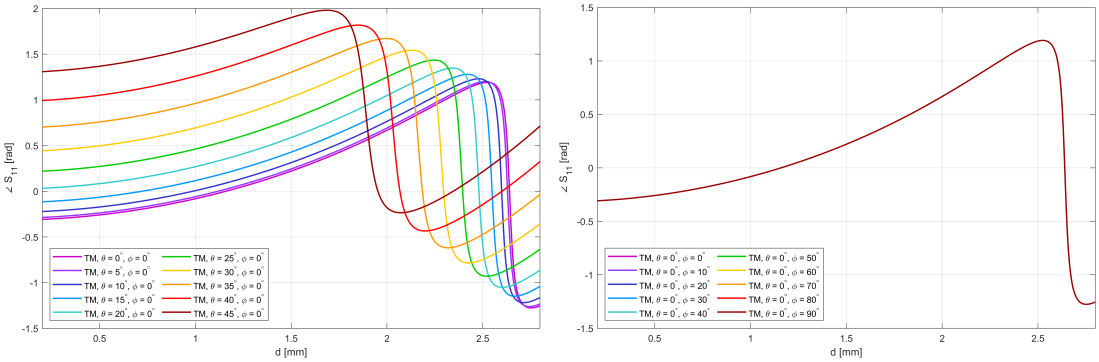
(c) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of θ)

(d) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of φ)



(e) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of θ)

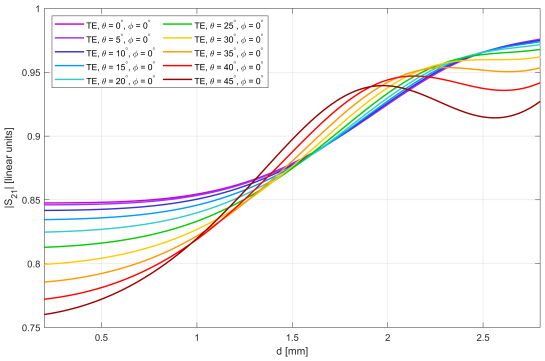
(f) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of φ)



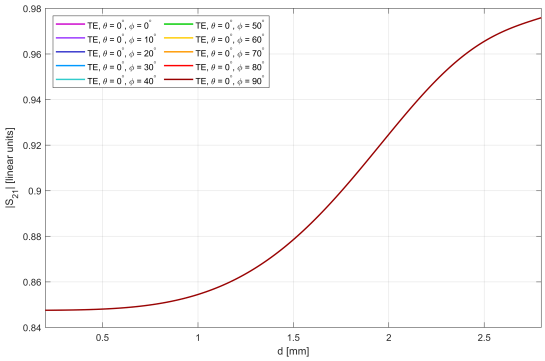
(g) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of θ)

(h) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of φ)

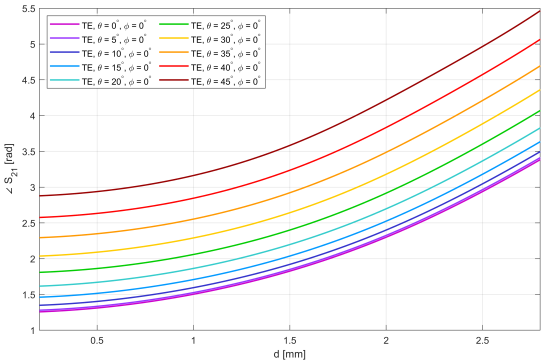
Figure 3.10: Reflection coefficient vs hole size d (UC-2, $T = 0.8\lambda_0$, TE & TM modes).



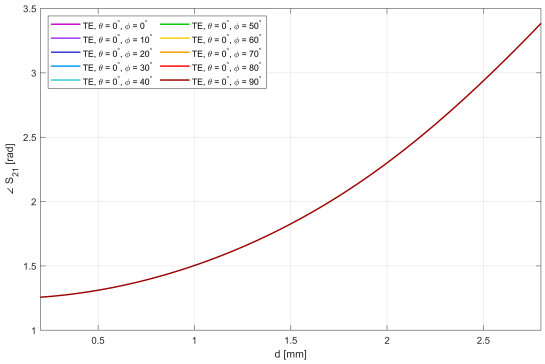
(a) $d \mapsto |S_{21}(d)|$ (TE mode, sweep of θ)



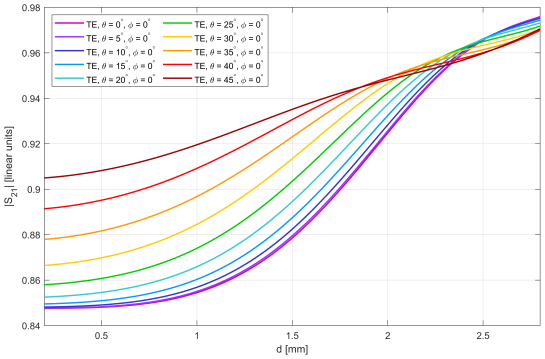
(b) $d \mapsto |S_{21}(d)|$ (TE mode, sweep of φ)



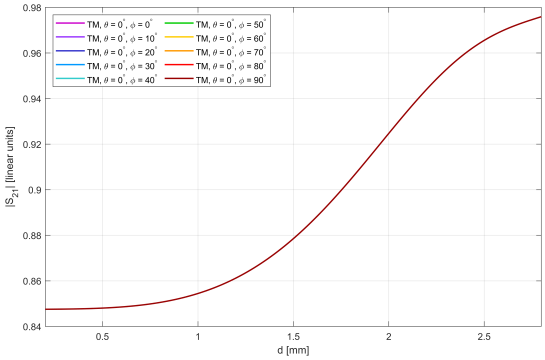
(c) $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of θ)



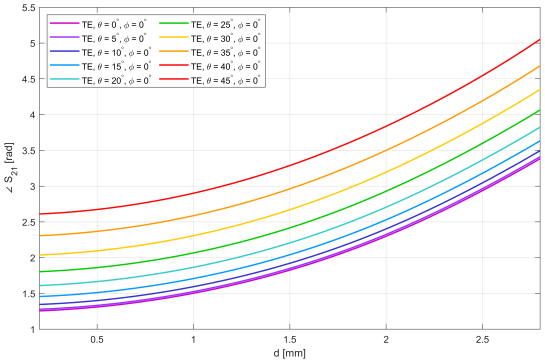
(d) $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of φ)



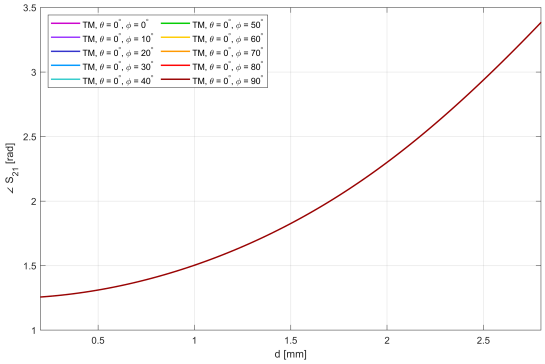
(e) $d \mapsto |S_{21}(d)|$ (TM mode, sweep of θ)



(f) $d \mapsto |S_{21}(d)|$ (TM mode, sweep of φ)

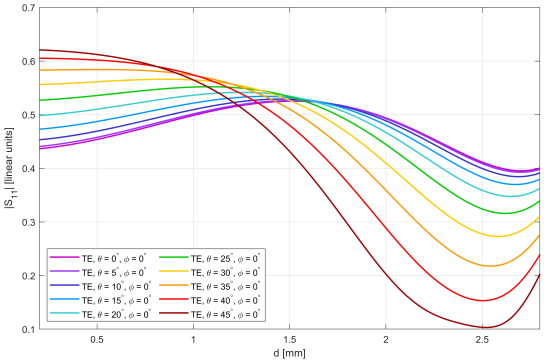


(g) $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of θ)

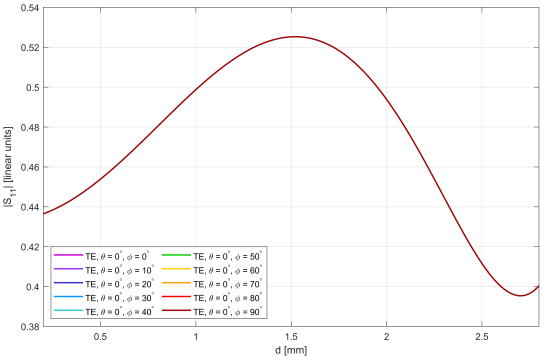


(h) $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of φ)

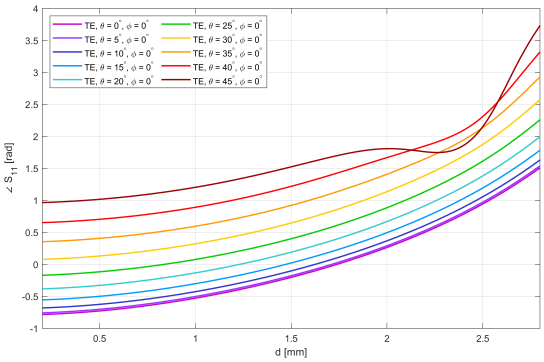
Figure 3.11: Transmission coefficient vs hole size d (UC-2, $T = 0.8\lambda_0$, TE & TM modes).



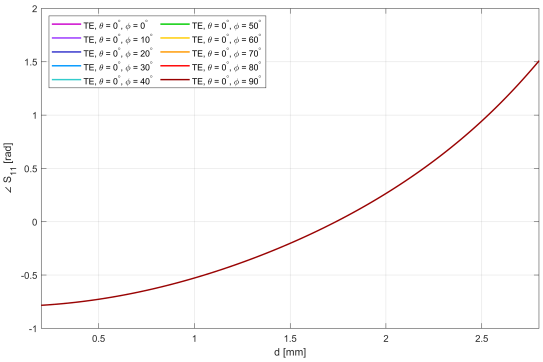
(a) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of θ)



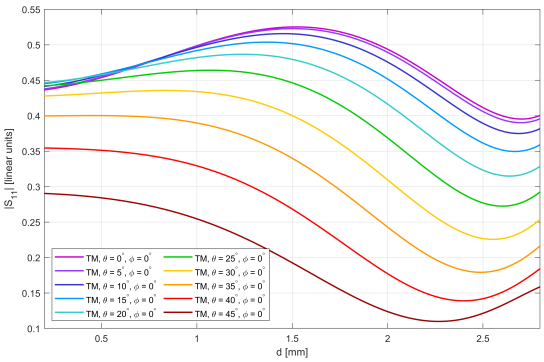
(b) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of φ)



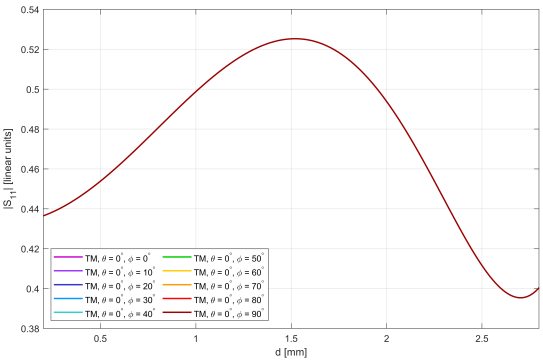
(c) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of θ)



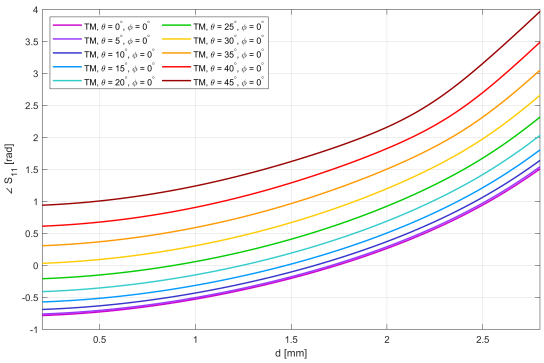
(d) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of φ)



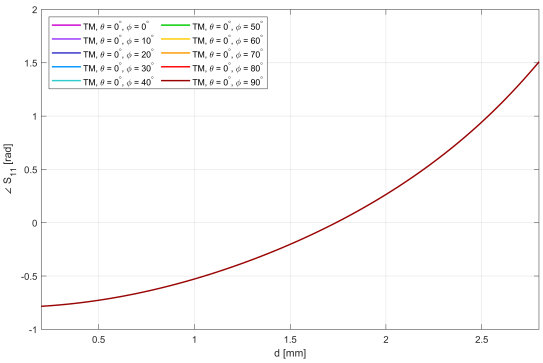
(e) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of θ)



(f) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of φ)

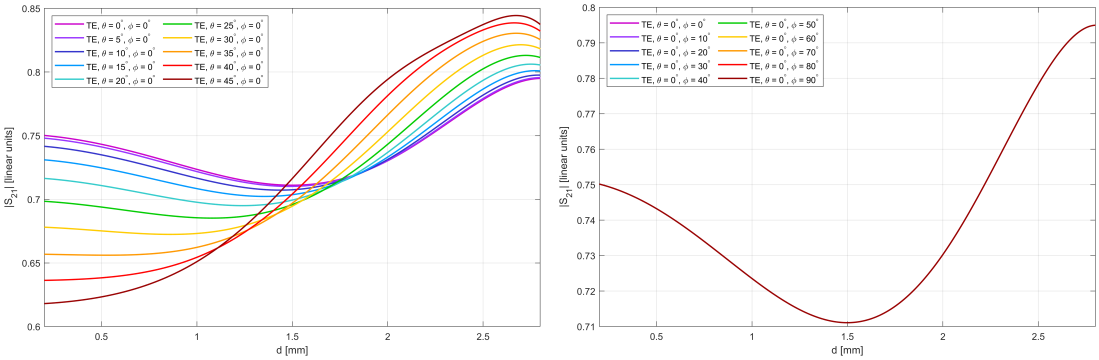


(g) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of θ)

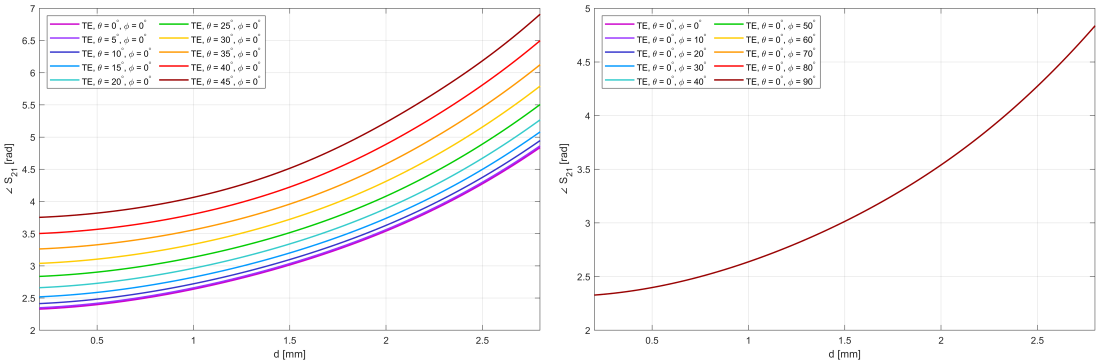


(h) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of φ)

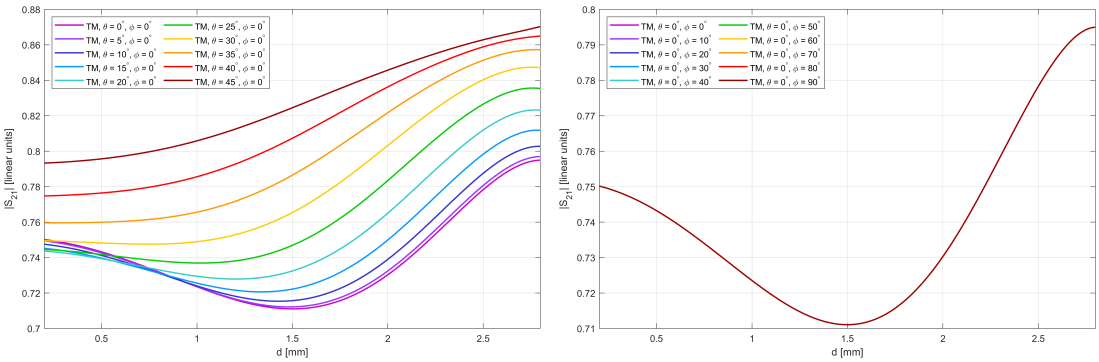
Figure 3.12: Reflection coefficient vs hole size d (UC-3, $T = 0.8\lambda_0$, TE & TM modes).



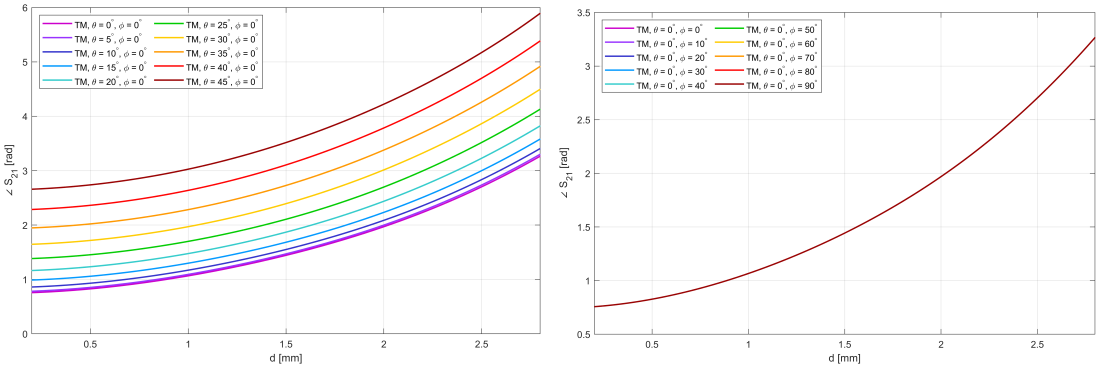
(a) $d \mapsto |S_{21}(d)|$ (TE mode, sweep of θ) **(b)** $d \mapsto |S_{21}(d)|$ (TE mode, sweep of φ)



(c) $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of θ) **(d)** $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of φ)

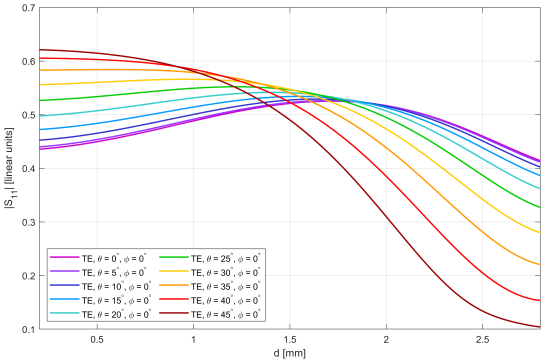


(e) $d \mapsto |S_{21}(d)|$ (TM mode, sweep of θ) **(f)** $d \mapsto |S_{21}(d)|$ (TM mode, sweep of φ)

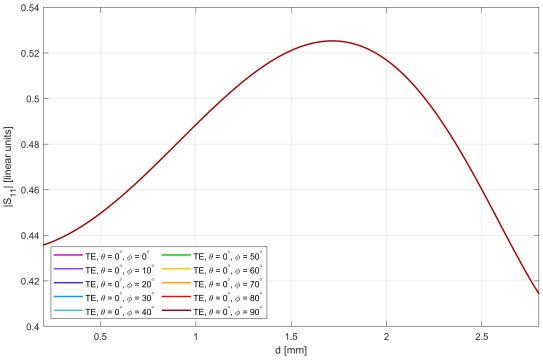


(g) $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of θ) **(h)** $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of φ)

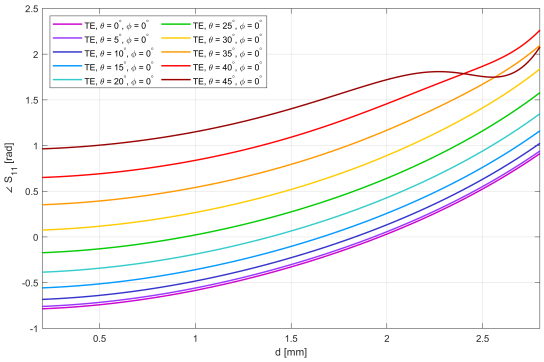
Figure 3.13: Transmission coefficient vs hole size d (UC-3, $T = 0.8\lambda_0$, TE & TM modes).



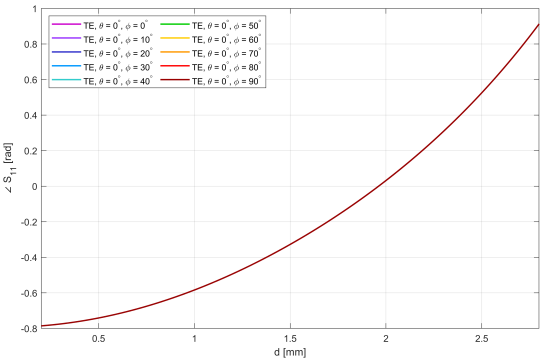
(a) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of θ)



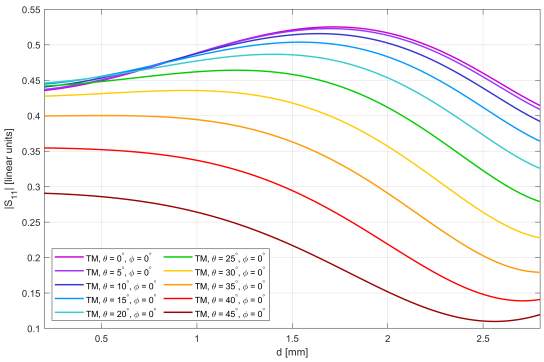
(b) $d \mapsto |S_{11}(d)|$ (TE mode, sweep of φ)



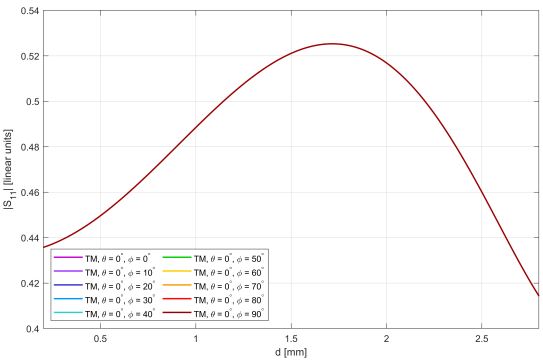
(c) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of θ)



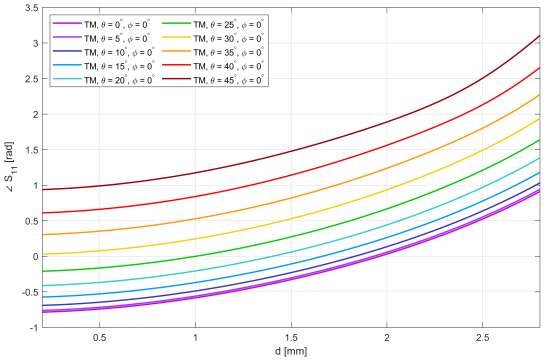
(d) $d \mapsto \angle S_{11}(d)$ (TE mode, sweep of φ)



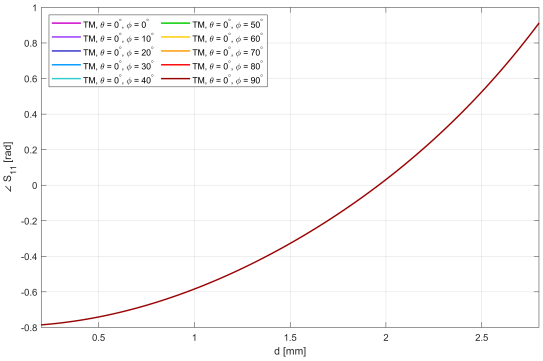
(e) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of θ)



(f) $d \mapsto |S_{11}(d)|$ (TM mode, sweep of φ)

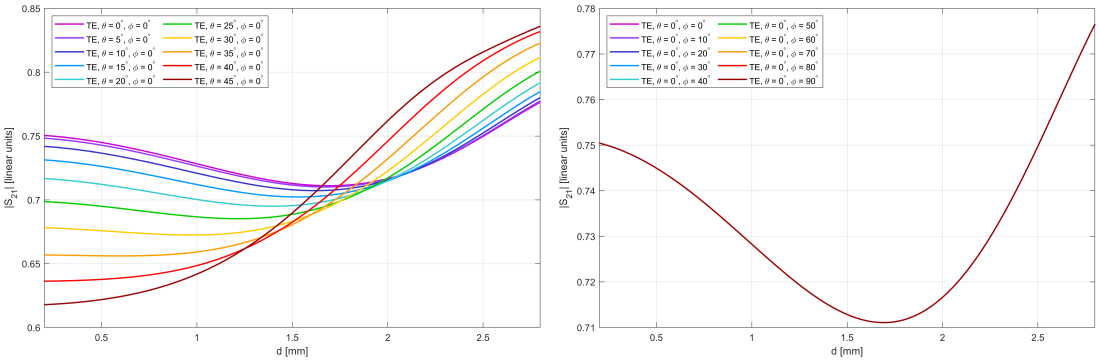


(g) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of θ)



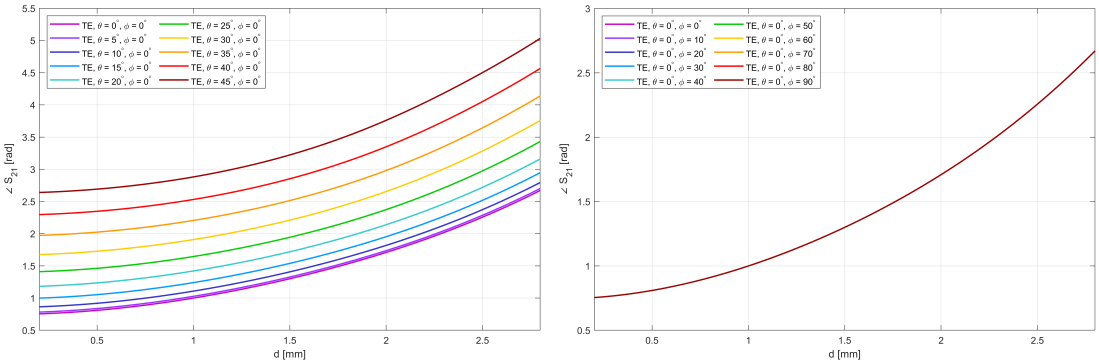
(h) $d \mapsto \angle S_{11}(d)$ (TM mode, sweep of φ)

Figure 3.14: Reflection coefficient vs hole size d (UC-4, $T = 0.8\lambda_0$, TE & TM modes).



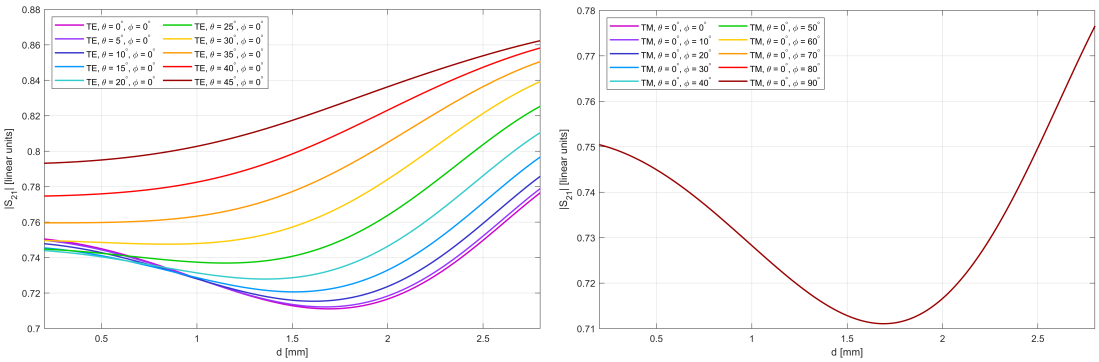
(a) $d \mapsto |S_{21}(d)|$ (TE mode, sweep of θ)

(b) $d \mapsto |S_{21}(d)|$ (TE mode, sweep of φ)



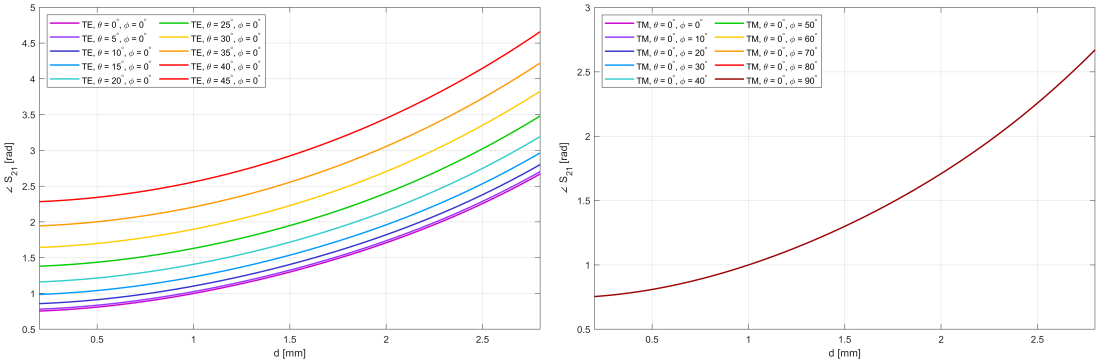
(c) $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of θ)

(d) $d \mapsto \angle S_{21}(d)$ (TE mode, sweep of φ)



(e) $d \mapsto |S_{21}(d)|$ (TM mode, sweep of θ)

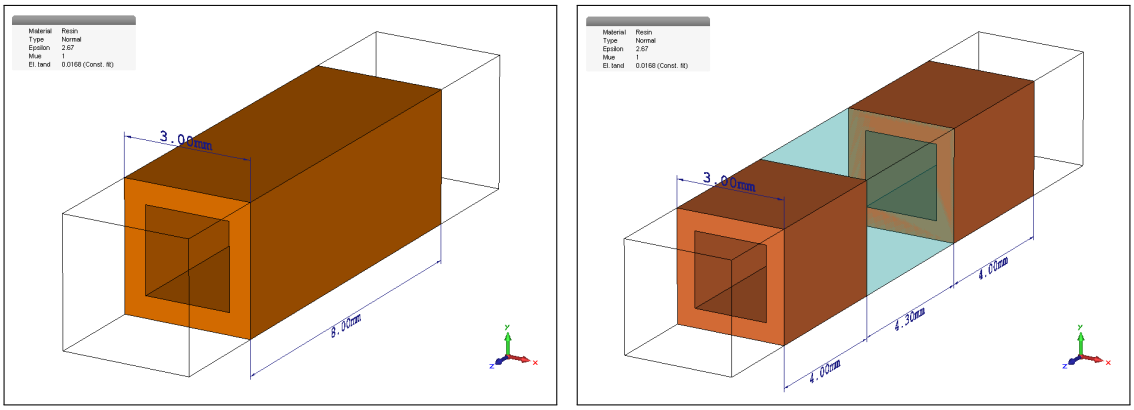
(f) $d \mapsto |S_{21}(d)|$ (TM mode, sweep of φ)



(g) $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of θ)

(h) $d \mapsto \angle S_{21}(d)$ (TM mode, sweep of φ)

Figure 3.15: Transmission coefficient vs hole size d (UC-4, $T = 0.8\lambda_0$, TE & TM modes).



(a) Simulated single-layer demo structure (UC-1) **(b)** Simulated three-layer demo structure (UC-3)

Figure 3.16: Demo structures simulated with CST for the validation of the UC model.

- In all the structures, $|S_{21}|$ increases with d because, for small values of the hole size, the losses of the material are non-neglibible and degrade the overall performance of the cell.
- In general, the three-layer structures (UC-3 and UC-4) have worse performance with respect to the single-layer structures because of the losses introduced by the glass layer.
- In general, in the square-hole cells (UC-1 and UC-3) losses have a lower impact with respect to the equivalent circular-hole structures (UC-2 and UC-4) due to the geometrical considerations already made in the previous paragraphs.

  For this reason, both the CST simulations used for the validation of the model (Section 3.2) and the analytical optimization (Section 3.4) have been carried out considering only the former structures, which are more promising in view of the practical realization of a TA or a SES.

3.2 Validation of the Model

The model presented in the previous paragraphs has been successfully validated against simulations performed with Computer Simulation Technology (CST) MW Studio Suite.

In order to check its effectiveness for a general structure, the simulations have been performed for several unit cells consisting of different number of layers, each with different characteristics in terms of material and presence, shape and size of the hole.

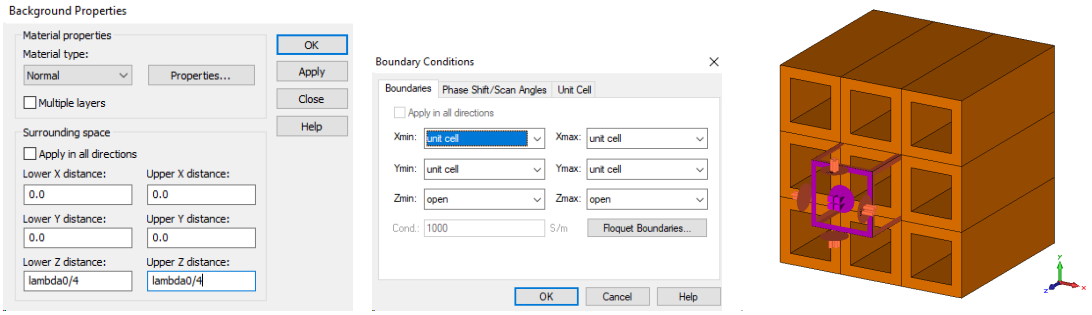
In particular, the most promising results presented in this work are those relative to the configurations UC-1 and UC-3, already analyzed in the previous paragraphs from an analytical point of view.

The simulated structures, which can be observed in Figure 3.16, are characterized by

$$W = 0.3\lambda_0 = 3\,\text{mm}, \quad T = 0.8\lambda_0 = 8\,\text{mm}, \quad T_g = 4.3\,\text{mm}. \tag{3.27}$$

In order to ensure the repeatibility of the performed simulations, the most important rules followed during the setup of the CST solver are here summarized:

- **Workflow and solver**: the chosen project template is *MW & RF & Optical/Periodic Structures*, with the *Frequency Domain* solver.



(a) Background properties **(b)** Boundary conditions **(c)** Boundary conditions, 3D

Figure 3.17: Setup for UC simulations in CST MW Studio®.

In this way, the mesh is automatically adapted to the dimensions of the structure and it is not necessary to manually refine it.

- **Background material**: the lower and upper distances along the three main axes are set in the following way, as shown in Figure 3.17a:

$$\begin{cases} x_{\text{low}} = x_{\text{up}} = 0 \\ y_{\text{low}} = y_{\text{up}} = 0 \\ z_{\text{low}} = z_{\text{up}} = \dfrac{\lambda_0}{4} \end{cases} \tag{3.28}$$

  Together with the proper settings of the boundary conditions, the choice $z_{\text{low}} = z_{\text{up}} \neq 0$ is necessary for the correct setup of the Floquet ports.

- **Boundary conditions**: both `Zmin` and `Zmax` should be set to `open`, as shown in Figure 3.17b; in this way, two Floquet ports are automatically detected at distance $\lambda_0/4$ from the structure and there is no need for the user to manually set up an excitation source.

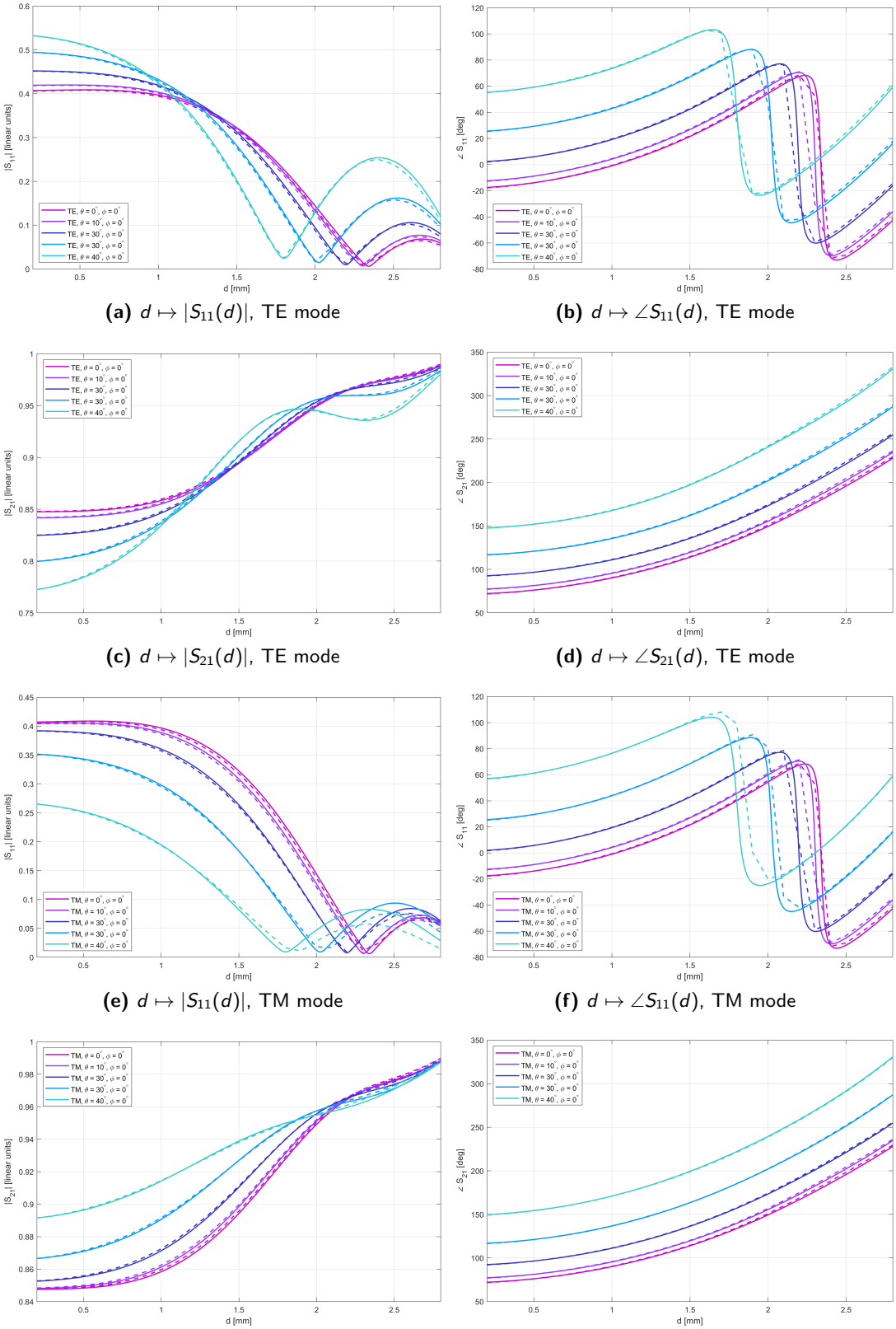
  Figure 3.17c shows how CST interprets the above settings: it is possible to notice that the array is treated as a fully periodic structure, which will not be true during the design of the entire antenna.

- **Setup solver**: to evaluate how the reflection and transmission coefficients change with θ, φ and d, a two-step procedure is needed:

  1. First of all, a *Parametric Sweep* of the desired parameters (`theta`, `phi` and `d`) must be properly set;
  2. Then, the calculation of the S parameters can be set up from the menu *Result templates*/*S-parameters*/*S parameters OD*, accessible from the *Parametric Sweep* window. Here, it is easily possible to select magnitude and phase of S_{11} and S_{21} for modes TE_{00} and TM_{00}.

The results of the simulations are shown in Figures 3.18 and 3.19, respectively for UC-1 and UC-3.

Coherently with the theoretical analysis performed in Section 3.1.6, d is swept in the range $0.2 \div 2.8\,\text{mm}$, while θ and φ are simulation parameters: θ varies from 0 to 40° with 10° steps, while $\varphi = 0°$ in all cases; this choice is justified by the considerations made at the end of the prevous paragraph.



(a) $d \mapsto |S_{11}(d)|$, TE mode

(b) $d \mapsto \angle S_{11}(d)$, TE mode

(c) $d \mapsto |S_{21}(d)|$, TE mode

(d) $d \mapsto \angle S_{21}(d)$, TE mode

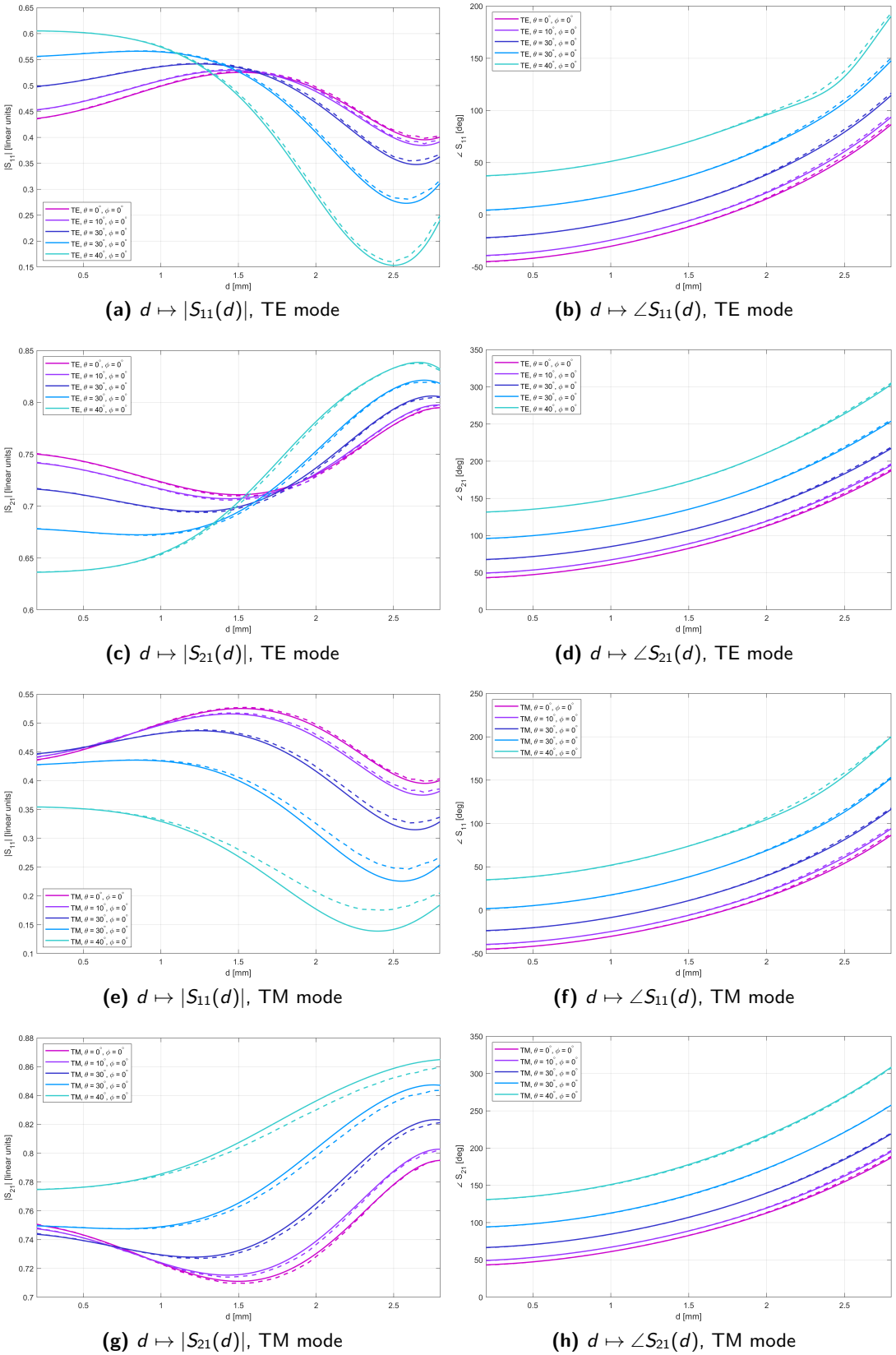
(e) $d \mapsto |S_{11}(d)|$, TM mode

(f) $d \mapsto \angle S_{11}(d)$, TM mode

(g) $d \mapsto |S_{21}(d)|$, TM mode

(h) $d \mapsto \angle S_{21}(d)$, TM mode

Figure 3.18: Validation of the UC model for UC-1 (TE & TM modes). Continuous lines: analytical model; dashed lines: CST simulation.



(a) $d \mapsto |S_{11}(d)|$, TE mode

(b) $d \mapsto \angle S_{11}(d)$, TE mode

(c) $d \mapsto |S_{21}(d)|$, TE mode

(d) $d \mapsto \angle S_{21}(d)$, TE mode

(e) $d \mapsto |S_{11}(d)|$, TM mode

(f) $d \mapsto \angle S_{11}(d)$, TM mode

(g) $d \mapsto |S_{21}(d)|$, TM mode

(h) $d \mapsto \angle S_{21}(d)$, TM mode

Figure 3.19: Validation of the UC model for UC-3 (TE & TM modes). Continuous lines: analytical model; dashed lines: CST simulation.

The curves derived from the EM solver are compared with their corresponding analytical values, demonstrating that the model achieves a high level of accuracy in predicting the UC behavior with the variation of its geometric parameters. In most cases, the agreement is more than acceptable, confirming the model's reliability for this purpose.

3.3 Extension of the Model to Tapered Structures

Since the model developd for the cells with "standard" geometries has proved to be highly accurate, this Section is devoted to its extension more complex and innovative structures, like the ones studied in [29].

In particular, a new unit cell, from now on referred to as UC-5, is considered starting from the basic UC-1; its 3D structure and equivalent circuit model can be observed in Figure 3.20:

- Single square-hole layer with thickness H_{sqh};
- Two tapering sections with thickness H_{tap}, that connect the central layer to free space; tapered transitions aim at improving the matching bandwidth between transmission lines without the need for multi-section transformers.

The resin used for the realization of the UC is the same as the previous Sections; however, since the structure is at least three times thicker than the original UC-1, frequency is halved with respect to the previous calculations:

$$f_0 = 15\,\text{GHz}, \quad \lambda_0 = \frac{c_0}{f_0} = 20\,\text{mm}. \tag{3.29}$$

The geometrical parameters of the cell are the following:

$$\begin{cases} W = 0.3\lambda_0 = 6\,\text{mm} \\ H_{\text{sqh}} = H_{\text{tap}} = 0.6\lambda_0 = 12\,\text{mm} \end{cases} \tag{3.30}$$

so the total cell thickness is $T = 2H_{\text{sqh}} + H_{\text{tap}} = 36\,\text{mm}$.

In principle, the tapered sections should be modeled as non-uniform transmission lines with z-dependent characteristic impedance $Z^{\text{tap}}_{\text{TE|TM}}(z)$; in particular, a wise choice to achieve good matching is the employment of linearly-tapered transmission lines (LTTLs), which provide a good matching between the adjacent layers.

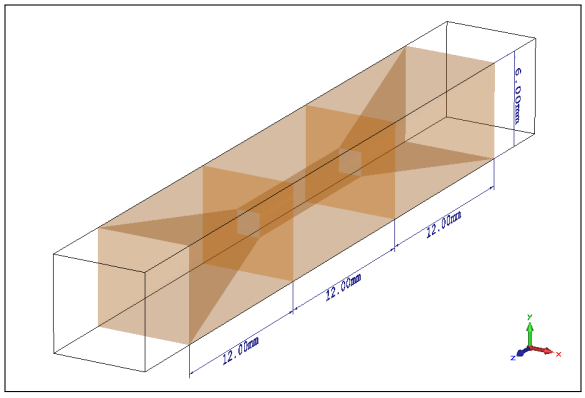
Non-uniform TLs, however, are very difficult to handle from a mathematical standpoint since, strictly speaking, their characteristic impedance should be expressed in terms of Bessel functions, as explained in [4].

To avoid introducing further complexity, in this work the tapered part is modeled as a sequence of steps with very small thickness $H_{\text{step}} \ll H_{\text{tap}}$, each of them modeled as a uniform TL section. In this way, the complete cell consists in the cascade of a very large number of layers, which can be easily modeled with the scheme introduced in the previous paragraphs.

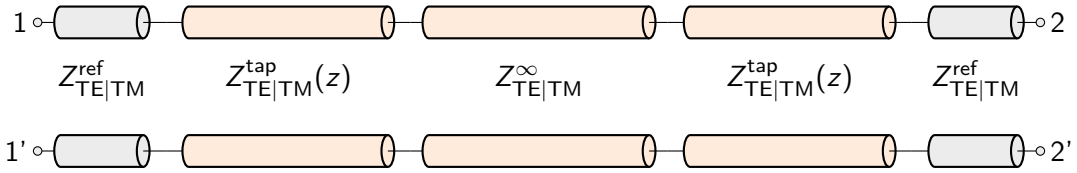
In this case, the accuracy of the results does not depend only on the employed physical model, but also on the amplitude of each step: in view of this, a trade-off is necessary because, as H_{step} decreases, the computational cost obviously increases.

The MATLAB® function used to model the tapered cell is reported in Section B.4.5; after several trials, the step size has been set to the following value:

$$H_{\text{step}} = 1\,\mu\text{m}, \tag{3.31}$$

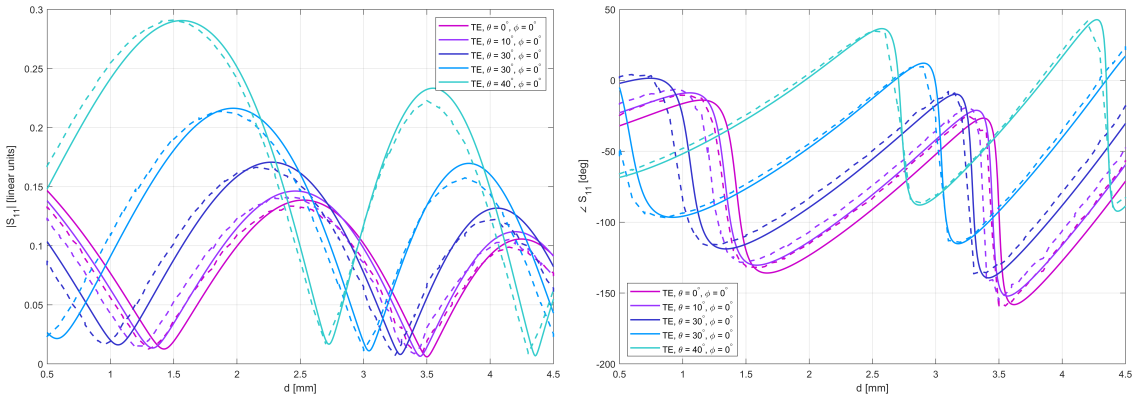


(a) Simulated tapered structure



(b) Equaivalent TL model

Figure 3.20: 3D model and equivalent TL model of the three-layer tapered UC (UC-5).



(a) $d \mapsto |S_{11}|(d)$ **(b)** $d \mapsto \angle S_{11}(d)$

(c) $d \mapsto |S_{21}|(d)$ **(d)** $d \mapsto \angle S_{21}(d)$

Figure 3.21: Validation of the UC model for a three-layer demo structure (TE mode). Continuous lines: analytical model; dashed lines: CST simulation.

meaning that each tapered layer is modeled with $N_{step} = H_{tap}/H_{step} = 12\,000$ steps. In this way, the results are reasonably accurate, as shown in Figure 3.21, without introducing excessive computational overhead; if N_{step} further increases, the MATLAB® risks getting stuck because of the significant complexity of the calculations.

Despite the results being encouraging, such structure is not further investigated for UC optimization and TA/SES design because, taking into account the presence of the glass layer, its thickness would be excessive for the successful realization of a smart skin. Tapered cells, however, have good potentialities in terms of both transmission coefficient and phase range, so the validation of the corresponding model is a good starting point for the exploration of novel research applications.

3.4 Unit Cell Optimization

In this section, the model developed in the previous paragraphs is further refined with the aim of designing a complete TA based solely on analytical considerations.

In particular, given the geometric configuration of the structure and the parameters of the materials of interest, an **optimization method** is developed and tested in order to calculate the values of the geometric parameters d and T that maximize the cell performance in terms of reflection and transmission coefficients.

The steps followed to reach this goal are here summarized:

1. First of all, the ranges where d and T can change are determined using mostly empirical considerations; this problem is addressed in Section 3.4.1;

2. Then, the actual optimization method is developed: the adopted procedure, which is described in details in Section 3.4.2, exploits a single- or multi-objective Genetic Algorithm (GA); given the incidence angles θ and φ and the required value for $\angle S_{21} =: \Phi_0$, the aims of the optimization process are the following:

   - minimization of the reflection coefficient $|S_{11}|$;
   - maximization of the transmission coefficient $|S_{21}|$.

   As just said, this task is carried out using different optimization strategies and the results obtained with the various approaches are compared in terms of both performance and computational cost.

   The advantages and disadvantages of each method are discussed not only in this Chapter, concerning the single-cell design, but also in Chapter 4, where the realization of a complete TA is addressed.

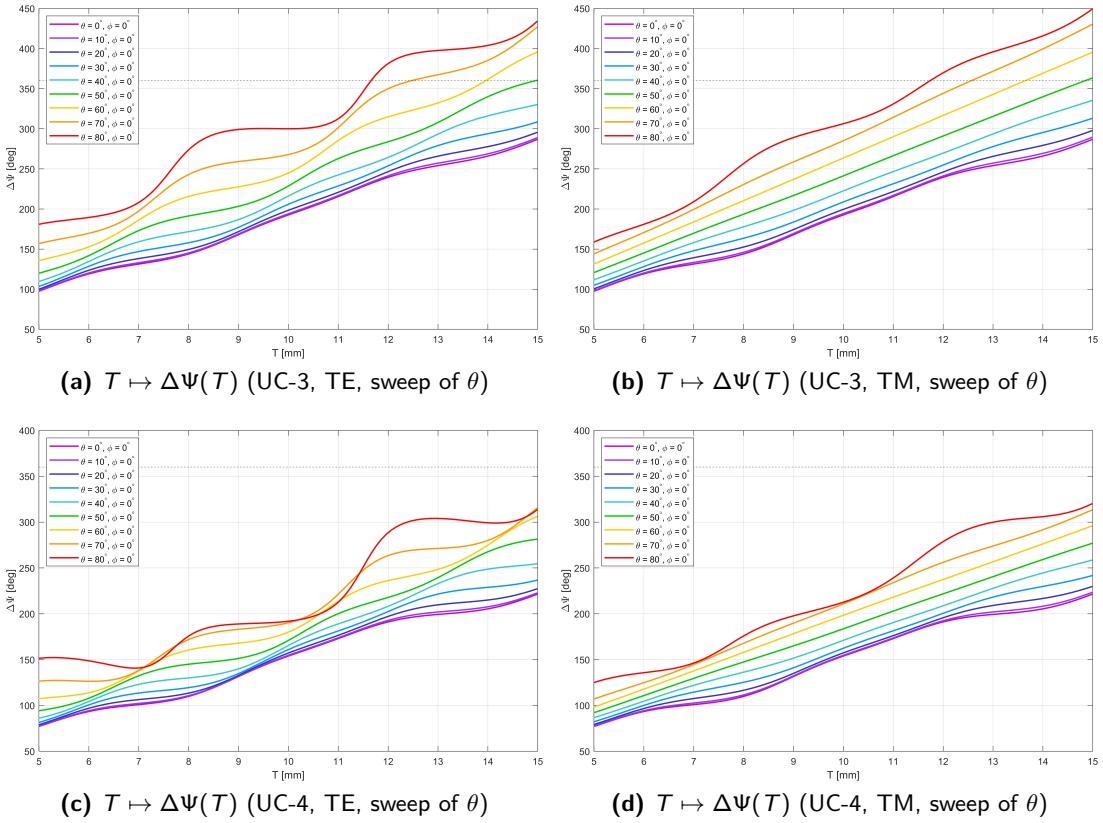
3.4.1 Choice of Minimum and Maximum UC Height

The first step to optimize the performance of the TA unit cell is to fix the variation ranges for the hole size d and the cell height T.

As far as d is concerned, there are not particular constraints apart from the technological feasibility of the structure; for this reason, the d range

$$d = d_{min} \div d_{max} \equiv 0.2 \div 2.8\,\text{mm}, \tag{3.32}$$

already exploited in the first part of the chapter, is adopted.



(a) $T \mapsto \Delta\Psi(T)$ (UC-3, TE, sweep of θ)

(b) $T \mapsto \Delta\Psi(T)$ (UC-3, TM, sweep of θ)

(c) $T \mapsto \Delta\Psi(T)$ (UC-4, TE, sweep of θ)

(d) $T \mapsto \Delta\Psi(T)$ (UC-4, TM, sweep of θ)

Figure 3.22: Phase range of the transmission coefficient vs cell height T for UC-3 and UC-4 (variable cell height T, TE & TM modes).

Table 3.2: Phase range of the transmission coefficient with $T = T_{\text{max}}$, UC-3 & UC-4.

	UC-3		UC-4	
θ [deg]	$\Delta\Psi_{\text{TE}}$ [rad]	$\Delta\Psi_{\text{TM}}$ [rad]	$\Delta\Psi_{\text{TE}}$ [rad]	$\Delta\Psi_{\text{TM}}$ [rad]
0	4.07	4.07	3.86	3.86
10	4.10	4.09	3.89	3.90
20	4.17	4.17	3.97	4.01
30	4.29	4.32	4.13	4.22
40	4.49	4.59	4.44	4.51
50	4.86	4.95	4.91	4.83
60	5.39	5.39	5.34	5.16
70	5.93	5.86	5.51	5.47
80	6.35	6.26	5.47	5.59

Concerning T, on the other hand, more complex considerations are required. In fact, it is important to point out that, in the final application, the structure (either TA or SES) will be integrated in a window pane; for this reason, its visual impact must be minimal.

In light of this, it is not only important to choose a **transparent material** and handle carefully the 3D-printing and post-processing phase, as discussed in Section 3.1.

It is also essential to design a **conformal structure** that integrates seamlessly with the glass pane, minimizing any protrusion and preserving its sleek appearance. Obviously, a reasonable trade-off must be found between this visual requirement and the structure's performance, which deteriorates as thickness decreases due to greater challenges in achieving satisfactory phase compensation.

Therefore, the choice of the T range

$$T = T_{\text{min}} \div T_{\text{max}} \tag{3.33}$$

is fundamental for the design of a practically feasible structure.

As an initial estimation, the range $T = 5 \div 15\,\text{mm}$ is chosen. To refine the boundaries of this interval, the MATLAB® function `Tmax.m`, reported in Appendix B.5, has been exploited.

This subroutine is suitably parametrized so as to be exploitable for all the UC configurations under exam (single/three-layer, square/circular hole), and it operates in the following way:

1. Loop over cell heights with fixed incidence angle θ: for each value of $T \in [T_{\text{min}}, T_{\text{max}}]$, $\angle S_{21}$ is evaluated as function of d and the *phase range* $\Delta\Psi$ is evaluated as follows:

$$\Delta\Psi|_\theta := \angle S_{21}(d_{\text{max}})|_\theta - \angle S_{21}(d_{\text{min}})|_\theta \,. \tag{3.34}$$

   The reason why the calculation of $\Delta\Psi$ is necessary is that, in the design of the complete array, the phase of the transmission coefficient will be constrained by the design Equation (4.19), which provides the required phase compensation for each UC element depending on the scan angle of the incident field.

   If $\Delta\Psi > 2\pi$, it is possible to achieve the required phase compensation irrespectively of the geometrical parameters of the cell (m, n), because at least a value of d and T exists providing the required Ψ_{mn}.

2. Calculation of T_{max}: the values of $\Delta\Psi$ for each value of T are stored in a vector and $T_{\text{max}}|_\theta$ for the given θ is chosen solving the following minimization problem:

$$T_{\text{max}}|_\theta = \min\{\, T \;:\; \Delta\Psi|_\theta - 2\pi > 0 \,\}\,. \tag{3.35}$$

The above procedure must be repeated sweeping θ in a predefined range and the lowest possible value of T_{max} must be selected:

$$T_{\text{max}} = \min_\theta \; T_{\text{max}}|_\theta \,. \tag{3.36}$$

If no value of T exists such that a 2π-phase range is ensured, namely

$$\{\, T \;:\; \Delta\Psi|_\theta - 2\pi > 0 \,\} = \emptyset, \tag{3.37}$$

then the original value for T_{max} is maintained, i.e., $T_{\text{max}} = 15\,\text{mm}$.

The results obtained for the configurations UC-3 and UC-4 are summarized in Figure 3.22 and Table 3.2.

The values of T_{max} found for the two configurations are

$$\begin{cases} T_{max,3} = 11.71\,\text{mm} \\ T_{max,4} = 15\,\text{mm} \end{cases} \tag{3.38}$$

The obtained results show that, due to the losses introduced by the glass layer, a 2π-range can only be obtained with very high values of T and θ which, however, are completely unfeasible from a practical standpoint; furthermore, the circular geometry introduces additional losses in the resin layer and makes it impossible to reach $\Delta\Psi = 2\pi$ even when the incidence angle is very high.

To reach a reasonable trade-off between thickness and performance is thus necessary, the configuration adopted in all the subsequent design steps is the following:

- Structure: UC-3 (three-layer cell with square-hole external resin layers and internal glass layer);
- Hole size (d) range: $0.2 \div 2.8\,\text{mm}$;
- Resin layer height (T) range: $5 \div 11.71\,\text{mm}$.

3.4.2 GA-based Optimization

At this point, the Genetic Algorithm (GA) described in Chapter 2 from a theoretical point of view can be implemented in order to find the values of the geometric parameters d and T that maximize the performance of the UC of interest.

This task can be achieved in MATLAB® exploiting the tools `ga()` and `gamultiobj()`, which are built-in functions provided within the Global Optimization Toolbox [36], offering users an accessible way to implement, respectively, single- or multi-objective genetic algorithms without extensive customization.

Before addressing the actual optimization problem, it is interesting and useful to give a quick overview of `ga()` and `gamultiobj()` features.

Both functions serve as an efficient tool for solving optimization problems, particularly in engineering and computational research; in fact, as an evolutionary algorithm, GA is highly effective in handling complex, non-linear, and multi-modal problems.

The performance of `ga()` and `gamultiobj()` has been tested in [43] using 29 benchmark problems, including 14 single-objective functions and 15 multi-objective functions. These tests were conducted with default settings, simulating a *black-box optimization scenario* where the algorithm operates without specific parameter tuning.

The results of this research are here summarized:

Single-Objective Optimization Performance. the `ga()` function demonstrates high accuracy and fast convergence for problems with two variables, successfully reaching global optima. However, when tested on problems with more than thirty variables, performance varies:

- Most functions exhibit steady convergence, but optimization speed is slower;
- The algorithm may settle at local optima rather than reaching the global solution.

Multi-Objective Optimization Performance. the `gamultiobj()` function faces greater challenges when tested on multi-objective problems. The algorithm has difficulty achieving both convergence to the global Pareto front and maintaining diversity across solutions:

- Performance is better if the complexity of the problem is lower;

- Several problems result in poor exploration of solution spaces, particularly in regions where extreme Pareto solutions are located.

In conclusion, despite its limitations the single-objective algorithm `ga()` provides reliable results for a wide range of cost functions, making it an effective tool for general-purpose optimization tasks, especially if the number of optimization variables is reasonably low.

On the other hand, the default settings of `gamultiobj()` are not sufficiently robust to ensure optimal results across all cases. As a result, users relying on this function should verify outputs carefully and consider manual parameter tuning for better results.

The optimization problem that must be solved for the application of interest is the following:

- **Physical framework**: UC-3, $d = 0.2 \div 2.8\,\text{mm}$, $T = 5 \div 11.71\,\text{mm}$;
- **Optimization variables**: hole size d, resin layer height T;
- **Optimization results**: optimal hole size d_{opt}, optimal resin layer height T_{opt};
- **Cost functions**, i.e., optimization goals:

$$\begin{cases} f_1(d, T) = |S_{11}(d, T)| \\ f_2(d, T) = -|S_{21}(d, T)| \end{cases} \tag{3.39}$$

  $-|S_{21}|$ is used instead of $|S_{21}|$ because MATLAB's GA only support minimization problems, but the aim of the project is to *maximize* the transmission coefficient.

- **Optimization constraints**: $|\angle S_{21} - \Phi_0| < \varepsilon$ (inequality constraint) or $\angle S_{21} = \Phi_0$ (equality constraint), depending whether an error on the phase delay is accepted or not.

Using MATLAB's GA, three different optimization approaches are investigated, depending on the formulation adopted from the mathematical point of view to find the optimal geometrical parameters:

- **Phase-only project** (`'p'`): optimizes transmission coefficient $|S_{21}|$:

$$(d_{\text{opt}}, T_{\text{opt}}) = (d, T) \;:\; |S_{21}(d_{\text{opt}}, T_{\text{opt}})| = \max_{d,T} |S_{21}(d, T)| \tag{3.40}$$

- **Reflection-only project** (`'r'`): optimizes reflection coefficient $|S_{11}|$:

$$(d_{\text{opt}}, T_{\text{opt}}) = (d, T) \;:\; |S_{11}(d_{\text{opt}}, T_{\text{opt}})| = \min_{d,T} |S_{11}(d, T)| \tag{3.41}$$

- **Complete project** (`'c'`): performs multi-objective optimization for both $|S_{21}|$ and $|S_{11}|$:

$$(d_{\text{opt}}, T_{\text{opt}}) = (d, T) \;:\; \begin{cases} |S_{21}(d_{\text{opt}}, T_{\text{opt}})| = \max_{d,T} |S_{21}(d, T)| \\ |S_{11}(d_{\text{opt}}, T_{\text{opt}})| = \min_{d,T} |S_{11}(d, T)| \end{cases} \tag{3.42}$$

The MATLAB® subroutines used to implement the problems presented above are reported in Appendix B.6.1 and B.6.2, respectively for the equality and inequality constraint.

These functions operate in the following way:

1. **Input arguments**:

- Scan angles (`theta`, `phi`);
- Desired phase (`phi0`), i.e., required phase delay for the transmission coefficient, which is crucial for the correct operation of the array;
- Tolerance (`tol`) accepted to satisfy the requirements of the optimization constraint. If an equality constraint is required, $\texttt{tol} = 0$; in the case of an inequality constraint, all the results reported in this thesis are obtained with $\texttt{tol} = 0.3$: this choice ensures a phase error not bigger than $5 \div 10^{\circ}$;
- Project type (`'p'`, `'r'`, `'c'`);
- Centerband frequency (`f0`);
- Material properties (`eps_r`, `tan_delta`);
- Unit cell periodicity (`W`);
- Height and hole size ranges (`T`, `d`);
- Additional height parameter (`H`), i.e., distance of the Floquet ports from the edges of the cell;
- Number of grid points (`dim_grid`) to tune the accuracy of the research algorithm; for the scope of this work, the choice $\texttt{dim\_grid} = 600$ is a reasonable trade-off between the precision of the calculations and the computational cost of the algorithm.

2. **Definition of the lower and upper bounds for d and T**:

$$d \in [d_{\text{lb}}, d_{\text{ub}}] \subseteq [d_{\text{min}}, d_{\text{max}}], \quad T \in [T_{\text{lb}}, T_{\text{ub}}] \subseteq [T_{\text{min}}, T_{\text{max}}] \tag{3.43}$$

   - First of all, S-parameters (S_{11} and S_{21}) and phase delay ($\angle S_{21}$) are evaluated for both TE_{00} and TM_{00} modes.
   - Then, the function identifies possible intersections between phase delay and desired phase Φ_0 within a small tolerance:

$$|\angle S_{21}(d, T) - \Phi_0| < 1 \times 10^{-3}. \tag{3.44}$$

     If the intersection exists, the corresponding d and T ranges are selected, otherwise $\angle S_{21}$ is shifted by $\pm 2\pi$ to check for intersections in adjacent phase cycles:

$$|\angle S_{21}(d, T) \pm 2\pi - \Phi_0| < 1 \times 10^{-3}. \tag{3.45}$$

   - If there is still no direct intersection, the function finds the closest match by minimizing the distance of $\angle S_{21}$ from Φ_0.
   - The result is a discrete set of points in the space $(d, T) \in [d_{\text{min}}, d_{\text{max}}] \times [T_{\text{min}}, T_{\text{max}}]$. The lower and upper bounds of this set identify, respectively, $(d_{\text{lb}}, T_{\text{lb}})$ and $(d_{\text{ub}}, T_{\text{ub}})$.

3. **Definition of the optimization constraint**: the set of values (d, T) identified in the previous point is interpolated using a 5^{th}-order polynomial fitting (`polyfit` and `polyval`):

$$\begin{cases} q_{\text{TE}}(d) := p_{\text{TE1}} d^5 + p_{\text{TE2}} d^4 + p_{\text{TE3}} d^3 + p_{\text{TE4}} d^2 + p_{\text{TE5}} d + p_{\text{TE6}} \\ q_{\text{TM}}(d) := p_{\text{TM1}} d^5 + p_{\text{TM2}} d^4 + p_{\text{TM3}} d^3 + p_{\text{TM4}} d^2 + p_{\text{TM5}} d + p_{\text{TM6}} \end{cases} \tag{3.46}$$

   Depending whether an error on the phase delay is accepted or not, the non-linear optimization constraint can now be defined:

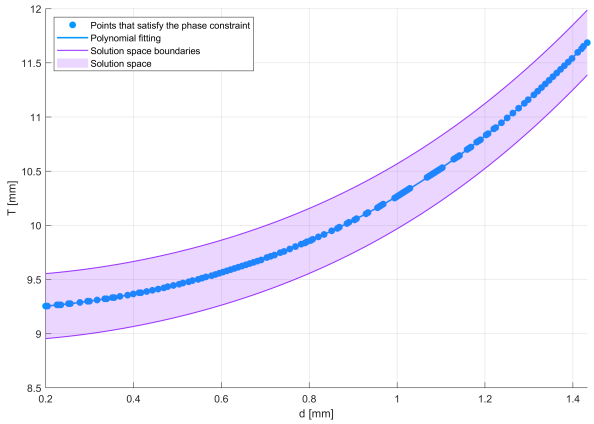


Figure 3.23: Optimization constraints for the set of angles $\theta = 0°$, $\varphi = 0°$, $\Phi_0 = 45°$, TE mode.

- Equality constraint (error on the phase delay not accepted):

$$\begin{cases} T = q_{\text{TE}}(d) & \text{TE mode optimization} \\ T = q_{\text{TM}}(d) & \text{TM mode optimization} \end{cases} \tag{3.47}$$

- Inequality constraint (error $\varepsilon_{\text{tol}} = \texttt{tol}$ on the phase delay accepted):

$$\begin{cases} |T - q_{\text{TE}}(d)| < \varepsilon_{\text{tol}} & \text{TE mode optimization} \\ |T - q_{\text{TM}}(d)| < \varepsilon_{\text{tol}} & \text{TM mode optimization} \end{cases} \tag{3.48}$$

These concepts can be better understood observing Figure 3.23, showing an overview of the optimization constraints for the set of angles $\theta = 0°$, $\varphi = 0°$, $\Phi_0 = 45°$ and TE-mode incidence: the blue points represent the set of values satisfying the phase constraint, while the blue curve shows the polynomial fitting.

When using the equality constraint, the solutions of the optimization problem lie on the blue fitting curve; when using the inequelity constraint, on the other hand, the solution space is represented by the lilac region, whose boundaries are

$$q_{\text{TE}}(d) - \varepsilon_{\text{tol}} \quad \text{and} \quad q_{\text{TE}}(d) + \varepsilon_{\text{tol}}. \tag{3.49}$$

4. **Definition of the objective function**: depending on the type of project, `'p'`, `'r'` or `'c'`, the objective functions are $|S_{11}|$ and/or $-|S_{21}|$, which must both be minimized.

5. **Implementation of the genetic algorithm**: to run the GA function, it is necessary to set the `optimoptions`:

   - `'PopulationSize', 50`: this parameter defines the number of individuals in each generation. Setting it to 50 is a good trade-off for maintaining diversity without excessive computational overhead.
   - `'MaxGenerations', 100`: this parameter sets the limit the number of evolutionary cycles for optimization. 100 generations should be sufficient for convergence in most problems.
   - `'ParetoFraction', 0.7`: this number controls the fraction of solutions retained in the Pareto front. The default value of `ParetoFraction` in MATLAB is 0.35: this means that 35% of the population is retained in each generation as part of the Pareto front. Increasing it to 0.7 increases the granularity of the solution space leading to a broader set of trade-off solutions and potentially improving diversity.

- `'Display', 'iter'`: this option prints information at each generation, allowing for a constant monitoring of the optimization progress.
- `'UseParallel', false`: this option determines whether computations run in parallel. Parallel computation (`true`) can speed up optimization, especially on high-performance machines. But since gamultiobj often requires function evaluations with nonlinear constraints, single-threaded processing (`false`) avoids memory bottlenecks.

To test and validate the optimization process just described, the algorithm has been executed for the cell UC-3, varying the incidence angles, the phase constraint, and the type of project.

A brief overview of the reported results is here provided:

- Tables 3.3-3.8 show the optimization results:
  1. Tables 3.3, 3.4 and 3.5 for the set $(\theta, \varphi, \Phi_0) = (0, 0, 45)^\circ$, considering both TE and TM mode incidence;
  2. Tables 3.6, 3.7 and 3.8 for the set $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$, considering both TE and TM incidence.
- The display information shown on MATLAB® Command Line while running the code is shown for the set of angles $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$ with inequality constraint. The `stdout` of the other performed tests is omitted to avoid unnecessary redundancy.
- Figure 3.24 depicts the results of a two-parameter analysis on the scattering parameters of UC-3, i.e., the behavior of S_{11} and S_{21} when both d and T are left as DoF. The results are shown for the sets $(\theta, \varphi, \Phi_0) = (0, 0, 45)^\circ$ and $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$. Again, only the TE-mode case is reported for the sake of simplicity.

Table 3.3: Optimization results, UC-3, `'p'` project ($\theta = 0^\circ$, $\varphi = 0^\circ$, $\Phi_0 = 45^\circ$).

Quantity	Unit	Equality (TE)	Inequality (TE)	Equality (TM)	Inequality (TM)
d	mm	0.72	0.72	0.20	0.70
T	mm	9.72	9.42	9.25	9.40
$\lvert S_{11}\rvert$	dB	-19.31	-29.73	-24.83	-30.11
$\lvert S_{21}\rvert$	dB	-1.61	-1.54	-1.58	-1.54
$\angle S_{21}$	deg	45.00	52.87	44.98	52.92
$\lvert S_{11}\rvert^2 + \lvert S_{21}\rvert^2$	dB	-1.54	-1.53	-1.56	-1.53

Table 3.4: Optimization results, UC-3, `'r'` project ($\theta = 0^\circ$, $\varphi = 0^\circ$, $\Phi_0 = 45^\circ$).

Quantity	Unit	Equality (TE)	Inequality (TE)	Equality (TM)	Inequality (TM)
d	mm	0.20	0.64	0.20	0.68
T	mm	9.25	9.33	9.25	9.37
$\lvert S_{11}\rvert$	dB	-24.83	-30.39	-24.83	-30.36
$\lvert S_{21}\rvert$	dB	-1.58	-1.54	-1.58	-1.54
$\angle S_{21}$	deg	44.98	52.71	44.98	52.96
$\lvert S_{11}\rvert^2 + \lvert S_{21}\rvert^2$	dB	-1.56	-1.53	-1.56	-1.53

Table 3.5: Optimization results, UC-3, 'c' project ($\theta = 0°$, $\varphi = 0°$, $\Phi_0 = 45°$).

Quantity	Unit	Equality (TE)	Inequality (TE)	Equality (TM)	Inequality (TM)
d	mm	0.60	0.51	0.24	0.54
T	mm	9.56	9.26	9.27	9.29
$\lvert S_{11}\rvert$	dB	-20.72	-29.66	-24.60	-29.35
$\lvert S_{21}\rvert$	dB	-1.59	-1.55	-1.58	-1.55
$\angle S_{21}$	deg	45.02	50.53	45.03	50.48
$\lvert S_{11}\rvert^2 + \lvert S_{21}\rvert^2$	dB	-1.54	-1.54	-1.55	-1.54

Table 3.6: Optimization results, UC-3, 'p' project ($\theta = 30°$, $\varphi = 90°$, $\Phi_0 = 180°$).

Quantity	Unit	Equality (TE)	Inequality (TE)	Equality (TM)	Inequality (TM)
d	mm	0.39	0.79	0.31	0.73
T	mm	5.54	5.57	5.60	5.57
$\lvert S_{11}\rvert$	dB	-18.97	-18.84	-21.41	-22.46
$\lvert S_{21}\rvert$	dB	-1.67	-1.64	-1.51	-1.46
$\angle S_{21}$	deg	181.31	187.48	180.10	187.59
$\lvert S_{11}\rvert^2 + \lvert S_{21}\rvert^2$	dB	-1.59	-1.55	-1.47	-1.43

Table 3.7: Optimization results, UC-3, 'r' project ($\theta = 30°$, $\varphi = 90°$, $\Phi_0 = 180°$).

Quantity	Unit	Equality (TE)	Inequality (TE)	Equality (TM)	Inequality (TM)
d	mm	0.51	0.95	0.28	0.73
T	mm	5.60	5.73	5.59	5.57
$\lvert S_{11}\rvert$	dB	-19.15	-19.47	-21.43	-22.46
$\lvert S_{21}\rvert$	dB	-1.68	-1.64	-1.51	-1.46
$\angle S_{21}$	deg	180.83	187.48	1879.97	187.59
$\lvert S_{11}\rvert^2 + \lvert S_{21}\rvert^2$	dB	-1.60	-1.57	-1.47	-1.43

Table 3.8: Optimization results, UC-3, 'c' project ($\theta = 30°$, $\varphi = 90°$, $\Phi_0 = 180°$).

Quantity	Unit	Equality (TE)	Inequality (TE)	Equality (TM)	Inequality (TM)
d	mm	0.98	0.95	0.60	0.74
T	mm	6.06	5.73	5.75	5.58
$\lvert S_{11}\rvert$	dB	-15.26	-19.47	-19.83	-22.45
$\lvert S_{21}\rvert$	dB	-1.80	-1.64	-1.53	-1.46
$\angle S_{21}$	deg	179.42	187.48	180.16	187.56
$\lvert S_{11}\rvert^2 + \lvert S_{21}\rvert^2$	dB	-1.61	-1.57	-1.47	-1.43

```
Theta [deg]: 30
Phi [deg]: 90
Desired phase [deg]: 180
Tolerance: 0.3
Type of project (p = phase-only, r = reflection-only, c = complete): p
```

```
Single objective optimization:
2 Variables
1 Nonlinear inequality constraints

Options:
CreationFcn:       @gacreationuniform
CrossoverFcn:      @crossoverscattered
SelectionFcn:      @selectionstochunif
MutationFcn:       @mutationadaptfeasible

                            Best       Max        Stall
Generation  Func-count       f(x)     Constraint  Generations
    1           2500     -0.745701            0      0
    2           4950     -0.745937            0      0
    3          14403     -0.827079            0      0
Optimization finished: average change in the fitness value less than options.FunctionTolerance and
↪  constraint violation is less than options.ConstraintTolerance.
```

```
Theta [deg]: 30
Phi [deg]: 90
Desired phase [deg]: 180
Tolerance: 0.3
Type of project (p = phase-only, r = reflection-only, c = complete): r

Single objective optimization:
2 Variables
1 Nonlinear inequality constraints

Options:
CreationFcn:       @gacreationuniform
CrossoverFcn:      @crossoverscattered
SelectionFcn:      @selectionstochunif
MutationFcn:       @mutationadaptfeasible

                            Best       Max        Stall
Generation  Func-count       f(x)     Constraint  Generations
    1           2500      0.111573            0      0
    2           4950       0.11157            0      0
    3           7400      0.111475            0      0
    4           9850      0.106447            0      0
    5          12300      0.106267    0.0008807      0
Optimization finished: average change in the fitness value less than options.FunctionTolerance and
↪  constraint violation is less than options.ConstraintTolerance.
```

```
Theta [deg]: 30
Phi [deg]: 90
Desired phase [deg]: 180
Tolerance: 0.3
Type of project (p = phase-only, r = reflection-only, c = complete): c
Multi-objective optimization:
2 Variables
1 Nonlinear inequality constraints

Options:
CreationFcn:       @gacreationuniform
CrossoverFcn:      @crossoverintermediate
SelectionFcn:      @selectiontournament
MutationFcn:       @mutationadaptfeasible

                              Average            Average
Generation   Func-count    Pareto distance    Pareto spread
    1            51                1                 1
    2           101                0                 0
    3           151                0                 0
```

```
     4           201                  0          0.0055973
     5           251                  0                  0
     6           301                  0                  0
     7           351                  0                  0
     8           401                  0        0.000404442
     9           451                  0                  0
    10           501                  0                  0
    11           551                  0                  0
    12           601                  0        0.000343513
    13           651                  0                  0
    14           701                  0        5.35249e-05
    15           751                  0                  0
    16           801                  0        3.09363e-05
    17           851                  0        1.15995e-05
    18           901                  0                  0
    19           951                  0                  0
    20          1001                  0                  0
    21          1051                  0                  0
    22          1101                  0         4.1322e-06
    23          1151                  0                  0
    24          1201                  0        6.56701e-07
    25          1251                  0                  0
    26          1301                  0        1.34188e-08
    27          1351                  0                  0
    28          1401                  0                  0
    29          1451                  0         3.1376e-08
    30          1501                  0                  0

                              Average            Average
 Generation   Func-count    Pareto distance    Pareto spread
    31          1551                  0                  0
    32          1601                  0                  0
    33          1651                  0                  0
    34          1701                  0                  0
    35          1751                  0                  0
    36          1801                  0                  0
    37          1851                  0                  0
    38          1901                  0                  0
    39          1951                  0                  0
    40          2001                  0                  0
    41          2051                  0                  0
    42          2101                  0                  0
    43          2151                  0         4.1762e-09
    44          2201                  0                  0
    45          2251                  0                  0
    46          2301                  0                  0
    47          2351                  0                  0
    48          2401                  0                  0
    49          2451                  0                  0
    50          2501                  0        1.82868e-10
    51          2551                  0        2.74788e-09
    52          2601                  0                  0
    53          2651                  0                  0
    54          2701                  0                  0
    55          2751                  0                  0
    56          2801                  0        3.43143e-10
    57          2851                  0                  0
    58          2901                  0                  0
    59          2951                  0                  0
    60          3001                  0        1.87871e-10

                              Average            Average
 Generation   Func-count    Pareto distance    Pareto spread
    61          3051                  0         4.4347e-11
    62          3101                  0                  0
    63          3151                  0                  0
    64          3201                  0        6.04156e-11
    65          3251                  0                  0
```

```
    66          3301                  0                  0
    67          3351                  0                  0
    68          3401                  0                  0
    69          3451                  0       2.38261e-12
    70          3501                  0                  0
    71          3551                  0                  0
    72          3601                  0                  0
    73          3651                  0       9.36147e-12
    74          3701                  0                  0
    75          3751                  0                  0
    76          3801                  0                  0
    77          3851                  0                  0
    78          3901                  0                  0
    79          3951                  0                  0
    80          4001                  0                  0
    81          4051                  0                  0
    82          4101                  0                  0
    83          4151                  0                  0
    84          4201                  0                  0
    85          4251                  0                  0
    86          4301                  0                  0
    87          4351                  0                  0
    88          4401                  0                  0
    89          4451                  0                  0
    90          4501                  0                  0

                              Average            Average
Generation    Func-count    Pareto distance    Pareto spread
    91          4551                  0                  0
    92          4601                  0                  0
    93          4651                  0                  0
    94          4701                  0                  0
    95          4751                  0                  0
    96          4801                  0                  0
    97          4851                  0                  0
    98          4901                  0                  0
    99          4951                  0                  0
   100          5001                  0                  0
   101          5051                  0                  0
   102          5101                  0                  0
gamultiobj stopped because the average change in the spread of Pareto solutions is less than
↪  options.FunctionTolerance.
```

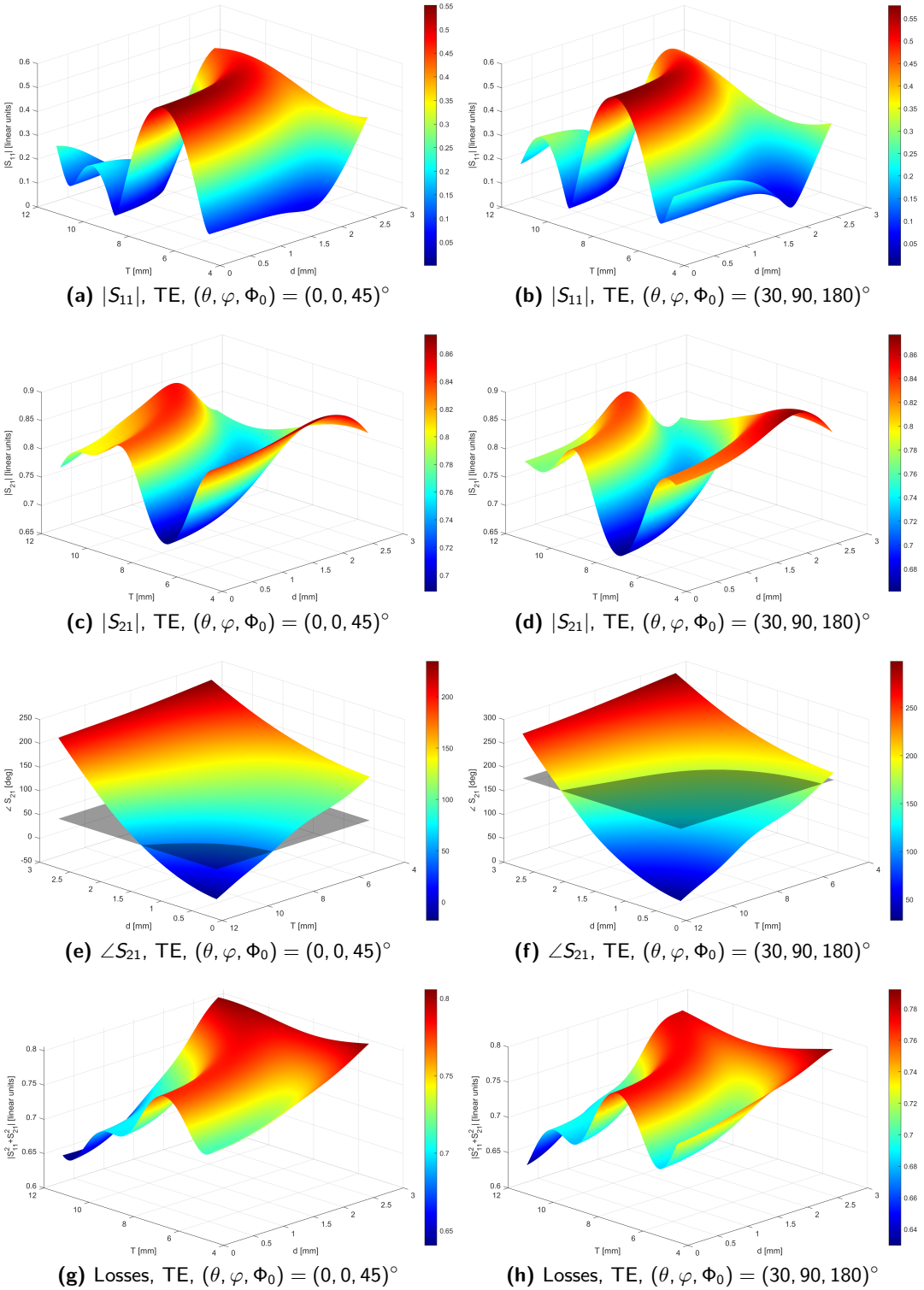
It is possible to notice the following facts:

- As Figure 3.24 shows, the maxima of $|S_{21}|$ roughly correspond to the minima of $|S_{11}|$. Therefore, the different optimization projects, namely `'p'`, `'r'` and `'c'`, provide very similar outputs, with the results even being identical in some of the explored angle configurations.

  This fact will be even more evident when analyzing the radiation pattern of the TA designed with the different methods; see Section 4.4.3 and, in particular, Figure 4.20 for further details on this aspect.

  Due to the algorithmic differences in the way MATLAB® implements `ga` and `gamultiobj`, the multi-objective optimization (`'c'`) usually reaches convergence faster than the single-objective methods (`'p'` and `'r'`), which is the reason why the former is chosen as the standard for TA and SES design in the rest of this work.

- When a small phase error is accepted, the performance of the cell can be significantly improved with respect to carrying out the optimization with an equality constraint. In fact, the results shown in Tables 3.3-3.8 show that



(a) $|S_{11}|$, TE, $(\theta, \varphi, \Phi_0) = (0, 0, 45)^\circ$

(b) $|S_{11}|$, TE, $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$

(c) $|S_{21}|$, TE, $(\theta, \varphi, \Phi_0) = (0, 0, 45)^\circ$

(d) $|S_{21}|$, TE, $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$

(e) $\angle S_{21}$, TE, $(\theta, \varphi, \Phi_0) = (0, 0, 45)^\circ$

(f) $\angle S_{21}$, TE, $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$

(g) Losses, TE, $(\theta, \varphi, \Phi_0) = (0, 0, 45)^\circ$

(h) Losses, TE, $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$

Figure 3.24: Results of the two-parameter analysis of UC-3 for two sets of angles $(\theta, \varphi, \Phi_0)$.

- Depending the on the chosen set of angles $(\theta, \varphi, \Phi_0)$, the results of the optimization can differ significantly: with the configuration $(\theta, \varphi, \Phi_0) = (0, 0, 45)^\circ$, it is possible to obtain both $|S_{11}|$ close to $-30\,\text{dB}$ and $|S_{21}| \approx -1.5\,\text{dB}$; on the other hand, with $(\theta, \varphi, \Phi_0) = (30, 90, 180)^\circ$ the obtained results are slightly worse, especially as far as the reflection coefficient is concerned ($|S_{11}| \approx -20\,\text{dB}$).

  This issue, probably, is caused by the imposition on the value of $\angle S_{21}$: in fact, even if a tolerance on Φ_0 is accepted, for certain values of the incidence angles d and T may be constrained in a region where $|S_{11}|$ and $|S_{21}|$ are difficult to minimize.

  Although this fact does not affect the overall quality of TA and SES projects, it is important to keep it into account because it can make it more difficult to minimize the back lobe level (BLL) in the antenna radiation pattern.

3.5 Detected Problems

The analytical model developed and validated in this chapter is highly satisfactory and paves the way for promising advancements in optimizing the performance of the unit cell and in the design of TA and SES, as the continuation of this work clearly demonstrates.

For the sake of completeness, however, it is necessary to point out that small inaccuracies appear in the curves of the reflection coefficient for high values of θ, especially when simulating the incidence of a TM-mode field.

Since the errors have been only detected when $|S_{11}| \ll -20\,\text{dB}$, they have been neglected in the subsequent design steps.

For a further development of the model, it is interesting to investigate the possible origin of these problems:

Material model. As said in Section 3.1.1, the adopted model for the description of the effective permittivities of the cells is developed in the quasi-static approximation and completely ignores the anisotropic behavior of the structure; furthermore, the dependency of the material dielectric constant on frequency is neglected, thus introducing possible inaccuracies.

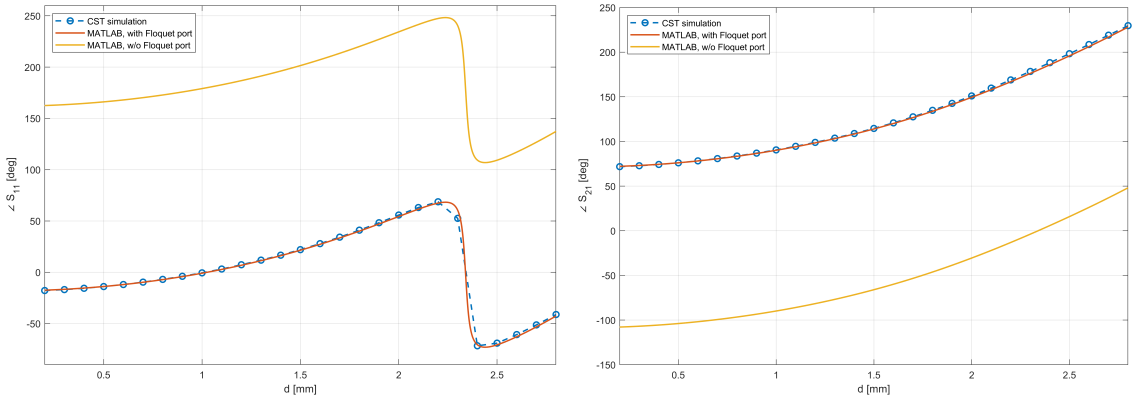
Single-mode excitation. The S-parameter calculation has been carried out under the hypothesis that only the fundamental mode (TE_{00} or TM_{00}) is propagating in the structure, while all the other Floquet modes are evanescent. However, as the incidence angle θ increases, resonances introduced by higher-order modes start playing a non-negligible role.

Furthermore, the assumptions made during the analysis of the UC model must be properly addressed when dealing with the design of the complete array:

Floquet ports. As pointed out in Equation (3.28), the Floquet ports used to excite the structure are at distance $\lambda_0/4$ from the boundaries of the cell, thus modifying the phase of the reflection and transmission coefficients.

This phenomoenon is clearly visible from the plots in Figure 3.25, representing $\angle S_{11}$ and $\angle S_{21}$ of UC-1 as function of the hole size: the analytical curves are matched with the simulation results (blue dashed curves) only if the presence of the Floquet ports is taken into account in the theoretical calculations.

This is the reason why, in the equivalent circuits shown in Figures 3.4 and 3.5, the ports are modeled by two additional TL sections at the edges of the structures, characterized by $\epsilon_r = 1$, $\tan\delta = 0$ (free space) and $d = 0$ (no hole).



(a) $d \mapsto \angle S_{11}(d)$ **(b)** $d \mapsto \angle S_{21}(d)$

Figure 3.25: Effect of the amplitude of Floquet ports on the phase of S_{11} and S_{21}.

This aspect is crucial for a successful TA design because, if the contribution of the Floquet ports to the phase is not kept into account, it is impossible to provide the correct phase compensation to the various cells.

Aperiodicities and finite size. Both the theoretical analysis and CST simulations have been carried out assuming to be dealing with an infinite and fully periodic structure, which is the necessary hypothesis to apply Floquet's theorem.

This assumption, however, is not true: in fact, an actual array is obviously not infinite and its cells are characterized by variable hole size and, possibly, variable height. For this reason, design errors may be introduced, thus leading to slightly imprecise results.

Chapter 4

TA Design and Optimization

In this chapter, the most promising unit cell models previously developed and validated are exploited for the design and simulation of several Transmitarrays (TA) structures, implemented as planar arrays of cells illuminated by a known feed source.

The organization of the chapter is briefly summarized here:

1. In Section 4.1, the model of the feed horn used to illuminate the array is presented, together with its far-field radiation pattern;

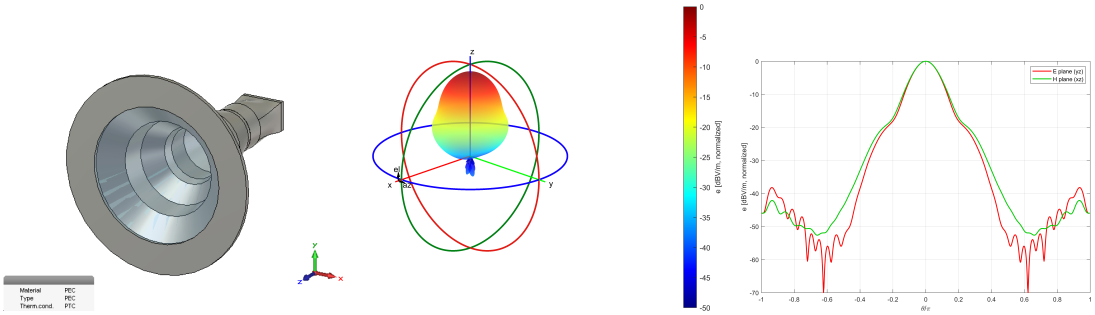
2. In Section 4.2, a simplistic model for the design of TAs is presented, which allows to calculate the phase compensation required at each unit cell (UC) in order to steer the main beam in a desired direction and to predict the far-field radiation pattern of the antenna;

3. In Section 4.3, a more accurate model based on the Physical Optics (PO) approximation is presented, which allows to take into account the actual geometry of the UCs and their interaction with the feed horn;

4. In Section 4.4, the design flow for the simulation of a Transmitarrays in CST MW Studio® is presented, the results of the simulations performed on several structures are shown, and the performance of the TAs is compared with the theoretical predictions obtained with the PO-based model:

   - At first, in Sections 4.4.1 and 4.4.2 the design of **broadside arrays** is presented, where the main beam is steered in the same direction of the incident field generated by the feed horn. These basic results simply aim at veryfying the correctness of the design flow and the accuracy of the PO theoretical model.
   - Then, in Section 4.4.3 the results obtained with **different design approaches** for the most promising broadside structures are compared, showing the impact of accepting a given error in the phase compensation and highlighting the differences in the far-field radiation pattern when a single or multi-objective optimization of the reflection and transmission coefficient of each cell is performed.

     In particular, it is shown how a careful UC design can lead to a significant improvement of the side lobe level (SLL) and back lobe level (BLL) of the antenna.
   - Finally, in Section 4.4.4, the design model is extended to **non-broadside arrays**, where the main beam is steered in a different direction with respect to the direction of incidence. In addition, the performance of one of the designed antennas is investigated when **mechanical beam steering** is applied, i.e., when the antenna is rotated around its main axis.



(a) Horn structure **(b)** Horn far-field (3D) **(c)** Horn far-field (1D)

Figure 4.1: Feed horn structure and radiation pattern.

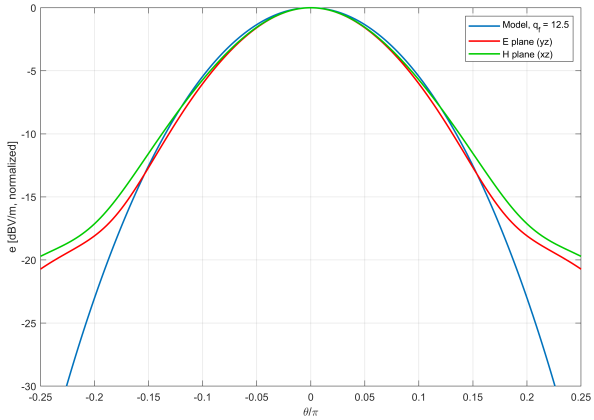


Figure 4.2: Radiation pattern of the theoretical cosine source for $n = 12.5$, along with the radiation pattern of the employed feed horn in the E and H planes.

4.1 Feed Horn

The focal source of a Transmitarray is expressed by a **single point**, which typically represents the phase center of the feed.

The source can be either a theoretical point-like element or a real structure, such as a feed horn, which is a common choice in practice. The feed horn is a device that converts the electromagnetic energy from a waveguide into a propagating wave in free space, and it is characterized by a well-defined radiation pattern [21].

In this work, the feed horn is modeled as a rectangular waveguide with a circular aperture, excited by a linearly-polarized TE_{00} mode. The horn structure is shown in Figure 4.1a, while the far-field radiation pattern in 3D and 1D (E and H planes) is shown in Figures 4.1b and 4.1c, respectively.

The design, prototyping and validation of the horn is thoroughly shown in [7]: the antenna, which has overall dimensions 52 mm × 52 mm × 53.7 mm, is manufactured with the Selective Laser Melting (SLM) process.

The chosen material is $AlSi_{10}Mg$, a specific aluminum alloy widely exploited in metal 3D printing due to its excellent strength, good thermal conductivity, corrosion resistance and good casting properties, which make it suitable for producing parts with thin walls and complex geometries.

The structure is designed to operate in the K_a-band, with a center frequency of 30 GHz. To model its radiation pattern, a theoretical cosine source is exploited, similarly to the approach

presented in [21]. In particular, the radiation intensity $(\theta, \varphi) \mapsto U_f(\theta, \varphi)$ is defined as follows:

$$U_f(\theta, \varphi) = \frac{q_f + 1}{2\pi} \cos^{q_f}(\theta). \tag{4.1}$$

Simulating the structure in CST Microwave Studio, the value of q_f is set to 12.5, which corresponds to a half-power beamwidth of approximately

$$\text{HPBW}_E \approx 25.4^\circ, \quad \text{HPBW}_H \approx 25.1^\circ, \tag{4.2}$$

respectively in the E and H planes.

The position of the phase center at the design frequency $f_0 = 30\,\text{GHz}$, instead, is at distance

$$d_f = 37.2\,\text{mm} \tag{4.3}$$

from the horn mouth, which must be kept into account when preparing the CST simulations in order to correctly set the relative position of the feed with respect to the TA.

Finally, the maximum gain at $f_0 = 30\,\text{GHz}$ is approximately

$$\mathcal{G} \approx 17\,\text{dB}. \tag{4.4}$$

It is possible to notice that the radiation pattern of the theoretical source is very similar to the one of the feed horn only for small values of θ, while for large angles the two patterns diverge. This is due to the fact that the theoretical source is a point-like element, while the feed horn has a finite aperture and therefore its radiation pattern is not isotropic. This discrepancy, however, does not affect the overall quality of the design, since the maximum value of the scan angle θ_f is, for geometrical reasons, no larger than 45°.

4.2 Simplistic Model

In this Section, a detailed analysis of the design of Transmitarray antennas is presented starting from the simplistic model explained in [1], that allows calculating the **phase compensation** required at each unit cell in order to steer the main beam in a desired direction.

Then, the radiation pattern of the antenna is predicted by means of the **array factor**, which is calculated as the superposition of the contributions of all the UCs, each one modeled as a point-like source.

The presented model is based on initial assumptions which make it partially incomplete and therefore not suitable for practical applications: for instance, the array factor provides information only on the main beam direction and the side lobes, but it does not predict the amplitude of the back lobe, thus giving no feedback on the quality of the optimization of the reflection coefficients of the UCs.

It is, however, useful to understand the basic principles of the design of TAs and to validate the more accurate model presented in Section 4.3.

First of all, the geometric structure of a Transmitarray is shown in Figure 4.3, where the reference frame $Oxyz$ introduced in Section 3.1 is shown and the characteristic size D and the feed focal length F are defined.

As partially anticipated at the end of the previous paragraph, F represents the distance between the center of the array O and the *phase center* of the feed horn; therefore, the distance of the horn mouth from O is

$$F_{\text{tot}} = F + d_f. \tag{4.5}$$

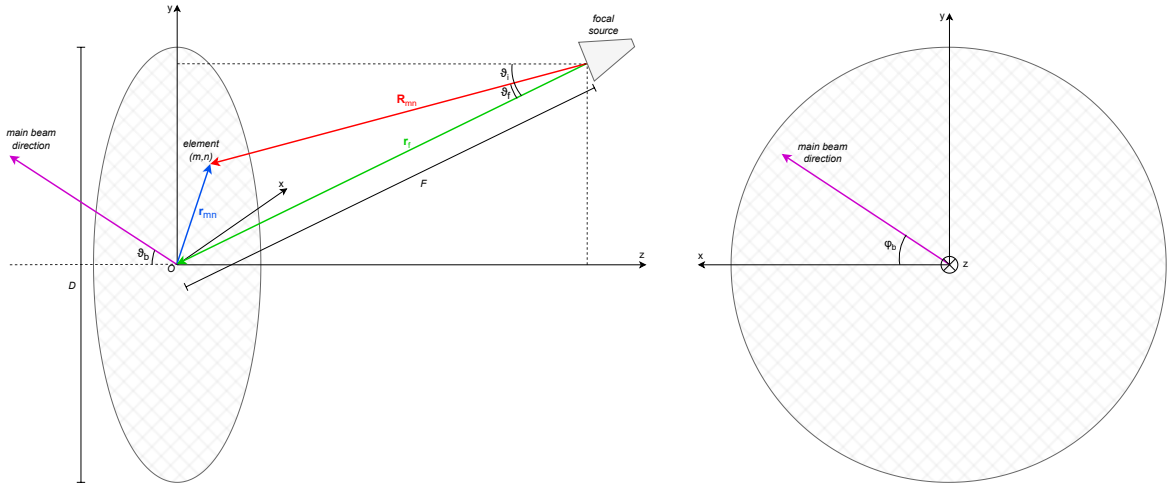


Figure 4.3: Geometric structure of a Transmitarray with main diameter D and focal length F.

The analysis of a TA is based on the assumption that the antenna elements are in the far-field region of the feed source. Under this approximation, the field generated by the feed and incident on each UC can be modeled as a spherical wave, that is, a plane wave with a phase proportional to the distance from the feed source [1].

4.2.1 Design Parameters

Before entering into the details of the design, it is useful to define some parameters that will be used in the following. All these quantities can be graphically observed in Figure 4.3 and are briefly listed below:

- F/D: focal length to antenna size ratio.
- M: number of UCs along each side of the array:

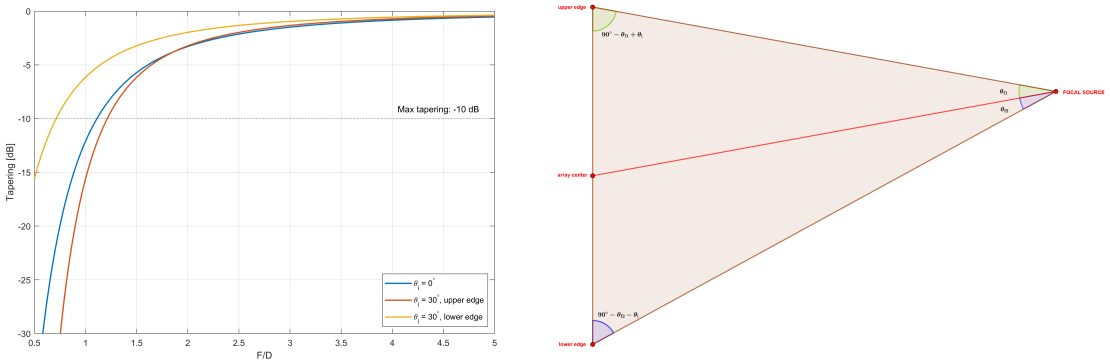
$$M := \frac{D}{W}. \tag{4.6}$$

- θ_i: angle formed between the feed horn axis and the z axis.
- $(\theta_\mathrm{b}, \varphi_\mathrm{b})$: angles of the main beam direction with respect to the z axis and the x axis, respectively.
- $\mathbf{r}_{mn}$: position vector of the mn^{th} UC element in the array with respect to the origin O of the reference system, coincident with the center of the array:

$$\mathbf{r}_{mn} = x_{mn}\,\widehat{\mathbf{x}} + y_{mn}\,\widehat{\mathbf{y}} = W\left[\left(m - \frac{M+1}{2}\right)\widehat{\mathbf{x}} + \left(n - \frac{M+1}{2}\right)\widehat{\mathbf{y}}\right]. \tag{4.7}$$

The magnitude of this vector can be calculated with the following formula:

$$r_{mn} = \|\mathbf{r}_{mn}\| = \sqrt{x_{mn}^2 + y_{mn}^2} = W\sqrt{\left(m - \frac{M+1}{2}\right)^2 + \left(n - \frac{M+1}{2}\right)^2}. \tag{4.8}$$



(a) $x \mapsto \mathcal{T}(x)$ **(b)** Geometry

Figure 4.4: Tapering of a Transmitarray antenna with 30×30 cells as function of the F/D ratio and geometrical overview of the system with variable θ_i.

- $\mathbf{r}_f$: position vector of the phase center of the feed horn with respect to the origin O of the reference system. If the focal point is centered with respect to the array, then:

$$\mathbf{r}_f = -F\,\hat{\mathbf{z}}. \tag{4.9}$$

  In the general case, instead, $\mathbf{r}_f$ is calculated as follows:

$$\mathbf{r}_f = -\left(F \sin\theta_i\,\hat{\mathbf{y}} + F\cos\theta_i\,\hat{\mathbf{z}}\right), \tag{4.10}$$

- $\mathbf{R}_{mn}$: distance between the mn^{th} UC and the phase center of the feed horn:

$$\mathbf{R}_{mn} = \mathbf{r}_{mn} + \mathbf{r}_f, \tag{4.11}$$

  and

$$R_{mn} = \sqrt{F^2 + r_{mn}^2} \overset{(4.8)}{=} \sqrt{F^2 + W^2\left[\left(m - \frac{M+1}{2}\right)^2 + \left(n - \frac{M+1}{2}\right)^2\right]} \tag{4.12}$$

- $\theta_{f,mn}$: angle formed between $\mathbf{R}_{mn}$ and $\mathbf{r}_f$ in the feed horn reference system:

$$r_{mn} = R_{mn}\sin\theta_{f,mn} \implies \theta_{f,mn} := \sin^{-1}\left(\frac{r_{mn}}{R_{mn}}\right). \tag{4.13}$$

The parameter $\theta_{f,mn}$ is extremely important since it is used to calculate the **tapering** of the TA, which is defined as the ratio between the amplitude of the incident field at the array edge and the amplitude at the center of the structure:

$$\mathcal{T} = \frac{U_f(\text{edge})}{U_f(\text{center})} = \frac{\cos^{q_f}\left(\theta_{f,\text{edge}}\right)}{\cos^{q_f}\left(\theta_{f,\text{center}}\right)}. \tag{4.14}$$

Assuming that the feed horn axis is aligned with the z axis ($\theta_i = 0$), the angles $\theta_{f,\text{center}}$ and $\theta_{f,\text{edge}}$ are defined as follows:

$$\theta_{f,\text{center}} = 0 \quad \text{and} \quad \theta_{f,\text{edge}} = \tan^{-1}\left(\frac{D/2}{F}\right) = \tan^{-1}\left(\frac{1}{2(F/D)}\right). \tag{4.15}$$

Therefore, it is possible to express the tapering as a function of the ratio $F/D =: x$, obtaining an expression whose graphical representation is shown by the blue curve in Figure 4.4a:

$$\mathcal{T}(x) = cos^{q_f}\left[\tan^{-1}\left(\frac{1}{2x}\right)\right]. \tag{4.16}$$

In the general case $\theta_i \neq 0$, the definition of $\theta_{f,edge}$ is a bit more complicated, as Figure 4.4b shows:

$$\frac{F}{\sin\left(90^\circ - (\theta_{f,edge} \pm \theta_i)\right)} = \frac{D/2}{\sin\left(\theta_{f,edge}\right)} \implies \frac{\cos\left(\theta_{f,edge} \pm \theta_i\right)}{\sin\left(\theta_{f,edge}\right)} = 2x, \tag{4.17}$$

where the sign $\pm$ depends whether the lower or upper edge of the array is considered, respectively.

Solving Equation (4.17) numerically, one finds the orange and yellow curves shown in Figure 4.4a: it is immediate and intuitive to see that the upper and lower tapering are different and the tapering is larger for larger values of θ_i, since the feed horn is more distant from the lower edge and therefore the amplitude of the incident field is smaller.

The tapering is a crucial parameter for the design of TAs, since it strongly affects both the side lobe level and the back lobe level of the antenna.

In particular, a larger tapering leads to a lower SLL and BLL, but it also requires a larger number of UCs to achieve the desired phase compensation.

A reasonable trade-off must therefore be found: in this work, the maximum acceptable tapering, which sets the maximum focal distance, is set to

$$\mathcal{T}_{max} = -10\,\text{dB}. \tag{4.18}$$

4.2.2 Calculation of the Array Factor

The main goal of the design is to compensate the phase of the incident field at each UC, so that the total field radiated by the antenna has a desired phase distribution. This is achieved by introducing a suitable *phase shift* at each UC playing on the value of the transmission coefficient $S_{21,mn}$ of the antenna elements.

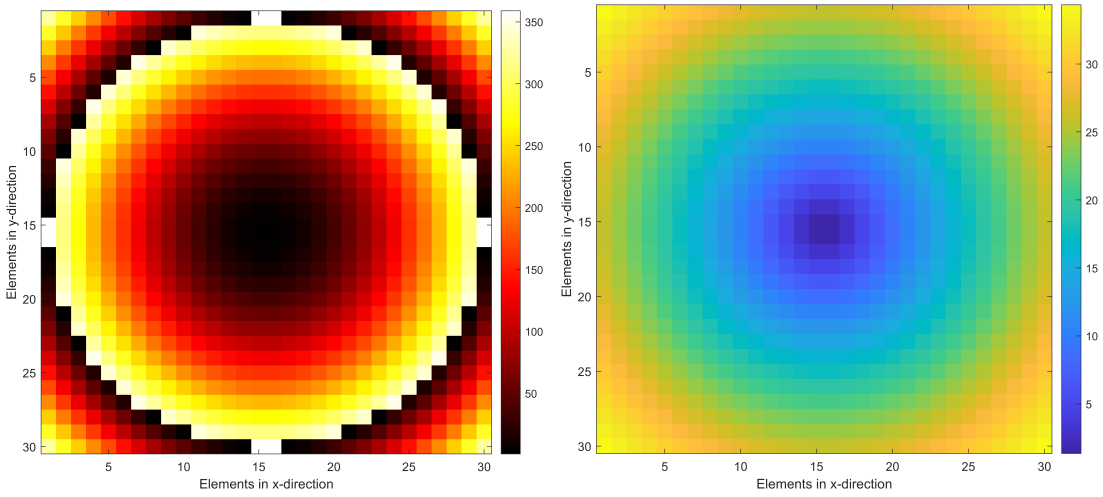
The required **phase compensation** at the mn^{th} cell, denoted with Ψ_{mn} and calculated as the difference between the phase of the incident field and the phase of the desired field [1], is given by:

$$\Psi_{mn} = \Psi_0 + k_0\left(R_{mn} - \mathbf{r}_{mn}\cdot\hat{\mathbf{u}}_O\right) \equiv \Psi_0 + \frac{2\pi}{\lambda_0}\left(R_{mn} - \mathbf{r}_{mn}\cdot\hat{\mathbf{u}}_O\right), \tag{4.19}$$

where:

- Ψ_0 is a reference phase term whose presence indicates that a relative transmission phase rather that the absolute one is required for a correct design; for the sake of simplicity, it is set to zero in the following;
- $k_0 = \dfrac{2\pi}{\lambda_0}$ is the free-space propagation constant at the design frequency f_0;
- R_{mn} is the distance between the mn^{th} UC and the phase center of the feed horn, defined in Equation (4.12);
- $\hat{\mathbf{u}}_O$ is a unit vector pointing in the direction of the main beam:

$$\hat{\mathbf{u}}_O = \sin\theta_b\cos\varphi_b\,\hat{\mathbf{x}} + \sin\theta_b\sin\varphi_b\,\hat{\mathbf{y}} + \cos\varphi_b\,\hat{\mathbf{z}}. \tag{4.20}$$



(a) Phase distribution **(b)** Scan angle distribution

Figure 4.5: Phase map and scan angle distribution for a 30×30 TA with $F/D = 1$.

For a TA with the main beam in the BS direction, $\theta_\mathrm{b} = \varphi_\mathrm{b} = 0$; therefore:

$$\mathbf{r}_{mn} \cdot \hat{\mathbf{u}}_O = 0, \quad \forall m, n \in \{1, \dots, M\}, \tag{4.21}$$

and Equation (4.19) reduces to

$$\Psi_{mn} =: \Delta\Phi_\mathrm{sp} = \Psi_0 + \frac{2\pi}{\lambda_0} R_{mn}. \tag{4.22}$$

In the general case, instead, the total phase compensation at each UC will be:

$$\Psi_{mn} = \Delta\Phi_\mathrm{sp} - \mathbf{r}_{mn} \cdot \hat{\mathbf{u}}_O = \Delta\Phi_\mathrm{sp} - \frac{2\pi}{\lambda_0} \sin\theta_\mathrm{b} \left(x_{mn} \cos\varphi_\mathrm{b} + y_{mn} \sin\varphi_\mathrm{b}\right). \tag{4.23}$$

The phase and scan angle distributions $\{m, n\} \mapsto \Psi_{mn}$ and $\{m, n\} \mapsto \theta_{\mathrm{f},mn}$ for a 30×30 broadside TA with $F/D = 1$ are shown in Figures 4.5a and 4.5b, respectively.

Before going on with the analysis, let us also define the radial unit vector $\mathbf{u}(\theta, \varphi)$, coherently with Appendix A.1.1:

$$\mathbf{u}(\theta, \varphi) = \sin\theta \cos\varphi\, \hat{\mathbf{x}} + \sin\theta \sin\varphi\, \hat{\mathbf{y}} + \cos\theta\, \hat{\mathbf{z}}. \tag{4.24}$$

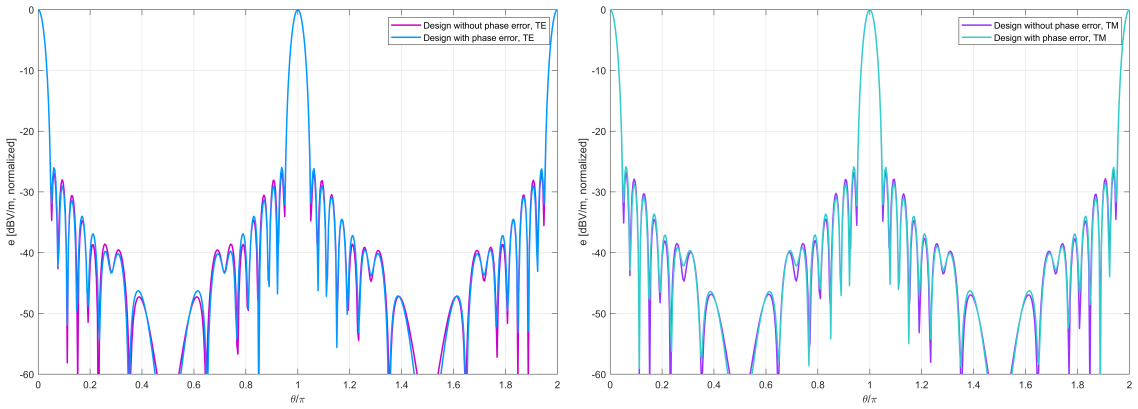
The projection of $\mathbf{r}_{mn}$ in the radial direction is thus given by

$$\mathbf{r}_{mn} \cdot \mathbf{u}(\theta, \varphi) = W \sin\theta \left[\left(m - \frac{M+1}{2}\right) \cos\varphi + \left(n - \frac{M+1}{2}\right) \sin\varphi\right]. \tag{4.25}$$

It is now possible to exploit the conventional array theory to calculate the **array factor** of a 2D planar array of $M \times M$ elements, which is defined as the superposition of the contributions of all the UCs, each one modeled as a point-like source.

In particular, the **element pattern vector function** A_{mn} and the **element current vector function** I_{mn} are defined as follows:

$$A_{mn}(\theta, \varphi) = \cos^{q_\mathrm{e}}(\theta) \exp\left[\mathrm{j} k_0 \left(\mathbf{r}_{mn} \cdot \mathbf{u}(\theta, \varphi)\right)\right] \tag{4.26a}$$



(a) TE mode **(b)** TM mode

Figure 4.6: Array factor of a 30×30 square TA with $F/D = 1$ according to the simplistic model developed in Section 4.2.

$$I_{mn} = \frac{\exp\left(-\mathrm{j}k_0 R_{mn}\right)}{R_{mn}} \cos^{q_\mathrm{f}}\left(\theta_{\mathrm{f},mn}\right) S_{21,mn} \tag{4.26b}$$

To simplify the analysis, both A_{mn} and I_{mn} are assumed to be independent of the azimuth angle φ, which is a reasonable assumption for a symmetric array, and they are modeled as scalar functions of the polar angle θ only.

In particular:

- $A_{mn}(\theta, \varphi) = A_{mn}(\theta)$, which describes the radiation pattern of the mn^{th} UC in the direction (θ, φ), is obtained with a cosine q_e model, where $q_\mathrm{e} \approx 1$ can be tuned to control the beamwidth of the antenna;
- I_{mn} is the current flowing through the mn^{th} UC, which is proportional to the transmission coefficient $S_{21,mn}$ of the element and to the cosine q_f model of the feed horn radiation pattern.

At this point, the field radiated by each element is

$$\mathcal{E}_{mn}(\theta, \varphi) \simeq A_{mn}(\theta, \varphi) I_{mn}. \tag{4.27}$$

In conclusion, the **total field** radiated by the antenna can be expressed as

$$\mathcal{E}(\theta, \varphi) = \sum_{m=1}^{M} \sum_{n=1}^{M} \mathcal{E}_{mn}(\theta, \varphi) \simeq \sum_{m=1}^{M} \sum_{n=1}^{M} A_{mn}(\theta, \varphi) I_{mn} \tag{4.28}$$

The MATLAB® to implement this model is reported in Appendix B.8.1 and the results obtained for a 30×30 broadside square TA with $F/D = 1$ are shown in Figures 4.6a and 4.6b, where the array factor is calculated for both TE and TM modes exploiting the two UC optimization methods developed in the previous chapter (without/with phase error).

Since, as said before, the quantities appearing in the array factor have no azimuthal dependence, the radiation pattern is shown only in the E plane ($\varphi = 0$, yz plane); the results in the H plane ($\varphi = 90°$, xz plane) are, at least theoretically, exactly the same.

It is possible to notice that the main beam is steered in the desired direction, which is the BS direction in this case, and that the side lobes are present but very low, as expected from such a basic modeling.

However, the back lobe is not predicted by the model, which is a limitation of this simplistic approach. To overcome this issue, a more accurate model based on the Physical Optics (PO) approximation is presented in the next Section.

4.3 Physical Optics Model

This approach has already been tested to model the interaction of an incident field with Reflectarrays [19], metascreens [37], PCB-based TA antennas [21] and parallel-plate lens antennas [34], proving to be successful and accurate in a variety of scenarios.

In this work, the same approach is applied to the design of dielectric-only Transmitarray antennas, with the goal of providing a more accurate model that can predict the back lobe level and the amplitude of the side lobes, which are not considered in the simplistic model presented in Section 4.2.

The geometry of the model, depicted in Figure 4.3, is the same as the one used in the simplistic model and the same parameters defined in Section 4.2.1 are used.

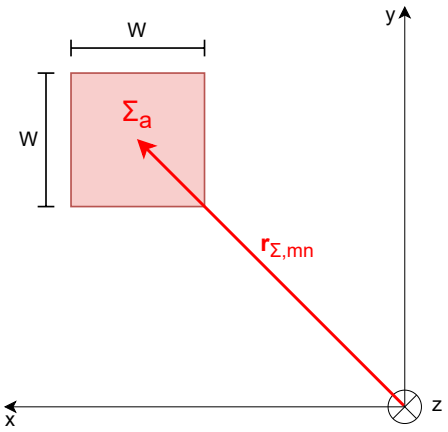


Figure 4.7: Geometry of the mn^{th} UC aperture Σ_a, defined as a square of side W centered at position $\mathbf{r}_{\Sigma,mn}$.

The main difference with respect to the previous model is that the UCs are not considered as point-like sources, but rather as extended structures that interact with the incident field. In particular, let us denote with Σ_a the surface of the mn^{th} UC, which a square of side W centered at position $\mathbf{r}_{\Sigma,mn}$, whose coordinates can be expressed in this way:

$$x \in \left[-\frac{W}{2}, \frac{W}{2}\right] + x_{mn} \implies x \in W\left[m - \frac{M+2}{2}, m - \frac{M}{2}\right] \subset \mathbb{R} \tag{4.29a}$$

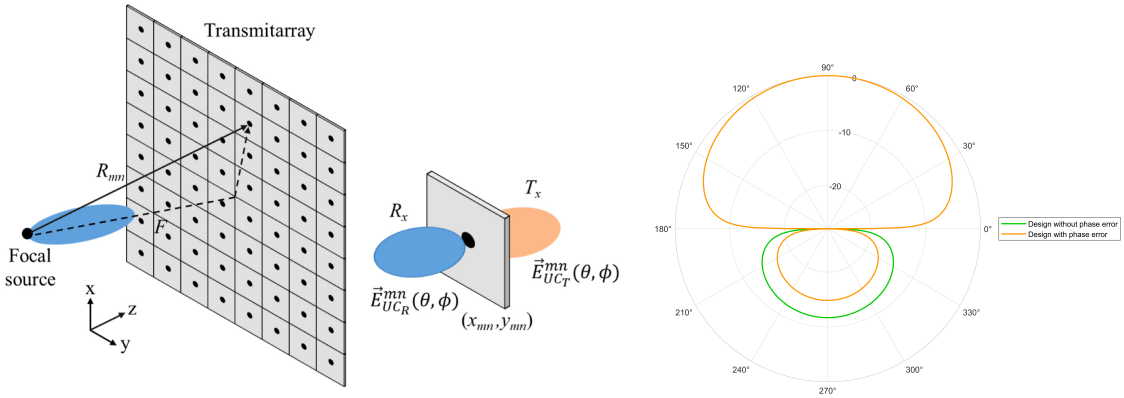
$$y \in \left[-\frac{W}{2}, \frac{W}{2}\right] + y_{mn} \implies y \in W\left[n - \frac{M+2}{2}, n - \frac{M}{2}\right] \subset \mathbb{R} \tag{4.29b}$$

$$\mathbf{r}_{\Sigma,mn} = x\,\hat{\mathbf{x}} + y\,\hat{\mathbf{y}} \implies \mathbf{r}_{\Sigma,mn} \cdot \mathbf{u}(\theta,\varphi) = x\sin\theta\cos\varphi + y\sin\theta\sin\varphi. \tag{4.30}$$

As a consequence, the surface of the aperture $\Sigma_a \subset \mathbb{R}^2$ is given by

$$\Sigma_a = W\left[m - \frac{M+2}{2}, m - \frac{M}{2}\right] \times W\left[n - \frac{M+2}{2}, n - \frac{M}{2}\right] \subset \mathbb{R}^2. \tag{4.31}$$

A schematic representation of the geometry presented above is shown in Figure 4.7, where the position of the mn^{th} unit cell aperture Σ_a is highlighted in red.



(a) General scheme **(b)** Polar coordinate plot of $e_{mn,\mathrm{UC}}$ vs θ

Figure 4.8: Scheme of the focal source and unit cell radiation patterns (a) and element field model in polar coordinates (b).

At this point, it is possible to introduce the **surface current density** $\theta_{\mathrm{f},mn} \mapsto \mathbf{J}_{\mathrm{s},mn}(\theta_{\mathrm{f},mn})$ at the mn^{th} UC, which is defined as the current flowing through the surface of the aperture Σ_{a} in the direction of the unit vector $\widehat{\boldsymbol{\theta}}$:

$$\mathbf{J}_{\mathrm{s},mn}(\theta_{\mathrm{f},mn}) = \cos^{q_{\mathrm{f}}}(\theta_{\mathrm{f},mn}) \, \frac{\exp -\mathrm{j}k_0 R_{mn}}{R_{mn}} \, \widehat{\boldsymbol{\theta}}. \tag{4.32}$$

Exploiting the definition of the surface current density, presented in Equation (4.32), and the **equivalence theorem**, presented in Appendix A.3, it is possible to calculate the **radiation integral** $\widetilde{\mathbf{J}}_{mn}(\theta,\varphi)$:

$$\widetilde{\mathbf{J}}_{mn}(\theta,\varphi) = \iint_{\Sigma_{\mathrm{a}}} \mathrm{d}x\,\mathrm{d}y \; S_{21,mn}\, \mathbf{J}_{\mathrm{s},mn}(\theta_{\mathrm{f}}) \, \exp\left[\mathrm{j}k_0 \left(\mathbf{r}_{\Sigma,mn} \cdot \mathbf{u}(\theta,\varphi)\right)\right], \tag{4.33}$$

where the exponential function is a phase term related to the Green function:

$$\exp\left[\mathrm{j}k_0 \left(\mathbf{r}_{\Sigma,mn} \cdot \mathbf{u}(\theta,\varphi)\right)\right] = \exp\left[\mathrm{j}k_0 \left(x \sin\theta\cos\varphi + y \sin\theta \sin\varphi\right)\right] \tag{4.34}$$

The components of the electric field along $\widehat{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\varphi}}$ directions, denoted with $e_{\theta,mn}(\theta,\varphi)$ and $e_{\varphi,mn}(\theta,\varphi)$, respectively, are thus given by

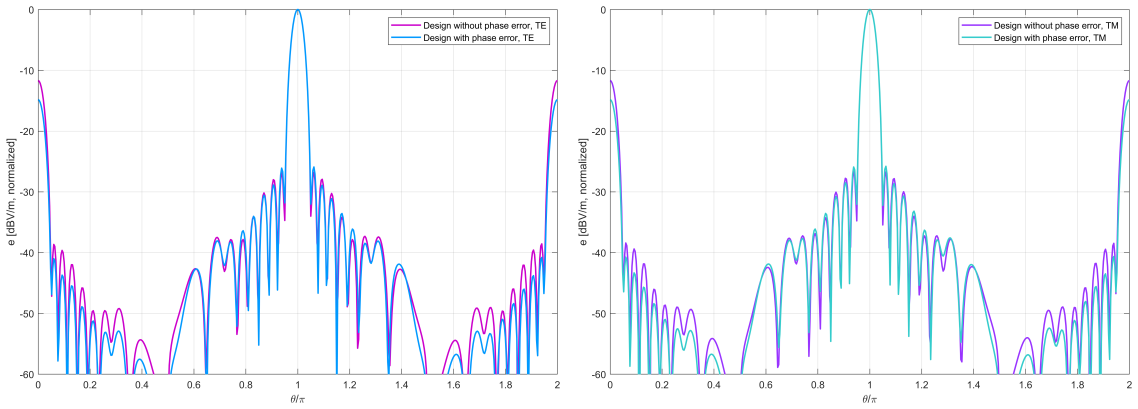
$$\begin{cases} \mathbf{e}_{\theta,mn}(\theta,\varphi) = e_{\theta,mn}(\theta,\varphi)\,\widehat{\boldsymbol{\theta}} = \left(-\mathrm{j}k_0 Z_0 \widetilde{\mathbf{J}}_{mn}(\theta,\varphi) \cdot \widehat{\boldsymbol{\theta}}\right)\widehat{\boldsymbol{\theta}} \\ \mathbf{e}_{\varphi,mn}(\theta,\varphi) = e_{\varphi,mn}(\theta,\varphi)\,\widehat{\boldsymbol{\varphi}} = \left(-\mathrm{j}k_0 Z_0 \widetilde{\mathbf{J}}_{mn}(\theta,\varphi) \cdot \widehat{\boldsymbol{\varphi}}\right)\widehat{\boldsymbol{\varphi}} = \mathbf{0} \end{cases} \tag{4.35}$$

meaning that the total electric field radiated by the mn^{th} UC can be expressed as

$$\mathbf{e}_{mn}(\theta,\varphi) = \mathbf{e}_{\theta,mn}(\theta,\varphi) + \mathbf{e}_{\varphi,mn}(\theta,\varphi) \equiv \mathbf{e}_{\theta,mn}(\theta,\varphi) = e_{\theta,mn}(\theta,\varphi)\,\widehat{\boldsymbol{\theta}}. \tag{4.36}$$

The fields radiated by the mn^{th} UC in reception and transmission side, shown in Figure 4.8, are denoted with $\mathbf{e}^{(\mathrm{R})}_{mn,\mathrm{UC}}(\theta,\varphi)$ and $\mathbf{e}^{(\mathrm{T})}_{mn,\mathrm{UC}}(\theta,\varphi)$ and can be calculated multiplying the cosine q_{e} model presented in the previous Section by the reflection and transmission coefficients $S_{11,mn}$ and $S_{21,mn}$, respectively:

$$\mathbf{e}^{(\mathrm{T})}_{mn,\mathrm{UC}}(\theta,\varphi) = |S_{21,mn}| \cos^{q_{\mathrm{e}}}(\theta)\,\widehat{\boldsymbol{\theta}}, \quad \mathbf{e}^{(\mathrm{R})}_{mn\mathrm{UC}}(\theta,\varphi) = |S_{11,mn}| \cos^{q_{\mathrm{e}}}(\theta)\,\widehat{\boldsymbol{\theta}} \tag{4.37}$$



(a) TE mode **(b)** TM mode

Figure 4.9: Array factor of a 30×30 square TA with $F/D = 1$ according to the PO-based model developed in Section 4.3.

Therefore, the actual electric field radiated by the cell is

$$\mathbf{e}_{mn,\mathrm{UC}}(\theta,\varphi) = \begin{cases} \mathbf{e}^{(\mathrm{T})}_{mn,\mathrm{UC}}(\theta,\varphi) & \theta \in \left[\dfrac{\pi}{2}, \dfrac{3\pi}{2}\right] \\ \mathbf{e}^{(\mathrm{R})}_{mn,\mathrm{UC}}(\theta,\varphi) & \theta \in \left[0, \dfrac{\pi}{2}\right) \cup \left(\dfrac{3\pi}{2}, 2\pi\right) \end{cases} \tag{4.38}$$

Finally, the total electric field radiated by the antenna can be expressed as the sum of the contributions of all the UCs, each one given by the Friis equation [15] between the focal source and the single antenna element:

$$\boldsymbol{\mathcal{E}}(\theta,\varphi) \equiv \boldsymbol{\mathcal{E}}_{\theta}(\theta,\varphi) = \left(\sum_{m=1}^{M} \sum_{n=1}^{M} \mathbf{e}_{mn}(\theta,\varphi) \cdot \mathbf{e}_{mn,\mathrm{UC}}(\theta,\varphi) \right) \widehat{\boldsymbol{\theta}}. \tag{4.39}$$

The MATLAB® code to implement this model is reported in Appendix B.8.2 and the results obtained for the same TA as the one analyzed in Section 4.2.2 are shown in Figures 4.9a and 4.9b. Unlike the simplistic model, the PO-based model is able to predict both the back lobe level and the amplitude of the side lobes, which makes it a more accurate and reliable tool for the design of TAs.

4.4 Validation of the Model

4.4.1 Design Flow

The validation of the model presented in Section 4.3 is carried out through a design flow consisting of **three steps**, thoroughly described in Sections 4.4.1.1, 4.4.1.2 and 4.4.1.3.

The first two steps are supported by several MATLAB® scripts, which must be called following a precise sequence: in order to make the design process more manageable and to reduce the risk of errors, the execution of the code is automatized thanks to a simple C++ project, reported in Appendix D. In particular, the class `Simulation` is defined, which contains three public methods:

- `editCode()`: this method allows the user to edit the MATLAB® scripts that will be executed in the next step, in order to set the desired design parameters, whose values are passed to the C++ program as command line arguments;

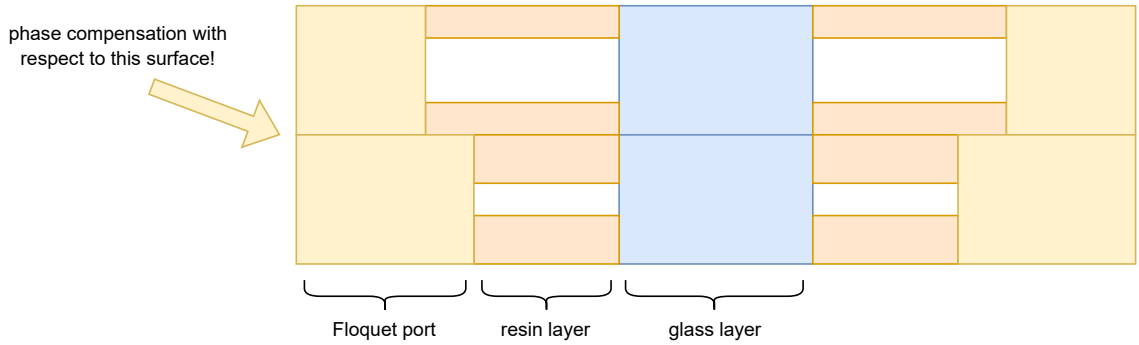


Figure 4.10: Scheme for TA design.

- `run()`: this method executes, in the correct order, the MATLAB® code that has been edited in the previous step, which is responsible for the optimization of the unit cells, the evaluation of the fer-field radiation pattern and the generation of the dxf files;
- `restoreCode()`: this method restores the original MATLAB® code, so that the user can run the program again with different parameters without having to edit the scripts manually.

4.4.1.1 Step 1: optimization of the unit cells

The first step of the design flow is the optimization of the unit cells, which is performed by means of a MATLAB® script that implements the optimization methods presented in Chapter 3.

The optimization is performed for both TE and TM modes, with and without phase error, in order to obtain the transmission coefficients $S_{21,mn}$ of the UCs that will be used in the next steps.

The MATLAB® function to achieve this task, namely `optimizer_TA_glass_sqh`, is reported in Appendix B.7.

It is worth pointing out that this script does not work only for the design of TAs, but also for the realization of SESs, where the feed horn is not present and the incident field is assumed to be a plane wave. This issue, however, will be addressed in Chapter 5.

The inputs arguments of the script are the following:

- `tol`: tolerance on phase error (0 if no phase error is considered, $\neq 0$ otherwise);
- `project`: type of optimization project, which can be either `p`, `r` or `c` depending whether the optimization is performed for $S_{21,mn}$, $S_{11,mn}$ or both;
- `f0`: design frequency, expressed in Hz;
- `eps_r, tan_delta`: dielectric constant and loss tangent of the resin layers; the glass parameters, instead, are already set in the functions that optimize each unit cell;
- `W`: UC periodicity, expressesd in m;
- `T, d`: ranges for thickness and hole size, expressed in mm;
- `H`: total height of the resin layer, expressed in mm, which keeps into account not only the thickness of the structure but also the additional distance introduced by the Floquet ports, as Figure 4.10 shows; in this case, it is set to 6 mm;

- `dim_grid`: resolution of the optimization grid, set to 600 in all the examples presented in this work;
- `F`: focal length, expressed in m;
- `M`: number of elements along one axis ($M \times M$ array)
- `theta_inc, phi_inc`: direction of incident in spherical coordinates, expressed in rad;
- `theta_beam, phi_beam`: desired beam direction, expressed in rad.
- `outfile_feed, outfile_plane`: names of the files where the results will be stored.

After calculating the phase map and scan angle distributions Ψ_{mn} and $\theta_{\mathrm{f},mn}$, the UC optimizer is run for each antenna element, choosing the correct routine depending on the values of the input arguments `project` and `tol`. The results are stored in the output files `outfile_feed` and `outfile_plane`, which in the end contain the quantities summarized in Table 4.1.

Table 4.1: Outputs of the TA optimizer.

Variable	Unit	Description
`Psi`	deg	Required phase delay per cell
`d_TE_opt`, `T_TE_opt`	mm	Optimized hole size and thickness for TE mode
`d_TM_opt`, `T_TM_opt`	mm	Optimized hole size and thickness for TM mode
`S11_TE_opt`, `S21_TE_opt`	–	Reflection and transmission coefficients for TE mode
`S11_TM_opt`, `S21_TM_opt`	–	Reflection and transmission coefficients for TM mode
`r`, `R`	m	Radial and slant distances from feed
`theta_f_feed`	deg	Scan angle in the RF of the feed horn

4.4.1.2 Step 2: generation of the DXF files

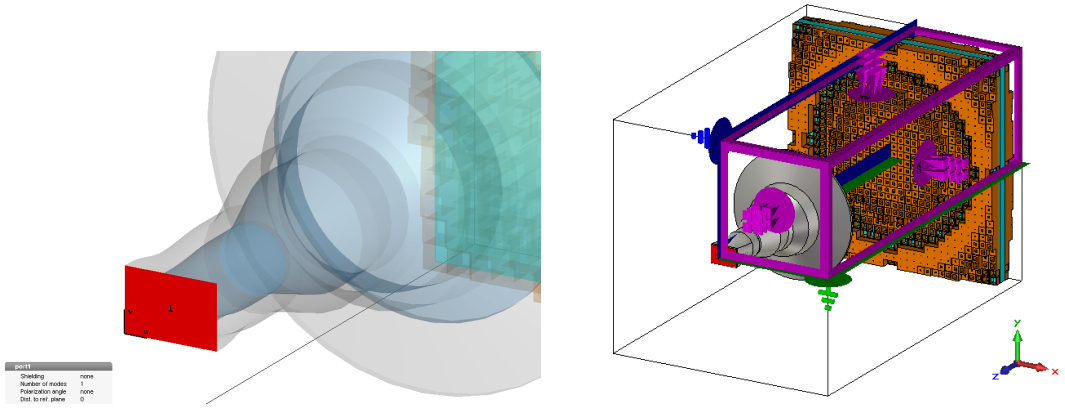
After the optimization has been performed and the distributions of the values of d and T have been obtained, the second step of the design flow is to generate the DXF files [13] that will be used in CST MW Studio® to simulate the antenna.

DXF stands for **Drawing Exchange Format**, a vector file type developed to enable seamless sharing of 2D and 3D CAD drawings across different software platforms.

Introduced in 1982, DXF was created to bridge the gap between various CAD programs. Its open-source nature made it a go-to format for interoperability long before formats like GIF or JPEG became mainstream. It is widely exploited for cross-platform collaboration as it can be read without problems by most CAD software in various fields, including architecture, engineering, and manufacturing. Furthermore, since this DXF files have a text-based format, they can be read and edited in plain text, which adds flexibility.

The MATLAB® functions to achieve this task are reported in Appendix C. The DXF files are generated bor both circular and square arrays, without and with phase error, TE and TM modes; for each structure, three files are generated:

1. `d.dxf` file: it stores the values of the hole size d for each unit cell, which are used to generate the holes in the 3D structure;
2. `WT.dxf` file: it stores the value of the thickness T (with + sign) for each unit cell, which is used to generate the variable heights in the *external* resin layer (the one that points towards the feed horn);



(a) Waveguide port **(b)** Symmetry planes

Figure 4.11: Setup for TA simulations in CST MW Studio®.

3. `WTm.dxf` file: it stores the value of the thickness T (with $-$ sign) for each unit cell, which is used to generate the variable heights in the *internal* resin layer (the one that points away from the feed horn).

4.4.1.3 Step 3: preparation of the CST simulation

Once the DXF files have been generated, the final step consists in preparing the full-wave CST simulation, to evaluate the far-field radiation pattern of the designed array.

In order to ensure the repeatibility of the performed simulations, the most important rules followed during the setup of the CST solver are here summarized:

- **Workflow and solver**: the chosen project template is *MW & RF & Optical/Antennas*, with the *Time Domain* solver.

  Contrary to the UC simulations performed in the previous chapter, the *Frequency Domain* solver is not used here because the mesh refinement would have an excessive computational cost and would dramatically increase the simulation time (several days for a single simulation instead of a few hours).

- **Background material**: the lower and upper distances along the three main axes are set to their default values:

$$\begin{cases} x_{\text{low}} = x_{\text{up}} = 0 \\ y_{\text{low}} = y_{\text{up}} = 0 \\ z_{\text{low}} = z_{\text{up}} = 0 \end{cases} \tag{4.40}$$

- **Boundary conditions**: all the boundaries are set to `open (add space)`;

- **Modeling of the structure**: the DXF files generated in the previous step are imported in CST MW Studio®, which automatically generates the 3D structure of the antenna thanks to a sequence of boolean operations;

- **Modeling of the feed horn**: the feed horn is imported as a CST subproject and, after being properly rotated and translated to place the phase center at the right distance from the surface of the array, it is excited by means of a waveguide port, shown in Figure 4.11a.

- **Symmetry planes**: to speed up the simulations, the optimization of the structure can be performed on a quarter or on half of the structure, depending whether the array is broadside or not. In this case, the symmetry planes are set as follows:

  1. **Broadside arrays**: symmetry planes on *xz* (magnetic) and *yz* (electric) planes, so that only one quadrant of the array is simulated (see Figure 4.11b);
  2. **Non-broadside arrays**: the symmetry planes are set on the *yz* (electric) plane only, so that only half of the array is simulated.

  In this way, not only the simulation time is strongly reduced, but also the computational cost of the optimization process performed in MATLAB® because only a fraction of the structure needs to be actually designed.

- **Setup solver**: the defalt hexahedral mesh should give enough accuracy for the simulations, but it is possible to refine it if necessary by increasing the number of cells per wavelenght. Before starting the simulation, the accuracy should be set at least to $-40\,\text{dB}$.

4.4.2 Broadside Arrays

To verify the correctness of the model presented in the previous paragraphs, the design and simulation of broadside arrays is initially addressed, imposing that the directions of both incident and scattered beams are orthogonal to the structure:

$$\theta_\text{i} = \varphi_\text{i} = \theta_\text{b} = \varphi_\text{b} \equiv 0^\circ. \tag{4.41}$$

In order to understand the differences between different shapes of the antenna and different optimization methods, **four TA structures** are initially considered:

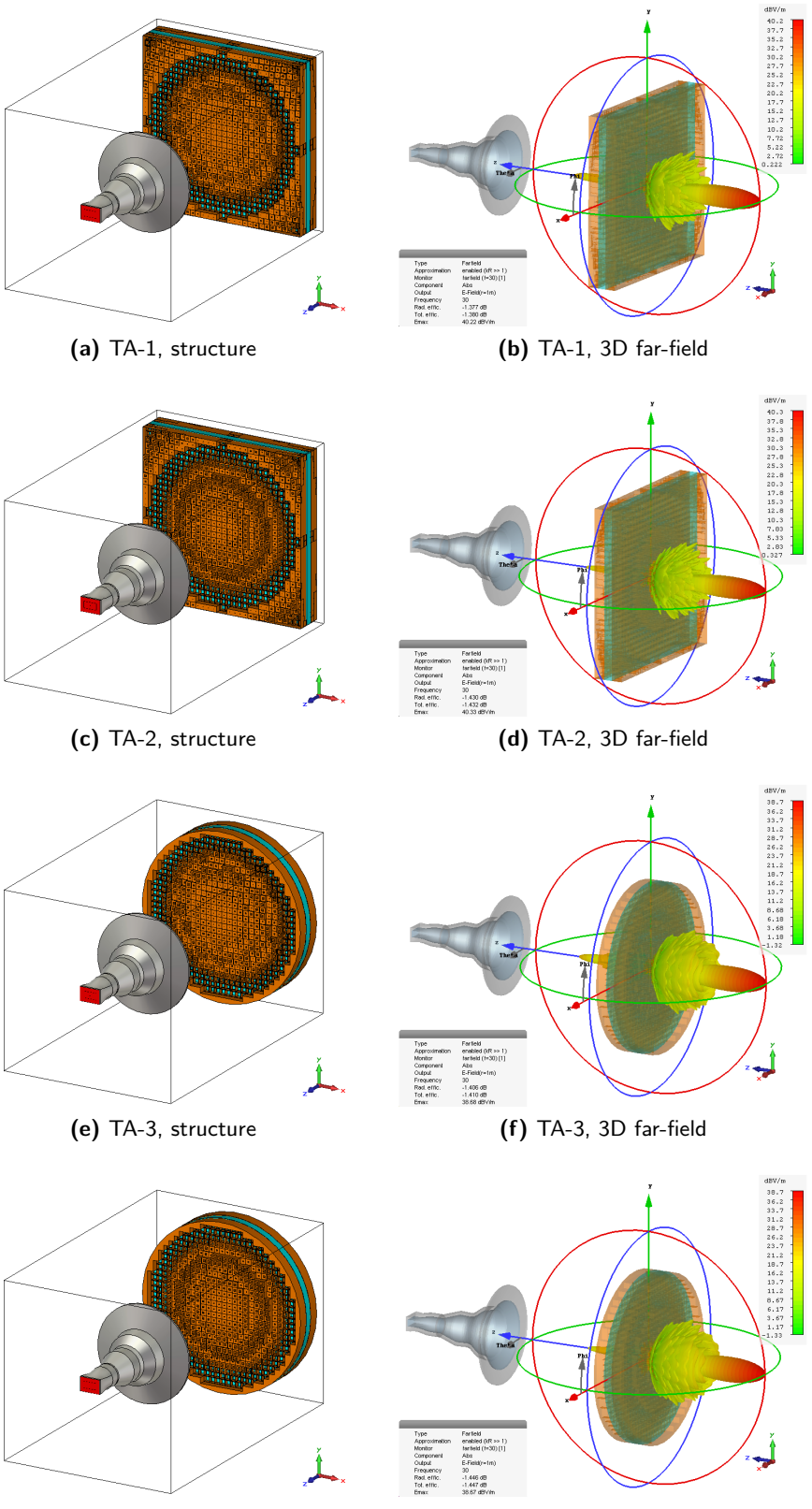
1. **TA-1**: square array with 30×30 cells, $F/D = 1$, design without phase error ($\texttt{tol} = 0$);
2. **TA-2**: square array with 30×30 cells, $F/D = 1$, design with phase error ($\texttt{tol} = 0.15$);
3. **TA-3**: circular array with 30×30 cells, $F/D = 1$, design without phase error ($\texttt{tol} = 0$);
4. **TA-4**: circular array with 30×30 cells, $F/D = 1$, design with phase error ($\texttt{tol} = 0.15$).

The setup of the simulations in CST MW Studio® is similar for all the antennas:

- **Frequency**: working band $\Delta f = 28 \div 32\,\text{GHz}$, central frequency $f_0 = 30\,\text{GHz}$;
- **Characteristic dimensions**: $F = D = MW = 30W = 9\lambda_0 = 90\,\text{mm}$;
- **Boundary conditions**: since the array is broadside, it is possible to set
  - a magnetic symmetry plane on the *xz* plane;
  - an electric symmetry plane on the *yz* plane.
- **Field monitors**: single-frequency far-field monitors at $f = 28, 29, 30, 31, 32\,\text{GHz}$.

The results of the simulations are reported in the following pages and their organization is the following:

- Figure 4.12 depicts the 3D structures (left) and far-field radiation patterns (right) of the designed TAs;



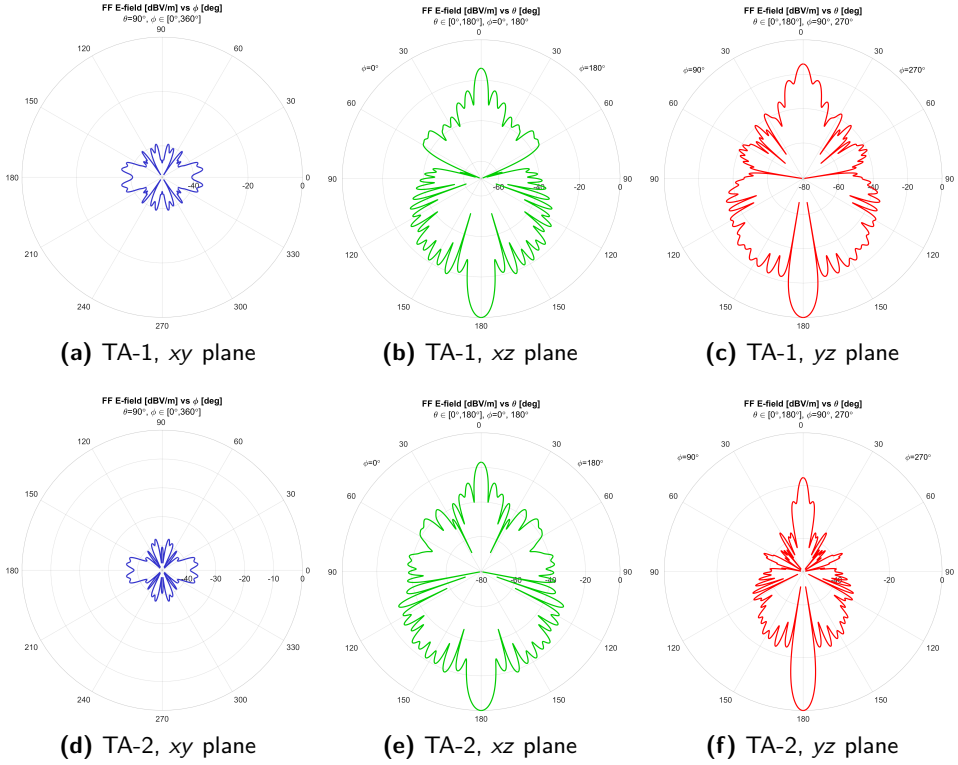
(a) TA-1, structure **(b)** TA-1, 3D far-field

(c) TA-2, structure **(d)** TA-2, 3D far-field

(e) TA-3, structure **(f)** TA-3, 3D far-field

(g) TA-4, structure **(h)** TA-4, 3D far-field

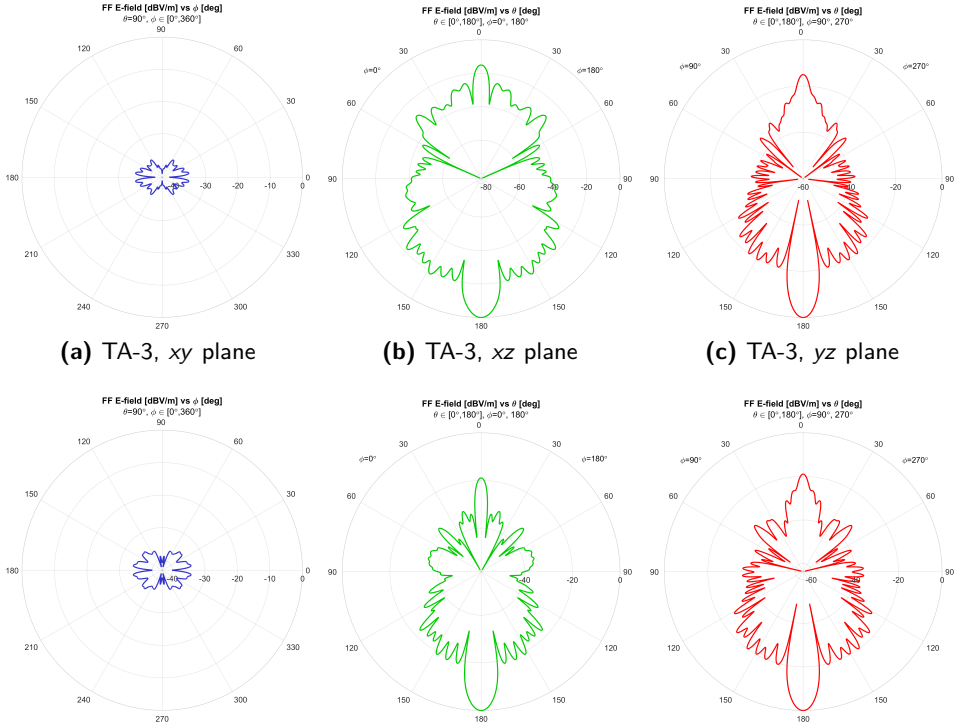
Figure 4.12: Simulated square and circular broadside TA. Left: 3D structures; right: 3D radiated far-fields (dB units).



(a) TA-1, *xy* plane **(b)** TA-1, *xz* plane **(c)** TA-1, *yz* plane

(d) TA-2, *xy* plane **(e)** TA-2, *xz* plane **(f)** TA-2, *yz* plane

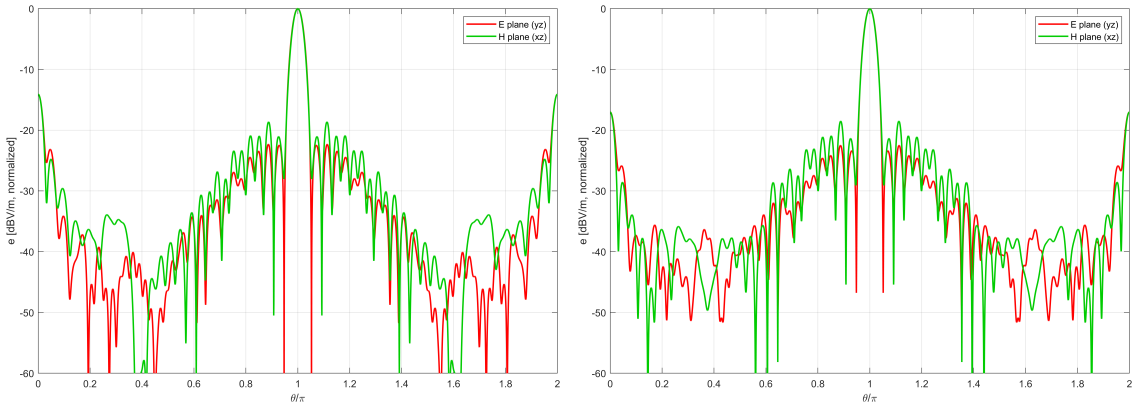
Figure 4.13: Radiated far field of TA-1 and TA-2 at center band (polar 1D view, dB units).



(a) TA-3, *xy* plane **(b)** TA-3, *xz* plane **(c)** TA-3, *yz* plane

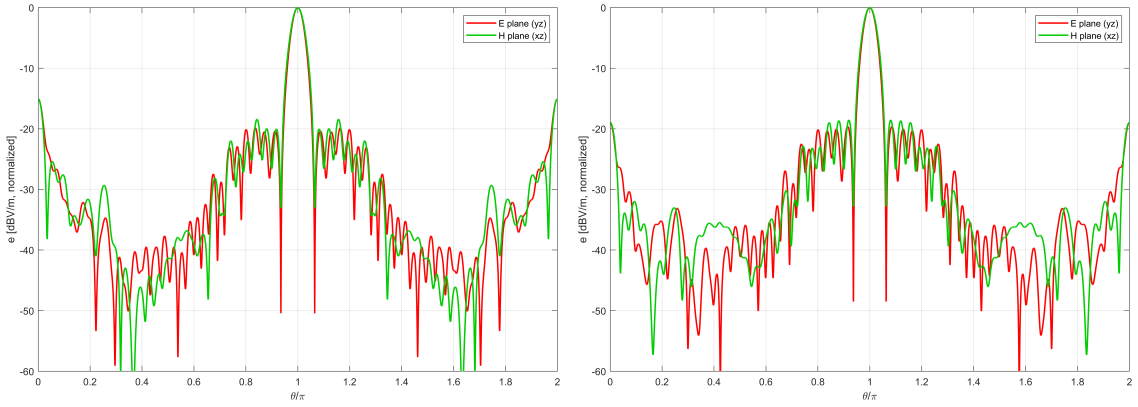
(d) TA-4, *xy* plane **(e)** TA-4, *xz* plane **(f)** TA-4, *yz* plane

Figure 4.14: Radiated far field of TA-3 and TA-4 at center band (polar 1D view, dB units).



(a) TA-1 **(b)** TA-2

Figure 4.15: Radiated far field of TA-1 and TA-2 on E (yz, red curves) and H (xz, green curves) planes.



(a) TA-3 **(b)** TA-4

Figure 4.16: Radiated far field of TA-3 and TA-4 on E (yz, red curves) and H (xz, green curves) planes.

Table 4.2: Performance of the designed BS TAs at centerband in terms of SLL, BLL, HPBW and efficiency in both E and H planes.

Structure	SLL_E [dB]	SLL_H [dB]	BLL_E [dB]	BLL_H [dB]	$HPBW_E$ [deg]	$HPBW_H$ [deg]	η_{tot} [dB]
TA-1	−22.32	−18.65	−14.08	−14.08	7.00	6.80	−1.380
TA-2	−22.50	−18.62	−17.02	−17.02	7.00	6.80	−1.432
TA-3	−19.84	−18.42	−15.07	−15.07	8.00	8.40	−1.410
TA-4	−19.64	−18.56	−18.96	−18.96	8.00	8.60	−1.447

- Figures 4.13 and 4.14 show the 1D far-field radiation patterns of the designed TAs in polar coordinates in the three-principal planes (xy in blue, xz in green and yz in red);
- Figures 4.15 and 4.16 show the 1D far-field radiation patterns of the designed TAs in cartesian coordinates in the E (xz) and H (yz) planes;
- Table 4.2 summarize the performance of the designed TAs in terms of side-lobe level (SLL), back-lobe level (BLL) and half-power beamwidth (HPBW) in both E and H planes. The total efficiency η_{tot} is reported as well.

Analyzing the results, it is possible to notice the following facts:

- Since broadside arrays inherently have a symmetric structure, which allow including two symmetry planes in the CST project, the results of the simulations are very similar in the E and H planes - theoretically they should be identical.

  This expectation is verified by the plots in Figures 4.15 and 4.16, where the E and H plane patterns overlap almost perfectly: in particular, BLL and HPBW are perfectly matched, while small differences arise in the values of SLL.

- The performance of square arrays (TA-1 and TA-2) is better than circular arrays (TA-3 and TA-4), especially when it comes to the value of the SLL [38], which is significantly lower for square structures.

  On the other hand, BLL improves when moving from square to circular design as circular arrays can offer more isotropic beam thanks to their radial symmetry.

  Based on this, only square-shaped arrays will be designed and simulated in the following in order to avoid issues deriving from reduced scanning range or increased sidelobes.

- Accepting a small error on the phase of the transmission coefficient of the UCs, in symbols $\angle S_{21,mn}$, the overall quality of the design improves: in fact, while the SLL is only subject to a small variation, the BLL significantly decreases, coherently with the enhancement of the UC performance noticed in Section 3.4.

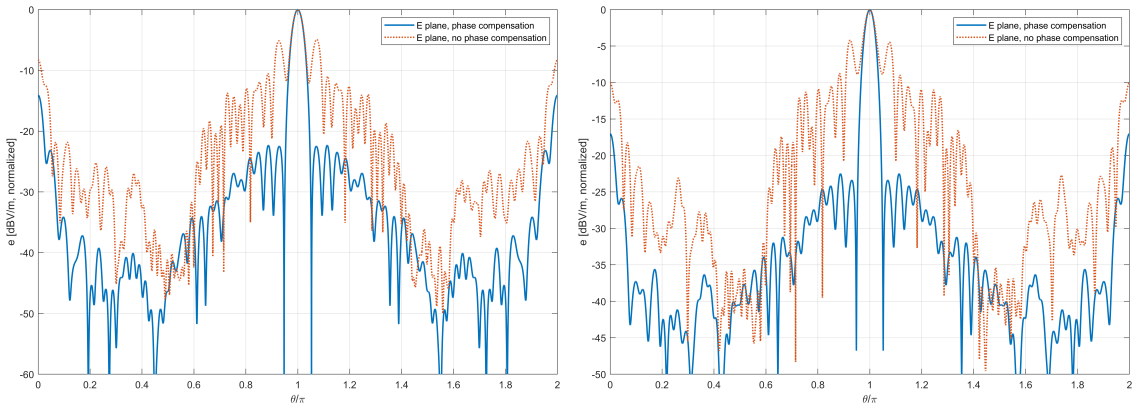
- While the side-lobe level, at least in the case of the square arrays, is well below $-20\,\text{dB}$, the back-lobe level is a bit less satisfactory: in order to improve it, a tuning of the optimization technique will be performed in Section 4.4.3.

- As already pointed out many times, keeping into account the spatial extension of the Floquet ports is crucial in order to achieve the correct phase compensation because all the cells appear to have the same height.

  This is evident observing Figure 4.17, where two different design flows are compared: the blue curves are obtained with the correct procedure, which includes the additional height introduced by the Floquet ports in the calculation of the phase compensation, while the orange curves are obtained without this correction.

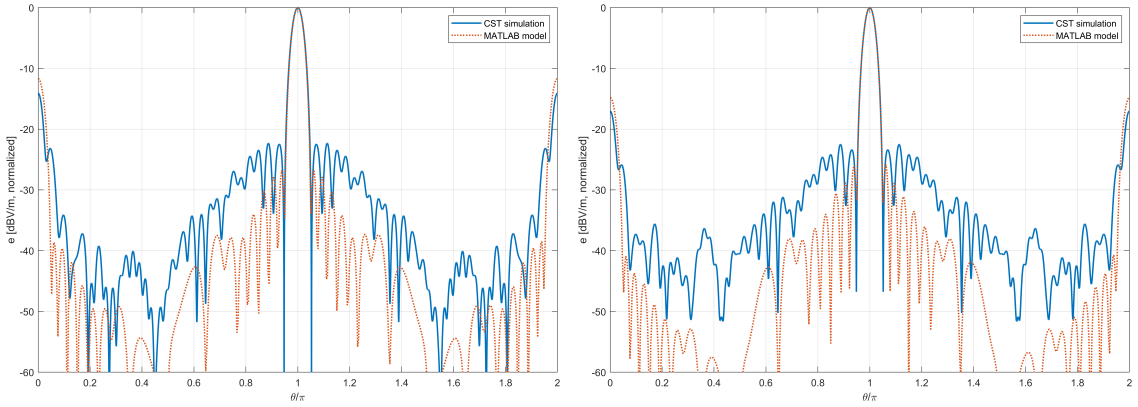
  It is evident that neglecting the Floquet port height leads to a significant degradation of the far-field pattern, with a clear reduction in the main beam amplitude and an increase in the side lobe and back lobe levels. This highlights the importance of accurately modeling all geometrical aspects of the unit cell in the design process.

- In Figure 4.18, the results of the CST full-wave simulations (blue continuous lines) are compared with the output of the analytical PO-based model (orange dashed lines) presented in Section 4.3.



(a) TA-1 **(b)** TA-2

Figure 4.17: Comparison between different design flows: the blue curves are obtained with the correct procedure, that ensures good phase compensation on the entire surface of the array, while the orange ones are obtained without compensating the phase in the correct way.



(a) TA-1 **(b)** TA-2

Figure 4.18: Comparison between simulation results (blue curves) and analytical model (orange curves) for the two simulated square TAs.

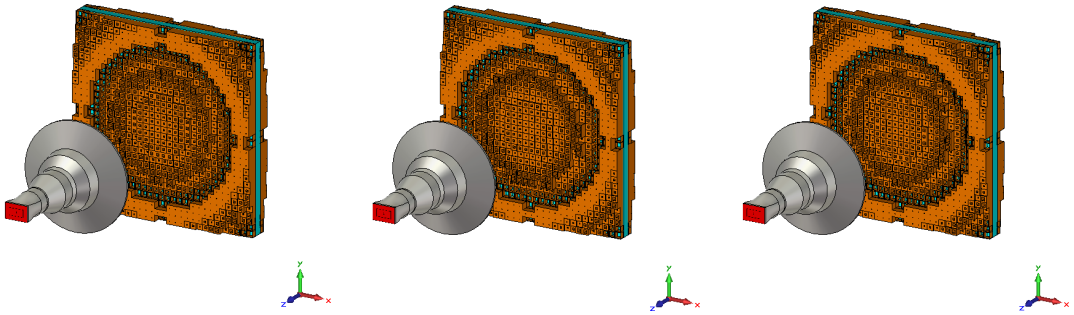
Understandably, the analytical curves are far more optimistic than the actual ones in terms of side-lobe level but, unexpectedly, the results of the simulations are better when it comes to the back-lobe level, probably due to the naïve modeling of the UC element presented in Equation (4.38).

Based on all the above considerations, the model can be considered **successfully validated** and, in the following of this work, the array selected for further development of the design process is **TA-2**, i.e., the **square array** optimized with a small-tolerance **inequality constraint**.

4.4.3 Improvement of SLL and BLL

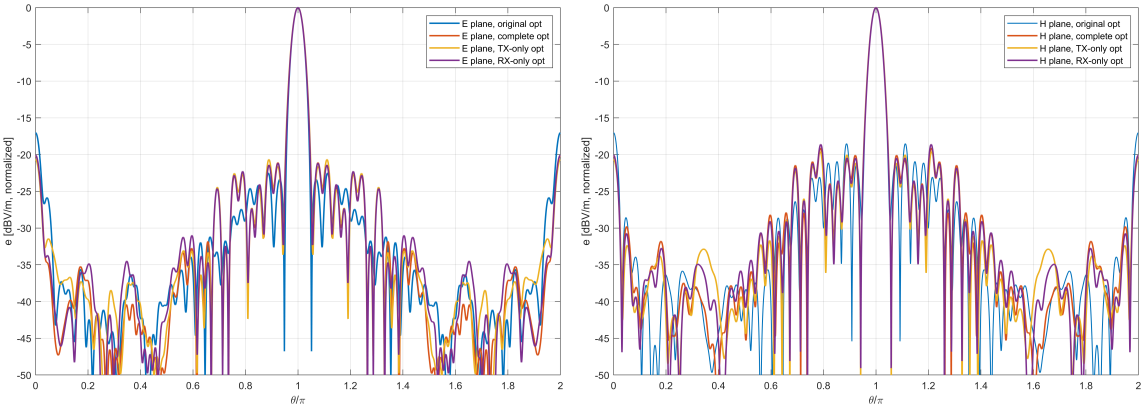
The aim of this Section is to tune and improve the optimization technique presented in the previous paragraph in order to improve the performance of the Transmitarray, especially in terms of back-lobe level, which in the simulations performed up to now are not very satisfactory.

In particular, three cases are considered starting from the project of the array TA-2, presented in Section 4.4.2:



(a) Complete optimization **(b)** TX-only optimization **(c)** RX-only optimization

Figure 4.19: 30×30 TAs designed with three different optimization methods.

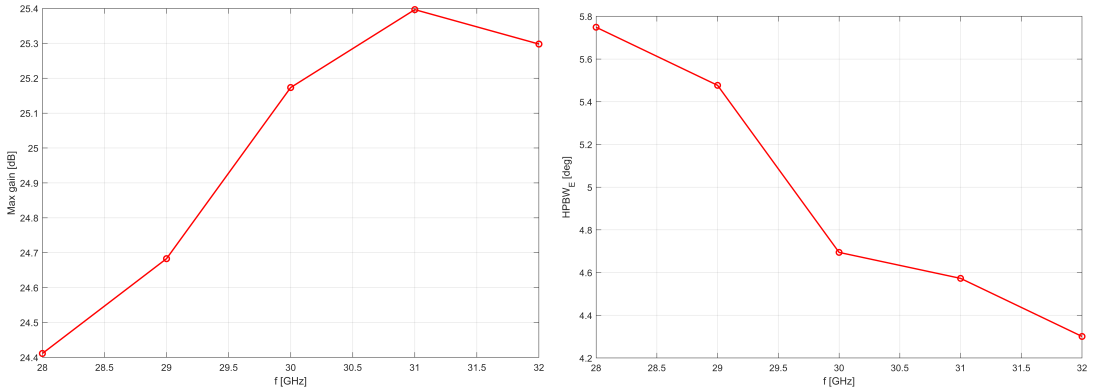


(a) E plane **(b)** H plane

Figure 4.20: Comparison between four different optimization approaches for TA design.

Table 4.3: Performance of TA-2 at centerband with tuning of the optimization technique.

Structure	SLL_E [dB]	SLL_H [dB]	BLL_E [dB]	BLL_H [dB]	$HPBW_E$ [deg]	$HPBW_H$ [deg]	η_{tot} [dB]
TA-2, c	−21.11	−19.12	−20.19	−20.19	7.00	6.80	−1.431
TA-2, p	−20.67	−19.60	−20.67	−20.67	7.00	6.80	−1.487
TA-2, r	−21.18	−18.62	−20.07	−20.07	7.00	6.80	−1.372



(a) $f \mapsto \mathcal{G}_{\max}(f)$ **(b)** $f \mapsto \mathrm{HPBW}_E(f)$

Figure 4.21: Maximum gain (a) and HPBW in the E plane (b) vs frequency for the array TA-2, c.

1. **TA-2, c** (*complete optimization*): optimization of both the transmission and reflection coefficients accepting a phase error with $\texttt{tol} = 0.3$;
2. **TA-2, p** (*TX-only optimization*): optimization of the transmission coefficient alone accepting a phase error with $\texttt{tol} = 0.3$;
3. **TA-2, r** (*RX-only optimization*): optimization of the reflection coefficient alone accepting a phase error with $\texttt{tol} = 0.3$.

All the results are reported in Figure 4.19 (3D structures) and 4.20 (radiated far-field in cartesian coordinates in the E and H planes) and in Table 4.3, where the values of SLL, BLL, HPBW and η_{tot} in both E and H planes are summarized.

As pointed out at the end of Section 3.4, in particular during the analysis and discussion of Tables 3.3-3.8, the results obtained with the three methods are very similar.

With respect to the previous simulations, the BLL is significantly better, being lower than $-20\,\text{dB}$ in all the cases of interest but, on the other hand, the SLL is slightly higher due to a necessary **trade-off** between the different optimization goals.

As far as the total efficiency is concerned, no significant variations are observed, while the behavior of the maximum gain $\mathcal{G}_{\max}$ and the E-plane half-power beamwidth as frequency spans over the design bandwidth are shown in Figure 4.21.

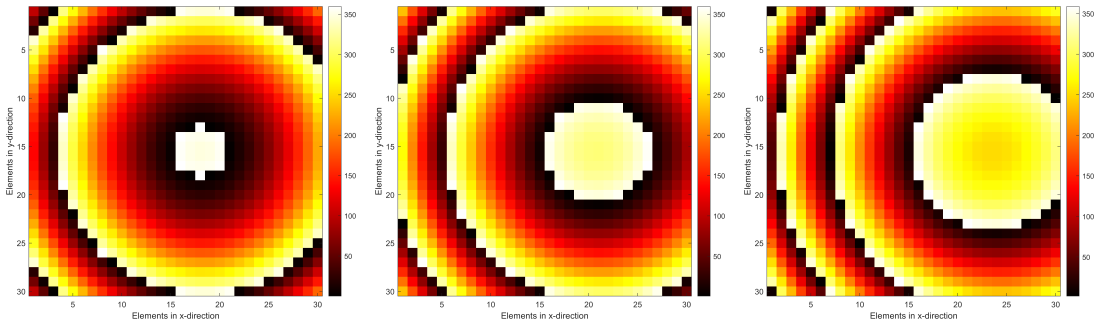
4.4.4 Generalization to Non-broadside Arrays

4.4.4.1 Beam-scanning Arrays

To conclude the analysis on Transmitarray structures, the design method explored in the previous Sections is now extended to the general case of non-broadside structures:

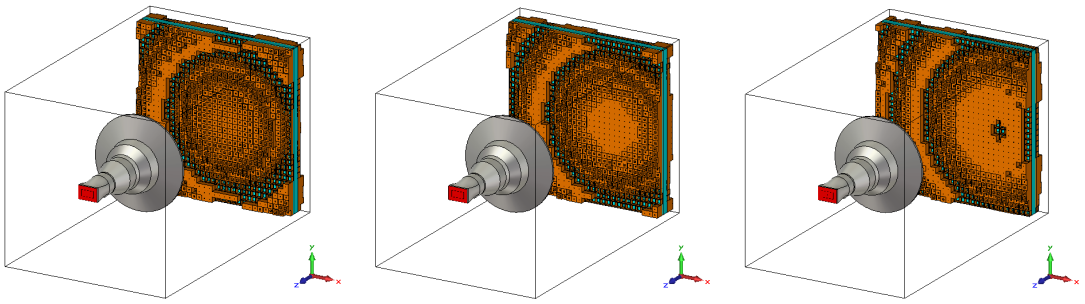
In this case, the phase compensation at each unit cell is calculated with Equation (4.23), while all the other design steps remain unchanged. The results of the simulations, performed for $\theta_{\text{b}} = 5, 10, 15^{\circ}$ are here reported; in particular:

- Figure 4.22 shows the theoretical phase map distributions of the three arrays;
- Figure 4.23 shows the 3D structures designed and simulated with CST MW Studio®;
- In Figures 4.24 and 4.25, the results of the simulations, i.e., the 3D and 1D far-field radiation patterns, are reported;



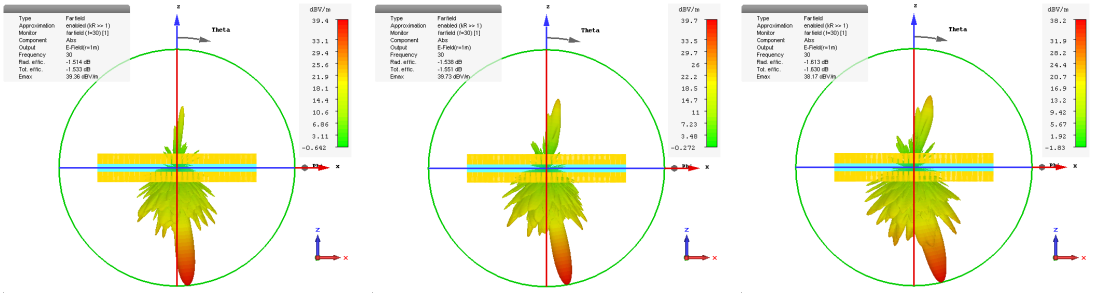
(a) $(\theta_b, \varphi_b) = (5°, 0°)$ **(b)** $(\theta_b, \varphi_b) = (10°, 0°)$ **(c)** $(\theta_b, \varphi_b) = (15°, 0°)$

Figure 4.22: Phase maps for the simulated non-broadside TAs, (30×30 cells, $F/D = 1$, variable beam angles).



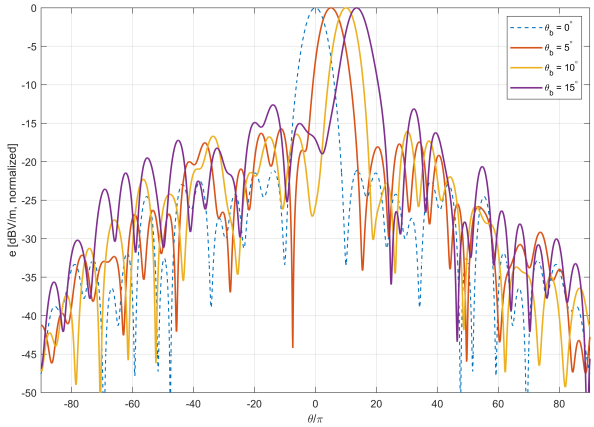
(a) $(\theta_b, \varphi_b) = (5°, 0°)$ **(b)** $(\theta_b, \varphi_b) = (10°, 0°)$ **(c)** $(\theta_b, \varphi_b) = (15°, 0°)$

Figure 4.23: Simulated non-broadside TAs.

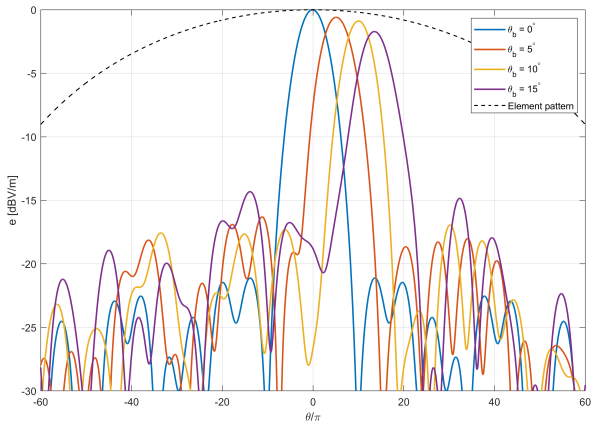


(a) $(\theta_b, \varphi_b) = (5°, 0°)$ **(b)** $(\theta_b, \varphi_b) = (10°, 0°)$ **(c)** $(\theta_b, \varphi_b) = (15°, 0°)$

Figure 4.24: Radiated far field of the simulated non-broadside TAs (3D view).

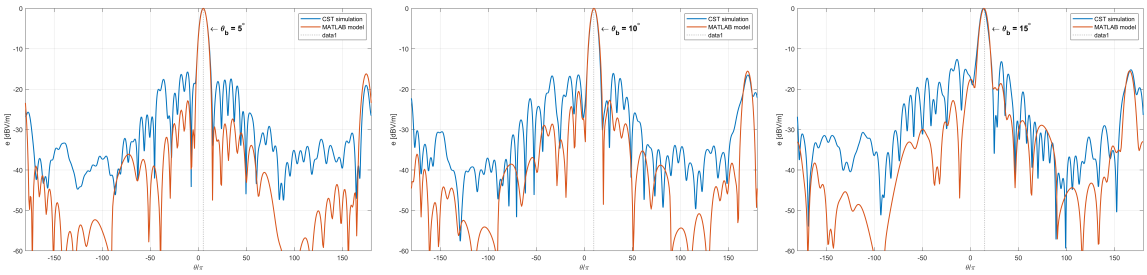


(a) Simulation results



(b) Scan loss

Figure 4.25: Radiated far field of the simulated non-broadside TAs (1D view, *E* plane).



(a) $\theta_b = 5°$ **(b)** $\theta_b = 10°$ **(c)** $\theta_b = 15°$

Figure 4.26: Comparison between the CST simulations of non-BS arrays (blue curves) and the MATLAB analytical model (orange curves).

- Figure 4.25b highlights how the amplitude of the main beam changes with respect to the BS case as the scan angle increases;
- In Figures 4.26a, 4.26b anad 4.26c, the results of the CST simulations are compared with the PO-based analytical model presented in Section 4.3 and implemented in MATLAB® in Appendix B.8.2.

In general, maintaining high gain and low side-lobe levels is a key challenge in the design of Transmitarray antennas. As discussed in [3], beam scanning in Transmitarray is typically achieved by applying appropriate phase compensation across the array elements, enabling the main beam to be steered electronically without physically moving the antenna.

The article highlights that the achievable scan range is fundamentally limited by the array geometry, element spacing, and the phase-shifting capabilities of the unit cells.

Similarly, [23] analyzes the impact of scan angle on array performance, noting that as the beam is steered away from broadside, effects such as scan loss, increased side-lobe levels, and potential grating lobes become more pronounced. Both works emphasize the importance of optimizing unit cell design and array parameters to maximize the effective scan range while minimizing performance degradation.

Indeed, the simulations performed with increasing values of θ_b show that:

- The amplitude of the main beam monotonically decreases as θ_b increases with respect to the results obtained for a broadside array; this is clearly visible in Figure 4.25b.
- Similarly, the amplitude of side lobes and back lobe increases as θ_b increases, as well as the half-power beamwidth.
- Gain and efficiency slightly decrease as θ_b increases.

In practice, as the scan angle θ_b increases, the array experiences a phenomenon known as **scan loss**, which is primarily due to the reduction in effective aperture and the non-uniform illumination of the array elements.

According to [2], scan loss manifests as a decrease in the maximum achievable gain when the main beam is steered away from broadside, which occurs because the projected aperture area in the scan direction becomes smaller and the phase errors across the array increase, leading to a less efficient constructive interference in the desired direction.

From the mathematical point of view, this is due to the fact that the "ideal" radiated far-field is tapered by the element pattern, which, as already said, can modeled by a power of the cosine function:

$$e_{UC}(\theta) = \cos^{q_e}\theta, \tag{4.42}$$

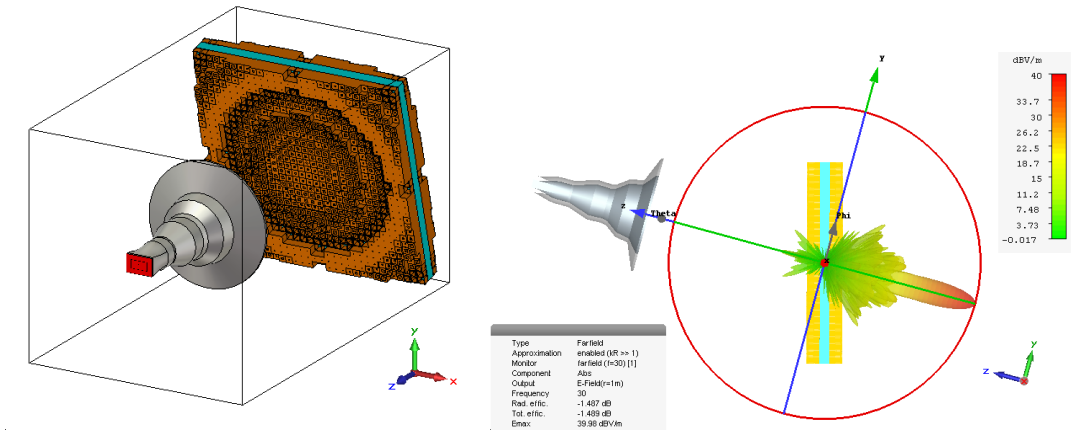
where the value of $q_e \in \mathbb{R}^+$, fixed in this work to 1, determines how wide the UC field is. The higher the tapering, the higher the mismatch between the element pattern and the array factor.

As a result, not only does the main beam gain decrease, but the side-lobe and back-lobe levels rise as well, further degrading the overall radiation pattern of the Transmitarray.

4.4.4.2 Mechanical Beam Steering

To conclude the analysis on non-broadside Transmitarrays, a brief overview of **mechanical beam steering** is here presented.

Mechanical beam steering is a technique in which the main beam direction of an antenna is changed by physically rotating or tilting the entire antenna structure, rather than by electronic phase shifting. As discussed in [47], this approach is particularly attractive for high-frequency



(a) Structure **(b)** 3D far-field

Figure 4.27: Simulated TA with mechanical beam steering.

and high-gain antennas, where implementing electronic beam steering can be complex and costly due to the need for tunable or active elements. Mechanical steering offers a simple, robust, and cost-effective solution, especially for applications where the beam direction does not need to be changed rapidly or frequently.

In [31], the authors demonstrate that mechanical rotation of the antenna allows for continuous beam scanning over a wide angular range while maintaining high efficiency and low side-lobe levels. Though the work is focused on Reflectarrays, the same considerations hold for Transmitarrays.

The main limitation of this approach is the mechanical **complexity** and potential increase in system size and weight, as well as the slower response time compared to electronic steering. However, for many practical scenarios — such as fixed wireless access, satellite communications, or point-to-point links — mechanical beam steering remains a viable and effective solution, enabling high-performance beam control without the need for complex electronic circuitry.

Furthermore, it has been recently demonstrated [47] that mechanically rotated Transmitarrays can maintain high aperture efficiency and low side-lobe levels over a broad scan range, with minimal degradation in gain compared to electronically steered solutions.

The article also points out that mechanical steering avoids the bandwidth limitations and losses associated with tunable elements, making it particularly suitable for high-frequency and broadband applications. However, it is important to consider the slower response time and potential mechanical wear, which may limit its use in applications requiring rapid or continuous beam reconfiguration.

In summary, mechanical beam steering provides a robust and effective approach for applications where high performance and wide-angle scanning are required, and where the speed of beam reconfiguration is not a critical constraint.

To understand whether the model developed in the previous Sections is applicable in this context, a mechanically steered Transmitarray with a fixed phase distribution was simulated by rotating the entire antenna by 15° with respect to the feed horn.

Figures 4.27 and 4.28 show the resulting structure and far-field pattern. The obtained results are extremely promising; in fact, the most important antenna parameters are equal to

$$\begin{cases} \mathrm{SLL}_E = -21.54\,\mathrm{dB} \\ \mathrm{SLL}_H = -21.64\,\mathrm{dB} \end{cases} \quad \begin{cases} \mathrm{BLL}_E = -32.10\,\mathrm{dB} \\ \mathrm{BLL}_H = -32.10\,\mathrm{dB} \end{cases} \quad \begin{cases} \mathrm{HPBW}_E = 7.00\,\mathrm{dB} \\ \mathrm{HPBW}_H = 6.90\,\mathrm{dB} \end{cases} \tag{4.43}$$

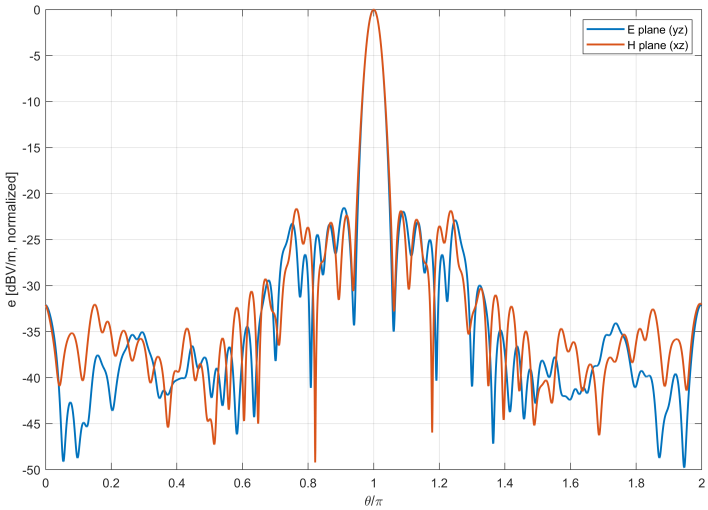


Figure 4.28: Radiated far field of the simulated TA with mechanical beam steering (1D view, E and H plane).

which makes this antenna one of the best designs of the entire Thesis, especially concerning the value of the back lobes.

Chapter 5

SES Design

In this chapter, the same design and validation method already exposed for TA antennas is extended to the realization of transmitting **Smart Electromagnetic Skins**, i.e., window-integrated structures which should steer in a desired direction the field generated from a base station, here modeled as an ideal plane wave for the sake of simplicity.

This chapter is divided into two parts:

1. In Section 5.1, an approximated PO-based analytical model that predicts the radiated far-field of the smart skin is developed and implemented in MATLAB®; several test cases are presented in order to demonstrate its validity. The mathematical approach is very similar to the one adopted for TA design in Section 4.3.

2. In Section 5.2, some simulations with incident plane wave are performed and discussed.

Contrary to TAs, Smart Electromagnetic Skins operating in transmission mode still remain a highly underexplored area of research.

At present, scientific literature addressing this topic remains sparse, and experimental validation is limited. Due to the absence of a standardized simulation methodology, several simulation approaches that yield divergent results have been explored, aiming to assess their reliability and gaining insight into the modeling of these emerging technologies.

In the end, the methodology proposed in [48] has been selected, and the results obtained with this approach are the only ones reported in the Thesis. This decision is primarily driven by the article's clarity and step-by-step precision — qualities lacking in other examined sources — and further reinforced by results that align closely with theoretical expectations, confirming the promising potential of the proposed approach for future development.

5.1 Physical Optics Model

Let us consider a $M \times M$ SES illuminated by a plane wave linearly polarized along the y axis, meaning that

$$\mathbf{E}_{mn} \equiv \mathbf{E} = E_0\,\widehat{\mathbf{y}}, \quad E_0 = \text{constant}\ \forall m, n = \{\,1, \dots, M\,\}\,. \tag{5.1}$$

In order to steer the transmitted beam from the incident direction $(\theta_\mathrm{i}, \varphi_\mathrm{i})$ to the desired direction $(\theta_\mathrm{b}, \varphi_\mathrm{b})$, the required phase distribution for the antenna elements is

$$\Psi_{mn} = \Psi_0 - \frac{2\pi}{\lambda_0}\Big[x_{mn}\left(\sin\theta_\mathrm{i}\cos\varphi_\mathrm{i} + \sin\theta_\mathrm{b}\cos\varphi_\mathrm{b}\right) + y_{mn}\left(\sin\theta_\mathrm{i}\sin\varphi_\mathrm{i} + \sin\theta_\mathrm{b}\sin\varphi_\mathrm{b}\right)\Big]. \tag{5.2}$$

Similarly to Section 4.3, let us model the mn^{th} unit cell as a square aperture $\Sigma_{\text{a}} \subset \mathbb{R}^2$ identified by the following coordinates:

$$x \in \left[-\frac{W}{2}, \frac{W}{2}\right] + x_{mn} \implies x \in W\left[m - \frac{M+2}{2}, m - \frac{M}{2}\right] \equiv [x_\ell, x_u] \subset \mathbb{R} \tag{5.3a}$$

$$y \in \left[-\frac{W}{2}, \frac{W}{2}\right] + y_{mn} \implies y \in W\left[n - \frac{M+2}{2}, n - \frac{M}{2}\right] \equiv [y_\ell, y_u] \subset \mathbb{R} \tag{5.3b}$$

meaning that

$$\Sigma_{\text{a}} = W\left[m - \frac{M+2}{2}, m - \frac{M}{2}\right] \times W\left[n - \frac{M+2}{2}, n - \frac{M}{2}\right] \equiv [x_\ell, x_u] \times [y_\ell, y_u] \subset \mathbb{R}^2 \tag{5.4}$$

and

$$\mathbf{r}_{\Sigma,mn} = x\,\hat{\mathbf{x}} + y\,\hat{\mathbf{y}} \implies \mathbf{r}_{\Sigma,mn} \cdot \mathbf{u}(\theta,\varphi) = x\sin\theta\cos\varphi + y\sin\theta\sin\varphi. \tag{5.5}$$

Contrary to the TA case, the surface current density can be considered *constant* over the entire aperture because the antenna elements are small with respect to the operating wavelenght λ_0, thus the field distribution is reasonably uniform over each cell. To prove this property formally, let us use the formal definition of $\mathbf{J}_{\text{s},mn}$ according to the equivalence theorem (see Appendix A.3):

$$\begin{aligned}
\mathbf{J}_{\text{s},mn} &= \hat{\mathbf{n}}_{mn} \wedge \mathbf{H}_{mn} = \hat{\mathbf{z}} \wedge \left(\frac{1}{Z_0}\,\mathbf{u}(\theta,\varphi) \wedge \mathbf{E}_{mn}\right) = \\
&= \hat{\mathbf{z}} \wedge \left[(\sin\theta\cos\varphi\,\hat{\mathbf{x}} + \sin\theta\sin\varphi\,\hat{\mathbf{y}} + \cos\theta\,\hat{\mathbf{z}}) \wedge E_0\,\hat{\mathbf{y}}\right] = \\
&\propto \hat{\mathbf{z}} \wedge \left[\sin\theta\cos\varphi\,(\hat{\mathbf{x}} \wedge \hat{\mathbf{y}}) + \cos\theta\,(\hat{\mathbf{z}} \wedge \hat{\mathbf{y}})\right] = \\
&= \hat{\mathbf{z}} \wedge \left[\sin\theta\cos\varphi\,\hat{\mathbf{z}} - \cos\theta\,\hat{\mathbf{x}}\right] = -\cos\theta\,(\hat{\mathbf{z}} \wedge \hat{\mathbf{x}}) = \cos\theta\,\hat{\mathbf{y}} \propto \hat{\mathbf{y}},
\end{aligned} \tag{5.6}$$

where the magnetic field has been evaluated with the FF impedance relation, defined in Equation (A.5). Therefore, we can write that

$$\mathbf{J}_{\text{s},mn} \equiv J_0\,\hat{\mathbf{y}}, \quad J_0 = \text{constant}. \tag{5.7}$$

The **radiation integral** $\widetilde{\mathbf{J}}_{mn}(\theta,\varphi)$ can be easily calculated analytically:

$$\begin{aligned}
\widetilde{\mathbf{J}}_{mn}(\theta,\varphi) &= \iint_{\Sigma_{\text{a}}} \mathrm{d}x\,\mathrm{d}y\; S_{21,mn}\,\mathbf{J}_{\text{s},mn}\,\exp\left[\mathrm{j}k_0\,(\mathbf{r}_{\Sigma,mn} \cdot \mathbf{u}(\theta,\varphi))\right] = \\
&= \left(S_{21,mn}J_0 \int_{x_\ell}^{x_u}\int_{y_\ell}^{y_u} \mathrm{d}x\,\mathrm{d}y\,\exp\left[\mathrm{j}k_0\,(x\sin\theta\cos\varphi + y\sin\theta\sin\varphi)\right]\right)\hat{\mathbf{y}} = \\
&= \left(S_{21,mn}J_0 \int_{x_\ell}^{x_u} e^{\mathrm{j}k_0 x\sin\theta\cos\varphi}\,\mathrm{d}x \int_{y_\ell}^{y_u} e^{jk_0 y\sin\theta\sin\varphi}\,\mathrm{d}y\right)\hat{\mathbf{y}} = \\
&= -S_{21,mn}J_0\,\mathcal{I}_x\mathcal{I}_y\,\hat{\mathbf{y}},
\end{aligned}$$

where the following quantities have been defined:

$$\begin{cases}
\mathcal{I}_x := \displaystyle\int_{x_\ell}^{x_u} e^{\mathrm{j}k_0 x\sin\theta\cos\varphi}\,\mathrm{d}x = \frac{e^{\mathrm{j}k_0 x_u\sin\theta\cos\varphi} - e^{\mathrm{j}k_0 x_\ell\sin\theta\cos\varphi}}{jk_0\sin\theta\cos\varphi} \\
\mathcal{I}_y := \displaystyle\int_{y_\ell}^{y_u} e^{jk_0 y\sin\theta\sin\varphi}\,\mathrm{d}y = \frac{e^{\mathrm{j}k_0 y_u\sin\theta\sin\varphi} - e^{\mathrm{j}k_0 y_\ell\sin\theta\sin\varphi}}{jk_0\sin\theta\sin\varphi}
\end{cases} \tag{5.8}$$

Table 5.1: Directions of the incident and transmitted beams of the designed Smart Skins.

Structure	θ_i [deg]	φ_i [deg]	θ_b [deg]	φ_b [deg]
SES-1	0	0	10.46	0.00
SES-2	0	0	19.84	0.00
SES-3	0	0	29.94	0.00
SES-4	0	0	40.04	0.00
SES-5	10	0	20.56	0.00
SES-6	10	0	31.38	0.00
SES-7	0	0	19.84	45.09
SES-8	0	0	19.84	90.18

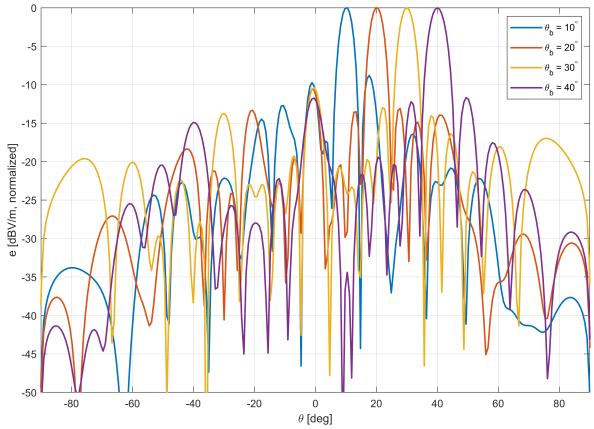


Figure 5.1: PO-based far-field radiation pattern in the E-plane for SESs with normal plane-wave incidence, variable θ_b and $\varphi_\mathrm{b} = 0°$.

The components of the electric field along $\widehat{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\varphi}}$ directions are given by

$$\begin{cases} \mathbf{e}_{\theta,mn}(\theta,\varphi) = e_{\theta,mn}(\theta,\varphi)\,\widehat{\boldsymbol{\theta}} = \left(-\mathrm{j}k_0 Z_0 \widetilde{\mathbf{J}}_{mn}(\theta,\varphi)\cdot\widehat{\boldsymbol{\theta}}\right)\widehat{\boldsymbol{\theta}} \\ \mathbf{e}_{\varphi,mn}(\theta,\varphi) = e_{\varphi,mn}(\theta,\varphi)\,\widehat{\boldsymbol{\varphi}} = \left(-\mathrm{j}k_0 Z_0 \widetilde{\mathbf{J}}_{mn}(\theta,\varphi)\cdot\widehat{\boldsymbol{\varphi}}\right)\widehat{\boldsymbol{\varphi}} \end{cases} \tag{5.9}$$

meaning that the total field radiated by the mn^{th} UC can be expressed as

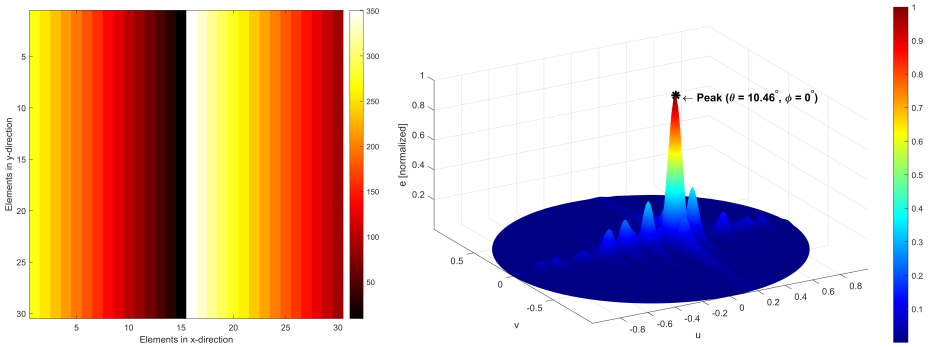
$$\mathbf{e}_{mn}(\theta,\varphi) = \mathbf{e}_{\theta,mn}(\theta,\varphi) + \mathbf{e}_{\varphi,mn}(\theta,\varphi) \equiv \mathbf{e}_{\theta,mn}(\theta,\varphi) = e_{\theta,mn}(\theta,\varphi)\,\widehat{\boldsymbol{\theta}}. \tag{5.10}$$

Therefore, the total electric field radiated by the antenna can be expressed as the superposition of the fields radiated by all the UCs, each one weighed by the corresponding cosine element pattern defined in Equation (4.38).

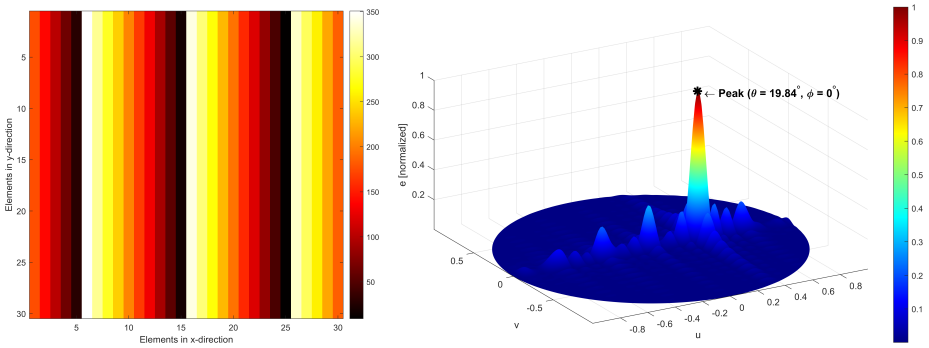
The MATLAB® code to implement the calculations above is reported in Appendix B.8.3; the results obtained for eight SES configurations are reported below in order to validate the model in different operating conditions:

1. Figures 5.1 and 5.2 (SES-1, SES-2, SES-3, SES-4) show the far-field radiation patterns, the phase distributions and the UV plots obtained with the PO-based model for SESs with:

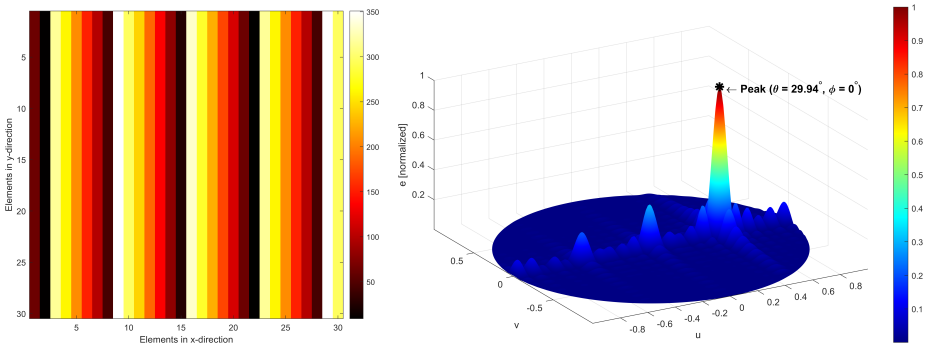
$$\begin{cases} \theta_\mathrm{i} = 0° \\ \varphi_\mathrm{i} = 0° \end{cases} \quad \begin{cases} \theta_\mathrm{b} = 10°, 20°, 30°, 40° \\ \varphi_\mathrm{b} = 0° \end{cases} \tag{5.11}$$



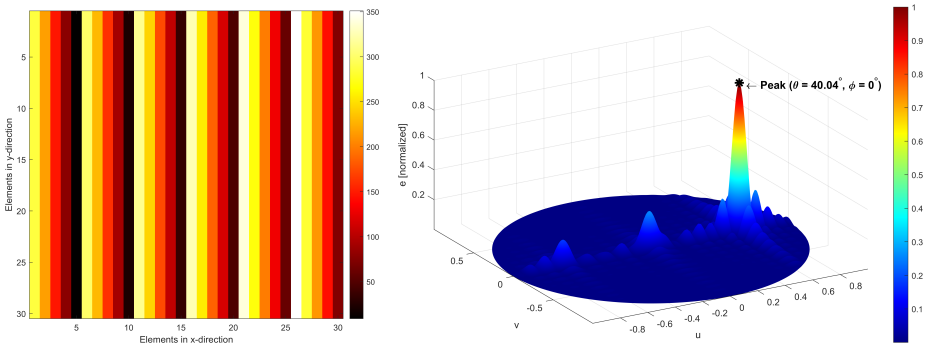
(a) SES-1, $(\theta_b, \varphi_b) = (10°, 0°)$ **(b)** SES-1, $(\theta_b, \varphi_b) = (10°, 0°)$



(c) SES-2, $(\theta_b, \varphi_b) = (20°, 0°)$ **(d)** SES-2, $(\theta_b, \varphi_b) = (20°, 0°)$

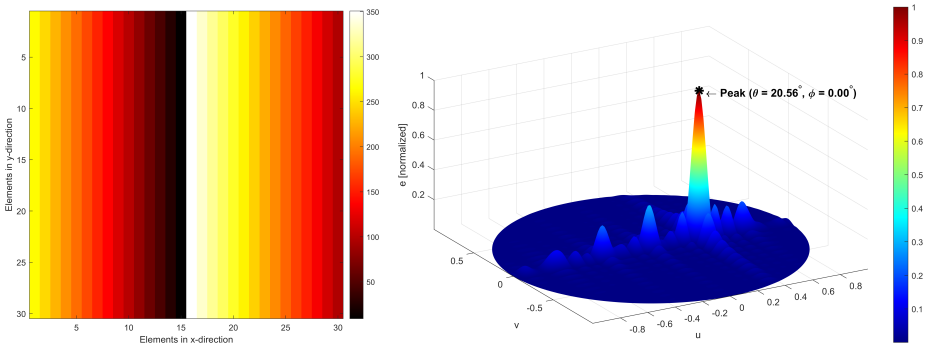


(e) SES-3, $(\theta_b, \varphi_b) = (30°, 0°)$ **(f)** SES-3, $(\theta_b, \varphi_b) = (30°, 0°)$

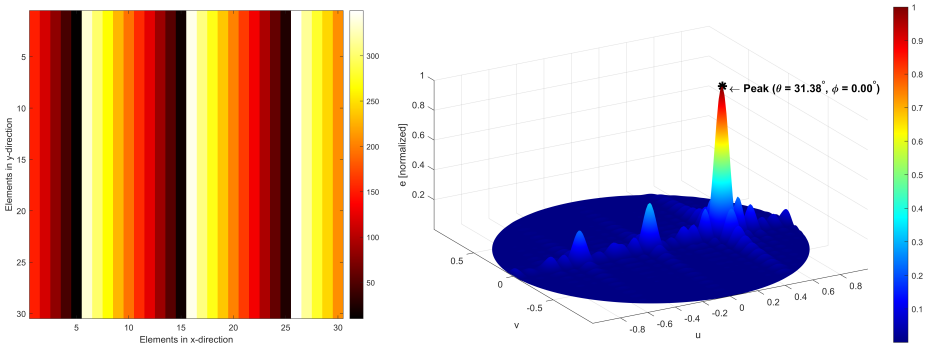


(g) SES-4, $(\theta_b, \varphi_b) = (40°, 0°)$ **(h)** SES-4, $(\theta_b, \varphi_b) = (40°, 0°)$

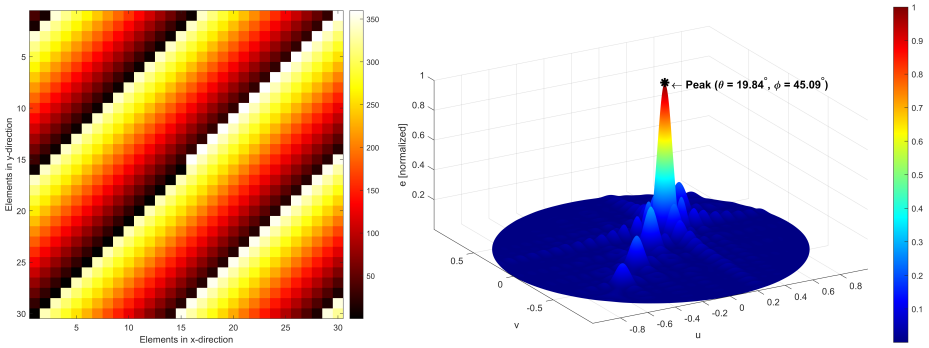
Figure 5.2: Phase distributions and PO-based UV plots for SESs with normal plane-wave incidence, variable θ_b and $\varphi_b = 0°$.



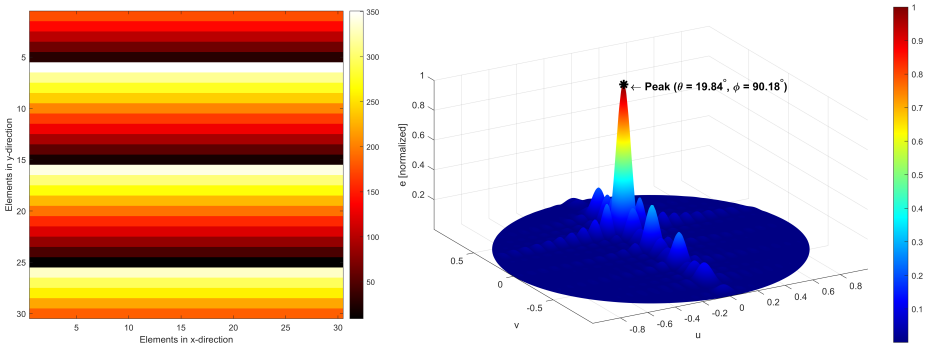
(a) SES-5, $(\theta_i, \theta_b) = (10°, 20°)$ **(b)** SES-5, $(\theta_i, \theta_b) = (10°, 20°)$



(c) SES-6, $(\theta_i, \theta_b) = (10°, 30°)$ **(d)** SES-6, $(\theta_i, \theta_b) = (10°, 30°)$



(e) SES-7, $(\theta_b, \varphi_b) = (20°, 45°)$ **(f)** SES-7, $(\theta_b, \varphi_b) = (20°, 45°)$



(g) SES-8$(\theta_b, \varphi_b) = (20°, 90°)$ **(h)** SES-8$(\theta_b, \varphi_b) = (20°, 90°)$

Figure 5.3: Phase distributions and PO-based UV plots for SESs with $\theta_i = 10°$ and variable θ_b(a)-(d); normal incidence and variable θ_b and φ_b (e)-(h).

2. Figures 5.3a-5.3d (SES-5, SES-6) show the phase distributions and the UV plots obtained with the PO-based model for SESs with:

$$\begin{cases}\theta_{\mathrm{i}} = 10^{\circ} \\ \varphi_{\mathrm{i}} = 0^{\circ}\end{cases} \quad \begin{cases}\theta_{\mathrm{b}} = 20^{\circ}, 30^{\circ} \\ \varphi_{\mathrm{b}} = 0^{\circ}\end{cases} \tag{5.12}$$

3. Figures 5.3e-5.3h (SES-7, SES-8) show the phase distributions and the UV plots obtained with the PO-based model for SESs with:

$$\begin{cases}\theta_{\mathrm{i}} = 0^{\circ} \\ \varphi_{\mathrm{i}} = 0^{\circ}\end{cases} \quad \begin{cases}\theta_{\mathrm{b}} = 20^{\circ} \\ \varphi_{\mathrm{b}} = 45^{\circ}, 90^{\circ}\end{cases} \tag{5.13}$$

4. Table 5.1 summarizes the directions of the transmitted beams generated by the designed structures.

From the obtained results, it is evident that the PO-based analytical model accurately predicts the main beam direction and the overall shape of the far-field radiation patterns for the different SES configurations. The phase distributions imposed on the SESs lead to the desired beam steering, as confirmed by the UV plots and the positions of the main lobes.

It is important to notice that the phase compensation is performed with respect to the reference frame of the *incident beam* and not with respect to the angles defined in the "standard" coordinate system *Oxyz*: in fact, the phase distributions obtained for SES-5 and SES-6, which have both $\theta_{\mathrm{i}} \neq 0$ and $\theta_{\mathrm{b}} \neq 0$, are equivalent to the phase maps of smart skins with orthogonal incidence and

$$\theta'_{\mathrm{b}} = \theta_{\mathrm{b}} - \theta_{\mathrm{i}}. \tag{5.14}$$

5.2 Results of CST Simulations

In order to validate the PO-based model developed in Section 5.1, in this paragraph two CST MW Studio® simulations are presented. The proposed setup is similar to the one adopted for Transmitarrays and described in Section 4.4.1.3; it is, however, important to point out some differences:

Excitation. The feed source and the waveguide port are replaced by a **plane-wave excitation source**, linearly polarized (TE mode) along the *y* axis:

$$\widehat{\mathbf{n}} = (0, 0, -1)\,, \quad \widehat{\mathbf{e}} = (0, 1, 0)\,. \tag{5.15}$$

In physics, plane waves $(\mathbf{r}, t) \mapsto \boldsymbol{\mathcal{E}}(\mathbf{r}, t)$ and $(\mathbf{r}, t) \mapsto \boldsymbol{\mathcal{H}}(\mathbf{r}, t)$ are solutions of the D'Alambert equation characterized by perfectly flat and parallel wavefronts[1]; here, without loss of generality, only the equations for the electric field are reported [33]:

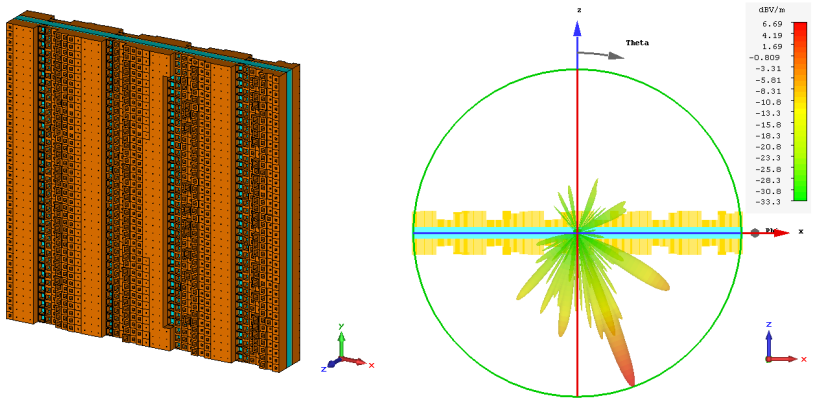
$$\Box\boldsymbol{\mathcal{E}}(\mathbf{r}, t) \equiv \left(\Delta - \epsilon\mu\,\frac{\partial^2}{\partial t^2}\right)\boldsymbol{\mathcal{E}}(\mathbf{r}, t) = \mathbf{0}, \tag{5.16}$$

$$\boldsymbol{\mathcal{E}}(\mathbf{r}, t) = \Re\left\{\mathbf{E}(\mathbf{r})e^{-\mathrm{j}\omega_0 t}\right\} \equiv \Re\left\{\mathbf{E}_0 e^{\mathrm{j}(\mathbf{k}_0\cdot\mathbf{r}-\omega_0 t)}\right\}, \tag{5.17}$$

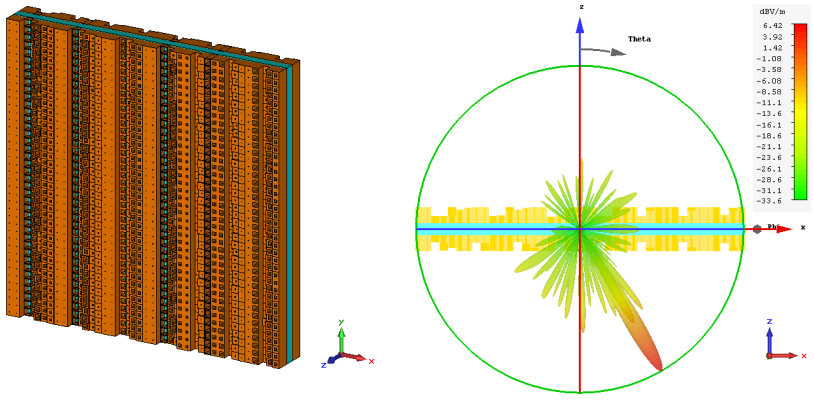
where

$$\mathbf{E}(\mathbf{r}) := \mathbf{E}_0 e^{\mathrm{j}\mathbf{k}_0\cdot\mathbf{r}}. \tag{5.18}$$

[1]The *wavefront* of a time-varying wave field is an imaginary surface defined as the locus of all points having the same phase.

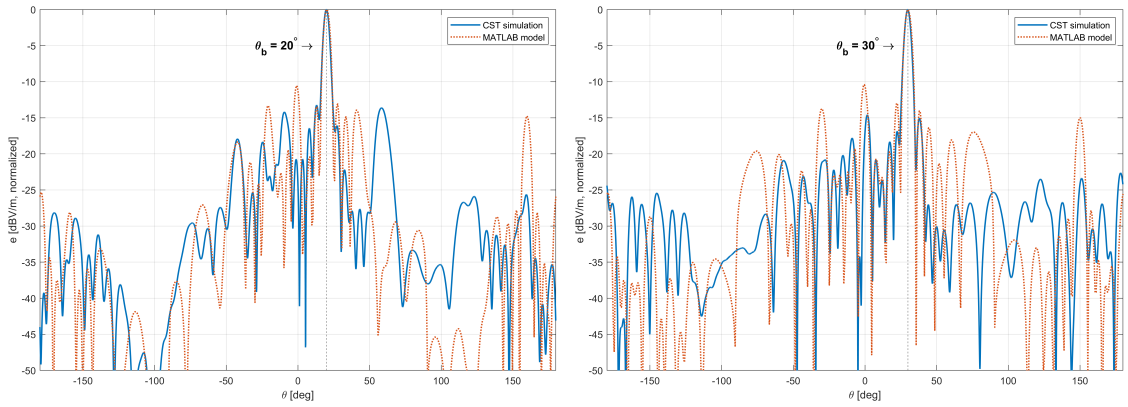


(a) SES-2, $(\theta_b, \varphi_b) = (20^\circ, 0^\circ)$ **(b)** SES-2, $(\theta_b, \varphi_b) = (20^\circ, 0^\circ)$



(c) SES-3, $(\theta_b, \varphi_b) = (30^\circ, 0^\circ)$ **(d)** SES-3, $(\theta_b, \varphi_b) = (30^\circ, 0^\circ)$

Figure 5.4: Simulated transmitting SESs. Left: 3D structures; right: 3D radiated far-fields (dB units).



(a) SES-2, $(\theta_b, \varphi_b) = (20^\circ, 0^\circ)$ **(b)** SES-3, $(\theta_b, \varphi_b) = (30^\circ, 0^\circ)$

Figure 5.5: Radiated far field of the simulated 40×40 SESs (E plane).

For this reason, a plane wave is the typical approximation of the radiation pattern of a source located at very high distance from the field monitor, where spherical waves flatten out and begin to resemble planes.

In CST MW Studio®, for obvious computational reasons it is not possible to set up a simulation domain with infinite extension: therefore, the results obtained with a plane-wave excitation are only approximations of the real behavior of the structure.

Boundary conditions. `open (add space)` boundary conditions are used in all directions to simulate free-space radiation; along the z axis, the amplitude of the simulation domain is extended to 100 mm in both directions to mitigate border effects, as pointed out in [40] and [48].

Although, in principle, the designed structures are symmetric with respect to the xz plane, no symmetry planes are used to avoid reflections and other undesired effects.

Post-processing. When monitoring the far-field, CST should already calculate the scattered field, contrary to what happens when monitoring the near-field [48]. Therefore, in principle no post-processing of the results is needed.

The results of the simulations can be observed in Figure 5.4 and 5.5, showing respectively the 3D structures, the 3D radiated far-field and the 1D cut in the E plane.

The agreement between the PO-based analytical model and the full-wave simulations is satisfactory, with the main beam direction and the HPBW being predicted in the correct way and thus suggesting the correctness of the adopted approach despite the already mentioned absence of validated simulation methods.

Minor discrepancies between the analytical and simulated results, especially concerning the side-lobe and back-lobe levels, can be attributed to the idealizations in the PO model, such as the assumption of uniform current distribution and neglecting mutual coupling effects. Nevertheless, the model provides a reliable and efficient tool for the preliminary design of Transmissive Smart Electromagnetic Skins.

Chapter 6

Conclusions

6.1 Summary and Key Findings

This Thesis has explored the design, modeling, and optimization of innovative dielectric-only Smart Electromagnetic Skins and Transmitarray antennas for next-generation wireless communication systems. The work is motivated by the growing demand for high data rates, low latency, and reliable connectivity in 5G, 6G and beyond, where traditional approaches face significant challenges due to the propagation characteristics of millimeter-wave and sub-THz frequencies.

The introductory chapter outlined the context of Smart Radio Environments, highlighting the limitations of conventional wireless systems and the transformative potential of electromagnetic metasurfaces. The Thesis then reviewed some state-of-the-art technologies, emphasizing the role of passive and active intelligent surfaces, such as SES and RIS, in shaping electromagnetic wave propagation to overcome coverage gaps and improve spectral efficiency.

Then, advanced mathematical optimization techniques, such as single- and multi-objective Genetic Algorithms, were introduced and applied to the optimization of Unit Cell geometrical parameters. These methods enabled the efficient exploration of large design spaces, balancing potentially conflicting requirements such as minimizing reflection and maximizing transmission, and achieving desired phase responses across a range of incidence angles. The use of genetic algorithms was validated through benchmark problems and then tailored to the specific needs of metasurface design.

The core of the Thesis focused on the development of an innovative Unit Cell model for TSESs and TAs (Chapter 3) — since the latter are similar to smart skins in transmission mode and can thus be used for their validation — based on an extended equivalent circuit approach and effective medium theory (Maxwell Garnett model). This model allowed for the analytical prediction of the electromagnetic response of multi-layer dielectric structures, significantly reducing the reliance on time-consuming full-wave simulations. The model was validated against CST MW Studio® simulations, demonstrating good agreement and providing a solid foundation for further optimization and array-level design.

Building on the validated Unit Cell models, the Thesis presented a comprehensive design flow for Transmitarray antennas (Chapter 4). Both simplistic (basic antenna array theory) and advanced (Physical Optics-based) models were developed to predict the far-field radiation patterns and optimize the phase compensation required for beam steering, with successful validation against full-wave simulations for a number of different configurations. The impact of different design strategies, including single- and multi-objective optimization, was analyzed, showing that careful UC design can lead to significant improvements in side lobe and back lobe levels. The models were further extended to non-broadside arrays and mechanical beam steering scenarios,

demonstrating the versatility and robustness of the proposed approach.

The final technical chapter (Chapter 5) extended the developed methodologies to the design of Smart Electromagnetic Skins operating in transmission mode. Despite the limited literature and lack of standardized simulation methodologies for SES, the Thesis successfully adapted the PO-based analytical model and validated it through a series of test cases and two full-wave simulations. The results confirmed the potential of SES for enhancing outdoor-to-indoor (O2I) communication links without the need for active amplification or complex control circuitry.

6.2 Contributions and Impact

The main contributions of this Thesis can be summarized as follows:

- Development of an analytical, circuit-based model for multi-layer dielectric unit cells, enabling rapid and accurate prediction of their electromagnetic properties.
- Implementation of advanced optimization techniques (Genetic Algorithms) for the multi-objective design of metasurface elements, balancing transmission, reflection, and phase compensation requirements.
- Comprehensive design and validation of Transmitarray antennas, including both theoretical modeling and full-wave simulation, with a focus on practical design flows and performance optimization.
- Extension of the design methodology to Smart Electromagnetic Skins, demonstrating their feasibility and effectiveness for O2I communication enhancement.
- Critical analysis of the challenges and limitations encountered, providing guidelines for future research and development in the field of passive intelligent surfaces.

The work presented in this Thesis advances the state of the art in metasurface antenna design, offering a set of analytical and computational tools that can accelerate the development of next-generation wireless infrastructure. By reducing the reliance on brute-force simulations and enabling systematic optimization, the proposed methods pave the way for more efficient, cost-effective, and scalable solutions.

6.3 Future Perspectives

While the results achieved are promising, several avenues for future research remain open:

Experimental Validation. The analytical and simulation-based results should be complemented by experimental prototyping and measurement campaigns to assess real-world performance and identify practical implementation issues.

Integration with Active and Reconfigurable Elements. The integration of passive SES and TA structures with active components (e.g., varactors, MEMS, or phase-change materials) could enable dynamic reconfiguration and adaptive beamforming, further enhancing the flexibility and functionality of Smart Radio Environments.

Advanced Materials and Fabrication. The exploration of novel materials, such as low-loss dielectrics, 3D-printed composites, or tunable meta-atoms, could lead to improved performance and new application scenarios.

System-Level Optimization. The joint optimization of metasurface design with network-level parameters (e.g., placement, control algorithms, and integration with AI/ML techniques) represents a promising direction for maximizing the impact of intelligent surfaces in future wireless networks.

Standardization and Simulation Methodologies. The development of standardized simulation and measurement methodologies for SESs and related structures will be crucial for the widespread adoption and comparison of different design approaches prior to empirical validation.

6.4 Final Remarks

In conclusion, this Thesis has demonstrated the feasibility and potential of analytical and optimization-driven approaches for the design of advanced metasurface antennas and Smart Electromagnetic Skins.

The methodologies developed herein provide a solid foundation for further research and innovation, contributing to the realization of smarter, more efficient, and more adaptable wireless communication environments.

As the field continues to evolve, the integration of analytical modeling, advanced optimization, and experimental validation will be key to unlocking the full potential of smart radio environments and metasurface-based technologies.

Appendix A

Antenna Theory

A.1 Antenna Basics

A.1.1 Antenna Far-field Radiation

An **antenna** is any device aimed at transmitting and receiving electromagnetic waves in an efficient way; contrary to what many people think, in principle any object can be considered an antenna and, as this thesis shows, the presence of a conductive material is not compulsory for their correct operation!

Typically, the reference frame exploited by antenna engineers is the **Spherical Coordinate System** $\mathcal{S} = (r, \theta, \varphi)$, where:

- $r \in [0, +\infty)$ is the distance between center of the antenna **O** and observation point **P**;
- $\theta \in [0, \pi)$ is the polar angle, i.e., the angle between the position vector $\mathbf{r} = \mathbf{OP}$ and the z axis;
- $\varphi \in [0, 2\pi)$ is the azimuthal angle, i.e., the angle between the projection of $\mathbf{r}$ on the xy plane and the x axis.

The three unit vectors $\left\{ \widehat{\mathbf{r}}, \widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{\varphi}} \right\}$ that define the spherical orthonormal basis are thus related to the cartesian unit vectors $\{ \widehat{\mathbf{x}}, \widehat{\mathbf{y}}, \widehat{\mathbf{z}} \}$ by the following expressions:

$$\begin{cases} \widehat{\mathbf{r}} \equiv \mathbf{u} = \sin\theta\cos\varphi\,\widehat{\mathbf{x}} + \sin\theta\sin\varphi\,\widehat{\mathbf{y}} + \cos\theta\,\widehat{\mathbf{z}} \\ \widehat{\boldsymbol{\theta}} = \cos\theta\cos\varphi\,\widehat{\mathbf{x}} + \cos\theta\sin\varphi\,\widehat{\mathbf{y}} - \sin\theta\,\widehat{\mathbf{z}} \\ \widehat{\boldsymbol{\varphi}} = -\sin\varphi\,\widehat{\mathbf{x}} + \cos\varphi\,\widehat{\mathbf{y}} \end{cases} \tag{A.1}$$

Alternatively, it is possible to use **UV spherical coordinates**, which are defined in the following way:

$$u(\theta, \varphi) = \sin\theta\cos\varphi, \quad v(\theta, \varphi) = \sin\theta\sin\varphi. \tag{A.2}$$

To conclude this brief overview, it is worth defining the *elementary area* of a spherical surface element $\mathrm{d}\Sigma$ and the *elementary solid angle* $\mathrm{d}\Omega$:

$$\mathrm{d}\Sigma = r^2 \sin\theta\,\mathrm{d}\theta\,\mathrm{d}\varphi, \quad \mathrm{d}\Omega = \frac{\mathrm{d}\Sigma}{r^2} = \sin\theta\,\mathrm{d}\theta\,\mathrm{d}\varphi. \tag{A.3}$$

To assess the performance of an antenna, it is usually sufficient to focus only on the **Far-Field (FF)** or **Fraunhofer region** $\mathcal{F}$, whose definition is derived from the formal solution of the

Maxwell's equations:

$$\mathcal{F} := \left\{ \mathbf{r} \in \mathbb{R}^3 \ : \ r \gg \lambda, r \gg D, r > \frac{2D^2}{\lambda} \right\}, \tag{A.4}$$

where:

- D is the characteristic size of the antenna;
- λ is the wavelength of the transmitted wave.

It is important to point out that, when the FF condition holds, the electric and magnetic fields radiated by the antenna, namely $\mathbf{r} \mapsto \mathbf{E}(\mathbf{r})$ and $\mathbf{r} \mapsto \mathbf{H}(\mathbf{r})$, satisfy the so-called FF impedance relations:

$$\begin{cases} \mathbf{H}(\mathbf{r}) \simeq \dfrac{1}{Z_0} \hat{\mathbf{r}} \wedge \mathbf{E}(\mathbf{r}) \\ \mathbf{E}(\mathbf{r}) \simeq -Z_0 \hat{\mathbf{r}} \wedge \mathbf{H}(\mathbf{r}) \end{cases} \tag{A.5}$$

where

$$Z_0 := \sqrt{\frac{\mu_0}{\epsilon_0}} \equiv \mu_0 c_0 \equiv \frac{1}{c_0 \epsilon_0} = 120\pi\,\Omega \approx 377\,\Omega \tag{A.6}$$

is a fundamental constant known as *free-space impedance*.

In practice, the far-field $\mathbf{E}$ and $\mathbf{H}$ are linearly dependent: if one of the two vectors is known, the other can be automatically derived as well. Two further advantages of operating in FF are the following:

1. The radiation pattern is stable, i.e., it does not change significantly with distance, allowing for accurate characterizations of gain, directivity, and efficiency.
2. Near-field complexities can be neglected: in fact, the near-field region is dominated by reactive fields, that do not radiate efficiently but instead store and exchange energy with the antenna, which requires a more intricate analysis.

A.1.2 Radiation Pattern Parameters

Antenna parameters are commonly defined according to the **IEEE Standard for Definitions of Terms for Antennas**, a.k.a. **IEEE Std 145™**. In this work, collisions with the standard have been avoided as much as possible and any exception has been clearly noted.

The fundamental quantities that must be defined to characterize the **far-field radiation** of an antenna are listed below.

Directivity. This quantity is denoted with $(\theta, \varphi) \mapsto d(\theta, \varphi)$ and it is used to characterize the amplitude of the radiated field in a given direction. It is the distribution function of the power density $S(r, \theta, \varphi)$ over angular directions, normalized over the average power density $S_{\text{av}}(r)$ over a sphere of radius r:

$$d(\theta, \varphi) := \frac{S(r, \theta, \varphi)}{S_{\text{av}}(r)}, \tag{A.7}$$

where

$$S(r, \theta, \varphi) = \left. \frac{\mathrm{d}P}{\mathrm{d}\Sigma} \right|_{(r,\theta,\varphi)} \quad \text{and} \quad S_{\text{av}}(r) = \frac{1}{4\pi r^2} \iint_{4\pi} S(r, \theta, \varphi)\, \mathrm{d}\Omega \equiv \frac{P_{\text{rad}}}{4\pi r^2}. \tag{A.8}$$

By construction, the expression of the electric field can be separated into a radial and angular component, as Equation (A.16) clearly shows; therefore, the dependency on r is the same for $S(r,\theta,\varphi)$ and $S_{\text{av}}(r)$ and the directivity eventually turns out to depend only on the angular variables.

Furthermore, exploiting Equation (A.8), the definition of directivity can be rewritten in an alternative form, which can be useful for practical calculations but is not compliant with IEEE standard:

$$d(\theta,\varphi) = \frac{\left.\dfrac{\mathrm{d}P}{\mathrm{d}\Sigma}\right|_{(r,\theta,\varphi)}}{\dfrac{P_{\text{rad}}}{4\pi r^2}}. \tag{A.9}$$

Radiation efficiency. This quantity, denoted with η, is the ratio between the radiated power P_{rad} and the accepted power P_{in}:

$$\eta := \frac{P_{\text{rad}}}{P_{\text{in}}} \leq 1. \tag{A.10}$$

Gain. This quantity is denoted with $(\theta,\varphi) \mapsto g(\theta,\varphi)$ and it is a measure of how well the input power is converted to radiated field strength over a given direction:

$$g(\theta,\varphi) := \eta d(\theta,\varphi). \tag{A.11}$$

Directivity and gain satisfy a well-known normalization relation, that can be easily derived from Equation (A.8):

$$\frac{1}{S_{\text{av}}(r)} \iint_{4\pi} S(r,\theta,\varphi)\,\mathrm{d}\Omega = \int_0^{\pi}\int_0^{2\pi} \underbrace{\frac{S(r,\theta,\varphi)}{S_{\text{av}}(r)}}_{=d(\theta,\varphi)} r^2 \sin\theta\,\mathrm{d}\theta\,\mathrm{d}\varphi \equiv 4\pi r^2 \tag{A.12}$$

$$\int_0^{\pi}\int_0^{2\pi} d(\theta,\varphi)\sin\theta\,\mathrm{d}\theta\,\mathrm{d}\varphi = 4\pi \tag{A.13a}$$

$$\int_0^{\pi}\int_0^{2\pi} g(\theta,\varphi)\sin\theta\,\mathrm{d}\theta\,\mathrm{d}\varphi = 4\pi\eta \tag{A.13b}$$

It is also important to point out that, generally, *maximum* directivity and gain are implied, i.e., in the direction of maximum radiation $(\theta_{\text{max}},\varphi_{\text{max}})$:

$$\mathcal{D} := \max_{\theta,\varphi} d(\theta,\varphi) \equiv d(\theta_{\text{max}},\varphi_{\text{max}}) \tag{A.14a}$$

$$\mathcal{G} := \max_{\theta,\varphi} g(\theta,\varphi) \equiv g(\theta_{\text{max}},\varphi_{\text{max}}) \tag{A.14b}$$

Radiation pattern. The normalized radiation pattern of an antenna $(\theta,\varphi) \mapsto e(\theta,\varphi)$ is defined as follows:

$$e(\theta,\varphi) \equiv \|\mathbf{e}(\theta,\varphi)\| := \frac{g(\theta,\varphi)}{\mathcal{G}} \tag{A.15}$$

It can be shown that the far-field radiation pattern of any source in a lossless medium can be factored as

$$\mathbf{E}(\mathbf{r}) = g(r)\,\mathbf{e}(\theta,\varphi) \equiv \frac{\exp(-\mathrm{j}k_0 r)}{4\pi r}\,\mathbf{e}(\theta,\varphi), \tag{A.16}$$

where:

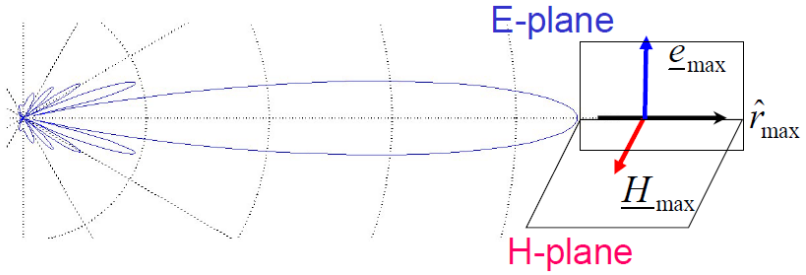


Figure A.1: Graphical representation of E and H planes for a linearly polarized antenna [50].

- $r \mapsto g(r)$ is the *universal spherical wave* or *Green function*, which does not depend on the physical structure of the antenna but only on the radial coordinate r;
- $(\theta, \varphi) \mapsto \mathbf{e}(\theta, \varphi)$ is the *radiation vector*, whose expression is determined by the features of the antenna of interest.

Side Lobe Level (SLL). Amplitude of the largest side lobe relative to the maximum. Side lobes are a very important feature of a directive antenna because they are an evidence of diffraction.

Back Lobe Level (BLL). Amplitude of the largest reflected lobe relative to the maximum.

First Null Beam Width (FNBW). Angular amplitude of the main beam emitted by an antenna, from maximum to first null.

Half Power Beam Width (HPBW). Also known as 3 dB beam width, it is the angular amplitude of the region around the maximum where $P/P_{max} > -3\,\text{dB}$.

A.2 Principal Planes

Normally, the radiation pattern of a *linearly polarized* antenna is not measured along all directions, but only along two *principal planes* that are considered particularly relevant.

Preliminarily, let us define $(\theta_{max}, \varphi_{max})$ as the angles (θ, φ) that maximize the magnitude of the radiation vector:

$$U(\theta, \varphi) := \|\mathbf{e}(\theta, \varphi)\|^2 \implies U(\theta_{max}, \varphi_{max}) = \max_{\theta,\varphi} \{U(\theta, \varphi)\}\,. \tag{A.17}$$

The principal planes, whose simplified graphical representation is shown in Figure A.1, can now be defined as follows:

E plane. It is the plane defined by

- the direction of maximum radiation:

$$\hat{\mathbf{r}}_{max} := \hat{\mathbf{r}}(\theta_{max}, \varphi_{max}); \tag{A.18}$$

- the direction of the electric field along the direction of maximum radiation:

$$\hat{\mathbf{e}}_{max} := \frac{\mathbf{e}(\theta_{max}, \varphi_{max})}{\|\mathbf{e}(\theta_{max}, \varphi_{max})\|}. \tag{A.19}$$

The normal to the E plane can thus be calculated as

$$\widehat{\mathbf{n}}_{\mathsf{e}} = \widehat{\mathbf{r}}_{\mathsf{max}} \wedge \mathbf{e}_{\mathsf{max}}. \tag{A.20}$$

H plane. It is the plane defined by

- the direction of maximum radiation $\widehat{\mathbf{r}}_{\mathsf{max}}$, defined in (A.18);
- the direction of the magnetic field along the direction of maximum radiation:

$$\mathbf{h}_{\mathsf{max}} := \frac{\mathbf{h}(\theta_{\mathsf{max}}, \varphi_{\mathsf{max}})}{\|\mathbf{h}(\theta_{\mathsf{max}}, \varphi_{\mathsf{max}})\|} \equiv \widehat{\mathbf{r}}_{\mathsf{max}} \wedge \mathbf{e}_{\mathsf{max}} \equiv \widehat{\mathbf{n}}_{\mathsf{e}}. \tag{A.21}$$

The normal to the H plane can thus be calculated as

$$\widehat{\mathbf{n}}_{\mathsf{h}} = \widehat{\mathbf{r}}_{\mathsf{max}} \wedge \mathbf{h}_{\mathsf{max}}. \tag{A.22}$$

It is important to highlight the following facts:

- The definition of $(\theta_{\mathsf{max}}, \varphi_{\mathsf{max}})$ depends on the choice of the coordinate system, but the E and H planes of an antenna are defined regardless of the reference frame;
- The principal planes can only be defined for linearly polarized antennas because, in the case of elliptically or circularly polarized antennas, the direction of maximum radiation is time-dependent, so the definition of the planes would not be univocal.

A.3 Equivalence Theorem

The equivalence theorem is a fundamental concept in electromagnetic theory that allows the representation of the radiation pattern of an antenna in terms of equivalent sources.

By strategically placing equivalent electric and magnetic currents on a boundary surface, this theorem enables the characterization of electromagnetic fields outside the source region without requiring detailed knowledge of the internal structure.

Let $\Omega_{\mathsf{a}} \subset \mathbb{R}^3$ be a volume with surface Σ_{a}, enclosing a known electric current distribution $\mathbf{J}_{\mathsf{src}}$. The electromagnetic fields $\mathbf{E}$ and $\mathbf{H}$ satisfy the following equations:

$$\nabla \wedge \mathbf{E}(\mathbf{r}) = \mathrm{j}\omega\mu\mathbf{H}(\mathbf{r}) + \delta(\mathbf{r} - \mathbf{r}_{\Sigma})\,(\mathbf{E}(\mathbf{r}) \times \widehat{\mathbf{n}}) \tag{A.23a}$$

$$\nabla \wedge \mathbf{H}(\mathbf{r}) = \mathrm{j}\omega\epsilon\mathbf{E}(\mathbf{r}) + \mathbf{J}_{\mathsf{src}}(\mathbf{r}) + \delta(\mathbf{r} - \mathbf{r}_{\Sigma})\,(\widehat{\mathbf{n}} \wedge \mathbf{H}(\mathbf{r})) \tag{A.23b}$$

Here, $\widehat{\mathbf{n}}$ is the outward normal to Σ_{a}, and $\delta(\mathbf{P} - \mathbf{P}_{\Sigma})$ represents the Dirac delta function, ensuring that the equivalent sources only exist on the boundary surface Σ_{a}.

A key consequence of the equivalence theorem states that the electromagnetic fields outside Ω can be described solely in terms of **electric and magnetic surface currents**:

$$\mathbf{J}_{\mathsf{s}}(\mathbf{r}) := \widehat{\mathbf{n}} \wedge \mathbf{H}(\mathbf{r}), \quad \mathbf{M}_{\mathsf{s}}(\mathbf{r}) := \mathbf{E}(\mathbf{r}) \wedge \widehat{\mathbf{n}}. \tag{A.24}$$

These equivalent surface currents act as virtual sources, effectively replacing the original volume distribution.

Equations (A.23a) and (A.23b) can thus be rewritten in a more compact form:

$$\nabla \wedge \mathbf{E}(\mathbf{r}) = \mathrm{j}\omega\mu\mathbf{H}(\mathbf{r}) + \delta(\mathbf{r} - \mathbf{r}_{\Sigma})\mathbf{J}_{\mathsf{s}}(\mathbf{r}) \tag{A.25a}$$

$$\nabla \wedge \mathbf{H}(\mathbf{r}) = \mathrm{j}\omega\epsilon\mathbf{E}(\mathbf{r}) + \mathbf{J}_{\mathrm{src}}(\mathbf{r}) + \delta(\mathbf{r} - \mathbf{r}_\Sigma)\mathbf{M}_{\mathrm{s}}(\mathbf{r}) \tag{A.25b}$$

Exploiting the invariance of the ME with respect to transformations that map **E** into **H** and **H** into **E** and the linearity of the electromagnetic field, it is possible to express the **electric** and **magnetic radiation integrals**, denoted with $\widetilde{\mathbf{J}}$ and $\widetilde{\mathbf{M}}$, as follows:

$$\widetilde{\mathbf{J}}(\theta,\varphi) = \iint_{\Sigma_{\mathrm{a}}} \mathbf{J}_{\mathrm{s}}(\mathbf{r}_\Sigma) \exp\left(\mathrm{j}k_0\,\widehat{\mathbf{r}}(\theta,\varphi)\cdot\mathbf{r}_\Sigma\right) \mathrm{d}\Sigma \tag{A.26a}$$

$$\widetilde{\mathbf{M}}(\theta,\varphi) = \iint_{\Sigma_{\mathrm{a}}} \mathbf{M}_{\mathrm{s}}(\mathbf{r}_\Sigma) \exp\left(\mathrm{j}k_0\,\widehat{\mathbf{r}}(\theta,\varphi)\cdot\mathbf{r}_\Sigma\right) \mathrm{d}\Sigma \tag{A.26b}$$

where k_0 is the free-space propagation constant and $\widehat{\mathbf{r}}(\theta,\varphi)$ is the radial unit vector, which can be expressed as function of the spherical angles θ and φ:

$$\widehat{\mathbf{r}}(\theta,\varphi) = \sin\theta\cos\varphi\,\widehat{\mathbf{x}} + \sin\theta\sin\varphi\,\widehat{\mathbf{y}} + \cos\theta\,\widehat{\mathbf{z}} \tag{A.27}$$

Once $\widetilde{\mathbf{J}}$ and $\widetilde{\mathbf{M}}$ are known, it is possible to calculate the components of the electric field along $\widehat{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\varphi}}$, namely $(\theta,\varphi) \mapsto e_\theta(\theta,\varphi)$ and $(\theta,\varphi) \mapsto e_\varphi(\theta,\varphi)$:

$$\begin{cases} e_\theta(\theta,\varphi) = -\mathrm{j}\omega\mu\widehat{\boldsymbol{\theta}}\cdot\widetilde{\mathbf{J}}(\theta,\varphi) - \mathrm{j}k_0\widehat{\boldsymbol{\varphi}}\cdot\widetilde{\mathbf{M}}(\theta,\varphi) \\ e_\varphi(\theta,\varphi) = -\mathrm{j}\omega\mu\widehat{\boldsymbol{\varphi}}\cdot\widetilde{\mathbf{J}}(\theta,\varphi) + \mathrm{j}k_0\widehat{\boldsymbol{\theta}}\cdot\widetilde{\mathbf{M}}(\theta,\varphi) \end{cases} \tag{A.28}$$

To find the magnetic field $\mathbf{h}(\theta,\varphi)$, it is simply necessary to apply the *impedance relation*:

$$\mathbf{e}(\theta,\varphi) = e_\theta(\theta,\varphi)\widehat{\boldsymbol{\theta}} + e_\varphi(\theta,\varphi)\widehat{\boldsymbol{\varphi}}, \quad \mathbf{h}(\theta,\varphi) = \frac{1}{Z_0}\,\widehat{\mathbf{r}} \wedge \mathbf{e}(\theta,\varphi), \tag{A.29}$$

Appendix B

MATLAB Analytical Model

B.1 Maxwell Garnett Model

```
%% Effective permittivity of a mixture of 2 materials according to the MG model
% er1, tand1 = relative dielectric constant and loss tangent of inclusion material
% er2, tand2 = relative dielectric constant and loss tangent of host material
% f1 = volume fraction of the inclusion (Vol_inclusion / Vol_tot)
function e_eff = eff_permittivity(er1, tand1, er2, tand2, f1)
    e1 = er1 .* ( 1 - 1i .* tand1 );
    e2 = er2 .* ( 1 - 1i .* tand2 );
    num = e2 .* ( ( 2 * e2 ) + e1 + ( 2 * f1 .* ( e1 - e2 ) ) );
    den = ( 2 * e2 ) + e1 - ( f1 .* ( e1 - e2 ) );
    e_eff = num ./ den;
end
```

B.2 Floquet Modal Analysis

B.2.1 Floquet Modal Wavenumber

```
%% Floquet modal wavenumber [m^-1]
% a, b = periodicities along x and y [m]
% er_med = RELATIVE dielectric constant of the material (NOT multiplied by eps0)
function kz = floquet_modes(k0, theta, phi, m, n, a, b, er_med)
    kx = ( k0 .* sin( deg2rad(theta) ) .* cos( deg2rad(phi) ) ) + ( ( 2*pi .* m) ./ a );
    ky = ( k0 .* sin( deg2rad(theta) ) .* sin( deg2rad(phi) ) ) + ( ( 2*pi .* n) ./ b );
    kz = sqrt( ( k0.^2 .* er_med ) - kx.^2 - ky.^2 );
end
```

B.2.2 Floquet Modal Admittance

```
%% Floquet modal admittance (1 = TE, 2 = TM)
% eps_med, mu_med = ABSOLUTE dielectric constant and magnetic permittivity of the medium
function Y = floquet_Y(omega0, kz, eps_med, mu_med, mode)
    if mode == 1 % TE mode
        Y = kz ./ (omega0 .* mu_med);
    elseif mode == 2 % TM mode
        Y = (omega0 .* eps_med) ./ kz;
    end
end
```

B.3 ABCD Formalism

B.3.1 ABCD Parameters of Useful Two-port Networks

While **Z**, **Y** and **S** parameter representations can be used to represent and characterize an arbitrarily complex microwave network, these matrix representations are impractical when dealing with a cascade of N two-port networks.

In this case, the **transmission** or **ABCD matrix** is typically used. Let us consider the two-port network shown in Figure B.1 and let V_i and I_i be the voltages and currents at ports $i = 1, 2$: V_1, V_2 and I_1 are defined according to the usual conventions of microwave electronics, while the sign of I_2 is changed since this current is flowing *out of* port 2.

This trick is useful because in a cascade network the current flowing out of one component is the same that flows into the adjacent one.

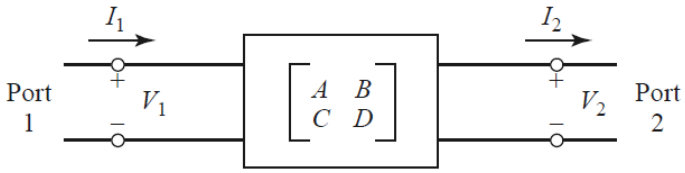


Figure B.1: Black-box two-port network [42].

The **ABCD** matrix of the two-port is defined as follows:

$$\begin{pmatrix} V_1 \\ I_1 \end{pmatrix} = \begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} V_2 \\ I_2 \end{pmatrix}, \tag{B.1}$$

meaning that

$$A = \left.\frac{V_1}{V_2}\right|_{I_2=0} \qquad B = \left.\frac{V_1}{I_2}\right|_{V_2=0} \tag{B.2}$$

$$C = \left.\frac{I_1}{V_2}\right|_{I_2=0} \qquad D = \left.\frac{I_1}{I_2}\right|_{V_2=0}. \tag{B.3}$$

From the above relations, it is possible to notice that A and C are dimensionless, B has the dimensions of an impedance and D has the dimensions of an admittance.

Let us now consider the cascade of two two-port networks shown in Figure B.2.

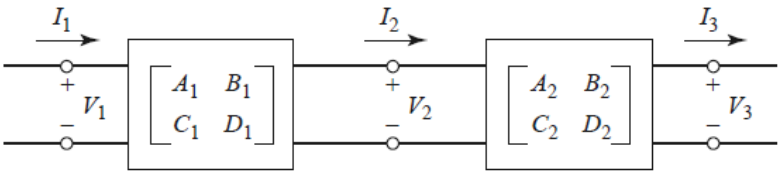


Figure B.2: Cascade of two black-box two-port networks [42].

Exploiting Equation (B.1), it is possible to write that

$$\begin{pmatrix} V_1 \\ I_1 \end{pmatrix} = \begin{pmatrix} A_1 & B_1 \\ C_1 & D_1 \end{pmatrix} \begin{pmatrix} V_2 \\ I_2 \end{pmatrix}, \quad \begin{pmatrix} V_2 \\ I_2 \end{pmatrix} = \begin{pmatrix} A_2 & B_2 \\ C_2 & D_2 \end{pmatrix} \begin{pmatrix} V_3 \\ I_3 \end{pmatrix}, \tag{B.4}$$

which can be rewritten as follows:

$$\begin{pmatrix} V_1 \\ I_1 \end{pmatrix} = \begin{pmatrix} A_1 & B_1 \\ C_1 & D_1 \end{pmatrix} \begin{pmatrix} A_2 & B_2 \\ C_2 & D_2 \end{pmatrix} \begin{pmatrix} V_3 \\ I_3 \end{pmatrix} \tag{B.5}$$

In other words, the ABCD representation of the entire network is obtained by simply multiplying the ABCD matrices of the individual components:

$$\mathbf{ABCD} \equiv \begin{pmatrix} A & B \\ C & D \end{pmatrix} = \begin{pmatrix} A_1 & B_1 \\ C_1 & D_1 \end{pmatrix} \begin{pmatrix} A_2 & B_2 \\ C_2 & D_2 \end{pmatrix} \tag{B.6}$$

The usefulness of this property lies in the fact that it can be easily extended to a cascade of N two-port networks:

$$\mathbf{ABCD} \equiv \begin{pmatrix} A & B \\ C & D \end{pmatrix} = \prod_{i=1}^{N} \begin{pmatrix} A_i & B_i \\ C_i & D_i \end{pmatrix} \tag{B.7}$$

It is extremely important to highlight that the matrices must be multiplied in the same order in which the networks are arranged, i.e., moving from port 1 to port 2, because matric product is not, in general, commutative.

The transmission matrix of basic two-ports can be easily found by inspection, as shown below for the case of a uniform transmission line.

Furthermore, it is possible to derive conversion formulas between transmission coefficients and other two-port network parameters, as shown in Appendix B.3.2 for the transmission-to-scattering conversion.

B.3.1.1 Transmission Line

Let us consider a TL with characteristic impedance Z_∞ and electrical length Θ. Its transmission matrix can be evaluated as follows:

$$\mathbf{ABCD} = \begin{pmatrix} \cos\Theta & \mathrm{j}Z_\infty \sin\Theta \\ \mathrm{j}Y_\infty \sin\Theta & \cos\Theta \end{pmatrix} \tag{B.8}$$

Equation (B.8) has been implemented in the following MATLAB function:

```
%% ABCD parameters of a transmission line
% Z0 = characteristic impedance of the line [Ohm]
% Theta = electrical length of the line [rad]
function [A, B, C, D] = ABCD_TL(Z0, Theta)
    A = cos(Theta);
    B = ( 1i .* Z0 ) .* sin(Theta);
    C = ( 1i ./ Z0 ) .* sin(Theta);
    D = cos(Theta);
end
```

B.3.2 Conversion from ABCD to S Parameters

The conversion formulas from transmission to scattering parameters are the following, having defined $Z_{r1}, Z_{r2} \in \mathbb{C}$ as the reference impedances of the two ports:

$$S_{11} = \frac{AZ_{r2} + B - CZ_{r1}^{\star}Z_{r2} - DZ_{r1}^{\star}}{AZ_{r2} + B + CZ_{r1}^{\star}Z_{r2} + DZ_{r1}^{\star}} \tag{B.9a}$$

$$S_{12} = \frac{2(AD - BC)\sqrt{Z_{r1}Z_{r2}}}{AZ_{r2} + B + CZ_{r1}^{\star}Z_{r2} + DZ_{r1}^{\star}} \tag{B.9b}$$

$$S_{21} = \frac{2\sqrt{Z_{r1}Z_{r2}}}{AZ_{r2} + B + CZ_{r1}^{\star}Z_{r2} + DZ_{r1}^{\star}} \tag{B.9c}$$

$$S_{22} = \frac{-AZ_{r2}^{\star} + B - CZ_{r1}Z_{r2}^{\star} + DZ_{r1}}{AZ_{r2} + B + CZ_{r1}^{\star}Z_{r2} + DZ_{r1}^{\star}} \tag{B.9d}$$

The above equations can be implemented in MATLAB, either using the following subroutine or exploiting the RF toolbox.

```
%% Conversion from ABCD to S parameters of a two-port network
% Zr1, Zr2 = reference impedances of the two ports [Ohm]
% A, B, C, D = ABCD parameters of the two-port network
function [S11, S12, S21, S22] = ABCD2S(Zr1, Zr2, A, B, C, D)
    S11 = ( ( Zr2 .* A ) + B - ( Zr1 .* Zr2 .* C ) - ( Zr1 .* D ) ) ./ ( ( Zr2 .* A ) +
    B + ( Zr1 .* Zr2 .* C ) + ( Zr1 .* D ) );
    S12 = sqrt(Zr2 ./ Zr1) .* ( 2 * Zr1 .* ( A .* D - B .* C ) ) ./ ( ( Zr2 .* A ) + B +
    ( Zr1 .* Zr2 .* C ) + ( Zr1 .* D ) );
    S21 = sqrt(Zr2 ./ Zr1) .* ( 2 * Zr1 ) ./ ( ( Zr2 .* A ) + B + ( Zr1 .* Zr2 .* C ) +
    ( Zr1 .* D ) );
    S22 = ( -( Zr2 .* A ) + B - ( Zr1 .* Zr2 .* C ) + ( Zr1 .* D ) ) ./ ( ( Zr2 .* A ) +
    B + ( Zr1 .* Zr2 .* C ) + ( Zr1 .* D ) );
end
```

B.4 S Parameters of the Unit Cell

B.4.1 Single-layer UC, Square Hole (UC-1)

```
%% S parameters unit cell, square hole
% theta, phi = scan angles [deg]
% f0 = centerband frequency [Hz]
% er, tand = relative dielectric constant and loss tangent of the material
% W = periodicity along x and y [m]
% T, d = cell height and hole size [m]
% Tf = Floquet port thickness [m]
function [S11TE, S21TE, S11TM, S21TM] = UC_TA_noGlass_sqh(theta, phi, m, n, f0, er,
tand, W, T, Tf, d)
    c0 = physconst('LightSpeed'); % speed of light in vacuum [m/s]
    eps0 = 8.8541878176e-12; % dielectric constant in vacuum [F/m]
    mu0 = pi * 4e-7; % magnetic permittivity in vacuum [H/m]

    omega0 = 2 * pi .* f0; % [rad s^-1]
    lambda0 = c0 ./ f0; % [m]
    k0 = 2 * pi ./ lambda0; % [m^-1]

    f1 = (d ./ W).^2; % volume fraction
    eps_eff = eff_permittivity(1, 0, er, tand, f1); % effective permittivity

    % modal wavenumbers
    kz_reference = floquet_modes(k0, theta, phi, m, n, W, W, 1);
    kz = floquet_modes(k0, theta, phi, m, n, W, W, eps_eff);

    % modal impedances
    Zr_TE = ( floquet_Y(omega0, kz_reference, eps0, mu0, 1) ).^(-1);
    Zr_TM = ( floquet_Y(omega0, kz_reference, eps0, mu0, 2) ).^(-1);
```

```
    ZC_TE = ( floquet_Y(omega0, kz, eps0*eps_eff, mu0, 1) ).^(-1);
    ZC_TM = ( floquet_Y(omega0, kz, eps0*eps_eff, mu0, 2) ).^(-1);

    Theta_air = kz_reference .* Tf;
    Theta_cell = kz .* T; % electrical length of the line
    [A_TE_air, B_TE_air, C_TE_air, D_TE_air] = ABCD_TL(Zr_TE, Theta_air); % ABCD matrix
    (TE)
    [A_TM_air, B_TM_air, C_TM_air, D_TM_air] = ABCD_TL(Zr_TM, Theta_air); % ABCD matrix
    (TM)
    [A_TE_cell, B_TE_cell, C_TE_cell, D_TE_cell] = ABCD_TL(ZC_TE, Theta_cell); % ABCD
    matrix (TE)
    [A_TM_cell, B_TM_cell, C_TM_cell, D_TM_cell] = ABCD_TL(ZC_TM, Theta_cell); % ABCD
    matrix (TM)

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE_air, B_TE_air, C_TE_air, D_TE_air,
    A_TE_cell, B_TE_cell, C_TE_cell, D_TE_cell);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM_air, B_TM_air, C_TM_air, D_TM_air,
    A_TM_cell, B_TM_cell, C_TM_cell, D_TM_cell);

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_air,
    B_TE_air, C_TE_air, D_TE_air);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_air,
    B_TM_air, C_TM_air, D_TM_air);

    [S11TE, ~, S21TE, ~] = ABCD2S(Zr_TE, Zr_TE, A_TE, B_TE, C_TE, D_TE); % S matrix (TE)
    [S11TM, ~, S21TM, ~] = ABCD2S(Zr_TM, Zr_TM, A_TM, B_TM, C_TM, D_TM); % S matrix (TM)
end
```

B.4.2 Single-layer UC, Circular Hole (UC-2)

```
%% S parameters unit cell, circular hole
% theta, phi = scan angles [deg]
% f0 = centerband frequency [Hz]
% er, tand = relative dielectric constant and loss tangent of the material
% W = periodicity along x and y [m]
% T, d = cell height and hole diameter [m]
% Tf = Floquet port thickness [m]
function [S11TE, S21TE, S11TM, S21TM] = UC_TA_noGlass_crh(theta, phi, m, n, f0, er,
tand, W, T, Tf, d)
    c0 = physconst('LightSpeed'); % speed of light in vacuum [m/s]
    eps0 = 8.8541878176e-12; % dielectric constant in vacuum [F/m]
    mu0 = pi * 4e-7; % magnetic permittivity in vacuum [H/m]

    omega0 = 2 * pi .* f0; % [rad s^-1]
    lambda0 = c0 ./ f0; % [m]
    k0 = 2 * pi ./ lambda0; % [m^-1]

    f1 = ( pi * (d/2).^2 ) ./ ( W.^2 ); % volume fraction
    eps_eff = eff_permittivity(1, 0, er, tand, f1); % effective permittivity

    % modal wavenumbers
    kz_reference = floquet_modes(k0, theta, phi, m, n, W, W, 1);
    kz = floquet_modes(k0, theta, phi, m, n, W, W, eps_eff);

    % modal impedances
    Zr_TE = ( floquet_Y(omega0, kz_reference, eps0, mu0, 1) ).^(-1);
    Zr_TM = ( floquet_Y(omega0, kz_reference, eps0, mu0, 2) ).^(-1);
```

```
    ZC_TE = ( floquet_Y(omega0, kz, eps0*eps_eff, mu0, 1) ).^(-1);
    ZC_TM = ( floquet_Y(omega0, kz, eps0*eps_eff, mu0, 2) ).^(-1);

    Theta_air = kz_reference .* lambda0 / 4;
    Theta_cell = kz .* T; % electrical length of the line
    [A_TE_air, B_TE_air, C_TE_air, D_TE_air] = ABCD_TL(Zr_TE, Theta_air); % ABCD matrix
    (TE)
    [A_TM_air, B_TM_air, C_TM_air, D_TM_air] = ABCD_TL(Zr_TM, Theta_air); % ABCD matrix
    (TM)
    [A_TE_cell, B_TE_cell, C_TE_cell, D_TE_cell] = ABCD_TL(ZC_TE, Theta_cell); % ABCD
    matrix (TE)
    [A_TM_cell, B_TM_cell, C_TM_cell, D_TM_cell] = ABCD_TL(ZC_TM, Theta_cell); % ABCD
    matrix (TM)

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE_air, B_TE_air, C_TE_air, D_TE_air,
    A_TE_cell, B_TE_cell, C_TE_cell, D_TE_cell);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM_air, B_TM_air, C_TM_air, D_TM_air,
    A_TM_cell, B_TM_cell, C_TM_cell, D_TM_cell);

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_air,
    B_TE_air, C_TE_air, D_TE_air);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_air,
    B_TM_air, C_TM_air, D_TM_air);

    [S11TE, ~, S21TE, ~] = ABCD2S(Zr_TE, Zr_TE, A_TE, B_TE, C_TE, D_TE); % S matrix (TE)
    [S11TM, ~, S21TM, ~] = ABCD2S(Zr_TM, Zr_TM, A_TM, B_TM, C_TM, D_TM); % S matrix (TM)
end
```

B.4.3 Three-layer UC, Square Hole (UC-3)

```
%% S parameters unit cell, square hole + glass layer
% f0 = centerband frequency
% er, tand = relative dielectric constant and loss tangent of the material
% W = periodicity along x and y [m]
% T, d = cell height and hole size [m]
% Tf = Floquet port thickness [m]
function [S11TE, S21TE, S11TM, S21TM] = UC_TA_glass_sqh(theta, phi, m, n, f0, er, tand,
W, T, Tf, d)
    c0 = physconst('LightSpeed'); % speed of light in vacuum [m/s]
    eps0 = 8.8541878176e-12; % dielectric constant in vacuum [F/m]
    mu0 = pi * 4e-7; % magnetic permittivity in vacuum [H/m]
    er_g = 6.4; % relative dielectric constant glass
    tand_g = 0.027; % loss tangent glass
    T_g = 4.3e-3; % glass thickness [m]

    omega0 = 2 * pi .* f0; % [rad s^-1]
    lambda0 = c0 ./ f0; % [m]
    k0 = 2 * pi ./ lambda0; % [m^-1]

    f1 = (d ./ W).^2; % volume fraction
    eps_sqh = eff_permittivity(1, 0, er, tand, f1); % effective permittivity
    eps_g = eff_permittivity(1, 0, er_g, tand_g, 0); % glass effective permittivity

    % modal wavenumbers
    kz_reference = floquet_modes(k0, theta, phi, m, n, W, W, 1);
    kz_sqh = floquet_modes(k0, theta, phi, m, n, W, W, eps_sqh);
    kz_g = floquet_modes(k0, theta, phi, m, n, W, W, eps_g);
```

```
27
28      % modal impedances
29      Zr_TE = ( floquet_Y(omega0, kz_reference, eps0, mu0, 1) ).^(-1);
30      Zr_TM = ( floquet_Y(omega0, kz_reference, eps0, mu0, 2) ).^(-1);
31      ZC_TE_sqh = ( floquet_Y(omega0, kz_sqh, eps0*eps_sqh, mu0, 1) ).^(-1);
32      ZC_TM_sqh = ( floquet_Y(omega0, kz_sqh, eps0*eps_sqh, mu0, 2) ).^(-1);
33      ZC_TE_g = ( floquet_Y(omega0, kz_g, eps0*eps_g, mu0, 1) ).^(-1);
34      ZC_TM_g = ( floquet_Y(omega0, kz_g, eps0*eps_g, mu0, 2) ).^(-1);
35
36      Theta_air = kz_reference .* Tf;
37      Theta_sqh = kz_sqh .* T/2; % electrical length square hole
38      Theta_g = kz_g * T_g; % electrical length glass layer
39
40      [A_TE_air, B_TE_air, C_TE_air, D_TE_air] = ABCD_TL(Zr_TE, Theta_air);
41      [A_TM_air, B_TM_air, C_TM_air, D_TM_air] = ABCD_TL(Zr_TM, Theta_air);
42      [A_TE_sqh, B_TE_sqh, C_TE_sqh, D_TE_sqh] = ABCD_TL(ZC_TE_sqh, Theta_sqh);
43      [A_TM_sqh, B_TM_sqh, C_TM_sqh, D_TM_sqh] = ABCD_TL(ZC_TM_sqh, Theta_sqh);
44      [A_TE_g, B_TE_g, C_TE_g, D_TE_g] = ABCD_TL(ZC_TE_g, Theta_g);
45      [A_TM_g, B_TM_g, C_TM_g, D_TM_g] = ABCD_TL(ZC_TM_g, Theta_g);
46
47      [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE_air, B_TE_air, C_TE_air, D_TE_air,
        A_TE_sqh, B_TE_sqh, C_TE_sqh, D_TE_sqh);
48      [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM_air, B_TM_air, C_TM_air, D_TM_air,
        A_TM_sqh, B_TM_sqh, C_TM_sqh, D_TM_sqh);
49
50      [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_g, B_TE_g,
        C_TE_g, D_TE_g);
51      [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_g, B_TM_g,
        C_TM_g, D_TM_g);
52
53      [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_sqh,
        B_TE_sqh, C_TE_sqh, D_TE_sqh);
54      [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_sqh,
        B_TM_sqh, C_TM_sqh, D_TM_sqh);
55
56      [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_air,
        B_TE_air, C_TE_air, D_TE_air);
57      [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_air,
        B_TM_air, C_TM_air, D_TM_air);
58
59      [S11TE, ~, S21TE, ~] = ABCD2S(Zr_TE, Zr_TE, A_TE, B_TE, C_TE, D_TE); % S matrix (TE)
60      [S11TM, ~, S21TM, ~] = ABCD2S(Zr_TM, Zr_TM, A_TM, B_TM, C_TM, D_TM); % S matrix (TM)
61  end
```

B.4.4 Three-layer UC, Circular Hole (UC-4)

```
1   %% S parameters unit cell, circular hole + glass layer
2   % f0 = centerband frequency
3   % er, tand = relative dielectric constant and loss tangent of the material
4   % W = periodicity along x and y [m]
5   % T, d = cell height and hole diameter [m]
6   % Tf = Floquet port thickness [m]
7   function [S11TE, S21TE, S11TM, S21TM] = UC_TA_glass_crh(theta, phi, m, n, f0, er, tand,
    W, T, Tf, d)
8       c0 = physconst('LightSpeed'); % speed of light in vacuum [m/s]
9       eps0 = 8.8541878176e-12; % dielectric constant in vacuum [F/m]
10      mu0 = pi * 4e-7; % magnetic permittivity in vacuum [H/m]
```

```
    er_g = 6.4; % relative dielectric constant glass
    tand_g = 0.027; % loss tangent glass
    T_g = 4.3e-3; % glass thickness [m]

    omega0 = 2 * pi .* f0; % [rad s^-1]
    lambda0 = c0 ./ f0; % [m]
    k0 = 2 * pi ./ lambda0; % [m^-1]

    f1 = ( pi * (d/2).^2 ) ./ ( W.^2 ); % volume fraction
    eps_crh = eff_permittivity(1, 0, er, tand, f1); % effective permittivity
    eps_g = eff_permittivity(1, 0, er_g, tand_g, 0); % glass effective permittivity

    % modal wavenumbers
    kz_reference = floquet_modes(k0, theta, phi, m, n, W, W, 1);
    kz_crh = floquet_modes(k0, theta, phi, m, n, W, W, eps_crh);
    kz_g = floquet_modes(k0, theta, phi, m, n, W, W, eps_g);

    % modal impedances
    Zr_TE = ( floquet_Y(omega0, kz_reference, eps0, mu0, 1) ).^(-1);
    Zr_TM = ( floquet_Y(omega0, kz_reference, eps0, mu0, 2) ).^(-1);
    ZC_TE_crh = ( floquet_Y(omega0, kz_crh, eps0*eps_crh, mu0, 1) ).^(-1);
    ZC_TM_crh = ( floquet_Y(omega0, kz_crh, eps0*eps_crh, mu0, 2) ).^(-1);
    ZC_TE_g = ( floquet_Y(omega0, kz_g, eps0*eps_g, mu0, 1) ).^(-1);
    ZC_TM_g = ( floquet_Y(omega0, kz_g, eps0*eps_g, mu0, 2) ).^(-1);

    Theta_air = kz_reference .* Tf;
    Theta_crh = kz_crh .* T/2; % electrical length circular hole
    Theta_g = kz_g * T_g; % electrical length glass layer

    [A_TE_air, B_TE_air, C_TE_air, D_TE_air] = ABCD_TL(Zr_TE, Theta_air);
    [A_TM_air, B_TM_air, C_TM_air, D_TM_air] = ABCD_TL(Zr_TM, Theta_air);
    [A_TE_crh, B_TE_crh, C_TE_crh, D_TE_crh] = ABCD_TL(ZC_TE_crh, Theta_crh);
    [A_TM_crh, B_TM_crh, C_TM_crh, D_TM_crh] = ABCD_TL(ZC_TM_crh, Theta_crh);
    [A_TE_g, B_TE_g, C_TE_g, D_TE_g] = ABCD_TL(ZC_TE_g, Theta_g);
    [A_TM_g, B_TM_g, C_TM_g, D_TM_g] = ABCD_TL(ZC_TM_g, Theta_g);

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE_air, B_TE_air, C_TE_air, D_TE_air,
    A_TE_crh, B_TE_crh, C_TE_crh, D_TE_crh);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM_air, B_TM_air, C_TM_air, D_TM_air,
    A_TM_crh, B_TM_crh, C_TM_crh, D_TM_crh);

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_g, B_TE_g,
    C_TE_g, D_TE_g);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_g, B_TM_g,
    C_TM_g, D_TM_g);

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_crh,
    B_TE_crh, C_TE_crh, D_TE_crh);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_crh,
    B_TM_crh, C_TM_crh, D_TM_crh);

    [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_air,
    B_TE_air, C_TE_air, D_TE_air);
    [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_air,
    B_TM_air, C_TM_air, D_TM_air);

    [S11TE, ~, S21TE, ~] = ABCD2S(Zr_TE, Zr_TE, A_TE, B_TE, C_TE, D_TE); % S matrix (TE)
    [S11TM, ~, S21TM, ~] = ABCD2S(Zr_TM, Zr_TM, A_TM, B_TM, C_TM, D_TM); % S matrix (TM)
end
```

B.4.5 Three-layer Tapered UC (UC-5)

```
%% S parameters unit cell, tapered sides
% f0 = centerband frequency
% er, tand = relative dielectric constant and loss tangent of the material
% W = periodicity along x and y [m]
% Htap, Hsqh = height of the taperede and holed part
% d = hole size [m]
function [S11TE, S21TE, S11TM, S21TM] = UC_TA_tapered(theta, phi, m, n, f0, er, tand, W,
Htap, Hsqh, d)
    c0 = physconst('LightSpeed'); % speed of light in vacuum [m/s]
    eps0 = 8.8541878176e-12; % dielectric constant in vacuum [F/m]
    mu0 = pi * 4e-7; % magnetic permittivity in vacuum [H/m]

    omega0 = 2 * pi .* f0; % [rad s^-1]
    lambda0 = c0 ./ f0; % [m]
    k0 = 2 * pi ./ lambda0; % [m^-1]

    T = (2 * Htap) + Hsqh;
    step_horiz = 1e-6; % height of each step [m]
    Nstep = Htap / step_horiz; % number of steps in the tapered part, each step is a
    uniform TL
    Vtot_step = step_horiz * W^2; % total volume of each step [m^3]
    step_vert = 0.5 * ( W - d ) / Nstep; % hole size of each step

    % Reference impedances and initial ABCD matrix (identity)
    kz_reference = floquet_modes(k0, theta, phi, m, n, W, W, 1);
    Zr_TE = ( floquet_Y(omega0, kz_reference, eps0, mu0, 1) ).^(-1);
    Zr_TM = ( floquet_Y(omega0, kz_reference, eps0, mu0, 2) ).^(-1);
    A_TE = 1; B_TE = 0; C_TE = 0; D_TE = 1;
    A_TM = 1; B_TM = 0; C_TM = 0; D_TM = 1;

    % Input tapering
    for i = 1:1:Nstep % for all the steps
        Vair_step = step_horiz .* ( W - 2 * i * step_vert ).^(2);
        f1_step = Vair_step ./ Vtot_step; % volume fraction
        eps_eff_step = eff_permittivity(1, 0, er, tand, f1_step); % effective relative
        dielectric constant

        kz = floquet_modes(k0, theta, phi, m, n, W, W, eps_eff_step);
        Theta = kz * step_horiz;
        ZC_TE = ( floquet_Y(omega0, kz, eps0*eps_eff_step, mu0, 1) ).^(-1);
        ZC_TM = ( floquet_Y(omega0, kz, eps0*eps_eff_step, mu0, 2) ).^(-1);

        [A_TE_new_layer, B_TE_new_layer, C_TE_new_layer, D_TE_new_layer] =
        ABCD_TL(ZC_TE, Theta);
        [A_TM_new_layer, B_TM_new_layer, C_TM_new_layer, D_TM_new_layer] =
        ABCD_TL(ZC_TM, Theta);

        [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE,
        A_TE_new_layer, B_TE_new_layer, C_TE_new_layer, D_TE_new_layer);
        [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM,
        A_TM_new_layer, B_TM_new_layer, C_TM_new_layer, D_TM_new_layer);
    end

    % Square hole
    f_sqh = (d ./ W).^2;
    eps_sqh = eff_permittivity(1, 0, er, tand, f_sqh);
```

```
50
51      kz_sqh = floquet_modes(k0, theta, phi, m, n, W, W, eps_sqh);
52      Theta_sqh = kz_sqh * Hsqh;
53      ZC_TE_sqh = ( floquet_Y(omega0, kz_sqh, eps0*eps_sqh, mu0, 1) ).^(-1);
54      ZC_TM_sqh = ( floquet_Y(omega0, kz_sqh, eps0*eps_sqh, mu0, 2) ).^(-1);
55
56      [A_TE_sqh, B_TE_sqh, C_TE_sqh, D_TE_sqh] = ABCD_TL(ZC_TE_sqh, Theta_sqh);
57      [A_TM_sqh, B_TM_sqh, C_TM_sqh, D_TM_sqh] = ABCD_TL(ZC_TM_sqh, Theta_sqh);
58
59      [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE, A_TE_sqh,
        B_TE_sqh, C_TE_sqh, D_TE_sqh);
60      [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM, A_TM_sqh,
        B_TM_sqh, C_TM_sqh, D_TM_sqh);
61
62      % Output tapering
63      for i = Nstep:-1:1 % for all the steps
64          Vair_step = step_horiz .* ( W - 2 * i * step_vert ).^(2);
65          f1_step = Vair_step ./ Vtot_step; % volume fraction
66          eps_eff_step = eff_permittivity(1, 0, er, tand, f1_step); % effective relative
            dielectric constant
67
68          kz = floquet_modes(k0, theta, phi, m, n, W, W, eps_eff_step);
69          Theta = kz * step_horiz;
70          ZC_TE = ( floquet_Y(omega0, kz, eps0*eps_eff_step, mu0, 1) ).^(-1);
71          ZC_TM = ( floquet_Y(omega0, kz, eps0*eps_eff_step, mu0, 2) ).^(-1);
72
73          [A_TE_new_layer, B_TE_new_layer, C_TE_new_layer, D_TE_new_layer] =
            ABCD_TL(ZC_TE, Theta);
74          [A_TM_new_layer, B_TM_new_layer, C_TM_new_layer, D_TM_new_layer] =
            ABCD_TL(ZC_TM, Theta);
75
76          [A_TE, B_TE, C_TE, D_TE] = multiplyMatrix(A_TE, B_TE, C_TE, D_TE,
            A_TE_new_layer, B_TE_new_layer, C_TE_new_layer, D_TE_new_layer);
77          [A_TM, B_TM, C_TM, D_TM] = multiplyMatrix(A_TM, B_TM, C_TM, D_TM,
            A_TM_new_layer, B_TM_new_layer, C_TM_new_layer, D_TM_new_layer);
78      end
79
80      % Evaluation of the scattering parameters
81      [S11TE, ~, S21TE, ~] = ABCD2S(Zr_TE, Zr_TE, A_TE, B_TE, C_TE, D_TE);
82      [S11TM, ~, S21TM, ~] = ABCD2S(Zr_TM, Zr_TM, A_TM, B_TM, C_TM, D_TM);
83  end
```

B.5 Maximum UC Height

```
1   %% Calculation of max cell height
2   % f0 = centerband frequency [Hz]
3   % eps_r, tan_delta =  relative dielectric constant and loss tangent of the material
4   % W = UC periodicity [m]
5   % T, d = cell height and hole size ranges [mm]
6   % theta = scan angle [deg]
7   % dim_grid = number of points
8   % n_layers = 1 (single-layer UC) or 3 (three-layer UC)
9   function [Tmax_sqh, Tmax_crh, DPsi_TE_sqh, DPsi_TM_sqh, DPsi_TE_crh, DPsi_TM_crh] =
    Tmax(f0, eps_r, tan_delta, W, T, d, theta, dim_grid, n_layers)
10      DPsi_TE_sqh = zeros(1, dim_grid);
11      DPsi_TM_sqh = zeros(1, dim_grid);
12      DPsi_TE_crh = zeros(1, dim_grid);
```

```
    DPsi_TM_crh = zeros(1, dim_grid);

    % phase range vs T
    for i = 1:dim_grid % for each value of T
        if ( n_layers == 3 ) % three-layer UC (2 dielectric layers + glass layer)
            S21_TE_sqh_phase = unwrap(angle(S21_TE_UC_TA_glass_sqh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
            S21_TM_sqh_phase = unwrap(angle(S21_TM_UC_TA_glass_sqh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
            S21_TE_crh_phase = unwrap(angle(S21_TE_UC_TA_glass_crh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
            S21_TM_crh_phase = unwrap(angle(S21_TM_UC_TA_glass_crh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
        elseif ( n_layers == 1 ) % single-layer UC (1 dielectric layer)
            S21_TE_sqh_phase = unwrap(angle(S21_TE_UC_TA_noGlass_sqh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
            S21_TM_sqh_phase = unwrap(angle(S21_TM_UC_TA_noGlass_sqh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
            S21_TE_crh_phase = unwrap(angle(S21_TE_UC_TA_noGlass_crh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
            S21_TM_crh_phase = unwrap(angle(S21_TM_UC_TA_noGlass_crh(theta, 0, 0, 0, f0,
            eps_r, tan_delta, W, T(i)*1e-3, d*1e-3)));
        else
            return
        end
        DPsi_TE_sqh(i) = max(S21_TE_sqh_phase) - min(S21_TE_sqh_phase); % phase range @
        T = T(j), TE mode, square hole
        DPsi_TM_sqh(i) = max(S21_TM_sqh_phase) - min(S21_TM_sqh_phase); % phase range @
        T = T(j), TM mode, square hole
        DPsi_TE_crh(i) = max(S21_TE_crh_phase) - min(S21_TE_crh_phase); % phase range @
        T = T(j), TE mode, circular hole
        DPsi_TM_crh(i) = max(S21_TM_crh_phase) - min(S21_TM_crh_phase); % phase range @
        T = T(j), TM mode, circular hole
    end

    % calculation of the max value of T that ensures DPsi = 2*pi
    distance_TE_sqh = abs(DPsi_TE_sqh - 2*pi);
    [~, pos_TE_sqh] = min(distance_TE_sqh);
    Tmax_TE_sqh = T(pos_TE_sqh);

    distance_TM_sqh = abs(DPsi_TM_sqh - 2*pi);
    [~, pos_TM_sqh] = min(distance_TM_sqh);
    Tmax_TM_sqh = T(pos_TM_sqh);

    distance_TE_crh = abs(DPsi_TE_crh - 2*pi);
    [~, pos_TE_crh] = min(distance_TE_crh);
    Tmax_TE_crh = T(pos_TE_crh);

    distance_TM_crh = abs(DPsi_TM_crh - 2*pi);
    [~, pos_TM_crh] = min(distance_TM_crh);
    Tmax_TM_crh = T(pos_TM_crh);

    Tmax_sqh = mean([Tmax_TE_sqh, Tmax_TM_sqh]);
    Tmax_crh = mean([Tmax_TE_crh, Tmax_TM_crh]);
end
```

B.6 UC Optimization

B.6.1 Optimization with no Phase Error (Equality Constraint)

```
%% Optimum hole size and cell height [mm]
% theta, phi = scan angles [deg]
% phi0 = desired phase [deg]
% project = phase-only/reflection-only/complete
% f0 = centerband frequency [Hz]
% eps_r, tan_delta =  relative dielectric constant and loss tangent of the material
% W = UC periodicity [m]
% T, d = total cell height and hole size ranges [mm]
% H = T/2 + Tf [mm]
% dim_grid = number of points
function [d_TE_opt, T_TE_opt, d_TM_opt, T_TM_opt] = optimizer_glass_sqh_equality(theta,
phi, phi0, project, f0, eps_r, tan_delta, W, T, H, d, dim_grid)

    if ( project ~= 'p' && project ~= 'r' && project ~= 'c' )
        fprintf('Invalid input, aborting execution ..\n');
        return
    end

    %% Calculation of S parameters
    S11_TE = zeros(dim_grid);
    S21_TE = zeros(dim_grid);
    S11_TM = zeros(dim_grid);
    S21_TM = zeros(dim_grid);

    for i=1:dim_grid
        [S11_TE(i,:), S21_TE(i,:), S11_TM(i,:), S21_TM(i,:)] = UC_TA_glass_sqh(theta,
        phi, 0, 0, f0, eps_r, tan_delta, W, T(i)*1e-3, (H - 0.5*T(i))*1e-3, d*1e-3);
    end

    S21_TE_phase = phase_rad(S21_TE);
    S21_TM_phase = phase_rad(S21_TM);

    %% Find d and T ranges (phase S21 = phi0)
    [d_grid, T_grid] = meshgrid(d, T); % [mm]

    % Check intersection between S21_TE_phase and phi0
    distance_TE = abs(S21_TE_phase - deg2rad(phi0));
    intersection_TE = distance_TE < 1e-3;

    if isempty(find(intersection_TE, 1)) % no intersection between S21_TE_phase and phi0
        % 1) Normalize phase shifting
        if ( isempty(find(S21_TE_phase < deg2rad(phi0), 1)) ) % S21_TE_phase bigger
            S21_TE_phase = S21_TE_phase - 2*pi;
        else
            S21_TE_phase = S21_TE_phase + 2*pi;
        end

        % 2) Recalculate distance and check intersection
        distance_TE = abs(S21_TE_phase - deg2rad(phi0));
        intersection_TE = distance_TE < 1e-3;

        if isempty(find(intersection_TE, 1)) % no intersection between S21_TE_phase and
        phi0
            % Find the closest match if no exact intersection exists
```

```
            [~, min_idx] = min(distance_TE(:));
            d_TE = d_grid(min_idx); % [mm]
            T_TE = T_grid(min_idx); % [mm]
        else % intersection found
            d_TE = d_grid(intersection_TE); % [mm]
            T_TE = T_grid(intersection_TE); % [mm]
        end

    else % intersection found
        d_TE = d_grid(intersection_TE); % [mm]
        T_TE = T_grid(intersection_TE); % [mm]
    end

    distance_TM = abs( S21_TM_phase - deg2rad(phi0) );
    intersection_TM = distance_TM < 1e-3;

    if isempty(find(intersection_TM, 1))
        % 1) Normalize phase shifting
        if ( isempty(find(S21_TM_phase < deg2rad(phi0), 1)) ) % S21_TE_phase bigger
            S21_TM_phase = S21_TM_phase - 2*pi;
        else
            S21_TM_phase = S21_TM_phase + 2*pi;
        end

        % 2) Recalculate distance and check intersection
        distance_TM = abs(S21_TM_phase - deg2rad(phi0));
        intersection_TM = distance_TM < 1e-3;

        if isempty(find(intersection_TM, 1))
            % Find the closest match if no exact intersection exists
            [~, min_idx] = min(distance_TM(:));
            d_TM = d_grid(min_idx); % [mm]
            T_TM = T_grid(min_idx); % [mm]
        else
            d_TM = d_grid(intersection_TM); % [mm]
            T_TM = T_grid(intersection_TM); % [mm]
        end
    else
        d_TM = d_grid(intersection_TM); % [mm]
        T_TM = T_grid(intersection_TM); % [mm]
    end

    %% Implementation of the genetic algorithm (TE mode)
    if ~isempty(find(intersection_TE, 1))
        % Define the bounds for d and T
        pTE = polyfit(d_TE, T_TE, 5);
        lb_TE = [d_TE(1) T_TE(1)]; % Lower bounds
        ub_TE = [d_TE(end) T_TE(end)]; % Upper bounds

        % Run the GA
        if project == 'p'
            options = optimoptions('ga', ...
                'PopulationSize', 50, ...     % Number of individuals in population
                'MaxGenerations', 100, ...    % Maximum number of generations
                'Display', 'iter', ...        % Display iteration information
                'UseParallel', false);        % Don't run // tasks
            x = ga(@singleobjectiveS21TE, 2, [], [], [], [], lb_TE, ub_TE,
            @constraintTE, options);
        elseif project == 'r'
```

```
            options = optimoptions('ga', ...
                'PopulationSize', 50, ...     % Number of individuals in population
                'MaxGenerations', 100, ...    % Maximum number of generations
                'Display', 'iter', ...        % Display iteration information
                'UseParallel', false);        % Don't run // tasks
            x = ga(@singleobjectiveS11TE, 2, [], [], [], [], lb_TE, ub_TE,
            @constraintTE, options);
        elseif project == 'c'
            options = optimoptions('gamultiobj', ...
                'PopulationSize', 50, ...     % Number of individuals in population
                'MaxGenerations', 100, ...    % Maximum number of generations
                'ParetoFraction', 0.7, ...    % Pareto fraction
                'Display', 'iter', ...        % Display iteration information
                'UseParallel', false);        % Don't run // tasks
            x = gamultiobj(@multiobjectiveTE, 2, [], [], [], [], lb_TE, ub_TE,
            @constraintTE, options);
        end
        d_TE_opt = x(1);
        T_TE_opt = x(2);
    else
        d_TE_opt = d_TE;
        T_TE_opt = T_TE;
    end

    %% Implementation of the genetic algorithm (TM mode)
    if ~isempty(find(intersection_TM, 1))
        % Define the bounds for d and T
        pTM = polyfit(d_TM, T_TM, 5);
        lb_TM = [d_TM(1) T_TM(1)]; % Lower bounds
        ub_TM = [d_TM(end) T_TM(end)]; % Upper bounds

        % Run the GA
        if project == 'p'
            options = optimoptions('ga', 'Display', 'iter', 'UseParallel', false);
            x = ga(@singleobjectiveS21TM, 2, [], [], [], [], lb_TM, ub_TM,
            @constraintTM, options);
        elseif project == 'r'
            options = optimoptions('ga', 'Display', 'iter', 'UseParallel', false);
            x = ga(@singleobjectiveS11TM, 2, [], [], [], [], lb_TM, ub_TM,
            @constraintTM, options);
        elseif project == 'c'
            options = optimoptions('gamultiobj', 'ParetoFraction', 0.7, 'Display',
            'iter', 'UseParallel', false);
            x = gamultiobj(@multiobjectiveTM, 2, [], [], [], [], lb_TM, ub_TM,
            @constraintTM, options);
        end
        d_TM_opt = x(1);
        T_TM_opt = x(2);
    else
        d_TM_opt = d_TM;
        T_TM_opt = T_TM;
    end

    %% Single-objective function (TE mode)
    function optfunc = singleobjectiveS21TE(x)
    [~, S21_TE_sobj, ~, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
    x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
    optfunc = -abs(S21_TE_sobj);
    end
```

```
162
163     function optfunc = singleobjectiveS11TE(x)
164     [S11_TE_sobj, ~, ~, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
        x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
165     optfunc = abs(S11_TE_sobj);
166     end
167
168     %% Multi-objective function (TE mode)
169     function [optfunc11, optfunc21] = multiobjectiveTE(x)
170     [S11_TE_mobj, S21_TE_mobj, ~, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r,
        tan_delta, W, x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
171     optfunc11 = abs(S11_TE_mobj);
172     optfunc21 = -abs(S21_TE_mobj);
173     end
174
175     %% Non-linear constraint function (TE mode)
176     function [c,ceq] = constraintTE(x)
177     c = []; % inequality constraints
178     ceq = x(2) - pTE(1) * x(1)^5 - pTE(2) * x(1)^4 - pTE(3) * x(1)^3 - pTE(4) * x(1)^2 -
        pTE(5) * x(1) - pTE(6); % equality constraints
179     end
180
181     %% Single-objective function (TM mode)
182     function optfunc = singleobjectiveS21TM(x)
183     [~, ~, ~, S21_TM_sobj] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
        x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
184     optfunc = -abs(S21_TM_sobj);
185     end
186
187     function optfunc = singleobjectiveS11TM(x)
188     [~, ~, S11_TM_sobj, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
        x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
189     optfunc = abs(S11_TM_sobj);
190     end
191
192     %% Multi-objective function (TM mode)
193     function [optfunc11, optfunc21] = multiobjectiveTM(x)
194     [~, ~, S11_TM_mobj, S21_TM_mobj] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r,
        tan_delta, W, x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
195     optfunc11 = abs(S11_TM_mobj);
196     optfunc21 = -abs(S21_TM_mobj);
197     end
198
199     %% Non-linear constraint function (TM mode)
200     function [c,ceq] = constraintTM(x)
201     c = []; % inequality constraints
202     ceq = x(2) - pTM(1) * x(1)^5 - pTM(2) * x(1)^4 - pTM(3) * x(1)^3 - pTM(4) * x(1)^2 -
        pTM(5) * x(1) - pTM(6); % equality constraints
203     end
204 end
```

B.6.2 Optimization with Phase Error (Inequality Constraint)

```
1 %% Optimum hole size and cell height [mm]
2 % theta, phi = scan angles [deg]
3 % phi0 = desired phase [deg]
4 % tol = tolerance on the phase delay
```

```
% project = phase-only/complete
% f0 = centerband frequency [Hz]
% eps_r, tan_delta =  relative dielectric constant and loss tangent of the material
% W = UC periodicity [m]
% T, d = total cell height and hole size ranges [mm]
% H = T/2 + Tf [mm]
% dim_grid = number of points
function [d_TE_opt, T_TE_opt, d_TM_opt, T_TM_opt] =
optimizer_glass_sqh_inequality(theta, phi, phi0, tol, project, f0, eps_r, tan_delta, W,
T, H, d, dim_grid)

    if ( project ~= 'p' && project ~= 'r' && project ~= 'c' )
        fprintf('Invalid input, aborting execution ..\n');
        return
    end

    %% Calculation of S parameters
    S11_TE = zeros(dim_grid);
    S21_TE = zeros(dim_grid);
    S11_TM = zeros(dim_grid);
    S21_TM = zeros(dim_grid);

    for i=1:dim_grid
        [S11_TE(i,:), S21_TE(i,:), S11_TM(i,:), S21_TM(i,:)] = UC_TA_glass_sqh(theta,
        phi, 0, 0, f0, eps_r, tan_delta, W, T(i)*1e-3, (H - 0.5*T(i))*1e-3, d*1e-3);
    end

    S21_TE_phase = phase_rad(S21_TE);
    S21_TM_phase = phase_rad(S21_TM);

    %% Find d and T ranges (phase S21 = phi0)
    [d_grid, T_grid] = meshgrid(d, T); % [mm]

    % Check intersection between S21_TE_phase and phi0
    distance_TE = abs(S21_TE_phase - deg2rad(phi0));
    intersection_TE = distance_TE < 1e-3;

    if isempty(find(intersection_TE, 1)) % no intersection between S21_TE_phase and phi0
        % 1) Normalize phase shifting
        if ( isempty(find(S21_TE_phase < deg2rad(phi0), 1)) ) % S21_TE_phase bigger
            S21_TE_phase = S21_TE_phase - 2*pi;
        else
            S21_TE_phase = S21_TE_phase + 2*pi;
        end

        % 2) Recalculate distance and check intersection
        distance_TE = abs(S21_TE_phase - deg2rad(phi0));
        intersection_TE = distance_TE < 1e-3;

        if isempty(find(intersection_TE, 1)) % no intersection between S21_TE_phase and
        phi0
            % Find the closest match if no exact intersection exists
            [~, min_idx] = min(distance_TE(:));
            d_TE = d_grid(min_idx); % [mm]
            T_TE = T_grid(min_idx); % [mm]
        else % intersection found
            d_TE = d_grid(intersection_TE); % [mm]
            T_TE = T_grid(intersection_TE); % [mm]
        end
```

```
60
61      else % intersection found
62          d_TE = d_grid(intersection_TE); % [mm]
63          T_TE = T_grid(intersection_TE); % [mm]
64      end
65
66      distance_TM = abs( S21_TM_phase - deg2rad(phi0) );
67      intersection_TM = distance_TM < 1e-3;
68
69      if isempty(find(intersection_TM, 1))
70          % 1) Normalize phase shifting
71          if ( isempty(find(S21_TM_phase < deg2rad(phi0), 1)) ) % S21_TE_phase bigger
72              S21_TM_phase = S21_TM_phase - 2*pi;
73          else
74              S21_TM_phase = S21_TM_phase + 2*pi;
75          end
76
77          % 2) Recalculate distance and check intersection
78          distance_TM = abs(S21_TM_phase - deg2rad(phi0));
79          intersection_TM = distance_TM < 1e-3;
80
81          if isempty(find(intersection_TM, 1))
82              % Find the closest match if no exact intersection exists
83              [~, min_idx] = min(distance_TM(:));
84              d_TM = d_grid(min_idx); % [mm]
85              T_TM = T_grid(min_idx); % [mm]
86          else
87              d_TM = d_grid(intersection_TM); % [mm]
88              T_TM = T_grid(intersection_TM); % [mm]
89          end
90      else
91          d_TM = d_grid(intersection_TM); % [mm]
92          T_TM = T_grid(intersection_TM); % [mm]
93      end
94
95      %% Implementation of the genetic algorithm (TE mode)
96      if ~isempty(find(intersection_TE, 1))
97          % Define the bounds for d and T
98          pTE = polyfit(d_TE, T_TE, 5);
99          lb_TE = [d_TE(1) T_TE(1)]; % Lower bounds
100         ub_TE = [d_TE(end) T_TE(end)]; % Upper bounds
101
102         % Run the GA
103         if project == 'p'
104             options = optimoptions('ga', ...
105                 'PopulationSize', 50, ...     % Number of individuals in population
106                 'MaxGenerations', 100, ...    % Maximum number of generations
107                 'Display', 'iter', ...        % Display iteration information
108                 'UseParallel', false);        % Don't run // tasks
109             x = ga(@singleobjectiveS21TE, 2, [], [], [], [], lb_TE, ub_TE,
                @constraintTE, options);
110         elseif project == 'r'
111             options = optimoptions('ga', ...
112                 'PopulationSize', 50, ...     % Number of individuals in population
113                 'MaxGenerations', 100, ...    % Maximum number of generations
114                 'Display', 'iter', ...        % Display iteration information
115                 'UseParallel', false);        % Don't run // tasks
116             x = ga(@singleobjectiveS11TE, 2, [], [], [], [], lb_TE, ub_TE,
                @constraintTE, options);
```

```
        elseif project == 'c'
            options = optimoptions('gamultiobj', ...
                'PopulationSize', 50, ...     % Number of individuals in population
                'MaxGenerations', 100, ...    % Maximum number of generations
                'ParetoFraction', 0.7, ...    % Pareto fraction
                'Display', 'iter', ...        % Display iteration information
                'UseParallel', false);        % Don't run // tasks
            x = gamultiobj(@multiobjectiveTE, 2, [], [], [], [], lb_TE, ub_TE,
            @constraintTE, options);
        end
        d_TE_opt = x(1);
        T_TE_opt = x(2);
    else
        d_TE_opt = d_TE;
        T_TE_opt = T_TE;
    end

    %% Implementation of the genetic algorithm (TM mode)
    if ~isempty(find(intersection_TM, 1))
        % Define the bounds for d and T
        pTM = polyfit(d_TM, T_TM, 5);
        lb_TM = [d_TM(1) T_TM(1)]; % Lower bounds
        ub_TM = [d_TM(end) T_TM(end)]; % Upper bounds

        % Run the GA
        if project == 'p'
            options = optimoptions('ga', 'Display', 'iter', 'UseParallel', false);
            x = ga(@singleobjectiveS21TM, 2, [], [], [], [], lb_TM, ub_TM,
            @constraintTM, options);
        elseif project == 'r'
            options = optimoptions('ga', 'Display', 'iter', 'UseParallel', false);
            x = ga(@singleobjectiveS11TM, 2, [], [], [], [], lb_TM, ub_TM,
            @constraintTM, options);
        elseif project == 'c'
            options = optimoptions('gamultiobj', 'ParetoFraction', 0.7, 'Display',
            'iter', 'UseParallel', false);
            x = gamultiobj(@multiobjectiveTM, 2, [], [], [], [], lb_TM, ub_TM,
            @constraintTM, options);
        end
        d_TM_opt = x(1);
        T_TM_opt = x(2);
    else
        d_TM_opt = d_TM;
        T_TM_opt = T_TM;
    end

    %% Single-objective function (TE mode)
    function optfunc = singleobjectiveS21TE(x)
    [~, S21_TE_sobj, ~, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
    x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
    optfunc = -abs(S21_TE_sobj);
    end

    function optfunc = singleobjectiveS11TE(x)
    [S11_TE_sobj, ~, ~, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
    x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
    optfunc = abs(S11_TE_sobj);
    end

```

```
    %% Multi-objective function (TE mode)
    function [optfunc11, optfunc21] = multiobjectiveTE(x)
    [S11_TE_mobj, S21_TE_mobj, ~, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r,
    tan_delta, W, x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
    optfunc11 = abs(S11_TE_mobj);
    optfunc21 = -abs(S21_TE_mobj);
    end

    %% Non-linear constraint function (TE mode)
    function [c,ceq] = constraintTE(x)
    c = abs(x(2) - pTE(1) * x(1)^5 - pTE(2) * x(1)^4 - pTE(3) * x(1)^3 - pTE(4) * x(1)^2
    - pTE(5) * x(1) - pTE(6)) - tol; % inequality constraints
    ceq = []; % equality constraints
    end

    %% Single-objective function (TM mode)
    function optfunc = singleobjectiveS21TM(x)
    [~, ~, ~, S21_TM_sobj] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
    x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
    optfunc = -abs(S21_TM_sobj);
    end

    function optfunc = singleobjectiveS11TM(x)
    [~, ~, S11_TM_sobj, ~] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r, tan_delta, W,
    x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
    optfunc = abs(S11_TM_sobj);
    end

    %% Multi-objective function (TM mode)
    function [optfunc11, optfunc21] = multiobjectiveTM(x)
    [~, ~, S11_TM_mobj, S21_TM_mobj] = UC_TA_glass_sqh(theta, phi, 0, 0, f0, eps_r,
    tan_delta, W, x(2)*1e-3, (H - 0.5*x(2))*1e-3, x(1)*1e-3);
    optfunc11 = abs(S11_TM_mobj);
    optfunc21 = -abs(S21_TM_mobj);
    end

    %% Non-linear constraint function (TM mode)
    function [c,ceq] = constraintTM(x)
    c = abs(x(2) - pTM(1) * x(1)^5 - pTM(2) * x(1)^4 - pTM(3) * x(1)^3 - pTM(4) * x(1)^2
    - pTM(5) * x(1) - pTM(6)) - tol; % inequality constraints
    ceq = []; % equality constraints
    end
end
```

B.7 TA Optimization

```
%% Optimum geometric parameters for the entire array
% tol = tolerance on the phase delay
% project = phase-only (p)/reflection-only (r)/complete (c)
% f0 = centerband frequency [Hz]
% eps_r, tan_delta = relative dielectric constant and loss tangent of the material
% W = UC periodicity [m]
% T, d = cell height and hole size ranges [mm]
% H = T/2 + Tf [mm]
% dim_grid = number of points
% F = focal length [m]
% M = number of elements along the main axis
```

```
% theta_inc, phi_inc = direction of incidence [rad]
% theta_beam, phi_beam = direction main beam [rad]
% outfile = name of the output file to save results
function optimizer_TA_glass_sqh(tol, project, f0, eps_r, tan_delta, W, T, H, d,
dim_grid, F, M, theta_inc, phi_inc, theta_beam, phi_beam, outfile_feed, outfile_plane)
    lambda0 = physconst('LightSpeed') / f0; % free-space wavelength [m]
    k0 = 2 * pi / lambda0; % propagation constant [m^-1]
    D = M * W; % TA diameter [m]
    theta_beam = theta_beam - theta_inc; % [rad]

    % Tapering
    qf = 12.5;
    e_inc = @(theta_f) ( cos( theta_f ) ).^qf; % normalized incident field
    tapering = e_inc(atan( D / (2 * F) ));
    fprintf('Chosen focal length: F = %.2f mm\n', F*1e3);
    fprintf('Tapering: %.2f dB\n', 20*log10(tapering));

    % Phase map
    [x, y] = meshgrid(1:1:M, 1:1:M);
    x_mn = @(m, n) W * ( m - (M+1)/2 );
    y_mn = @(m, n) W * ( n - (M+1)/2 );
    r_mn = @(m, n) W * sqrt( ( m - (M+1)/2 ).^2 + ( n - (M+1)/2 ).^2 ); % [m]
    r_mn_u = @(m, n) sin(theta_beam) * ( ( W * ( m - (M+1)/2 ) ) * cos(phi_beam) + W * (
    n - (M+1)/2 ) * sin(phi_beam));
    R_mn = @(m, n) sqrt( F^2 + r_mn(m, n).^2 ); % [m]

    Psi_mn_feed = @(m, n) mod(( k0 * ( R_mn(m, n) - r_mn_u(m, n) ) ), 2*pi); % [rad]
    Psi_mn_plane = @(m, n) mod(-k0 * ( ( x_mn(m,n) * ( sin(theta_inc) * cos(phi_inc) +
    sin(theta_beam) * cos(phi_beam) ) ) + ( y_mn(m,n) * ( sin(theta_inc) * sin(phi_inc)
    + sin(theta_beam) * sin(phi_beam) ) ) ), 2*pi); % [rad]
    Psi_mn_feed_deg = @(m, n) rad2deg( Psi_mn_feed(m, n) ); % [deg]
    Psi_mn_plane_deg = @(m, n) rad2deg( Psi_mn_plane(m, n) ); % [deg]

    r = zeros(M, M); % position vector of each element [m]
    R = zeros(M, M); % distance of each element from the feed horn [m]
    theta_f_feed = zeros(M, M); % spherical angle in feed coordinate system [deg]
    theta_f_plane = zeros(M, M); % spherical angle in feed coordinate system [deg]
    Psi = zeros(M, M); % required phase delay for each element [deg]

    d_TE_opt = zeros(M, M);
    T_TE_opt = zeros(M, M);
    d_TM_opt = zeros(M, M);
    T_TM_opt = zeros(M, M);
    S11_TE_opt = zeros(M, M);
    S11_TM_opt = zeros(M, M);
    S21_TE_opt = zeros(M, M);
    S21_TM_opt = zeros(M, M);

    %%%%%%%% Run the optimizer for each cell of the array (feed) %%%%%%%%
    if (tol == 0)
        fprintf('\nRunning optimizer (feed), no phase error ..\n');
    else
        fprintf('\nRunning optimizer (feed), phase error ..\n');
    end

    for m = 1:1:M
        for n = 1:1:M
            fprintf('-------------------------------------------------------\n');
            fprintf('CELL: m = %d, n = %d', m, n);
```

```
            r(m,n) = r_mn(m,n); % [m]
            R(m,n) = R_mn(m,n); % [m]
            theta_f_feed(m,n) = rad2deg( asin( r(m,n) / R(m,n) ) ); % [deg]

            Psi(m,n) = Psi_mn_feed_deg(m,n); % [deg]
            if (Psi(m,n) >= max(Psi_mn_feed_deg(x,y), [], "all"))
                Psi(m,n) = max(Psi_mn_feed_deg(x,y), [], "all");
            elseif (Psi(m,n) <= min(Psi_mn_feed_deg(x,y), [], "all"))
                Psi(m,n) = min(Psi_mn_feed_deg(x,y), [], "all");
            end

            if (tol == 0)
                [d_TE_opt(m,n), T_TE_opt(m,n), d_TM_opt(m,n), T_TM_opt(m,n)] =
                optimizer_glass_sqh_equality(theta_f_feed(m,n), 0, ...
                    Psi(m,n), project, f0, eps_r, tan_delta, W, T, H, d, dim_grid);
            else
                [d_TE_opt(m,n), T_TE_opt(m,n), d_TM_opt(m,n), T_TM_opt(m,n)] =
                optimizer_glass_sqh_inequality(theta_f_feed(m,n), 0, ...
                    Psi(m,n), tol, project, f0, eps_r, tan_delta, W, T, H, d, dim_grid);
            end

            [S11_TE_opt(m,n), S21_TE_opt(m,n), ~, ~] =
            UC_TA_glass_sqh(theta_f_feed(m,n), 0, 0, 0, ...
                f0, eps_r, tan_delta, W, T_TE_opt(m,n)*1e-3, (H -
                0.5*T_TE_opt(m,n))*1e-3, d_TE_opt(m,n)*1e-3);
            [~, ~, S11_TM_opt(m,n), S21_TM_opt(m,n)] =
            UC_TA_glass_sqh(theta_f_feed(m,n), 0, 0, 0, ...
                f0, eps_r, tan_delta, W, T_TM_opt(m,n)*1e-3, (H -
                0.5*T_TM_opt(m,n))*1e-3, d_TM_opt(m,n)*1e-3);
        end
    end

    % Save results
    fprintf('Project completed. Results stored in %s\n', outfile_feed);
    save(outfile_feed, 'theta_beam', 'phi_beam', 'r', 'R', 'theta_f_feed', 'Psi',
    'd_TE_opt', 'T_TE_opt', 'd_TM_opt', 'T_TM_opt', 'S11_TE_opt', 'S21_TE_opt',
    'S11_TM_opt', 'S21_TM_opt');

    %%%%%%%% Run the optimizer for each cell of the array (plane wave) %%%%%%%%
    if (theta_beam ~= 0) || (phi_beam ~= 0)
        % Plane wave ~ normal incidence --> theta = phi = 0 for all cells
        if (tol == 0)
            fprintf('\nRunning optimizer (plane wave), no phase error ..\n');
        else
            fprintf('\nRunning optimizer (plane wave), phase error ..\n');
        end

        for m = 1:1:M
            for n = 1:1:M
                fprintf('-------------------------------------------------------\n');
                fprintf('CELL: m = %d, n = %d', m, n);

                Psi(m,n) = Psi_mn_plane_deg(m,n); % [deg]
                if (Psi(m,n) >= max(Psi_mn_plane_deg(x,y), [], "all"))
                    Psi(m,n) = max(Psi_mn_plane_deg(x,y), [], "all");
                elseif (Psi(m,n) <= min(Psi_mn_plane_deg(x,y), [], "all"))
                    Psi(m,n) = min(Psi_mn_plane_deg(x,y), [], "all");
                end

```

```
                if (tol == 0)
                    [d_TE_opt(m,n), T_TE_opt(m,n), d_TM_opt(m,n), T_TM_opt(m,n)] =
                    optimizer_glass_sqh_equality(0, 0, ...
                        Psi(m,n), project, f0, eps_r, tan_delta, W, T, H, d, dim_grid);
                else
                    [d_TE_opt(m,n), T_TE_opt(m,n), d_TM_opt(m,n), T_TM_opt(m,n)] =
                    optimizer_glass_sqh_inequality(0, 0, ...
                        Psi(m,n), tol, project, f0, eps_r, tan_delta, W, T, H, d,
                        dim_grid);
                end

                [S11_TE_opt(m,n), S21_TE_opt(m,n), ~, ~] = UC_TA_glass_sqh(0, 0, 0, 0,
                ...
                    f0, eps_r, tan_delta, W, T_TE_opt(m,n)*1e-3, (H -
                    0.5*T_TE_opt(m,n))*1e-3, d_TE_opt(m,n)*1e-3);
                [~, ~, S11_TM_opt(m,n), S21_TM_opt(m,n)] = UC_TA_glass_sqh(0, 0, 0, 0,
                ...
                    f0, eps_r, tan_delta, W, T_TM_opt(m,n)*1e-3, (H -
                    0.5*T_TM_opt(m,n))*1e-3, d_TM_opt(m,n)*1e-3);

            end
        end

        % Save results
        fprintf('Project completed. Results stored in %s\n', outfile_plane);
        save(outfile_plane, 'theta_beam', 'phi_beam', 'Psi', 'd_TE_opt', 'T_TE_opt',
        'd_TM_opt', 'T_TM_opt', 'S11_TE_opt', 'S21_TE_opt', 'S11_TM_opt', 'S21_TM_opt');
    end
end
```

B.8 TA & SES Analytical Models

B.8.1 TA Simplistic Model

```
close all,
clearvars,
clc,
format long e

%% Constants
load('../constants.mat'); % physical constants
load('../free_space.mat'); % f0, lambda0, k0

%% Parameters
M = 30; % number of elements along the main axis
qe = 1; % element pattern power factor
qf = 12.5; % feed pattern power factor
dx = 0.3; % spacing between elements in x-direction (in wavelengths)
dy = 0.3; % spacing between elements in y-direction (in wavelengths)
theta = linspace(-pi, pi, 500); % Elevation angle [rad]
phi = linspace(-pi, pi, 500); % Azimuth angle [rad]

%% Array factor calculation (no phase error)
% Excitation coefficients
load('Output/TA_30x30_0d8_tbeam0_pbeam0/output_equality_30x30_0d8_feed.mat');
I_TE_transmit = zeros(M, M);
I_TM_transmit = zeros(M, M);
```

```
for m = 1:1:M
    for n = 1:1:M
        I_TE_transmit(m,n) = S21_TE_opt(m,n) * exp( -1j * k0 * R(m,n) ) * ( ( cos(
        deg2rad(theta_f_feed(m,n)) ) )^qf / R(m,n) );
        I_TM_transmit(m,n) = S21_TM_opt(m,n) * exp( -1j * k0 * R(m,n) ) * ( ( cos(
        deg2rad(theta_f_feed(m,n)) ) )^qf / R(m,n) );
    end
end

% Array factor
FN_TE_equality = FN(theta, phi, M, M, dx, dx, I_TE_transmit, qe);
FN_TM_equality = FN(theta, phi, M, M, dx, dx, I_TM_transmit, qe);

%% Array factor calculation (with phase error)
% Excitation coefficients
load('Output/TA_30x30_0d8_tbeam0_pbeam0/output_inequality_30x30_0d8_feed.mat');
for m = 1:1:M
    for n = 1:1:M
        I_TE_transmit(m,n) = S21_TE_opt(m,n) * exp( -1j * k0 * R(m,n) ) * ( ( cos(
        deg2rad(theta_f_feed(m,n)) ) )^qf / R(m,n) );
        I_TM_transmit(m,n) = S21_TM_opt(m,n) * exp( -1j * k0 * R(m,n) ) * ( ( cos(
        deg2rad(theta_f_feed(m,n)) ) )^qf / R(m,n) );
    end
end

% Array factor
FN_TE_inequality = FN(theta, phi, M, M, dx, dx, I_TE_transmit, qe);
FN_TM_inequality = FN(theta, phi, M, M, dx, dx, I_TM_transmit, qe);

%% Save results
save('Output/TA_30x30_0d8_tbeam0_pbeam0/TA_glass_sqh_simple_model_30x30_0d8.mat',
'theta', 'phi', 'FN_TE_equality', 'FN_TM_equality', 'FN_TE_inequality',
'FN_TM_inequality')
```

B.8.2 TA PO Model

```
close all,
clearvars,
clc,
format long e

%% Constants
load('../constants.mat'); % physical constants
load('../free_space.mat'); % f0, lambda0, k0

%% Parameters
M = 30;                             % number of elements along the main axis
qe = 1;                             % element pattern power factor
qf = 12.5;                          % feed pattern power factor
W = 0.3 * lambda0;                  % periodicity
theta = linspace(-pi, pi, 500);     % elevation angle [rad]
phi = linspace(-pi, pi, 500);       % azimuth angle [rad]

%% FF calculation (WITHOUT phase error)
load('Output/TA_30x30_0d8_tbeam0_pbeam0/output_equality_30x30_0d8_feed.mat');

eTE_theta = cell(M, M);
```

```
eTM_theta = cell(M, M);

for m = 1:1:M
    for n = 1:1:M
        xl = W * ( m - ( ( M + 2 ) / 2 ) );
        xu = W * ( m - ( M / 2 ) );
        yl = W * ( n - ( ( M + 2 ) / 2 ) );
        yu = W * ( n - ( M / 2 ) );
        int_x = @(theta, phi, x) exp( 1i * k0 .* sin(theta) .* cos(phi) .* x );
        int_y = @(theta, phi, y) exp( 1i * k0 .* sin(theta) .* sin(phi) .* y );
        Ix = @(theta, phi) integral(@(x) int_x(theta, phi, x), xl, xu, 'ArrayValued',
        true);
        Iy = @(theta, phi) integral(@(y) int_y(theta, phi, y), yl, yu, 'ArrayValued',
        true);
        Js_theta_mn = ( ( cos( deg2rad(theta_f_feed(m,n)) ) )^qf / R(m, n) ) .* exp( -1j
        * k0 * R(m,n) );
        JtildeTE_theta_mn = @(theta, phi) S21_TE_opt(m, n) .* Js_theta_mn .* Ix(theta,
        phi) .* Iy(theta, phi);
        eTE_theta{m, n} = @(theta, phi) -1i * k0 * Z0 * JtildeTE_theta_mn(theta, phi);

        JtildeTM_theta_mn = @(theta, phi) S21_TM_opt(m, n) .* Js_theta_mn .* Ix(theta,
        phi) .* Iy(theta, phi);
        eTM_theta{m, n} = @(theta, phi) -1i * k0 * Z0 * JtildeTM_theta_mn(theta, phi);

    end
end

eTE_theta_tot = @(theta, phi) sum(cell2mat(cellfun(@(f) f(theta, phi), eTE_theta(:),
'UniformOutput', false)));
eTM_theta_tot = @(theta, phi) sum(cell2mat(cellfun(@(f) f(theta, phi), eTM_theta(:),
'UniformOutput', false)));

%% Results
eEL_func = @(theta, phi) abs( cos(theta).^qe ) .* ( rms(abs(S11_TE_opt), 'all') .* (
theta < -pi/2 | theta > pi/2 ) + rms(abs(S21_TE_opt), 'all') .* ( theta >= -pi/2 & theta
<= pi/2 ) );
eTE_func = @(theta, phi) abs(eTE_theta_tot(theta, phi));
eTM_func = @(theta, phi) abs(eTM_theta_tot(theta, phi));
eTE_TOT_func = @(theta, phi) eTE_func(theta, phi) .* eEL_func(theta, phi);
eTM_TOT_func = @(theta, phi) eTM_func(theta, phi) .* eEL_func(theta, phi);

% E plane
eEl_Eplane = eEL_func(theta, 0);
eTE_Eplane = eTE_func(theta, 0);
eTM_Eplane = eTM_func(theta, 0);
eTE_Eplane_TOT = eTE_TOT_func(theta, 0);
eTM_Eplane_TOT = eTM_TOT_func(theta, 0);

eEl_Eplane_equality = eEl_Eplane / max(eEl_Eplane);
eTE_Eplane_equality = eTE_Eplane / max(eTE_Eplane);
eTM_Eplane_equality = eTM_Eplane / max(eTM_Eplane);
eTE_Eplane_TOT_equality = eTE_Eplane_TOT / max(eTE_Eplane_TOT);
eTM_Eplane_TOT_equality = eTM_Eplane_TOT / max(eTM_Eplane_TOT);

% H plane
eEl_Hplane = eEL_func(theta, 90);
eTE_Hplane = eTE_func(theta, 90);
eTM_Hplane = eTM_func(theta, 90);
eTE_Hplane_TOT = eTE_TOT_func(theta, 90);
```

```
eTM_Hplane_TOT = eTM_TOT_func(theta, 90);

eEl_Hplane_equality = eEl_Hplane / max(eEl_Hplane);
eTE_Hplane_equality = eTE_Hplane / max(eTE_Hplane);
eTM_Hplane_equality = eTM_Hplane / max(eTM_Hplane);
eTE_Hplane_TOT_equality = eTE_Hplane_TOT / max(eTE_Hplane_TOT);
eTM_Hplane_TOT_equality = eTM_Hplane_TOT / max(eTM_Hplane_TOT);

%% FF calculation (WITH phase error)
load('Output/TA_30x30_0d8_tbeam0_pbeam0/output_inequality_30x30_0d8_feed.mat');
eTE_theta = cell(M, M);
eTM_theta = cell(M, M);

for m = 1:1:M
    for n = 1:1:M
        xl = W * ( m - ( ( M + 2 ) / 2 ) );
        xu = W * ( m - ( M / 2 ) );
        yl = W * ( n - ( ( M + 2 ) / 2 ) );
        yu = W * ( n - ( M / 2 ) );
        int_x = @(theta, phi, x) exp( 1i * k0 .* sin(theta) .* cos(phi) .* x );
        int_y = @(theta, phi, y) exp( 1i * k0 .* sin(theta) .* sin(phi) .* y );
        Ix = @(theta, phi) integral(@(x) int_x(theta, phi, x), xl, xu, 'ArrayValued',
        true);
        Iy = @(theta, phi) integral(@(y) int_y(theta, phi, y), yl, yu, 'ArrayValued',
        true);
        Js_theta_mn = ( ( cos( deg2rad(theta_f_feed(m,n)) ) )^qf / R(m, n) );
        JtildeTE_theta_mn = @(theta, phi) S21_TE_opt(m, n) .* Js_theta_mn .* Ix(theta,
        phi) .* Iy(theta, phi) .* exp( -1j * k0 * R(m,n) );
        eTE_theta{m, n} = @(theta, phi) -1i * k0 * Z0 * JtildeTE_theta_mn(theta, phi);

        JtildeTM_theta_mn = @(theta, phi) S21_TM_opt(m, n) .* Js_theta_mn .* Ix(theta,
        phi) .* Iy(theta, phi) .* exp( -1j * k0 * R(m,n) );
        eTM_theta{m, n} = @(theta, phi) -1i * k0 * Z0 * JtildeTM_theta_mn(theta, phi);
    end
end

eTE_theta_tot = @(theta, phi) sum(cell2mat(cellfun(@(f) f(theta, phi), eTE_theta(:),
'UniformOutput', false)));
eTM_theta_tot = @(theta, phi) sum(cell2mat(cellfun(@(f) f(theta, phi), eTM_theta(:),
'UniformOutput', false)));

%% Results
eEL_func = @(theta, phi) abs( cos(theta).^qe ) .* ( mean(abs(S11_TE_opt), 'all') .* (
theta < -pi/2 | theta > pi/2 ) + mean(abs(S21_TE_opt), 'all') .* ( theta >= -pi/2 &
theta <= pi/2 ) );
eTE_func = @(theta, phi) abs(eTE_theta_tot(theta, phi));
eTM_func = @(theta, phi) abs(eTM_theta_tot(theta, phi));
eTE_TOT_func = @(theta, phi) eTE_func(theta, phi) .* eEL_func(theta, phi);
eTM_TOT_func = @(theta, phi) eTM_func(theta, phi) .* eEL_func(theta, phi);

% E plane
eEl_Eplane = eEL_func(theta, 0);
eTE_Eplane = eTE_func(theta, 0);
eTM_Eplane = eTM_func(theta, 0);
eTE_Eplane_TOT = eTE_TOT_func(theta, 0);
eTM_Eplane_TOT = eTM_TOT_func(theta, 0);

eEl_Eplane_inequality = eEl_Eplane / max(eEl_Eplane);
eTE_Eplane_inequality = eTE_Eplane / max(eTE_Eplane);
```

```
eTM_Eplane_inequality = eTM_Eplane / max(eTM_Eplane);
eTE_Eplane_TOT_inequality = eTE_Eplane_TOT / max(eTE_Eplane_TOT);
eTM_Eplane_TOT_inequality = eTM_Eplane_TOT / max(eTM_Eplane_TOT);

% H plane
eEl_Hplane = eEL_func(theta, 90);
eTE_Hplane = eTE_func(theta, 90);
eTM_Hplane = eTM_func(theta, 90);
eTE_Hplane_TOT = eTE_TOT_func(theta, 90);
eTM_Hplane_TOT = eTM_TOT_func(theta, 90);

eEl_Hplane_inequality = eEl_Hplane / max(eEl_Hplane);
eTE_Hplane_inequality = eTE_Hplane / max(eTE_Hplane);
eTM_Hplane_inequality = eTM_Hplane / max(eTM_Hplane);
eTE_Hplane_TOT_inequality = eTE_Hplane_TOT / max(eTE_Hplane_TOT);
eTM_Hplane_TOT_inequality = eTM_Hplane_TOT / max(eTM_Hplane_TOT);

%% Save results
save('Output/TA_30x30_1d0/TA_glass_sqh_PO_model_30x30_1d0_equality.mat', 'theta', ...
    'eEl_Eplane_equality', 'eTE_Eplane_equality', 'eTM_Eplane_equality',
    'eTE_Eplane_TOT_equality', 'eTM_Eplane_TOT_equality', ...
    'eEl_Hplane_equality', 'eTE_Hplane_equality', 'eTM_Hplane_equality',
    'eTE_Hplane_TOT_equality', 'eTM_Hplane_TOT_equality')

save('Output/TA_30x30_1d0/TA_glass_sqh_PO_model_30x30_1d0_inequality.mat', 'theta', ...
    'eEl_Eplane_inequality', 'eTE_Eplane_inequality', 'eTM_Eplane_inequality',
    'eTE_Eplane_TOT_inequality', 'eTM_Eplane_TOT_inequality', ...
    'eEl_Hplane_inequality', 'eTE_Hplane_inequality', 'eTM_Hplane_inequality',
    'eTE_Hplane_TOT_inequality', 'eTM_Hplane_TOT_inequality')
```

B.8.3 SES PO Model

```
close all,
clearvars,
clc,
format long e

%% Constants
load('../constants.mat'); % physical constants
load('../free_space.mat'); % f0, lambda0, k0

%% Parameters
M = 40;                          % number of elements along the main axis
qe = 1;                          % element pattern power factor
qf = 12.5;                       % feed pattern power factor
W = 0.3 * lambda0;               % periodicity
theta = linspace(-pi, pi, 500);  % elevation angle [rad]
phi = linspace(-pi, pi, 500);    % azimuth angle [rad]

%% FF calculation
load('PO/SES_40x40_100d0_0_0_30_0.mat');

eTE = cell(M, M);
for m = 1:1:M
    for n = 1:1:M
        xl = W * ( m - ( ( M + 2 ) / 2 ) );
        xu = W * ( m - ( M / 2 ) );
```

```
        yl = W * ( n - ( ( M + 2 ) / 2 ) );
        yu = W * ( n - ( M / 2 ) );
        int_x = @(theta, phi, x) exp( 1i * k0 .* sin(theta) .* cos(phi) .* x );
        int_y = @(theta, phi, y) exp( 1i * k0 .* sin(theta) .* sin(phi) .* y );
        Ix = @(theta, phi) integral(@(x) int_x(theta, phi, x), xl, xu, 'ArrayValued',
        true);
        Iy = @(theta, phi) integral(@(y) int_y(theta, phi, y), yl, yu, 'ArrayValued',
        true);
        Js_theta_mn = 1;
        JtildeTE_mn = @(theta, phi) S21_TE_opt(m, n) .* Js_theta_mn .* Ix(theta, phi) .*
        Iy(theta, phi);
        eTE{m, n} = @(theta, phi) JtildeTE_mn(theta, phi);
    end
end

eTE_theta_tot = @(theta, phi) sum(cell2mat(cellfun(@(f) f(theta, phi), eTE(:),
'UniformOutput', false)));

%% Results
eTE_func = @(theta, phi) abs(eTE_theta_tot(theta, phi));
eEL_func = @(theta, phi) abs( cos(theta).^qe ) .* ( mean(abs(S11_TE_opt), 'all') .* (
theta < (-pi/2) | theta > (pi/2) ) + mean(abs(S21_TE_opt), 'all') .* ( theta >= (-pi/2)
& theta <= (pi/2) ) );
eTE_TOT_func = @(theta, phi) eTE_func(theta, phi) .* eEL_func(theta, phi);

% E plane
eEl_Eplane = eEL_func(theta, 0);
eTE_Eplane = eTE_func(theta, 0);
eTE_Eplane_TOT = eTE_TOT_func(theta, 0);

save('PO/SES_PO_model_40x40_100d0_0_0_30_0.mat', 'theta', ...
    'eEl_Eplane', 'eTE_Eplane', 'eTE_Eplane_TOT')
```

```
close all;
clearvars;
clc;
format long e

%% Constants & Parameters
load('../constants.mat');           % Physical constants
load('../free_space.mat');          % f0, lambda0, k0
M = 40;                             % Number of elements per axis
qe = 1;                             % Element-level pattern exponent
W = 0.3 * lambda0;                  % Element periodicity
theta = linspace(-pi, pi, 500);     % Elevation angles
phi   = linspace(-pi, pi, 500);     % Azimuth angles
[THETA, PHI] = meshgrid(theta, phi);
U = sin(THETA) .* cos(PHI);
V = sin(THETA) .* sin(PHI);

%% Total Field Calculation
load('PO/SES_40x40_100d0_0_0_20_45.mat'); % S parameters
E_TOT = zeros(size(U));

for m = 1:M
    for n = 1:M
        xl = W * ( m - ( ( M + 2 ) / 2 ) );
        xu = W * ( m - ( M / 2 ) );
```

```
        yl = W * ( n - ( ( M + 2 ) / 2 ) );
        yu = W * ( n - ( M / 2 ) );
        kx = k0 * U; % element-wise on grid
        ky = k0 * V;
        Ix = ( exp(1i .* kx .* xu) - exp(1i .* kx .* xl) ) ./ (1i * kx);
        Iy = ( exp(1i .* ky .* yu) - exp(1i .* ky .* yl) ) ./ (1i * ky);
        Js_theta_mn = 1;
        JtildeTE_mn =  S21_TE_opt(m, n) .* Js_theta_mn .* Ix .* Iy;
        eTE_mn = JtildeTE_mn;
        E_TOT = E_TOT + eTE_mn;
    end
end

%% Element-level pattern modulation
eEL = abs(cos(THETA)).^qe;
reflected   = mean(abs(S11_TE_opt), 'all');
transmitted = mean(abs(S21_TE_opt), 'all');
backward_mask = (THETA < -pi/2) | (THETA > pi/2);
forward_mask  = ~backward_mask;
eEL = eEL .* (reflected * backward_mask + transmitted * forward_mask);

%% Apply modulation and normalize
E_TOT = abs(E_TOT .* eEL);
E_TOT = E_TOT / max(E_TOT(:));

%% Peak
[~, idx] = max(E_TOT(:));
[row, col] = ind2sub(size(E_TOT), idx);
u0 = U(row, col);
v0 = V(row, col);
theta_peak = asin(sqrt(u0.^2 + v0.^2));
phi_peak = atan2(v0, u0);
save('PO/SES_40x40_100d0_0_0_20_45_UV', 'U', 'V', 'E_TOT', 'theta_peak', 'phi_peak')
```

Appendix C

DXF Generation Files

C.1 Fixed-height Structure

```
% fid = output file name (.dxf)
% center = (x,y) coordinates UC center
% Width = UC width (x dimension)
% Height = UC height (y dimension)
function dxfRectangle_fixedT(fid, Center, Width, Height)
    X1 = Center(1) - Width/2;
    X2 = Center(1) + Width/2;
    Y1 = Center(2) + Height/2;
    Y2 = Center(2) - Height/2;

    fprintf(fid, '0\n');
    fprintf(fid, 'SECTION\n');
    fprintf(fid, '2\n');
    fprintf(fid, 'HEADER\n');
    fprintf(fid, '0\n');
    fprintf(fid, 'ENDSEC\n');

    fprintf(fid, '0\n');
    fprintf(fid, 'SECTION\n');
    fprintf(fid, '2\n');
    fprintf(fid, 'ENTITIES\n');
    fprintf(fid, '0\n');
    fprintf(fid, 'SOLID\n');
    fprintf(fid, '8\n');
    fprintf(fid, '0\n');

    % Write vertex (X1,Y1)
    fprintf(fid, '10\n');
    fprintf(fid, '%f\n', X1);
    fprintf(fid, '20\n');
    fprintf(fid, '%f\n', Y1);
    fprintf(fid, '30\n');
    fprintf(fid, '0.0\n');

    % Write vertex (X1,Y2)
    fprintf(fid, '11\n');
    fprintf(fid, '%f\n', X1);
    fprintf(fid, '21\n');
    fprintf(fid, '%f\n', Y2);
```

```
    fprintf(fid, '31\n');
    fprintf(fid, '0.0\n');

    % Write vertex (X2,Y1)
    fprintf(fid, '12\n');
    fprintf(fid, '%f\n', X2);
    fprintf(fid, '22\n');
    fprintf(fid, '%f\n', Y1);
    fprintf(fid, '32\n');
    fprintf(fid, '0.0\n');

    % Write vertex (X2,Y2)
    fprintf(fid, '13\n');
    fprintf(fid, '%f\n', X2);
    fprintf(fid, '23\n');
    fprintf(fid, '%f\n', Y2);
    fprintf(fid, '33\n');
    fprintf(fid, '0.0\n');

    fprintf(fid, '0\n');
    fprintf(fid, 'EOF\n');
end
```

C.2 Variable-height Structure

```
% fid = output file name (.dxf)
% Center = (x,y) coordinates UC center
% Width = UC width along x and y
% Thickness = UC thickness
function dxfRectangle_variableT(fid, Center, Width, Thickness)
    X1 = Center(1) - Width/2;
    X2 = Center(1) + Width/2;
    Y1 = Center(2) + Width/2;
    Y2 = Center(2) - Width/2;

    fprintf(fid, '0\n');
    fprintf(fid, 'SECTION\n');
    fprintf(fid, '2\n');
    fprintf(fid, 'HEADER\n');
    fprintf(fid, '0\n');
    fprintf(fid, 'ENDSEC\n');

    fprintf(fid, '0\n');
    fprintf(fid, 'SECTION\n');
    fprintf(fid, '2\n');
    fprintf(fid, 'ENTITIES\n');
    fprintf(fid, '0\n');
    fprintf(fid, 'SOLID\n');
    fprintf(fid, '8\n');
    fprintf(fid, '0\n');

    % Write vertex (X1,Y1)
    fprintf(fid, '10\n');
    fprintf(fid, '%f\n', X1);
    fprintf(fid, '20\n');
    fprintf(fid, '%f\n', Y1);
    fprintf(fid, '30\n');
```

```
33      fprintf(fid, '%f\n', Thickness);
34
35      % Write vertex (X1,Y2)
36      fprintf(fid, '11\n');
37      fprintf(fid, '%f\n', X1);
38      fprintf(fid, '21\n');
39      fprintf(fid, '%f\n', Y2);
40      fprintf(fid, '31\n');
41      fprintf(fid, '%f\n', Thickness);
42
43      % Write vertex (X2,Y1)
44      fprintf(fid, '12\n');
45      fprintf(fid, '%f\n', X2);
46      fprintf(fid, '22\n');
47      fprintf(fid, '%f\n', Y1);
48      fprintf(fid, '32\n');
49      fprintf(fid, '%f\n', Thickness);
50
51      % Write vertex (X2,Y2)
52      fprintf(fid, '13\n');
53      fprintf(fid, '%f\n', X2);
54      fprintf(fid, '23\n');
55      fprintf(fid, '%f\n', Y2);
56      fprintf(fid, '33\n');
57      fprintf(fid, '%f\n', Thickness);
58
59      fprintf(fid, '0\n');
60      fprintf(fid, 'EOF\n');
61  end
```

Appendix D

Automatization of the Design Flow

D.1 Simulation Class

D.1.1 Header File

```
#ifndef SIMULATION_H
#define SIMULATION_H

using namespace std;

class Simulation
{
public:
    // Constructor
    explicit Simulation(unsigned int = 0);

    // Public methods
    void editCode(unsigned int M, float FD, int theta, int phi);
    void run();
    void restoreCode(unsigned int M, float FD, int theta, int phi);

private:
    unsigned int correct;
    string filename_phase_map = "phase_map_TA_glass_sqh.m";
    string filename_test_opt = "test_opt_TA_glass_sqh.m";
    string filename_TA_simple_model = "test_TA_glass_sqh_simple_model.m";
    string filename_TA_PO_model = "test_TA_glass_sqh_PO_model.m";
    string filename_dxf_equality = "dxf_creation_equality.m";
    string filename_dxf_inequality = "dxf_creation_inequality.m";
    string filename_dxf_equality_circ = "dxf_creation_equality_circ.m";
    string filename_dxf_inequality_circ = "dxf_creation_inequality_circ.m";
};

#endif
```

D.1.2 Source File

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <filesystem>
```

```cpp
#include <cstdlib>
#include <string>
#include <cstring>
#include <vector>
#include <cmath>
#include <iomanip>
#include <regex>

#include "Simulation.hpp"

using namespace std;

Simulation::Simulation(unsigned int c)
    : correct{c} {}

void Simulation::editCode(unsigned int M, float FD, int theta, int phi)
{
    int FD_int = static_cast<int>(FD); // extract integer part
    int FD_dec = (FD - FD_int) * 10;   // extract fractional part

    string M_initial = "M = 30;";
    string FD_initial = "FD = 0.8;";
    string theta_beam_initial = "theta_beam = deg2rad(0);";
    string phi_beam_initial = "phi_beam = deg2rad(0);";

    string M_final = "M = " + to_string(M) + ";";
    string FD_final = "FD = " + to_string(FD_int) + "." + to_string(FD_dec) + ";";
    string theta_beam_final = "theta_beam = deg2rad(" + to_string(theta) + ");";
    string phi_beam_final = "phi_beam = deg2rad(" + to_string(phi) + ");";

    /* phase map ***************************************************/
    system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
    ↪  filename_phase_map).c_str());
    system(("sed -i -e 's/" + FD_initial + "/" + FD_final + "/g' " +
    ↪  filename_phase_map).c_str());
    system(("sed -i -e 's/" + theta_beam_initial + "/" + theta_beam_final + "/g' " +
    ↪  filename_phase_map).c_str());
    system(("sed -i -e 's/" + phi_beam_initial + "/" + phi_beam_final + "/g' " +
    ↪  filename_phase_map).c_str());

    system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
    ↪  filename_phase_map).c_str());
    system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
    ↪  filename_phase_map).c_str());
    system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
    ↪  filename_phase_map).c_str());
    system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
    ↪  filename_phase_map).c_str());

    /* TA optimizer ************************************************/
    system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
    ↪  filename_test_opt).c_str());
    system(("sed -i -e 's/" + FD_initial + "/" + FD_final + "/g' " +
    ↪  filename_test_opt).c_str());
    system(("sed -i -e 's/" + theta_beam_initial + "/" + theta_beam_final + "/g' " +
    ↪  filename_test_opt).c_str());
    system(("sed -i -e 's/" + phi_beam_initial + "/" + phi_beam_final + "/g' " +
    ↪  filename_test_opt).c_str());

```

```
52      system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
        ↪  filename_test_opt).c_str());
53      system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
        ↪  filename_test_opt).c_str());
54      system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
        ↪  filename_test_opt).c_str());
55      system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
        ↪  filename_test_opt).c_str());
56
57      /* theoretical model (simple) ********************************/
58      system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_TA_simple_model).c_str());
59      system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
        ↪  filename_TA_simple_model).c_str());
60      system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
        ↪  filename_TA_simple_model).c_str());
61      system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
        ↪  filename_TA_simple_model).c_str());
62      system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
        ↪  filename_TA_simple_model).c_str());
63
64      /* theoretical model (PO) ************************************/
65      system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_TA_PO_model).c_str());
66      system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
        ↪  filename_TA_PO_model).c_str());
67      system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
        ↪  filename_TA_PO_model).c_str());
68      system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
        ↪  filename_TA_PO_model).c_str());
69      system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
        ↪  filename_TA_PO_model).c_str());
70
71      /* dxf creation **********************************************/
72      system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_equality).c_str());
73      system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_inequality).c_str());
74      system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_equality_circ).c_str());
75      system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
76
77      system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
        ↪  filename_dxf_equality).c_str());
78      system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
        ↪  filename_dxf_inequality).c_str());
79      system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
        ↪  filename_dxf_equality_circ).c_str());
80      system(("sed -i -e 's/30x30/" + to_string(M) + "x" + to_string(M) + "/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
81
82      system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
        ↪  filename_dxf_equality).c_str());
83      system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
        ↪  filename_dxf_inequality).c_str());
84      system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
        ↪  filename_dxf_equality_circ).c_str());
```

```
85      system(("sed -i -e 's/0d8/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
86
87      system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
        ↪  filename_dxf_equality).c_str());
88      system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
        ↪  filename_dxf_inequality).c_str());
89      system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
        ↪  filename_dxf_equality_circ).c_str());
90      system(("sed -i -e 's/tbeam0/tbeam" + to_string(theta) + "/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
91
92      system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
        ↪  filename_dxf_equality).c_str());
93      system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
        ↪  filename_dxf_inequality).c_str());
94      system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
        ↪  filename_dxf_equality_circ).c_str());
95      system(("sed -i -e 's/pbeam0/pbeam" + to_string(phi) + "/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
96  }
97
98  void Simulation::run()
99  {
100     /* phase map ************************************************/
101     cout << "*********************************************************************" <<
        ↪  endl
102          << "Running MATLAB script " << filename_phase_map << " .." << endl
103          << endl;
104     string command = "matlab -batch \"run('" + filename_phase_map + "');\"";
105     int status = system(command.c_str());
106     if (status == 0)
107         cout << "MATLAB script " << filename_phase_map << " executed successfully!" <<
            ↪  endl;
108     else
109         cerr << "Failed to execute MATLAB script " << filename_phase_map << "." << endl;
110     cout << "*********************************************************************" <<
        ↪  endl
111          << endl;
112
113     /* TA optimizer *********************************************/
114     cout << "*********************************************************************" <<
        ↪  endl
115          << "Running MATLAB script " << filename_test_opt << " .." << endl
116          << endl;
117     command = "matlab -batch \"run('" + filename_test_opt + "');\"";
118     status = system(command.c_str());
119     if (status == 0)
120         cout << "MATLAB script " << filename_test_opt << " executed successfully!" <<
            ↪  endl;
121     else
122         cerr << "Failed to execute MATLAB script " << filename_test_opt << "." << endl;
123     cout << "*********************************************************************" <<
        ↪  endl
124          << endl;
125
126     /* theoretical model (simple) ***********************************/
127     cout << "*********************************************************************" <<
        ↪  endl
```

```
128             << "Running MATLAB script " << filename_TA_simple_model << " .." << endl
129             << endl;
130     command = "matlab -batch \"run('" + filename_TA_simple_model + "');\"";
131     status = system(command.c_str());
132     if (status == 0)
133         cout << "MATLAB script " << filename_TA_simple_model << " executed
            ↪  successfully!" << endl;
134     else
135         cerr << "Failed to execute MATLAB script " << filename_TA_simple_model << "." <<
            ↪  endl;
136     cout << "*********************************************************************" <<
        ↪  endl
137          << endl;
138
139     /* theoretical model (PO) ************************************/
140     cout << "*********************************************************************" <<
        ↪  endl
141          << "Running MATLAB script " << filename_TA_PO_model << " .." << endl
142          << endl;
143     command = "matlab -batch \"run('" + filename_TA_PO_model + "');\"";
144     status = system(command.c_str());
145     if (status == 0)
146         cout << "MATLAB script " << filename_TA_PO_model << " executed successfully!" <<
            ↪  endl;
147     else
148         cerr << "Failed to execute MATLAB script " << filename_TA_PO_model << "." <<
            ↪  endl;
149     cout << "*********************************************************************" <<
        ↪  endl
150          << endl;
151
152     /* dxf equality ******************************************/
153     cout << "*********************************************************************" <<
        ↪  endl
154          << "Running MATLAB script " << filename_dxf_equality << " .." << endl
155          << endl;
156     command = "matlab -batch \"run('" + filename_dxf_equality + "');\"";
157     status = system(command.c_str());
158     if (status == 0)
159         cout << "MATLAB script " << filename_dxf_equality << " executed successfully!"
            ↪  << endl;
160     else
161         cerr << "Failed to execute MATLAB script " << filename_dxf_equality << "." <<
            ↪  endl;
162     cout << "*********************************************************************" <<
        ↪  endl
163          << endl;
164
165     /* dxf inequality ****************************************/
166     cout << "*********************************************************************" <<
        ↪  endl
167          << "Running MATLAB script " << filename_dxf_inequality << " .." << endl
168          << endl;
169     command = "matlab -batch \"run('" + filename_dxf_inequality + "');\"";
170     status = system(command.c_str());
171     if (status == 0)
172         cout << "MATLAB script " << filename_dxf_inequality << " executed successfully!"
            ↪  << endl;
173     else
```

```
174         cerr << "Failed to execute MATLAB script " << filename_dxf_inequality << "." <<
            ↪ endl;
175     cout << "*********************************************************************" <<
        ↪ endl
176          << endl;
177
178     /* dxf equality circ ***********************************************/
179     cout << "*********************************************************************" <<
        ↪ endl
180          << "Running MATLAB script " << filename_dxf_equality_circ << " .." << endl
181          << endl;
182     command = "matlab -batch \"run('" + filename_dxf_equality_circ + "');\"";
183     status = system(command.c_str());
184     if (status == 0)
185         cout << "MATLAB script " << filename_dxf_equality_circ << " executed
            ↪ successfully!" << endl;
186     else
187         cerr << "Failed to execute MATLAB script " << filename_dxf_equality_circ << "."
            ↪ << endl;
188     cout << "*********************************************************************" <<
        ↪ endl
189          << endl;
190
191     /* dxf inequality circ *********************************************/
192     cout << "*********************************************************************" <<
        ↪ endl
193          << "Running MATLAB script " << filename_dxf_inequality_circ << " .." << endl
194          << endl;
195     command = "matlab -batch \"run('" + filename_dxf_inequality_circ + "');\"";
196     status = system(command.c_str());
197     if (status == 0)
198         cout << "MATLAB script " << filename_dxf_inequality_circ << " executed
            ↪ successfully!" << endl;
199     else
200         cerr << "Failed to execute MATLAB script " << filename_dxf_inequality_circ <<
            ↪ "." << endl;
201     cout << "*********************************************************************" <<
        ↪ endl
202          << endl;
203 }
204
205 void Simulation::restoreCode(unsigned int M, float FD, int theta, int phi)
206 {
207     int FD_int = static_cast<int>(FD); // extract integer part
208     int FD_dec = (FD - FD_int) * 10;   // extract fractional part
209
210     string M_initial = "M = " + to_string(M) + ";";
211     string FD_initial = "FD = " + to_string(FD_int) + "." + to_string(FD_dec) + ";";
212     string theta_beam_initial = "theta_beam = deg2rad(" + to_string(theta) + ");";
213     string phi_beam_initial = "phi_beam = deg2rad(" + to_string(phi) + ");";
214
215     string M_final = "M = 30;";
216     string FD_final = "FD = 0.8;";
217     string theta_beam_final = "theta_beam = deg2rad(0);";
218     string phi_beam_final = "phi_beam = deg2rad(0);";
219
220     /* phase map ***********************************************/
221     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪ filename_phase_map).c_str());
```

```
222     system(("sed -i -e 's/" + FD_initial + "/" + FD_final + "/g' " +
        ↪  filename_phase_map).c_str());
223     system(("sed -i -e 's/" + theta_beam_initial + "/" + theta_beam_final + "/g' " +
        ↪  filename_phase_map).c_str());
224     system(("sed -i -e 's/" + phi_beam_initial + "/" + phi_beam_final + "/g' " +
        ↪  filename_phase_map).c_str());
225
226     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_phase_map).c_str());
227     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_phase_map).c_str());
228     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_phase_map).c_str());
229     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_phase_map).c_str());
230
231     /* TA optimizer ************************************************/
232     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_test_opt).c_str());
233     system(("sed -i -e 's/" + FD_initial + "/" + FD_final + "/g' " +
        ↪  filename_test_opt).c_str());
234     system(("sed -i -e 's/" + theta_beam_initial + "/" + theta_beam_final + "/g' " +
        ↪  filename_test_opt).c_str());
235     system(("sed -i -e 's/" + phi_beam_initial + "/" + phi_beam_final + "/g' " +
        ↪  filename_test_opt).c_str());
236
237     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_test_opt).c_str());
238     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_test_opt).c_str());
239     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_test_opt).c_str());
240     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_test_opt).c_str());
241
242     /* theoretical model (simple) **********************************/
243     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_TA_simple_model).c_str());
244     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_TA_simple_model).c_str());
245     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_TA_simple_model).c_str());
246     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_TA_simple_model).c_str());
247     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_TA_simple_model).c_str());
248
249     /* theoretical model (PO) **************************************/
250     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_TA_PO_model).c_str());
251     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_TA_PO_model).c_str());
252     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_TA_PO_model).c_str());
253     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_TA_PO_model).c_str());
254     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_TA_PO_model).c_str());
255
```

```
256     /* dxf creation ***********************************************/
257     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_equality).c_str());
258     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_inequality).c_str());
259     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_equality_circ).c_str());
260     system(("sed -i -e 's/" + M_initial + "/" + M_final + "/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
261
262     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_dxf_equality).c_str());
263     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_dxf_inequality).c_str());
264     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_dxf_equality_circ).c_str());
265     system(("sed -i -e 's/" + to_string(M) + "x" + to_string(M) + "/30x30/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
266
267     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_dxf_equality).c_str());
268     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_dxf_inequality).c_str());
269     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_dxf_equality_circ).c_str());
270     system(("sed -i -e 's/" + to_string(FD_int) + "d" + to_string(FD_dec) + "/0d8/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
271     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_dxf_equality).c_str());
272     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_dxf_inequality).c_str());
273     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_dxf_equality_circ).c_str());
274     system(("sed -i -e 's/tbeam" + to_string(theta) + "/tbeam0/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
275
276     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_dxf_equality).c_str());
277     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_dxf_inequality).c_str());
278     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_dxf_equality_circ).c_str());
279     system(("sed -i -e 's/pbeam" + to_string(phi) + "/pbeam0/g' " +
        ↪  filename_dxf_inequality_circ).c_str());
280 }
```

D.2 Main Code

```
1  #include <iostream>
2  #include <fstream>
3  #include <sstream>
4  #include <filesystem>
5  #include <cstdlib>
6  #include <string>
7  #include <cstring>
8  #include <vector>
9  #include <cmath>
```

```cpp
#include <iomanip>
#include <regex>
#include "Simulation.hpp"

using namespace std;

// Function to check if a value is a positive integer
bool isPositiveInteger(double number)
{
    return (number > 0) && (std::fmod(number, 1) == 0);
}

// Function to check if a value is a positive float
bool isPositiveFloat(double number)
{
    return (number > 0) && (std::fmod(number, 1) != 0);
}

int main(int argc, char *argv[])
{
    int ret = 0;                    // Return code
    Simulation Simulator;           // Simulation instance
    unsigned int M = 30;            // Default value for M
    float FD = 1.0;                 // Default value for FD
    int theta_beam, phi_beam = 0; // Default value for beam direction angles

    // Check the number of arguments
    if (argc != 5)
    {
        cout << "Invalid arguments! Please provide exactly four arguments: <M> <F/D> <theta_beam> <phi_beam>" << endl;
        return 1; // Exit with error
    }

    // Convert and validate the first argument (M)
    char *endPtr1;
    double mValue = std::strtod(argv[1], &endPtr1);
    if (*endPtr1 == '\0' && isPositiveInteger(mValue))
    {
        M = static_cast<unsigned int>(mValue);
    }
    else
    {
        cout << "Invalid value for M! It must be a positive integer. Aborting execution." << endl;
        return 1;
    }

    // Convert and validate the second argument (F/D)
    char *endPtr2;
    double fdValue = std::strtod(argv[2], &endPtr2);
    if (*endPtr2 == '\0' && fdValue > 0)
    {
        FD = static_cast<float>(fdValue);
    }
    else
    {
        cout << "Invalid value for F/D! It must be a positive number. Aborting execution." << endl;
```

```
66          return 1;
67      }
68
69      // Convert and validate the third argument (beam direction, theta)
70      char *endPtr3;
71      double thetaValue = std::strtod(argv[3], &endPtr3);
72      if (*endPtr3 == '\0')
73      {
74          theta_beam = static_cast<int>(thetaValue);
75      }
76
77      // Convert and validate the fourth argument (beam direction, phi)
78      char *endPtr4;
79      double phiValue = std::strtod(argv[4], &endPtr4);
80      if (*endPtr4 == '\0')
81      {
82          phi_beam = static_cast<int>(phiValue);
83      }
84
85      // Call Simulation methods
86      system("clear");
87      cout << "**************************************************************************" <<
        ↪  endl;
88      cout << "OPTIMIZATION-BASED TA DESIGN" << endl
89           << "Number of elements: " << M << " x " << M << endl
90           << "Focal lenght: F/D = " << FD << endl
91           << "Theta beam: theta = " << theta_beam << endl
92           << "Phi beam: phi = " << phi_beam << endl
93           << endl;
94      Simulator.editCode(M, FD, theta_beam, phi_beam);
95      Simulator.run();
96      Simulator.restoreCode(M, FD, theta_beam, phi_beam);
97
98      // Return success code
99      return ret;
100 }
```

Acknowledgements

At the end of this Master's Thesis journey, my sincere gratitude goes to my supervisor, Prof. Paola Pirinoli, with whom I shared both enthusiasm when results were promising and anxiety when challenges arose. Her guidance, expertise, and support have been invaluable throughout this work.

I would also like to extend my appreciation to my co-supervisors, Dott. Michele Beccaria and Prof. Giuseppe Vecchi, for their precious advice, insightful discussions and continuous support throughout the development of the Thesis.

Furthermore, I cannot forget to thank Dott. Andrea Massaccesi and Prof. Daniele Milanesio for their expert consultations, which proved to be crucial in overcoming apparently impossible obstacles, respectively at the beginning and at the end of this work.

A special thank you goes to my parents, Enrica and Fulvio, for their deep love, patience, and support — especially for putting up with my not-so-easy personality: thank you for forcing me to go out for a walk whenever you noticed I needed to take a break but, above all, thank you for never giving up on me and teaching me that the easiest path is almost never the right one.

Letizia

Bibliography

[1] Abdelrhaman, A. H., Yang, F., Elsherbeni, A. Z. & Nayeri, P., (2017), *Analysis and Design of Transmitarray Antennas*, M&C Publishers (cit. on p. 7, 67, 68, 70).

[2] Adams, M. & Pour, M., (2021), *On the Gain Loss of Wide-Angle Scanning Phased Arrays with Narrow- and Wide-beam Element Patterns*, IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (APS/URSI), Singapore, 501-502 (cit. on p. 89).

[3] Ahmed, F., Singh, K., Esselle & K. P., (2023), *State-of-the-Art Passive Beam-Steering Antenna Technologies: Challenges and Capabilities*, IEEE access, **11**/69101-69116 (cit. on p. 89).

[4] Arnold, J. W. & Taylor, R. C., (1932), *Linearly Tapered Loaded Transmission Lines*, Proceedings of the Institute of Radio Engineers, **20**/11/1811-1817 (cit. on p. 48).

[5] Ashcroft, N. W. & Mermin, N. D., (1976), *Solid State Physics*, Cengage Learning, Inc (cit. on p. 9, 24).

[6] Banafaa, M., Shayea, I., Din, J., Azmi, M. H., Alashbi, A., Daradkeh, Y. I., Alhammadi, A., (2023), *6G Mobile Communication Technology: Requirements, Targets, Applications, Challenges, Advantages, and Opportunities*, Alexandria Engineering Journal, **64**/245-274 (cit. on p. V, 2).

[7] Beccaria, M., Addamo, G., Orefice, M., Peverini, O., Manfredi, D., Calignano, F., Virone, G. & Pirinoli, P., (2021), *Enhanced Efficiency and Reduced Side Lobe Level Convex Conformal Reflectarray*. Appl. Sci., **11**/9893 (cit. on p. 66).

[8] Bhattacharyya, A. K., (2006), *Phased Array Antennas: Floquet Analysis, Synthesis, BFNs, and Active Array Systems*, John Wiley & Sons, Inc, Publication, 61-73 (cit. on p. 25).

[9] Bhowmik, L.M., (2019), *Applications of Floquet Analysis to Modern Phased Array Antennas*, Doctoral Thesis, Electronics, University of Oklahoma (cit. on p. 26).

[10] Bilotti, F., Barbuto, M., Hamzavi-Zarghani, Z., Karamirad, M., Longhi, M., Monti, A., Ramaccia, D., Stefanini, L., Toscano, A. & Vellucci, S., (2024) *Reconfigurable intelligent surfaces as the key-enabling technology for smart electromagnetic environments*, Advances in Physics, arXiv, **10**/9 (cit. on p. VI, 2, 4).

[11] Chowdhury, M. Z., Shahjalal, Md., Ahmed, S. & Jang, Y. M., (2019), *6G Wireless Communication Systems: Applications, Requirements, Technologies, Challenges, and Research Directions*, IEEE Open Journal of the Communications Society, arXiv (cit. on p. 2, 3).

[12] Di Renzo, M., Debbah, M., Phan-Huy, D. T., Zappone, A., Alouini, M. S., Yuen, C., Sciancalepore, V., Alexandropoulos, G. C., Hoydis, J., Gacanin, H., de Rosny, J., Bounceur, A., Lerosey, G. & Fink, M., (2019), *Smart radio environments empowered by reconfigurable AI meta-surfaces: an idea whose time has come*, EURASIP Journal on Wireless Communications and Networking, **10**/1186 (cit. on p. 6).

[13] DXF files: `https://cadexchanger.com/dxf/` (cit. on p. 77).

[14] Francavilla, M. A., Martini, E., Maci, S. & Vecchi, G., (2015), *On the Numerical Simulation of Metasurfaces With Impedance Boundary Condition Integral Equations*, IEEE Transactions on Antennas and Propagation, **63**/5/2153-2161 (cit. on p. 23).

[15] Friis, H. T., (1946), *A Note on a Simple Transmission Formula*. Proceedings of the IEEE, **34**/5/254-256 (cit. on p. 75).

[16] Garnett, J. C. M., (1904), *Colours in metal glasses and in metallic films*, Philos. Trans. Roy. Soc. London A, Containing Papers Math. Phys. Character, **203**/359-371, 385-420 (cit. on p. 9, 24).

[17] Ghione, G. & Pirola, M., (2017), *Microwave Electronics*, 1-5 (cit. on p. VI, 3).

[18] Gómez, F., (2023), *Genetic algorithms for feature selection in machine learning*: `https://www.neuraldesigner.com/blog/genetic_algorithms_for_feature_selection/` (cit. on p. 18).

[19] Huang, J. & Encinar, J. A., (2008), *Reflectarray Antennas*, IEEE Press Editorial Board, John Wiley & Sons, Inc, Publication (cit. on p. 73).

[20] Jain, I. K., Kumar, R. & Panwar, S. S., (2019), *The impact of mobile blockers on millimeter-wave cellular systems*. IEEE J. Sel. Areas. Commun., **4**/37/854-868 (cit. on p. 3).

[21] Koutsos, M., (2022), *Study, modeling and design of fixed beam transmitarray antennas at* 300 GHz, Doctoral Thesis, Electronics, Université Rennes 1 (cit. on p. 66, 67, 73).

[22] Liang, J.C., Zhang, L., Luo, Z. *et al.*, (2024), *A filtering reconfigurable intelligent surface for interference-free wireless communications*, Nature Communications, **15**/3838 (cit. on p. 7).

[23] Li, M., Chen, S. L., Liu, Y. & Jay Guo, Y., (2023), *Wide-Angle Beam Scanning Phased Array Antennas: A Review*, IEEE Open Journal of Antennas and Propagation, **4**/695-712 (cit. on p. 89).

[24] Liu, G., Cruz, E. M., Pham, K., Ovejero, D. G. & Sauleau, R., (2018), *Low Scan Loss Bifocal Ka-band Transparent Transmitarray Antenna*, 2018 IEEE International Symposium on Antennas and Propagation & USNC/URSI National Radio Science Meeting, Boston, MA, USA (cit. on p. 7).

[25] Lupinacci, P. F., (2024), *Design of innovative Transparent Smart Electromagnetic Skins working in Transmitting Mode*, Master's Degree Thesis, Electronics Engineering, Politecnico di Torino (cit. on p. 8, 10).

[26] Lukianchuk, I., Tulashvili, Y., Podolyak, V., Horbariuk, R., Kovalchuk, V. & Bazyl, S., (2022), *Didactic Principles Of Education Students 3D-printing*, International Journal of Computer Science and Network Security, **22**/7 (cit. on p. 29).

[27] Madden, J., (2022), *The Ideal Band for 6G*, https://www.microwavejournal.com/articles/38670-the-ideal-band-for-6g (cit. on p. V, 2).

[28] Markel, V.A., (2016), *Introduction to the Maxwell Garnett approximation: tutorial*, J. Opt. Soc. Am., **33**/1244-1256 (cit. on p. 24).

[29] Massaccesi, A., (2019), *Dielectric Transmitarray Antennas: from Design to Realization using Additive Manufacturing Technique*. Doctoral thesis. Politecnico di Torino (cit. on p. 48).

[30] Massaccesi, A., Bertana, V., Beccaria, M., Marasso, S. L., Cocuzza, M., Dassano, G. & Pirinoli, P., (2023), *Three-Dimensional-Printed Wideband Perforated Dielectric-Only Reflectarray in Ka-Band*, IEEE Trans. Antennas Propag., **71**/10 (cit. on p. 23, 24, 25, 26, 28, 31).

[31] Massaccesi, A., Beccaria, M., Bertana, V., Marasso, S. L., Cocuzza, M., Dassano, G. & Pirinoli, P., (2024), *3D-printed wideband reflectarray antennas with mechanical beam-steering*, International Journal of Microwave and Wireless Technologies, **16**/1/21-29 (cit. on p. 90).

[32] Massaccesi, A., Pirinoli, P., Bertana, V., Scordo, G., Marasso, S. L., Cocuzza, M. & Dassano, G., (2018), *3D-Printable Dielectric Transmitarray with Enhanced Bandwidth at Millimeter-Waves*, IEEE Access, **6**/46407-46418 (cit. on p. 7, 30).

[33] Mazzoldi, P., Nigro, M. & Voci, C., (2021), *Fisica - Elettromagnetismo e Onde*, Volume II, Third Edition, EdiSES Università, 402-414 (cit. on p. 97).

[34] Mesa, F., Chen, M., Castillo-Tapia P. & Quevedo-Teruel, O., (2024), *Physical Optics Applied to Parallel-Plate Lens Antennas*, IEEE Open Journal of Antennas and Propagation, **5**/4/833-844 (cit. on p. 73).

[35] MathWorks®, MATLAB®: https://www.mathworks.com/products/matlab.html, version R2024b (cit. on p. 11).

[36] MATLAB® GA: https://it.mathworks.com/discovery/genetic-algorithm.html (cit. on p. 53).

[37] Mencagli, M., J., Martini, E., Maci, S. & Albani, M., (2020), *A Physical Optics Approach to the Analysis of Metascreens*, IEEE access, **8**/162634-162641 (cit. on p. 73).

[38] Mubeen, S., (2018), *Design of Uniform and Nonuniform Circular Arrays Comparison with FFA and RLS*, Progress of Electrical and Electronic Engineering, **1**/3 (cit. on p. 83).

[39] Oliveri, G., Zardi, F., Gottardi, G. & Massa, A., (2024), *Optically-Transparent EM Skins for Outdoor-to-Indoor mm-Wave Wireless Communications*, IEEE Access, **12**/65178-65191 (cit. on p. 7, 8).

[40] Perotoni, M. B., Andrade, L. A., Junqueira, C., (2016), *Design, Prototyping and Measurement of a Cascaded 6-GHz Frequency Selective Surface Array*, Journal of Aerospace Technology and Management, **8**/2, 137-142 (cit. on p. 99).

[41] Podczerwinski, J., *Optimization and How to Do it in CST*: https://wiki.physics.wisc.edu/ObsCos/images/8/82/Optimization_and_How_to_Do_it_in_CST.pdf (cit. on p. 16).

[42] Pozar, D. M., (1998), *Microwave Engineering*, Fourth Edition, John Wiley & Sons, Inc, Publication, 188-194 (cit. on p. VIII, 110).

[43] Punnathanam, V., Sivadurgaprasad, C. & Kotecha, P., (2016), *On the performance of MATLAB's inbuilt genetic algorithm on single and multi-objective unconstrained optimization problems*, International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, India, 3976-3981 (cit. on p. 53).

[44] Rastrigin, L. A., (1974), *Systems of extremal control*, Mir, Moscow (cit. on p. 19).

[45] Sahin, M. M., Arslan, H. & Chen, K. C., (2022), *Control of Electromagnetic Radiation on Coexisting Smart Radio Environment*, IEEE Open Journal of the Communication Society, **3**/557-573 (cit. on p. 5).

[46] Systèmes, Dassault, CST Studio Suite: `https://www.3ds.com/it/products/simulia/cst-studio-suite`, version 2015 (cit. on p. 8).

[47] Vaquero, A. F., Teixeira, J., Matos, S. A., Arrebola, M., Costa, J. R., Felício, J. M., Fernandes, C., A. & Fonseca, N. J. C., (2023), *Design of Low Profile Transmitarray Antennas with Wide Mechanical Beam Steering at Millimeter-Waves*, IEEE Transactions on Antennas and Propagation, **71**/4/3713-3718 (cit. on p. 89, 90).

[48] Xu, Q., (2018), *How to Obtain the Scattered Near Field by Using CST*, Research Gate (cit. on p. 92, 99).

[49] Zheng, Z., Ren, W., Li, W., & Xue, Z., (2024), *A New 1 Bit Electronically Reconfigurable Transmitarray*, Electronics, **13**/7 (cit. on p. 7).

[50] Lectures and personal notes from the course *Advanced Antenna Engineering* (01NVSOQ), Master's Degree in Electronic Engineering, Politecnico di Torino, Prof. Giuseppe Vecchi & Paola Pirinoli, A.A. 2024/25 (cit. on p. VIII, 106).

[51] Lectures from the course *Advanced Optimization Techniques* (01QFFRV), PhD Program in Electrical, Electronics and Communications Engineering, Politecnico di Torino, Prof. Paola Pirinoli, A.A. 2024/25 (cit. on p. VI, 12, 13, 14, 16, 17, 18).

[52] Lectures from the course *Operational Research: Theory and Applications* (01QWTBH), Master's Degree in ICT For Smart Societies, Politecnico di Torino, Prof. Edoardo Fadda, A.A. 2022/23 (cit. on p. 14).