UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
CAMPUS DE SÃO JOÃO DA BOA VISTA

MILENA FERREIRA PAGNI

INTEGRAÇÃO DE PYTHON COM HFSS PARA DISPOSITIVOS DE MICROONDAS

São João da Boa Vista
2023

Milena Ferreira Pagni

INTEGRAÇÃO DE PYTHON COM HFSS PARA DISPOSITIVOS DE MICROONDAS

Trabalho de Graduação apresentado ao Conselho de Curso de Graduação em Engenharia Eletrônica e de Telecomunicações do Campus de São João da Boa Vista, Universidade Estatual Paulista, como parte dos requisitos para obtenção do diploma de Graduação em Engenharia Eletrônica e de Telecomunicações .

Orientador: Prof. Dr. Ivan Aritz Aldaya Garde

São João da Boa Vista
2023

P139i

Pagni, Milena Ferreira
Integração de Python com HFSS para dispositivos de microondas / Milena Ferreira Pagni. -- São João da Boa Vista, 2023
47 p. : il., tabs., fotos

Trabalho de conclusão de curso (Bacharelado - Engenharia de Telecomunicações) - Universidade Estadual Paulista (Unesp), Faculdade de Engenharia, São João da Boa Vista
Orientador: Ivan Aritz Aldaya Garde

1. Simulação (Computadores). 2. Python (Linguagem de programação de computador). 3. Antenas (Eletrônica). 4. Telecomunicações. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Engenharia, São João da Boa Vista. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE ENGENHARIA - CÂMPUS DE SÃO JOÃO DA BOA VISTA
GRADUAÇÃO EM ENGENHARIA ELETRÔNICA E DE TELECOMUNICAÇÕES

TRABALHO DE CONCLUSÃO DE CURSO

INTEGRAÇÃO DE PYTHON COM HFSS PARA DISPOSITIVOS DE MICROONDAS

Aluna: Milena Ferreira Pagni
Orientador: Prof. Dr. Ivan Aritz Aldaya Garde

Banca Examinadora:

- Ivan Aritz Aldaya Garde (Orientador)
- Rafael Abrantes Penchel (Examinador)
- Hildo Guillardi Júnior (Examinador)

A ata da defesa com as respectivas assinaturas dos membros encontra-se no prontuário da aluna (Expediente nº 090/2022)

São João da Boa Vista, 26 de maio de 2023

Dedico à minha família e amigos.

AGRADECIMENTOS

Gostaria de agradecer aos meus pais, Sandra e Roberto, à minha irmã, Nádia, e aos meus amigos pela ajuda, apoio e incentivo. Aos meus professores pelos ensinamentos. Ao meu orientador Dr. Ivan Aritz Aldaya Garde, que me ajudou a realizar esse trabalho. A Deus por dar-me forças e por me guiar nessa jornada.

“Todas as recompensas na vida, seja em riqueza, relacionamentos ou conhecimento, provêm de juros compostos.“

(Naval Ravikant)

RESUMO

A simulação computacional revolucionou o processo de fabricação nas indústrias, diminuindo a quantidade necessária de construções dos protótipos, diminuindo o tempo e, consequentemente, o seu custo. No meio de telecomunicações, um *software* muito utilizado é o HFSS, por meio dele é possível realizar análises de parâmetros como: coeficiente de reflexão, impedância de entrada e ganho. Porém, apesar de realizar simulações bastante contingente com a realidade, possui uma grande desvantagem, principalmente no âmbito dos projetos comerciais: não apresenta flexibilidade para implementar as otimizações. A flexibilidade é fornecida por linguagens de programação de alto nível, entretanto, a sua solução não é muito eficiente comparando com os *softwares* computacionais de eletromagnetismo. A integração do *software* HFSS com uma linguagem de programação, como Python, solucionaria esse problema, porém, apesar da existência de uma biblioteca no Python para a integração, a documentação não é clara. Desta forma, este trabalho buscou demonstrar uma integração passo a passo do HFSS com o Python para um projeto de uma antena *patch microstrip*. Tal antena foi projetada com uma frequência central de 10 GHz, impedância de entrada de 50 Ω, substrato Duroid 5880 e a altura de 0,787 mm. Utilizando tais características foi descoberta as dimensões faltantes. Em seguida, foi realizado a preparação e instalação das bibliotecas necessárias. Posteriormente, foi criado, configurado e executada a análise da antena. Por fim, com os dados obtidos, realizou-se uma representação gráfica e uma discussão dos resultados obtidos.

PALAVRAS-CHAVE: Antena. Integração. HFSS. Python (Linguagem de programação de computador).

ABSTRACT

Computational simulation has changed many production processes in industry, then, less prototypes were needed, bringing a save of money and time. In the field of Telecommunications, HFSS is a software commonly used for parameters analysis, such as reflection coefficient, input impedance and gain. However, despite the excellent results obtained when compared to experimental analysis, optimization process is a challenge. Many software languages are used as the key, otherwise, their efficiency is not as high as others electromagnetism software's. Connecting a software language, such as Python, and the HFSS may solve this issue. Although, the documentation of a framework which links both methods, is not totally clear, bringing some challenges in Its usage. Considering this trouble, this present study aims to show an integration, step-by-step, of both tools by using a patch microstrip antenna. The antenna was developed with a natural frequency of 10 GHz, 50 Ω of impedance, 0.787 mm of height and Duroid 5880 as the substract material. Using all these parameters, the other dimensions of the antenna were found. After, frameworks for the connection between both tools were configured and installed, enabling the analysis in HFSS. Finally, some data related to the analysis was extracted, and some graphic representations were made.

KEYWORDS: Antenna. Integration. HFSS. Python.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

2D	*Two Dimensions* - Duas Dimensões
3D	*Three Dimensions* - Três Dimensões
AEDT	*Ansys Electronics Desktop*
AESA	*Active Electronically Scanned Array* - Radar de Varredura Eletrônica Ativa
CST MWS	*CST Microwave Studio*
FDTD	*Finite-Difference Time-Domain* - Técnica de Domínio do Tempo de Diferenças Finitas
FEM	*Finite Element Method* - Método de Elementos Finitos
FIT	*Finite Integration Technique* - Técnica de Integração Finita
GPS	*Global positioning system* - Sistema de Posicionamento Global
HFSS	*High-Frequency Structure Simulator*
IEC	*International Electrotechnical Commission* - Comissão Eletrotécnica Internacional
MoM	*Method of Moments* - Método dos Momentos
IDE	*Integrated Development Environment* - Ambiente de Desenvolvimento Integrado
IEEE	*Institute of Electrical and Electronics Engineers*
MIC	*Microwave Integrated Circuit* - Circuitos Integrados de Micro-ondas
MMIC	*Microwave Monolithic Integrated Circuit* - Circuitos Integrados Monolíticos de Micro-ondas
PCB	*Printed Circuit Board* - Circuito Impresso em Placa
TLM	*Transmission Line Matrix* - Matriz de Linha de Transmissão

LISTA DE SÍMBOLOS

$^\circ$	Unidade de ângulo: Graus
π	Letra grega referente a uma constante matemática
$Tan\,\delta$	Tangente de perdas
Ω	Unidade de medida de resistência: Ohms
υ_0	Velocidade da luz
λ	Comprimento de onda
ΔL	Acréscimo de comprimento do *patch*
ε_r	Constante dielétrica
ε_{reff}	Permissividade relativa efetiva
e_{cd}	Eficiência de radiação da antena na unidade adimensional
μm	Micrômetros
t	Espessura do *patch* e do terra
f_r	Frequência central
L	Comprimento do *patch*
dB	Decibel
D_0	Diretividade
G	Ganho
GHz	Giga hertz
GND	Terra da antena
j	Indicador de número imaginário
h	Altura do substrato
mm	Milímetros
P_{in}	Potência total de entrada
P_{rad}	Potência total radiada
U	Intensidade de radiação

U_0	Intensidade de radiação de uma fonte isotrópica
y_0	Comprimento do *gap*
S_{11}	Coeficiente de reflexão
sub_x	Comprimento do substrato e do terra da antena *patch*
sub_y	Largura do substrato e do terra da antena *patch*
R_A	Resistência da antena nos terminais a - b
X_A	Reatância da antena nos terminais a - b
Z_A	Impedância da antena nos terminais a - b
Z_0	Impedância de entrada
W	Largura do *patch*
W_0	Largura da linha

SUMÁRIO

1 INTRODUÇÃO

1.1 ANTENAS *PATCH* DE ALTO DESEMPENHO

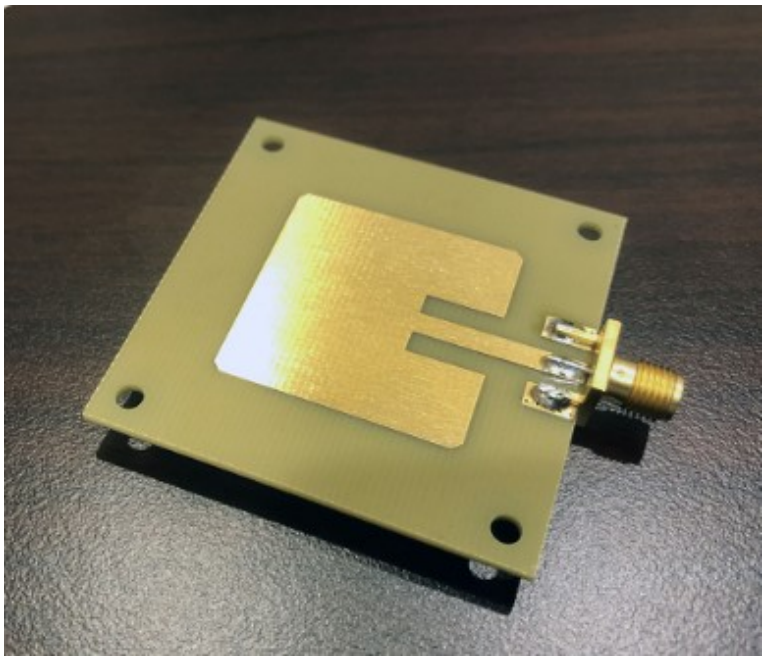
Uma antena é um dispositivo fundamental para um sistema de comunicação sem fio, sendo que a sua definição segundo a norma IEEE Definições Padronizadas de Termos para antenas (*IEE Standard Definitions of Terms for Antennas* – IEEE Std 145-19831) é dada como "um dispositivo para a radiação ou a recepção de ondas de rádio". Elas são classificadas em diversos tipos, sendo:

- Antena Filamentares: são muito utilizadas, devido ao fato da diversidade de aplicações, por exemplo, pode ser em navios, prédios e automóveis.
- Antena de Abertura: pelo motivo de ser fácil de ser colocada na fuselagem de aviões ou naves espaciais, são muito utilizadas por esses ramos.
- Conjuntos de Antena: são um grupo de elementos radiantes, com o intuito de gerar um diagrama de radiação, no qual não pode ser obtido apenas por um componente, podendo, por exemplo, aumentar a diretividade máxima.
- Antenas Refletoras: são empregadas no ramo espacial e tem a capacidade de transmitir e receber em grandes distâncias, por milhões de quilômetros, por esse motivo as antenas refletoras, apresentarem uma enorme dimensão, algumas chagando a mais de 300 metros de diâmetros.
- Antenas-Lentes: a sua forma geométrica ou o material são o que classificam, que podem utilizar lentes de todos os tipos, como côncavo-plana e convexo-plana, direcionando a radiação para onde deseja. Essa técnica resulta em perdas relativamente baixas, consequentemente, apresenta um comportamento próximo do ideal.
- Antena de *microstrip*: normalmente são feitos em forma retangulares (Figura 1) e circulares (Figura 2). Com aplicações em diversos ramos, são amplamente utilizados em razão de ter a capacidade de ser montadas em muitas superfícies (INC, 2021).

Detalhando mais sobre as antenas de *microstrip*, na década de 1970 ficou popular, sobretudo nas aplicações da indústria espacial, porém hoje em dia tem uma aplicação mais ampla, utilizados também para fins comerciais. Alguns exemplos de aplicações são: em comunicação moveis, comunicação via satélite, GPS e radar.

Esta antena específica, *microstrip*, é composta pelo plano terra e acima dele, encontra-se o substrato, pode ser composto por diferentes materiais, na qual a constante dielétrica (ε_r) varia como podemos visualizar na Tabela 1, mas tipicamente, apresenta um valor de $2{,}2 \leq \varepsilon_r \leq 12$. Por fim, na camada mais elevada, tem-se um elemento irradiante e uma linha de transmissão *microstrip* (POZAR, 2011). Estas antenas possuem diversos formatos, como triangular e elíptica, mas geralmente, como já comentado, são retangulares e circulares.

Figura 1 – Exemplo: antena *patch* retangular.



Fonte: (INC, 2021).

Figura 2 – Exemplo: antena *patch* circular.

Fonte: (AMITEC, 2018).

Segundo Patel, Narang e Jain (2013), as principais vantagens desse tipo de antena são a sua leveza, além de apresentam dimensões pequenas a um custo baixo para ser desenvolvidos, consequentemente, elas são fabricadas em massa. Em relação a sua frequência, podem suportar múltipla bandas, operar nas faixas onde antenas tradicionais teriam seu emprego inviabilizado, como exemplo a frequência de micro-ondas. Outro fator positivo, são a facilidade de ser gravadas no PCB, independentemente do formato, em funções disso, não apresentam dificuldade na integração com MICs ou MMICs.

Exemplificando mais dos principais benefícios citada acima, em cada uma das aplicações mencionadas anteriormente:

- Comunicação moveis: pontos positivos são as dimensões pequenas, custo baixo comparando

Tabela 1 – Aplicações de diferentes materiais na construção de antenas *patch*, incluindo a frequência de operação, permitividade relativa e tangente de perdas.

Material	Frequência	ε_r	Tan δ
Ceramic (A-35)	3 GHz	5,60	0,0041
Fused quartz	10 GHz	3,78	0,0001
Gallium arsenide	10 GHz	13,0	0,006
Glass (pyrex)	3 GHz	4,82	0,0054
Glazed ceramic	10 GHz	7,2	0,008
Lucite	10 GHz	2,56	0,005
Nylon (610)	3 GHz	2,84	0,012
Parafin	10 GHz	2,24	0,0002
Plexiglass	3 GHz	2,60	0,0057
Polyethylene	10 GHz	2,25	0,0004
Silicon	10 GHz	11,9	0,004
Styrofoam (103,7)	3 GHz	1.03	0,0001
Teflon	10 GHz	2,08	0,0004

Fonte: Adaptado do Pozar.

com outras antenas e “baixo perfil”;

- Comunicação via satélite: apresenta o diagrama de radiação circular, em razão de possuir uma baixa diretividade, se for retangular ou circular, caso seja outra forma, poderá ter uma resposta diferente no diagrama;
- GPS: também por causa da polarização ser circular, além de ser bastante compacto;
- Radar: esse tipo de antena tem o benefício de oferecer grande escala de fabricação, sempre com o mesmo desempenho, por um menor valor e tempo, comparando com as antenas tradicionais disponíveis no mercado, graças a tecnologia de fotolitografia utilizada. São amplamente utilizadas para a implementação de antenas de arranjos, que são requeridas em sistemas AESA. (PATEL; NARANG; JAIN, 2013).

No entanto, não são apenas vantagens, dependendo da aplicação do projeto, é preciso considerar e analisar as suas limitações antes de decidir se é a opção mais adequada. As suas limitações comparando com outras antenas são o baixo ganho, largura de banda estreita, além das perdas dielétricas e no condutor, resultarem em uma baixa eficiência (BHUNIA, 2013). De fato, muitas vezes é necessário projetar antenas *microstrip* com formas irregulares a fim de contornar algumas das limitações. Porém, a otimização topologia não trivial requer da utilização de algoritmos complexos.

1.2 PROBLEMA A RESOLVER

Os *softwares* computacionais de antenas são um ótimo meio de se assistir o *design* de antenas e realizar a análise do efeito de parâmetros importantes em figuras de mérito críticas com ganho, diretividade e o coeficiente de reflexão (S_{11}). Porém os algoritmos de otimização implementados nestes *softwares* geralmente carecem da flexibilidade requerida, por exemplo, para implementar a otimização topológica.

As linguagens de programação de alto nível, como podem ser, C, C++, Python ou Matlab, oferecem uma flexibilidade para se implementar algoritmos de otimização sofisticados, mas não contam com os métodos de integração eletromagnéticas eficientes dos simuladores específicos da área.

Assim, fazer a integração das simulações por meio da programação, é uma solução para uma melhor eficiência nos projetos eletromagnéticos. Realizando uma pesquisa sobre Python e o *software* HFSS, encontra-se uma bibliografia muito pequena sobre o assunto, pois apesar de existir uma biblioteca do Python, a documentação é confusa, ou seja, não é clara, faltando exemplos e podendo encontrar só o nome das funções sem outras informações importantes para o entendimento do tema.

1.3 INTEGRAÇÃO DO SOFTWARE HFSS COM A LINGUAGEM DE PROGRAMAÇÃO PYTHON PARA OTIMIZAÇÃO GEOMÉTRICA

Os software utilizam soluções numéricas, equações integrais e diferencias. A primeira, as equações integrais, faz o uso das equações de Maxwell com o intuito de resolver problemas eletromagnéticos com correntes desconhecidas. No outro procedimento, são utilizadas as derivas das equações de rotação de Maxwell ou equações de onda de Helmnholtz, por meio delas são empregados vários métodos, métodos dos elementos finitos (FEM - *Finite Element Method*), técnica de domínio do tempo de diferenças finitas (FDTD - *Finite-Difference Time-Domain*), método Matriz de Linha de Transmissão (TLM - *Transmission Line Matrix*), Técnica de Integração Finita (FIT - *Finite Integration Technique*) e métodos dos momentos (MoM - *Method of Moments*) (VANDENBOSCH; VASYLCHENKO, 2010).

Estas soluções são implementadas em *softwares*, sendo comerciais ou não, como MAGMAS 3D, que foi desenvolvido pela cooperação da Katholieke Universiteit Leuven da Bélgica em parceria com a Agência Espacial Europeia e baseia-se no IE-MoM (VANDENBOSCH; VASYLCHENKO, 2010). Descrevendo um pouco mais dos *softwares* comerciais, geralmente tem um alto valor, e cada um deles utilizam uma técnica para resolver questões de eletromagnetismo. O *CST Microwave Studio (CST MWS)*, por exemplo, é um *software* bastante usado nas empresas da área e baseia-se no FIT (VANDENBOSCH; VASYLCHENKO, 2010).

No entanto, o software mais conhecido mundialmente e empregado nas empresas é o HFSS, baseia-se no FEM, é versátil, podendo simular o comportamento de diversas topologias de antenas, filtros e outros dispositivos. Apresentando simulações eficientes com resultados realistas, como nos parâmetros (S, Y ou Z) ou na visualização em 3D do campo eletromagnético (VANDENBOSCH; VASYLCHENKO, 2010).

Como mencionado, os simuladores comerciais são extremamente eficientes para solucionar o problema eletromagnético, mas não são muito flexíveis na hora de implementar os algoritmos de otimização. Em razão disso, é interessante integrar o HFSS e Python. Existem duas maneiras de realizar a integração entre Python e HFSS, a primeira é executando *scripts* com comando pelo HFSS e a segunda, é o Python, por meio de uma IDE, executar todas as chamadas e comando no HFSS, assim, não é necessário o programa ser aberto manualmente, já que o Python irá fazer tudo, até mostrar os gráficos.

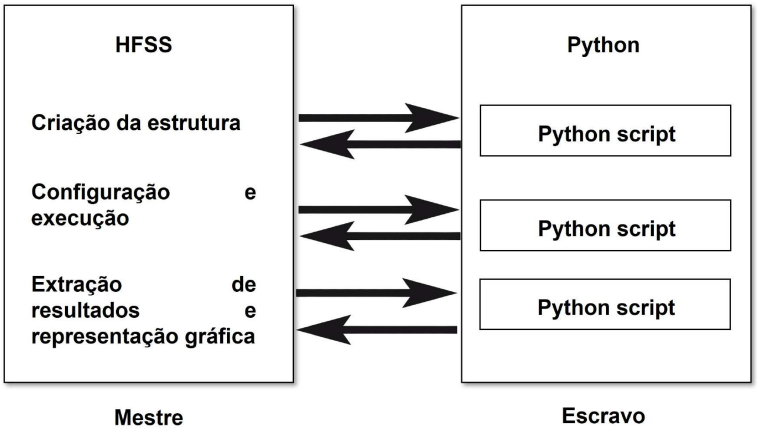
Nas Figura 3 e Figura 4 apresentamos a esquematização, descrevendo com alguns tópicos, os procedimentos da integração. Na Figura 3 tem-se a necessidade de abrir o *software* HFSS, o mestre

nesse caso, e executando o arquivo em Python para realizar a ação desejada.

Na Figura 4, a situação é inversa, os resultados sejam numéricos e/ou gráficos são apresentados pelo Python. Exemplificando, a informação para criar a estrutura é passada do Python para o HFSS, que desenvolve o design 3D com os materiais especificados e também pode ser visualizado no Python.

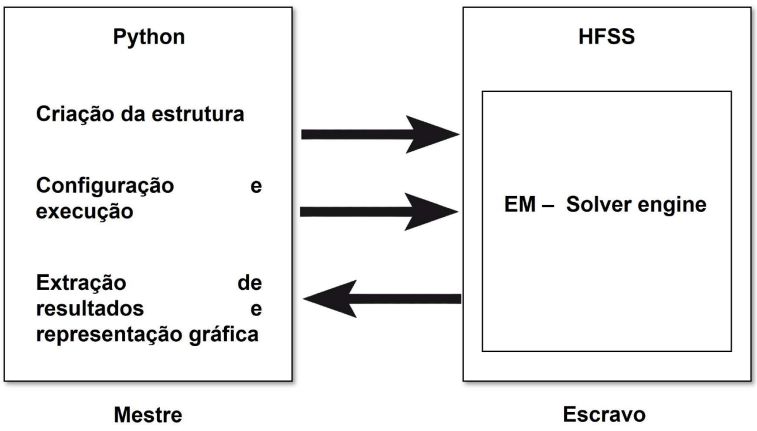
Consequentemente, no caso citado da integração do Python com HFSS, o código tem a vantagem de ser reutilizável e, com adaptação, pode ser usado para projetar outros dispositivos de maneira fácil. Entretanto, atualmente não encontra-se na documentação do site a maioria dos métodos com explicação de uso.

Figura 3 – HFSS executando arquivo Python



Fonte: Produção do próprio autor.

Figura 4 – Python executando o HFSS



Fonte: Produção do próprio autor.

1.4 OBJETIVOS

Esse trabalho final de curso tem o objetivo principal de contribuir com uma espécie de manual, contendo explicações passo a passo da integração do *software HFSS Electronics Desktop* do Ansys com a linguagem de programação Python, onde será executado todos os comandos, a fim de obter os resultados nele.

Para conseguir o objetivo principal precisam ser atingidos os seguintes objetivos particulares:

- As instalações dos pacotes requeridos e a configuração do ambiente de desenvolvimento;
- A implementação da estrutura para simular a antena *patch*, nesse caso, a 10 GHz;
- A configuração da simulação, como o *sweep*;
- A extração e análise dos dados a partir da simulação do HFSS.

1.5 ESTRUTURA DO DOCUMENTO

Este documento está organizado em sete capítulos, sendo que o primeiro é introdutório, contendo informações sobre antenas, a apresentação do problema a ser resolvido, as soluções e os objetivos.

No segundo capítulo, são descritos os conceitos básicos de antenas como: diretividade, ganho e largura de banda. Nessa etapa, também foram abordados os cálculos das dimensões da antena desenvolvida no trabalho.

No terceiro capítulo, explica-se os detalhes de configuração do ambiente para o projeto, tanto na parte de Python, como no *software* HFSS, com as bibliotecas necessárias e a versão usada.

A partir do quarto capítulo até o sexto, comenta-se sobre o código passo a passo para o desenvolvimento da antena, começando pela criação da estrutura da antena *patch* (capítulo quatro), configuração e execução da simulação (capítulo cinco). No capítulo seis é abordado a extração dos dados de simulação (impedância de entrada e coeficiente de reflexão) e as representações gráficas desses parâmetros, além do ganho.

Por último, a conclusão, que além das considerações finais, são comentadas algumas sugestões de trabalhos futuro a serem realizados sobre esse assunto.

2 CONCEITOS BÁSICOS DE ANTENAS

Neste capítulo, ressaltaremos os conceito fundamentais da teoria, antes da criação e análise da estrutura, além disso, contém os valores dos parâmetros obtido para a antena.

2.1 DIAGRAMA DE RADIAÇÃO

Pela definição encontrada no livro escrito por Balanis, diagrama de radiação é:

> Uma função matemática ou representação gráfica das propriedades de radiação da antena em função das coordenadas espaciais. Na maioria dos casos, o diagrama de radiação é determinado na região de campo distante e é representado como uma função das coordenadas direcionais. As propriedades de radiação incluem densidade de fluxo de potência, intensidade de radiação, intensidade de campo, diretividade, fase ou polarização. (BALANIS, 2009, p. 17)

Geralmente a unidade de medida utilizada é o decibéis (dB), na escala logarítmica. O diagrama de radiação pode ser de campo, que exibe a magnitude do campo elétrico, ou também de potência, se for em escala linear, exibe o quadrado da magnitude do campo elétrico, porém se a escala utilizada for dB, corresponde a magnitude do campo elétrico ou magnético em dB (BALANIS, 2009).

Com a representação gráfica podemos visualizar os lóbulos presentes, que são a região que começa e termina quando a intensidade de radiação for muito baixa, praticamente zero. Eles são subclassificados como:

Lóbulo principal: corresponde a parte de maior intensidade de radiação, significando a representação da máxima radiação. Caso a antena seja de feixe bifurcado, pode ocorrer múltiplos lóbulos principais.

Lóbulos secundário: excluindo a região do lóbulo principal, as outras são classificadas como lóbulos secundário, normalmente, são indesejáveis.

Lóbulos lateral: geralmente localiza-se adjacentemente do lóbulo principal e também são indesejáveis a sua presença.

Lóbulos posterior: são as regiões no qual cria um ângulo por volta de 180 ° com o feixe da antena.

Outra análise, é em relação se apresenta o diagrama isotrópicos, direcionais ou omnidirecionais. Nos isotrópicos, observa-se que em todas as direções exibem a mesma radiação, desse modo, essa antena não contém perdas e por esse motivo é uma antena ideal. No entanto, não temos a capacidade de replicar no mundo real.

A segunda, os direcionais, apresentam mais eficiências em algumas direção do que comparadas com outras e essa expressão é usado se a diretividade máxima é muito superior ao um dipolo de meia onda.

Por fim, os omnidirecionais são um caso especifico do diagrama direcional, pois apresenta um diagrama direcional e também um diagrama não direcional em algum plano ortogonal (BALANIS, 2009).

2.2 DIRETIVIDADE

Segundo a Comissão Eletrotécnica Internacional - IEC, o significado de diretividade é "a razão entre a intensidade de radiação em uma dada direção da antena e a intensidade de radiação média. A intensidade de radiação média é igual à potência total radiada pela antena dividida por 4 π. Se a direção não for especificada, a direção de máxima intensidade de radiação fica implícita". Portanto, a diretividade é dada por:

$$D = \frac{U}{U_0} = \frac{4\pi \cdot U}{P_{rad}}, \tag{1}$$

onde, D é a diretividade, U significa a intensidade de radiação, U_0 é a intensidade de radiação de uma fonte isotrópica e P_{rad} a potência radiada total (BALANIS, 2009).

2.3 GANHO

Um outro parâmetro considerável é o ganho, que é associado com a diretividade e eficiência da antena. Pode-se ser descrito matematicamente com a Equação 2.

$$G = 4\pi \cdot \frac{\textit{intensidade de radiação}}{\textit{potência total de entrada (aceita)}} = 4\pi \cdot \frac{U(\theta,\ \phi)}{P_{in}}\ \textit{(adimensional)}. \tag{2}$$

Usualmente a unidade do ganho é em decibéis e não adimensional, para realizar a transformação, segue a equação:

$$G(dB) = 10 \cdot log[e_{cd} \cdot D_0(em\ adimensional)]. \tag{3}$$

Pode-se relacionar a potência total radiada (P_{rad}) com a potência total de entrada (P_{in}):

$$P_{rad} = e_{cd} \cdot P_{in}, \tag{4}$$

sendo e_{cd} a eficiência de radiação da antena adimensional.

Como já mencionando anteriormente, o ganho é associado com a diretividade, assim, pode-se ser expresso pela Equação 5 e a diretividade máxima com o ganho máximo por meio da Equação 6.

$$G(\theta,\ \phi) = e_{cd} \cdot D(\theta,\ \phi) \tag{5}$$

$$G = G(\theta,\ \phi)|_{mx} = e_{cd} \cdot D(\theta,\ \phi)|_{max} = e_{cd} \cdot D_0 \tag{6}$$

Uma pertinente observação, estes ganhos não levam em conta as perdas com o descasamentos de impedância e perdas de polarização, apenas do elemento particular (BALANIS, 2009).

2.4 LARGURA DE BANDA

Outro parâmetro importante, a largura de banda, é uma faixa de frequência, que pode ser de duas categorias, banda larga ou banda estreita. Caso seja banda larga, a largura de banda, por exemplo,

de um valor X:1, sendo X a representação da frequência superior, indica que é X vezes maior que a inferior. Assim, a faixa tolerável de operação será calculada por essa razão (BALANIS, 2009).

A categoria banda estreita, apresenta uma frequência central e geralmente uma porcentagem referente a frequência de operação aceitável, ou seja, frequência central + frequência de operação, resulta na frequência superior e frequência central - frequência de operação, denomina-se, frequência inferior e portanto, as frequência entre a inferior e superior configura-se a largura de banda estreita (BALANIS, 2009).

2.5 IMPEDÂNCIA DE ENTRADA DE UMA ANTENA

A definição de impedância de entrada segundo o livro escrito por Balanis, é "a impedância apresentada pela antena em seus terminais ou a razão entre a tensão e corrente em um par de terminais, ou razão entre componentes apropriadas de campos elétricos e magnéticos em um ponto."

Para uma ocorrência de um par de terminais, sem a conexão da carga, chamando os terminais de *a* e *b*, expressa-se matematicamente com a fórmula:

$$Z_A = R_A + j \cdot X_A, \tag{7}$$

com todas as unidades de medidas em ohms. No qual Z_A é a impedância da antena nos terminais *a* - *b*, R_A é a resistência da antena nos terminais *a* - *b* e X_A é a reatância da antena nos terminais *a* - *b* (BALANIS, 2009).

2.6 MÉTODO DE ALIMENTAÇÃO

Para alimentar uma antena *patch* existe algumas possibilidades, sendo com contato ou sem contato, que chamamos de acoplamento.

- Acoplamento por abertura: apesar da sua fabricação não ser simples, comparado com a alimentação por cabo coaxial e por linha *microstrip*, possui uma menor radiação espúria. O seu design consiste em ter um patch, elemento irradiante, em baixo, dois substratos com permissividades diferentes, separados pelo plano terra, que contém uma abertura e por fim, tem a linha de transmissão que se posiciona na parte inferior do substrato que não está ligado ao *patch*;

- Acoplamento por proximidade: parecido com o anterior. Mas em vez do plano terra separar dois substratos, a linha *microstrip* faz isso e o plano terra localiza-se conectado ao substrato inferior (BEVELACQUA, 2022);

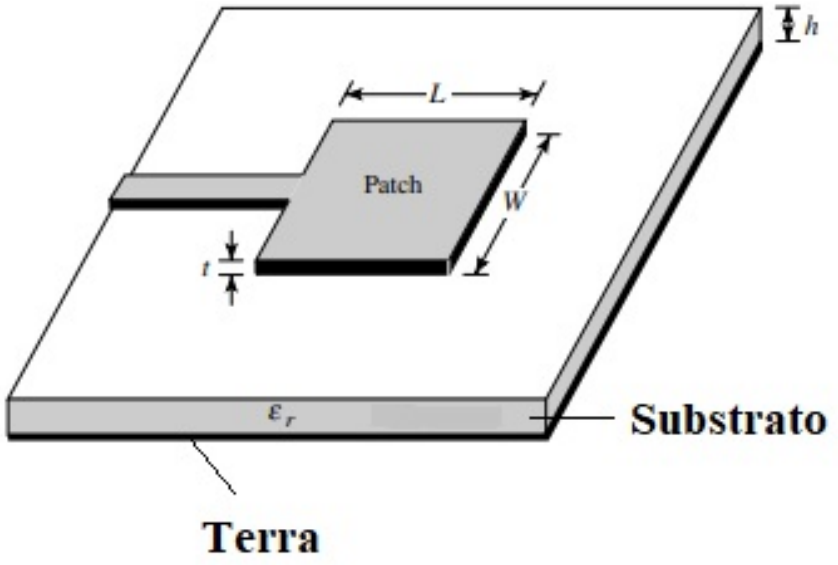
- Alimentação por cabo coaxial: é muito usada. O cabo coaxial interliga o plano terra com o elemento irradiante, sendo a parte externo do condutor conectado com o plano terra e a parte interna com o *patch*. Essa alimentação tem a vantagem de ter uma fabricação fácil e uma pequena radiação na linha. Entretanto, possui algumas desvantagem, como uma banda estreita e introduz uma indutância que necessita de atenção para o caso do aumento da altura do substrato;

- Alimentação por linha *microstrip*: o *patch* é conectado em um dos lados pela linha *microstrip*, por causa do descasamento de impedância, existe a necessidade de reverter, fazendo que ocorra o casamento de impedância, para isso é utilizado a técnica chamada *inset feed*, que simplificando, é inserir gaps entre o *patch* e a linha *microstrip* (BEVELACQUA, 2022). Os dimensionamentos dos *gaps* serão abordados na próxima subseção.

2.7 DEFINIÇÃO DOS PARÂMETROS

O desenvolvimento de uma antena *patch* envolve o dimensionamento geométrico, como podem ser observados na Figura 5.

Figura 5 – Parâmetros da antena *patch microstrip*.



Fonte: Adaptado (BALANIS, 2009)

Antes de mostrar as etapas para realizar os cálculos das dimensões, alguns valores dos parâmetros foram definidos, no substrato: a sua altura (h) e o material utilizado, que foi o duroid 5880, além da frequência central, como pode-se observar na Tabela 2.

Tabela 2 – Parâmetros inicial da antena *patch microstrip*.

Parâmetro	Valor
h	0,787 mm
t	18 μm
f_r	10 GHz
ε_r	2,2
Z_0	50 Ω

Fonte: Produção do Próprio Autor.

Por meio das Equações 8, 9, 10, 11 e da Tabela 2, é possível encontrar os valores para os demais dados necessários para a simulação.

$$W = \frac{\upsilon_0}{2 \cdot f_r}\sqrt{\frac{2}{\varepsilon_r + 1}} \tag{8}$$

Para o ε_{reff}, temos:

$$\varepsilon_{reff} = \frac{\varepsilon_r + 1}{2} + \frac{\varepsilon_r - 1}{2} \cdot \left(1 + 12 \cdot \frac{h}{W}\right)^{-1/2} \tag{9}$$

Para o ΔL, temos:

$$\frac{\Delta L}{h} = 0,412 \cdot \frac{(\varepsilon_{reff} + 0,3) \cdot \left(\frac{W}{h} + 0,264\right)}{(\varepsilon_{reff} - 0,258) \cdot \left(\frac{W}{h} + 0,8\right)} \tag{10}$$

Para o L, temos:

$$L = \frac{\upsilon_0}{2 \cdot f_r \cdot \sqrt{\varepsilon_{reff}}} - 2 \cdot \Delta L \tag{11}$$

Para encontrar o valor de W_0 da linha *microstrip* tem que satisfazer: (W_0 / h) < 2, usando-se a seguinte equação:

$$\frac{W_0}{h} = \frac{8 \cdot e^A}{e^{2 \cdot A - 2}} \tag{12}$$

Onde A é:

$$A = \frac{Z_0}{60} \cdot \sqrt{\frac{\varepsilon_r + 1}{2}} + \frac{\varepsilon_r - 1}{\varepsilon_r + 1} \cdot \left(0,23 + \frac{0,11}{\varepsilon_r}\right) \tag{13}$$

Se satisfazer, (W_0 / h) > 2, usa-se:

$$\frac{W_0}{h} = \frac{2}{\pi} \cdot \left(B - 1 - ln\,(2 \cdot B - 1) + \frac{\varepsilon_r - 1}{2 \cdot \varepsilon_r}\left(ln\,(B - 1) + 0,39 - \frac{0,61}{\varepsilon_r}\right)\right) \tag{14}$$

Sendo B:

$$B = \frac{377 \cdot \pi}{2 \cdot Z_0 \cdot \sqrt{\varepsilon_r}} \tag{15}$$

Uma outra maneira, na qual podem ser realizadas, é com o auxilio de um *software* ou site como o emtalk. Através do site, especificando os parâmetros: valor da constante dielétrica, altura do substrato, frequência e se for o caso da linha, a impedância de entrada, números já definido na Tabela 2, obtivemos as dimensões do *patch* e da linha *microstrip*, que podem ser visualizadas na Tabela 3.

Tabela 3 – Parâmetros da antena *patch* e da linha *microstrip* .

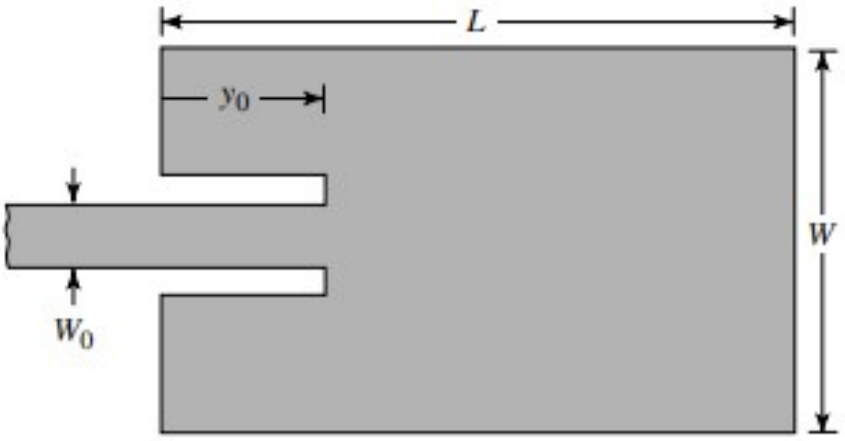
Parâmetro	Valor
W (patch)	11,858 mm
L (patch)	9,661 mm
W_0 (linha *microstrip*)	2,420 mm
L_0 (linha *microstrip*)	5,425 mm

Fonte: Produção do Próprio Autor.

O próximo passo foi fazer uma otimização no dispositivo, fazendo uso da técnica *inset feed*, acrescentou os *gaps*, como verifica-se na Figura 6, com o objetivo de obter uma resposta melhor no gráfico de S_{11}, os valores utilizados com os cálculos, Equação 16, são apresentados logo depois da figura.

$$y_0 = \frac{L_{linha\ microstrip}}{\pi} \cdot \cos^{-1}\sqrt{\frac{Z}{R_{in}}} \tag{16}$$

Figura 6 – Antena *microstrip* com gap, vista de cima.



Fonte: Adaptado (BALANIS, 2009)

$$y_0 = \frac{5425,85}{\pi} \cdot \cos^{-1} \sqrt{\frac{50}{144}}$$

$$y_0 = 1,812\ mm$$

De forma mais organizada, na Tabela 4 apresentam os resultados encontrados, junto com os dados pré-definidos anteriormente.

Tabela 4 – Todos os parâmetros da antena patch.

Parâmetro	Valor
h	0,787 mm
f_r	10 GHz
ε_r	2,2
Z_0	50 Ω
W	11,858 mm
L	9,661 mm
y_0	1,812 mm
W_0	2.420 mm
sub_x	20,512 mm
sub_y	23,717 mm
gnd	18 μm

Fonte: Produção do Próprio Autor.

3 CONFIGURAÇÃO DO ENTORNO DE CO-SIMULAÇÃO

Nesse tópico apresentam-se as configurações mais importante para a integração.

3.1 BIBLIOTECAS NECESSÁRIAS DO PYTHON

Python é uma linguagem de programação, criada por Guido Van Rossum em 1991. Seus objetivos são: ser *"open source"*, gratuita para qualquer pessoa, apresentar uma linguagem tão poderosa quanto as dos seus concorrentes, mas, ao mesmo tempo, ser intuitiva e fácil, tendo um código simples de ser compreendido e conseguindo realizar algumas tarefas com pouco tempo de desenvolvimento.

Os seus objetivos contribuíram para torna-se uma das mais populares do mundo, podendo ser aplicadas em inúmeros ramos, como em *data science*, automação e na visualização de dados. Atualmente, é utilizada por grandes companhias de tecnologia, como o Google, Netflix e Facebook (INSTITUTE, 2022).

Outro fator, é o grande número de bibliotecas existentes. São mais de 130 mil conjuntos de módulos e funções, os quais facilitam o desenvolvimento. Praticamente existe uma biblioteca específica para cada finalidade que o programador deseja (CATUNDA, 2022).

No projeto foram utilizados o Python na versão 3.9.7 e algumas bibliotecas. A versão do Python é facilmente verificada no *prompt* de comando por meio da execução da linha:

```
python -V
```

As bibliotecas necessárias para o projeto foram:

- Numpy: é o acrônimo de *"Numerical Python"* e é excelente para cálculo numérico, possuindo habilidade de executar código de uma linha com grandes cálculos. Basea-se em objetos denominados de *arrays* e são multidimensionais (JR., 2018).

  Para instalar o numpy, basta executar o código:

  ```
  pip install numpy
  ```

  Para usá-la, precisamos importar a biblioteca:

  ```
  import numpy as np
  ```

- Pandas: seu nome vem de *"Panel Data"* e essa biblioteca tem como foco as funções de limpeza, análise e tratamento dos dados, permitindo que o usuário faça a leitura, a manipulação, a adição e até mesmo a exibição dos gráficos, já que conta com a integração do Matplotlib, Plotly, Seaborn e muitos outros. As Series e os *DataFrames* são importantes conceitos para entender o funcionamento do pandas. De maneira simplificada, os *DataFrames* são tabelas com index no Python e as Series são uma única coluna com index, mostrando também o seu tamanho e o tipo da variável (ALMEIDA, 2022).

A instalação é feita através do comando:

```
pip install pandas
```

Importação:

```
import pandas as pd
```

- Matplotlib: uma biblioteca muito conhecida. É uma extensão do numpy, com foco na visualização dos dados, seja em 2D ou 3D. Tem a capacidade de criar gráficos de linhas, de dispersão, de barras, histogramas e outros tipos, existindo a possibilidade do programador estilizar vários detalhes, tal como, as cores, o título, tamanhos e a legenda (LOCAWEB, 2022).

  Instalação no Python:

  ```
  pip install matplotlib
  ```

  Importação no Python:

  ```
  import matplotlib.pyplot as plt
  ```

- Tempfile: esse módulo já vem instalado no Python e cria arquivos e diretórios temporários no caminho definido (GEEKS, 2020). Em virtude disso, após a execução e fechamento do HFSS pelo código, tem como consequência a exclusão do arquivo .aedt. Uma vantagem nesse caso específico, pois só precisamos dos dados e gráficos gerados, e no caso de alguma alteração, para adicionar ou deletar alguma caracterização da antena, sempre vai ser realizado diretamente no código. Gerando outro aspecto positivo, o espaço na memória de armazenamento. Um ponto relevante, todas as mensagens mostradas dentro do HFSS são gravadas em um arquivo .txt, contendo a data e hora da ocorrência, Figura 7.

  Como mencionado anteriormente, não precisa instalar, apenas importar:

  ```
  import tempfile
  ```

- OS: é uma biblioteca de comandos do sistema operacional. Por meio dela é possível verificar, por exemplo, se determinado caminho existe e se o nome do arquivo não. Utilizando o Tempfile, cria-se um arquivo, no qual não haverá conflito por conta do nome e nem de tentar criar em um lugar inexistente (CATUNDA, 2021).

  Para utilizar:

  ```
  import os
  ```

Figura 7 – HFSS - mensagens, logs

```
2023/03/07 18.36.08:Global:INFO    :Project Project71 has been created.
2023/03/07 18.36.08:Global:INFO    :No design is present. Inserting a new design.
2023/03/07 18.36.12:Global:INFO    :Added design 'HFSS_L9X' of type HFSS.
2023/03/07 18.36.12:Global:INFO    :Design Loaded
2023/03/07 18.36.12:Global:INFO    :Successfully loaded project materials !
2023/03/07 18.36.12:Global:INFO    :Materials Loaded
2023/03/07 18.36.15:Global:INFO    :Union of 2 objects has been executed.
2023/03/07 18.36.15:Global:INFO    :Union of 2 objects has been executed.
2023/03/07 18.36.15:Global:INFO    :Union of 2 objects has been executed.
2023/03/07 18.36.16:Global:INFO    :Union of 2 objects has been executed.
2023/03/07 18.36.16:Global:INFO    :Union of 2 objects has been executed.
2023/03/07 18.36.16:Global:INFO    :Boundary Radiation Rad__DRAB6M has been correctly created.
2023/03/07 18.36.16:Global:INFO    :Enabling Material Override
2023/03/07 18.36.16:Global:INFO    :Boundary AutoIdentify Port_Q0YJIK has been correctly created.
2023/03/07 18.36.24:Global:INFO    :Open Region correctly created.
2023/03/07 18.36.24:Global:INFO    :Linear step sweep sweep has been correctly created.
2023/03/07 18.36.25:Global:INFO    :Project AntennaPatchGap Saved correctly
2023/03/07 18.36.25:Global:INFO    :Solving design setup MySetup
2023/03/07 18.39.35:Global:INFO    :Design setup MySetup solved correctly
2023/03/07 18.39.36:Global:INFO    :Solution Data Correctly Loaded.
2023/03/07 18.39.36:Global:INFO    :Solution Data Correctly Loaded.
2023/03/07 18.39.38:Global:INFO    :Solution Data Correctly Loaded.
```

Fonte: Produção do próprio autor.

- PyAEDT: é um pacote do PyAnsys, com a finalidade de realizar a integração do *Ansys Electronics Desktop - (AEDT)* com o Python, permitindo assim, o comando das funcionalidades do *software* computacional pela linguagem de programação. Algumas das ferramentas que tem a possibilidade da integração são: o Icepack, Maxwell 2D, Maxwell 3D, layout 3D HFSS e HFSS (PYAEDT, 2023), sendo o último citado, o utilizado neste trabalho na versão 0.4.73.

  Em razão de ser um pacote do PyAnsys, deve-se instalar, através do comando:

  ```
  pip install pyaedt
  ```

  Antes de utilizar, nesse caso, deve-se executar a linha de código:

  ```
  from pyaedt import generate_unique_name
  from pyaedt import Desktop, Hfss, constants
  ```

3.2 CONFIGURAÇÃO DO HFSS

As bibliotecas essenciais mencionadas na sessão anterior não são os únicos detalhes para se ter sucesso na integração, precisa verificar qual versão o usuário dispõe. Pois não são todas as versões com a capacidade de realizar o objetivo pretendido. Devido a esse fator foi utilizado a versão 2022 R1.

O código a seguir são para criar um arquivo temporário para o projeto, até a quarta linha, e as demais para abrir o *software*. Se fosse apenas "non_graphical=True", não poderia vermos a janela do HFSS no computador, porém o Python executaria perfeitamente os comandos e exibiria as respostas.

```
tmpfold = tempfile.gettempdir()
temp_folder = os.path.join(
    tmpfold,
    generate_unique_name("Antenna")
```

```
)
if not os.path.exists(temp_folder):
    os.mkdir(temp_folder)

non_graphical = os.getenv(
    "PYAEDT_NON_GRAPHICAL",
    "False"
).lower() in ("true", "1", "t")

d = Desktop(
    "2022.1",
    non_graphical=non_graphical,
    new_desktop_session=True
)
```

Um detalhe importante, por meio do código do Python irá abrir e fechar do HFSS, não tendo a possibilidade de encerrar o HFSS através do *software*, botão *X*. A seguir o código utilizado para o encerramento. Executado apenas depois da análise dos resultados dos gráficos.

```
d.release_desktop()
```

4 CRIAÇÃO DA ESTRUTURA

Depois dos *imports* e de abrir o *software* HFSS, o próximo passo é desenvolver a estrutura e declarar as variáveis com os seus respectivos valores:

```
L= 9661.15
W= 11858.54
L0= 5425.85
W0= 2420.98
sub_x= 20512.85
sub_y= 23717.08
sub_z= 787
gnd= 18
L_gap = 1812
```

Logo em seguida, foi escolhido o tipo de solução do projeto, nesse caso foi terminal:

```
hfss = Hfss(solution_type="Terminal")
```

Mas em caso do projeto requerer ser modal, é só alterar para:

```
hfss = Hfss(solution_type="Modal")
```

Além de declarar as variáveis, não podemos esquecer de indicar a unidade que irá ser trabalhada:

```
hfss.modeler.model_units = "um"
```

A unidade “um” é o micrômetros, mas pode-se trabalhar com milímetros, “mm” ou outra unidade que desejar.

Para criar as estruturas, primeiro simplificamos o código, chamando de “p” a classe referente a “primitives”. O “create_box” cria uma caixa em 3D, o primeiro colchetes corresponde a posição de onde vai se iniciar. Já o segundo colchetes representa o tamanho das dimensões, em ambos, no plano de coordenadas, X, Y e Z, respectivamente. O “matname” significa o nome do material a ser utilizado e o “name” é o nome da caixa.

Para a porta, por ser um retângulo, duas dimensões, empregou o “create_rectangle”, posteriormente, especificou-se o plano YZ e as variáveis em parênteses respeitou-se a mesma lógica da caixa em 3D. Uma observação importante, as linhas que contém # são comentário, afim de somente facilitar a leitura.

```
p = hfss.modeler.primitives

# Microstrip:
microstrip = p.create_box(
    [L/2, -W0/2, sub_z],
```

```
    [L0, W0, gnd],
    matname='copper',
    name='microstrip'
)

# Porta:
p.create_rectangle(
    constants.PLANE.YZ,
    (sub_x/2, -W0/2, sub_z),
    (W0, -sub_z),
    name='port1'
)

# Substrato:
p.create_box(
    (-sub_x/2, -sub_y/2, 0),
    [sub_x, sub_y, sub_z],
    matname='Rogers RT/duroid 5880 (tm)',
    name='substrato'
)

# Terra:
p.create_box(
    [-sub_x/2, -sub_y/2, -gnd],
    [sub_x, sub_y, gnd],
    matname='copper',
    name='terra'
)

# Caixa de ar:
box = p.create_box(
    [(-sub_x/2),(-sub_y/2), (-gnd)],
    [(sub_x), (sub_y), sub_z*10],
    name='airbox',
    matname='air'
)
```

Foi utilizado o "unite" para unir a linha *microstrip* com o *patch*, que foi configurado por meio de um *array*. De acordo com o objetivo do projeto a ser desenvolvido, é necessário criar um *array* com números aleatório para o dispositivo.

```
N = 5
```

```
Warray = [11858.54, 11858.54, 11858.54, 11858.54, 11858.54]
for Waux in range(N):
    W = Warray[Waux]
    patch = p.create_box((-L/2+Waux*L/N, -W/2, sub_z),
    (L/N, W, gnd), matname='copper', name='patch')
    p.unite([microstrip, patch])
```

A inserção dos *gaps* é uma otimização e foi necessário utilizar o "subtract" para subtrair a região em comum do *patch* e das caixas de ar que nomeamos de gap_1 e gap_2. Na última etapa, criamos a caixa de radiação feita de ar.

```
# Gap:
gap_1 = p.create_box(
    (L/2, W0/2, sub_z),
    (-L_gap, W*0.07, gnd),
    matname='air',
    name='gap'
)
gap_2 = p.create_box(
    (L/2, -W0/2 , sub_z),
    (-L_gap, -W*0.07, gnd),
    matname='air',
    name='gap'
)

p.subtract(microstrip, gap_1)
p.subtract(microstrip, gap_2 )

hfss.assign_radiation_boundary_to_objects('airbox')
hfss.change_material_override()
```

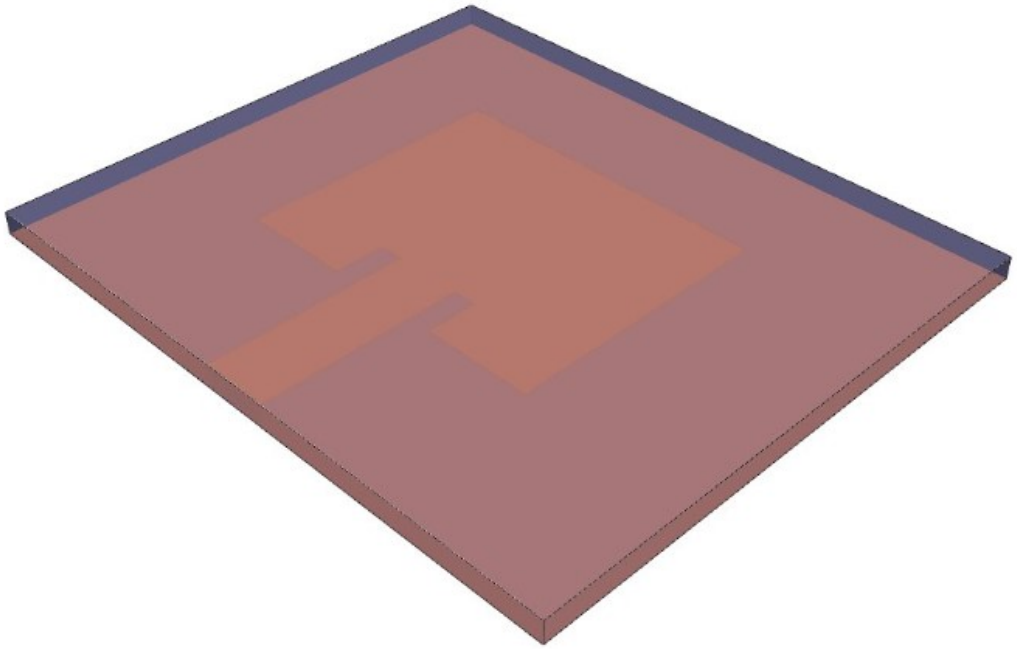
Último passo para gerar o design da antena é colocar a excitação na porta, nesse trabalho foi feito com o tipo *lumped port*. O "sheet_name" do código é referente ao nome que foi dado no retângulo e o "axisdir" é relativo em qual plano vai ser gerado.

```
#Porta:
hfss.create_lumped_port_to_sheet(
    sheet_name='port1',
    axisdir=constants.AXIS.Z
)
```

A visualização em perspectiva isométrica da estrutura da antena *patch* no Python, Figura 8, pode-se ser realizada através de código:

```
my_plot = hfss.plot(show=False, plot_air_objects=False)
my_plot.show_axes = False
my_plot.show_grid = False
my_plot.isometric_view = True
my_plot.plot(
    os.path.join(hfss.working_directory, "AntenaPatchGap.jpg")
)
```

Figura 8 – Antena *patch* exibido no Python.



Fonte: Produção do próprio autor.

5 CONFIGURAÇÃO E EXECUÇÃO DA SIMULAÇÃO

A próxima etapa é a criação do *setup*, na qual nomeamos de "MySetup":

```
setup = hfss.create_setup("MySetup")
```

Por critério de ter uma melhor organização no código, optou-se em criar funções. A primeira desenvolvida foi para configurar a opção de uma frequência única, com quatro parâmetros de entrada, a frequência central, número máximo de passos, máximo espaçamento dos passos e o nome, os dois últimos citados tem um valor padrão de 0,01 e “MySetup”, respectivamente.

```
def SingleFrequency (
    freq,
    maxPasses,
    maxDeltaS = 0.01,
    name="MySetup"
):
    hfss.create_open_region(Frequency=freq)
    setup.props["Frequency"] = freq
    setup.props["MaximumPasses"] = maxPasses
    setup.props["MaxDeltaS"] = maxDeltaS
    setup.props["UseIterativeSolver"] = True
    setup.props["SaveRadFieldsOnly"] = True
```

Em relação ao *sweep*, utilizou-se o modelo de passos linear, sendo necessários a frequência de inicio, final, o espaçamento (pré-configurado para 0,001), especificar a unidade (padrão: GHz) e as opções de salvar os campos (Elétrico e Magnético) e de radiação, ambos setados como verdadeiro. Assim configuraremos o *sweep* como: “sweep_type ="Interpolating”.

```
def LinearStepSweep(
    fStart,
    fStop,
    stepSize=0.001,
    units="GHz",
    saveFields=True,
    saveRadFields=True
):
    hfss.create_linear_step_sweep(
        setupname=setup.name,
        unit= units,
        freqstart = fStart,
        freqstop = fStop,
```

```
        step_size = stepSize,
        sweepname ="sweep",
        sweep_type ="Interpolating",
        save_fields = saveFields,
        save_rad_fields = saveRadFields
    )
```

Por fim, escreveu-se uma função com o objetivo de salvar o projeto e realizar a sua análise.

```
def SaveAnalyseSetup(
    nameFolder = "AntennaPatch.aedt",
    setup = "MySetup"
):
    hfss.save_project(os.path.join(temp_folder, nameFolder))
    hfss.analyze_setup(setup)
```

Chamando as funções criadas anteriormente, começando pela *SingleFrequency*, com a frequência central de 10 GHz e uma quantidade limite de 15 passos com o espaçamento de 0,01. Logo em seguida, a *LinearStepSweep*, configurada para realizar a análise entre 9 GHZ e 11 GHz.

```
# Setup:
SingleFrequency("10GHz", 15, 0.01)
# Sweep:
LinearStepSweep(9.0, 11.0)
# Save e executa:
SaveAnalyseSetup("AntennaPatchGap.aedt")
```

6 EXTRAÇÃO DOS DADOS DE SIMULAÇÃO E REPRESENTAÇÃO GRÁFICA

6.1 CASO I: CAIXA DE AR DIFERENTE DE UM QUARTO DE ONDA

Após a análise podemos extrair os dados, nesse caso, do Z_{real} e S_{11}. A extração desses dados seguem a mesma lógica. Usamos o *hfss.post.get_solution_data* e especificando a categoria, depois acessamos os dados real, por meio do método *_solution_data_real()*, retornando os valores da frequência junto com Z_{real} ou S_{11}. Como os dados não estão separados, criamos duas listas. A primeira para adicionar a frequência, através do *lista.append(valor)* e a segunda, acrescenta a categoria escolhida. A próxima etapa, é inserir as listas dentro de um *DataFrame* vazio e exibi-las. Na Figura 9, pode-se visualizar alguns dos resultados do Z_{real} e S_{11} lado a lado. O método *.to_csv* exporta as informações em um arquivo .csv.

Código referente ao Z_real:

```
solutions = hfss.post.get_solution_data(
    expressions=hfss.get_traces_for_plot(category="Z"),
)

Data = solutions._solution_data_real()
index = 0
real1  = Data[list(Data.keys())[index]]
lista = []
lista2 = []
dataVazioZGap = {}
dfVazioZGap = pd.DataFrame(dataVazioZGap)
for i in real1:
    a = i[0]
    lista.append(a)
    c=real1[i]
    lista2.append(c)
dfVazioZGap.insert(0, "Frequency [GHz]", lista, True)
dfVazioZGap.insert(1, "Re $Z_{in}$ (ohm)", lista2, True)
print(dfVazioZGap)

dfVazioZGap.to_csv('Z_Real_Gap.csv')
```

Extração do S_{11}:

```
solutions = hfss.post.get_solution_data(
    expressions=hfss.get_traces_for_plot(category="dB(S"),
)
```

```
solutions.enable_pandas_output = True

S_Real = solutions._solution_data_real()
index = 0
S_real1  = S_Real[list(S_Real.keys())[index]]
listaReal = []
listaReal2 = []
dataVazioGap = {}
dfRealGap = pd.DataFrame(dataVazioGap)
for i in S_real1:
    a = i[0]
    listaReal.append(a)
    c=S_real1[i]
    listaReal2.append(c)

dfRealGap.insert(0, "Frequency [GHz]", listaReal, True)
dfRealGap.insert(1, "|$S_{11}$| [dB]", listaReal2, True)
print (dfRealGap)

dfRealGap.to_csv('S11_Gap.csv')
```

Figura 9 – Resultados exibidos no Python.

	Frequency [GHz]	Re Z_{in} (Ohm)
0	9.000	5.000806
1	9.001	5.002837
2	9.002	5.004876
3	9.003	5.006921
4	9.004	5.008972
...	...	...
1996	10.996	20.932795
1997	10.997	20.858657
1998	10.998	20.784933
1999	10.999	20.711619
2000	11.000	20.638712

[2001 rows x 2 columns]

	Frequency [GHz]	|S_{11}| [dB]
0	9.000	-1.145754
1	9.001	-1.147169
2	9.002	-1.148587
3	9.003	-1.150007
4	9.004	-1.151430
...	...	...
1996	10.996	-2.179920
1997	10.997	-2.177175
1998	10.998	-2.174437
1999	10.999	-2.171705
2000	11.000	-2.168981

[2001 rows x 2 columns]

Fonte: Produção do próprio autor.

Para conseguir a informação do valor máximo de coeficiente de reflexão executou o código:

```
indexValueMin = dfRealGap['|$S_{11}$| [dB]'].idxmin()
print(dfRealGap.loc[indexValueMin])
```

Antes de gerar os gráficos 2D, criou-se a função Grafico, como pode-se visualizar abaixo, que contém como variáveis de entradas: os dados, o título do gráfico, nome do eixo X e do Y, e a cor padrão vermelho.

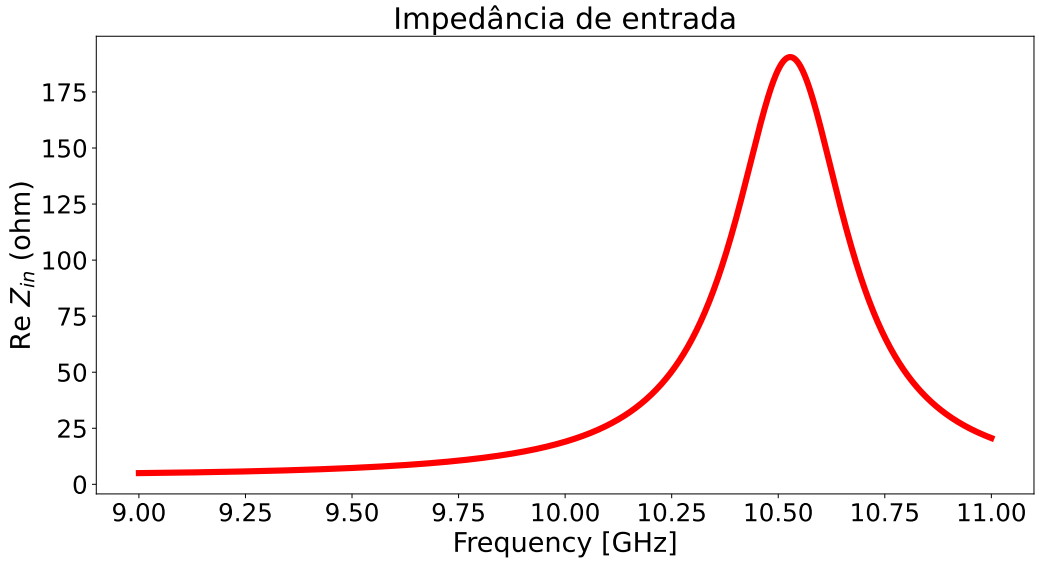
```
def Grafico(
```

```
    df,
    title,
    nameX='Frequency [GHz]',
    nameY='Result',
    colorGraf='red'
):
    valorEixoY = df[nameY].to_numpy()
    valorEixoX = df[nameX].to_numpy()
    fig, ax = plt.subplots()
    ax.plot(valorEixoX, valorEixoY, color='red', linewidth = 5)
    ax.set_xlabel(nameX, fontsize = 22)
    ax.set_ylabel(nameY, fontsize = 22)
    plt.tick_params(axis='both', which='major', labelsize=20)
    plt.title(title, fontsize = 24)
    plt.show()
```

Chamando as funções para gerar os gráficos, com os seus respectivos resultados, Figura 10 e Figura 11:

```
Grafico(
    dfVazioZGap,
    "Impedância de entrada",
    'Frequency [GHz]',
    'Re $Z_{in}$ (ohm)'
)
```

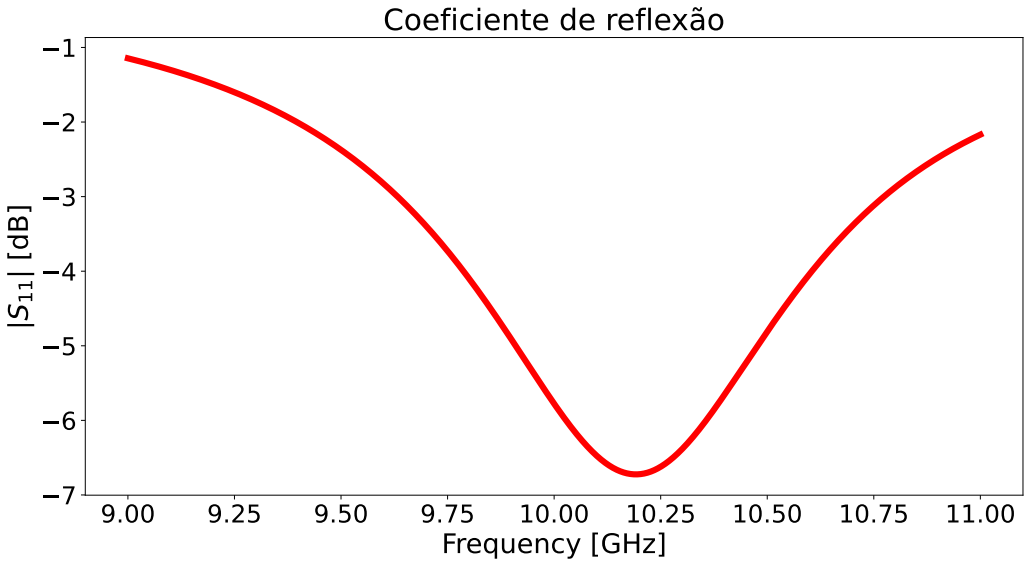
Figura 10 – Caixa de ar diferente de um quarto de onda (Caso I): Z_{real}.



Fonte: Produção do próprio autor.

```
Grafico(
    dfRealGap,
    "Coeficiente de reflexão",
    'Frequency [GHz]',
    '|$S_{11}$| [dB]'
)
```

Figura 11 – Caixa de ar diferente de um quarto de onda (Caso I): S_{11}.



Fonte: Produção do próprio autor.

Na Figura 10, representa o gráfico do valor da impedância de entrada versus a frequência. Verifica-se que em 50 Ω, valor da impedância de entrada nesse exemplo, a frequência (localizado no eixo x) é de aproximadamente 10,22 GHz. Esse resultado é muito próximo do esperado de 10 GHz.

Na Figura 11, visualiza-se o gráfico do coeficiente de reflexão. Observa-se que no eixo x corresponde ao intervalo da frequência, entre 9 GHz a 11 GHz. Podemos analisar, a frequência central, 10 GHz, fica muito perto do vale, ponto que refere-se ao valor máximo do coeficiente de reflexão. Esse ponto localiza-se na frequência de 10,192 GHz com um valor de -6,72415 dB. Outro ponto importante, nas extremidades do gráfico são onde tem os menores valores do eixo y. Resultado que era esperado, por causa de ser as regiões mais distantes da frequência central.

O código consecutivo, utiliza-se o método *hfss.post.reports_by_category.far_field* e corresponde ao ganho total mostrado em 2D. A resposta obtida é mostrada na Figura 12, que representa um corte com o plano *Phi* constante e o plano *Theta* variando. Analisando o corte e comparando com o *plot* do HFSS, percebe-se que é referente ao plano XZ, isso significa que pertence ao plano *E*, campo elétrico.

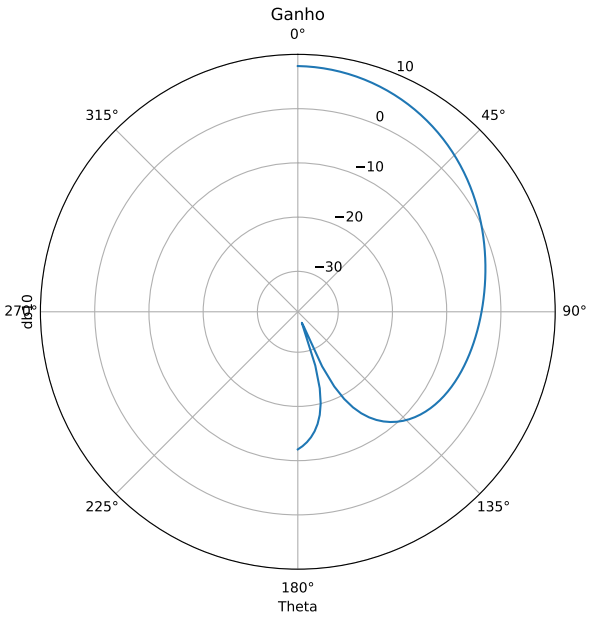
```
new_report = hfss.post.reports_by_category.far_field(
    "GainTotal",
    hfss.nominal_adaptive,
    "3D"
```

```
)
variations = hfss.available_variations.nominal_w_values_dict
variations["Theta"] = ["All"]
variations["Phi"] = ["All"]
variations["Freq"] = ["10GHz"]

new_report.primary_sweep = "Theta"
new_report.secondary_sweep ="Phi"
new_report.far_field_sphere = "3D"
new_report.variations = variations
solutions = new_report.get_solution_data()

solutions.plot(
    math_formula="db10",
    is_polar=True,
    show_legend = False,
    title='Ganho'
)
```

Figura 12 – Caixa de ar diferente de um quarto de onda (Caso I): Ganho em 2D.

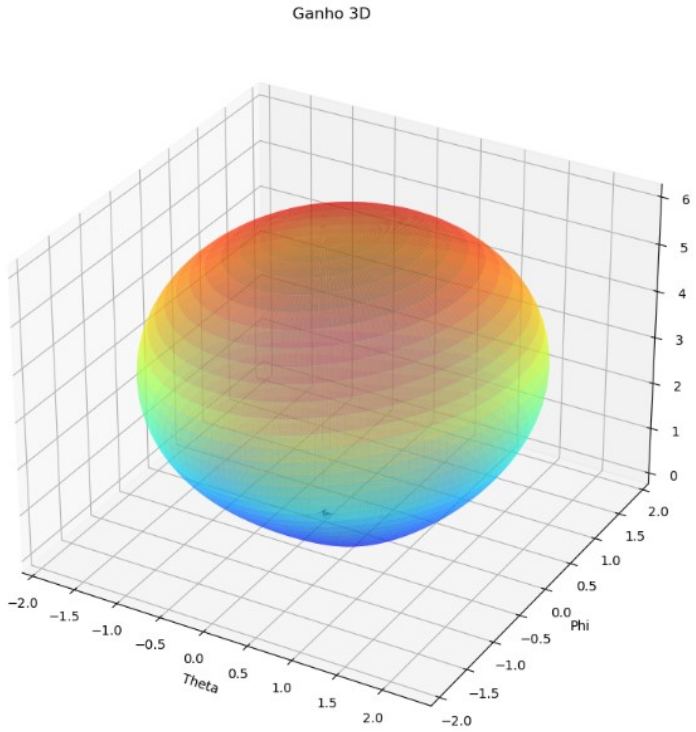


Fonte: Produção do próprio autor.

O ganho em 3D, Figura 13:

```
solutions.plot_3d(title='Ganho 3D')
```

Figura 13 – Caixa de ar diferente de um quarto de onda (Caso I): Ganho em 3D.



Fonte: Produção do próprio autor.

Para conseguir os valores do ganho, pode-se executar o código:

```
solutions.data_db10()
```

No caso de requerer apenas os valores máximo e mínimo, respectivamente:

```
max(solutions.data_db10())
min(solutions.data_db10())
```

Os resultados foram de 7,8598 dB para o máximo e o mínimo foi de -35,2957 dB. Comparando com a Figura 12, percebe-se que o máximo encontra-se em 0 ° e o mínimo próximo de 180 °.

6.2 CASO II: CAIXA DE AR COM UM QUARTO DE ONDA

A simulação foi realizada para o caso de a dimensão da caixa de ar ser um quarto do comprimento da onda. Primeiramente, calculou-se o valor por meio de:

$$novaCaixaAr = \frac{\lambda}{4} \tag{17}$$

$$novaCaixaAr = \frac{299.792.458}{10 \cdot 10^9 \cdot 4}$$

$$novaCaixaAr = 7494,81145\ \mu m$$

A próxima etapa foi copiar o arquivo Python e alterar o código, na parte da Criação da Estrutura, Capítulo 4, no começo, antes era:

```
# Caixa de ar:
box = p.create_box(
    [(-sub_x/2), (-sub_y/2), (-gnd)],
    [(sub_x), (sub_y), sub_z*10],
    name='airbox',
    matname='air'
)
```

Foi alterado para:

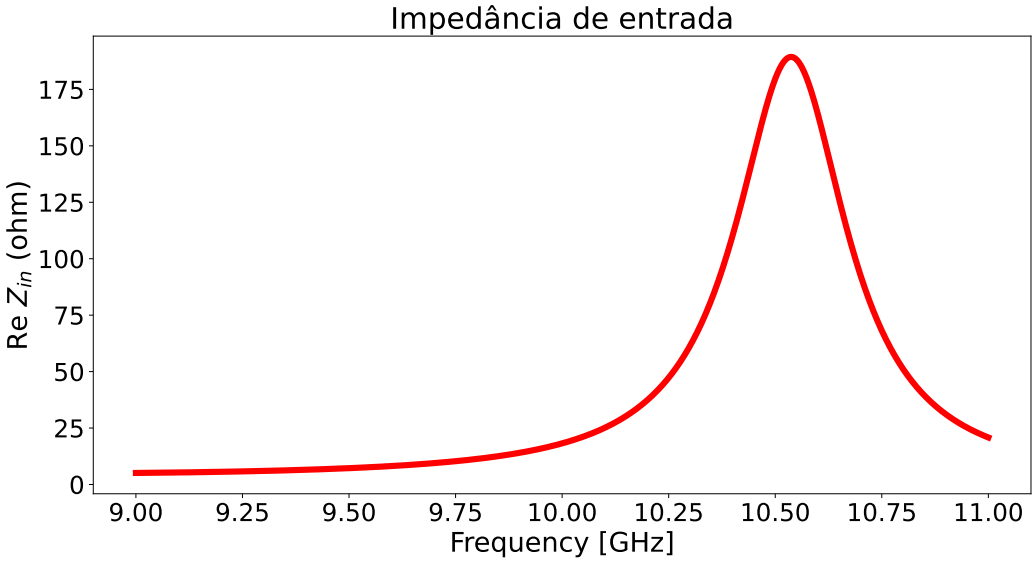
```
# Caixa de ar:
novaD = 7494.81145
box = p.create_box(
    [
        (-(sub_x + (2*novaD))/2),
        (-(sub_y + (2*novaD))/2),
        -((sub_z+gnd+gnd) + (2*novaD))
    ],
    [
        (sub_x + (2*novaD)),
        (sub_y + (2*novaD)),
        ((sub_z+gnd+gnd)+(2*novaD))
    ],
    name='airbox',
    matname='air'
)
```

Não foi necessário realizar mais ajustes, apenas executar a simulação para conferir os resultados dos gráficos.

Analisando, primeiramente, as Figura 14 e Figura 15, observou-se similaridade com o caso I. A Figura 14 apresenta 50 Ω na frequência de 10,26 GHz e a Figura 15 ocorre o maior coeficiente de reflexão em -6,7959 dB com 10,205 GHz de frequência.

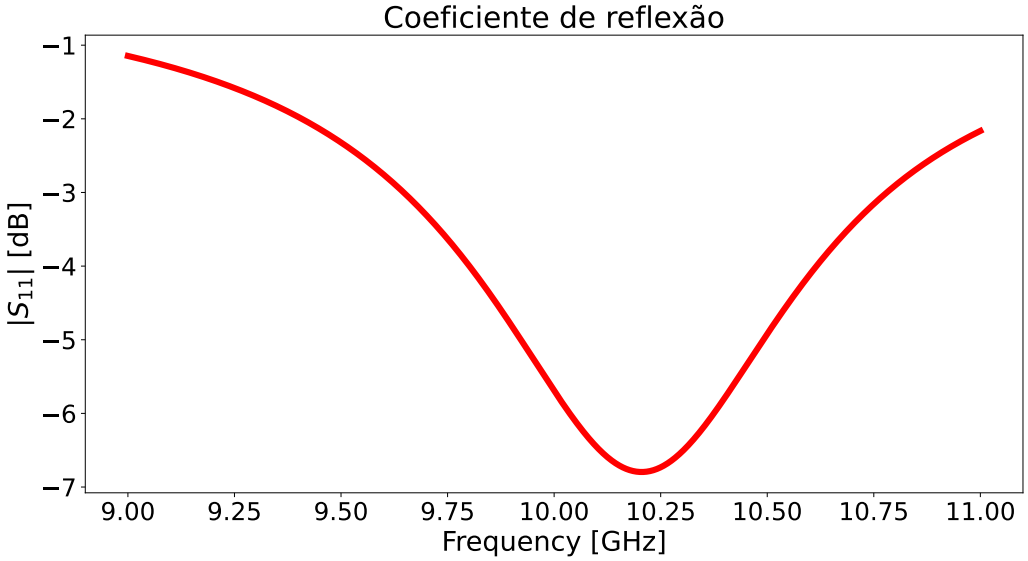
A Figura 16 e Figura 17 visualmente é idêntica ao caso anterior, sendo o ganho máximo em 8,11 dB e -30,9285 dB o mínimo.

Figura 14 – Caixa de ar com um quarto de onda (Caso II): Z_{real}.



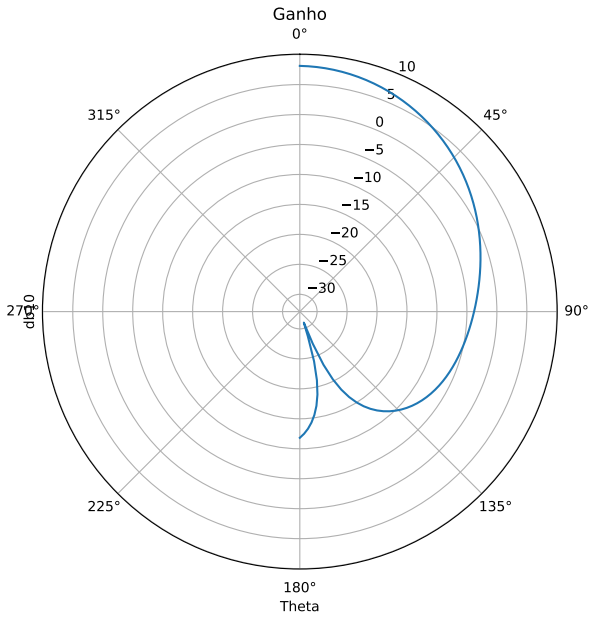
Fonte: Produção do próprio autor.

Figura 15 – Caixa de ar com um quarto de onda (Caso II): S_{11}.



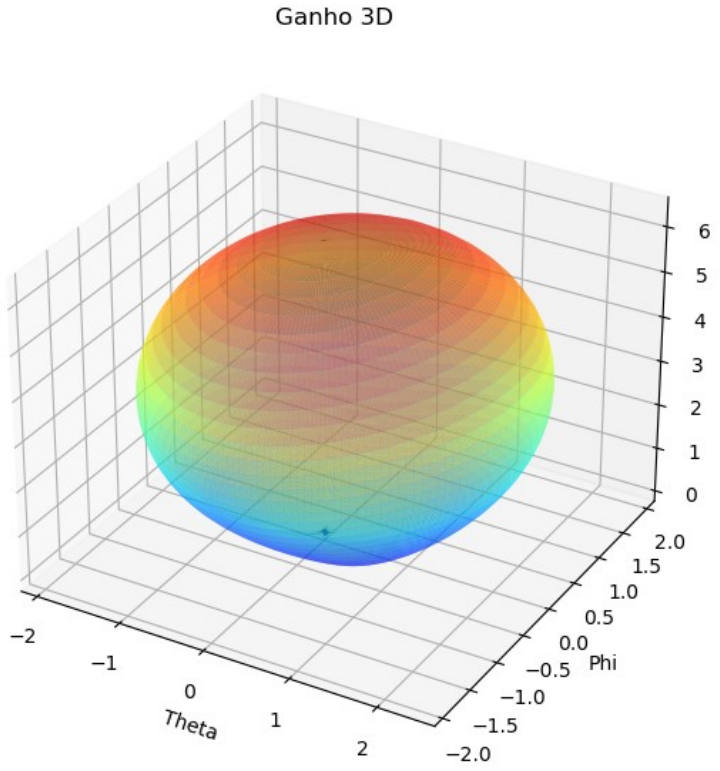
Fonte: Produção do próprio autor.

Figura 16 – Caixa de ar com um quarto de onda (Caso II): Ganho em 2D.



Fonte: Produção do próprio autor.

Figura 17 – Caixa de ar com um quarto de onda (Caso II): Ganho em 3D.



Fonte: Produção do próprio autor.

7 CONCLUSÃO

Nesse trabalho final, teve como objetivo principal contribuir com a integração do *software* HFSS com o Python, realizando explicações passo a passo, deste a sua criação, até a extração dos gráficos no Python, em razão da pouca quantidade sobre esse assunto, tanto da documentação oficial da biblioteca como em artigos escritos por pesquisadores e estudantes. Em particular, ocorreu o estudo, a implementação e análise dos resultados da antena *patch microstrip* com operação em 10 GHz, utilizando o material Duroid 5880 como subtrato, à uma altura de 0,787 mm e uma impedância de entrada de 50 Ω.

Observamos um resultado tanto do Z_{real} e S_{11}, próximo da impedância de entrada e da frequência central, respectivamente. Apesar de apresentar um valor baixo no S_{11}, é possível, visualizar a forma típica de um gráfico do coeficiente de reflexão, o ponto máximo, próximo da frequência central e a medida que for se distanciando, se aproxima de 0 dB. Em relação ao ganho, foi desenvolvido dois gráficos, o primeiro 2D e o segundo em 3D. Porém para a comparação do valor do ganho máximo e do mínimo com os resultados das funções (max e min), utilizou-se o gráfico em corte, 2D. Nos resultados das simulações dos casos I e II, verifica-se que existem uma pequena diferença nos valores de Z_{real} e S_{11}. Porém, analisando o ganho, a simulação com a caixa de ar de dimensão de um quarto de onda apresenta um valor de -30,9285 dB, enquanto para a primeira simulação obteve -35,2957 dB.

Apesar do projeto final não apresentar uma antena com boa resposta no mundo real, percebe-se que a integração foi realizada e os objetivos alcançados. Como o *software* é completo de diversas funcionalidades, para trabalhos futuros que podem ser desenvolvidos, podemos citar a continuação da integração, para a otimização e o uso de ferramentas não estudada nesse trabalho. Além disso, é possível utilizar para desenvolver códigos com *machine learning* de forma a automatizar a criação dos dispositivos, antenas, filtros, etc. Outro exemplo, a alteração automática das dimensões da antena com o intuito de buscar um maior ganho.

REFERÊNCIAS

ALMEIDA, M. **Pandas: o que é, para que serve e como instalar**. 2022. Disponível em: <https://www.alura.com.br/artigos/pandas-o-que-e-para-que-serve-como-instalar>.

AMITEC. **Microstrip Circular Patch**. 2018. Disponível em: <https://amitec.co/microstrip-circular-patch/>.

BALANIS, C. A. **Teoria de Antenas: Análise e Síntese**. Rio de Janeiro, RJ - Brasil: LTC, 2009.

BEVELACQUA, P. **Microstrip Antennas - Feeding Methods.** 2022. Disponível em: <www.antenna-theory.com/antennas/patches/patch3.php>.

BHUNIA, D. S. Microstrip patch antenna's limitation and some remedies. **InternatIonal Journal of electronics communication technology - IJECT**, w w w .i j e c t. o r g, v. 4, p. 30–39, 2013.

CATUNDA, H. **BIBLIOTECA OS NO PYTHON – COMO SAIR DO ZERO**. 2021. Disponível em: <https://www.hashtagtreinamentos.com/biblioteca-os-no-python>.

CATUNDA, H. **BIBLIOTECAS DO PYTHON: CONHEÇA AS MELHORES POR FINALIDADE!** 2022. Disponível em: <https://www.hashtagtreinamentos.com/bibliotecas-mais-importantes-do-python#:~:text=O%20que%20%C3%A9%20uma%20biblioteca,dos%20desenvolvedores%2C%20com%20diversas%20finalidades.>

GEEKS, G. for. **Python tempfile module**. 2020. Disponível em: <https://www.geeksforgeeks.org/python-tempfile-module/>.

INC, S. P. D. **2.45GHz Patch Antenna**. 2021. Disponível em: <https://saturnpcb.com/2-45ghz-patch-antenna/>.

INSTITUTE, P. **Python – the language of today and tomorrow**. 2022. Disponível em: <https://pythoninstitute.org/about-python#:~:text=Python%20was%20created%20by%20Guido,called%20Monty%20Python's%20Flying%20Circus.>

JR., L. S. **Entendendo a biblioteca NumPy**. 2018. Disponível em: <https://medium.com/ensina-ai/entendendo-a-biblioteca-numpy-4858fde63355>.

LOCAWEB, E. **Python Matplotlib: conheça a biblioteca de visualização de dados**. 2022. Disponível em: <https://blog.locaweb.com.br/temas/codigo-aberto/python-matplotlib-conheca-a-biblioteca-de-visualizacao-de-dados/>.

PATEL, B. D.; NARANG, T.; JAIN, S. Microstrip patch antenna - a historical perspective of the development. **Conference on Advances in Communication and Control Systems 2013**, Atlantis Press, p. 445–449, 2013.

POZAR, D. M. **Microwave Engineering**. Hoboken, NJ - Estados Unidos: John Wiley Sons, Inc., 2011.

PYAEDT. **PyAEDT documentation 0.6.48**. 2023. Disponível em: <https://aedt.docs.pyansys.com/release/0.6/>.

VANDENBOSCH, G. A. E.; VASYLCHENKO, A. **A Practical Guide to 3D Electromagnetic Software Tools**. 2010. Disponível em: <https://www.intechopen.com/chapters/14726>.