Departamento de Electrónica, Sistemas e Informática
Maestría en Diseño Electrónico

Antenna Design Optimization Through Input Space Mapping

TESIS que para obtener el **GRADO** de
MAESTRO EN DISEÑO ELECTRÓNICO

Presenta: **EDUARDO MICHEL CAMACHO**

Director **ZABDIEL BRITO BRITO**

Tlaquepaque, Jalisco a 1ro de Mayo de 2020

A Olga.

Abstract

We propose a generalized algorithm to optimize an antenna through Broyden based input space mapping. We elaborate on the details of creating an effective coarse model from a 2D coarse mesh method of moments simulation on Matlab™ Antenna Toolbox. The fine model consists of a 3D finite element simulation from Ansys™ EM2019 HFSS Antenna Toolbox. The coarse model is optimized using a minimax formulation to obtain a target response. The Broyden based input space mapping of the models is implemented through a generalized Matlab to HFSS driver. We design 2 antennas; a dipole bowtie and a microstrip patch using the antenna optimization algorithm. We elaborate on the adjustments needed for the implementation of the algorithm to the specific examples.

Contents

Introduction

The main contribution of this thesis dissertation is an antenna optimization algorithm through Broyden based input space mapping. We exploit the space mapping technique to be able to design antennas with a few simulations of a fine, very precise yet computationally expensive, model and many iterations of a coarse model , a less precise but computationally inexpensive simulation.

The fine model is implemented as a 3D finite element analysis simulation using *Ansys™ EM2019 HFSS Antenna toolbox (HFSS for short)* [1]. The coarse model is implemented as a coarse mesh 2D method of moments (MoM) simulation from Matlab™ Antenna toolbox [2]. We implement the algorithm in Matlab with a Matlab-Python™ driver for HFSS.

The algorithm is then applied to the design of 2 different antennas: A dipole bowtie and a microstrip patch. We elaborate on the challenges to implement the algorithm for each case and provide the Matlab and Python™ code necessary to reproduce our results.

We follow the literature on circuit optimization through space mapping that started with Bandler's seminal work on the subject , surveyed by himself in [3]. The literature using this technique specifically for antennas is relatively young and started with Bandler's alumni Zhu et al in [4], this is the first paper that exploits space mapping to optimize an antenna. The literature has since grown with several approaches to antenna optimization through space mapping, most of them by Bandler's alumni in [5]- [9]. The literature on the subject is sparse and there are many opportunities for contributions.

This work focusses on implementation of antenna simulations, optimization methods and the space mapping technique, rather than antenna theory.

Chapter 1 presents the challenges of designing antennas. Elaborates on the inner workings of the simulation techniques using for the fine and coarse models, 3D finite element analysis and coarse method of MoM. Introduces a generalized Matlab-Python™ to HFSS driver and introduces the reader to the 2 antennas designed using the algorithm.

Chapter 2 elaborates on the optimization techniques: the minimax formulation for antenna engineering specifications. Presents the Nelder-Mead algorithm that is used multiple times during the antenna design. It introduces the reader to the literature on the Broyden based input space mapping technique.

Chapter 3 presents the antenna frequency response optimization through input space mapping in general. The algorithm is analyzed in detail, describing the challenges and providing suggestions on how to tackle them. These suggestions are generalizations of the authors experience applying the algorithm to specific antenna designs.

Chapter 4 presents the implementation of the algorithm to design a bowtie dipole antenna to engineering specifications. It presents the mathematical formulations of every step of the algorithm and the results of the fine and coarse simulations. The formulation of an effective coarse model takes center stage.

Chapter 5 presents the implementation of the algorithm to design a microstrip patch antenna to engineering specifications. It presents the mathematical formulations of every step of the algorithm and the results of the fine and coarse simulations. The formulation of an effective coarse model takes center stage.

Finally, we conclude highlighting the original insights and contributions of this work and present suggestions to extend this line of research.

The author wishes to thank Dr. Zabdiel Brito for his patience and guidance through the development of this work and years of collaboration on this intellectual endeavor.

The authors original contributions presented in this thesis dissertation follow:

- Formulation and development of a generalized algorithm for antenna optimization through Broyden based input space mapping.
- A generalized Matlab™-Python™ to HFFS™ driver. This driver can be reused for any HFFS™ simulation and is not specific to antenna simulations.
- Formulation and development of a dipole bowtie antenna design using the Broyden based input space mapping algorithm.
- Formulation and development of a microstrip patch antenna design using the Broyden based input space mapping algorithm.
- Implementation in Matlab™, Python™ and HFSS™ of the antenna optimization algorithm that includes:
    - Matlab™ code for MoM coarse model formulation and implementation for bowtie dipole and microstrip patch antennas.
    - Matlab™ code for formulation and implementation of the minimax formulation and optimization of the coarse model for bowtie dipole and microstrip patch antennas.
    - Matlab™ code for the formulation and implementation of the Broyden based input space mapping algorithm.
    - Matlab™ and Python™ code for an HFSS™ driver for bowtie dipole and microstrip patch antennas.
    - HFSS™ Antenna toolbox implementation for bowtie dipole and microstrip patch antennas.

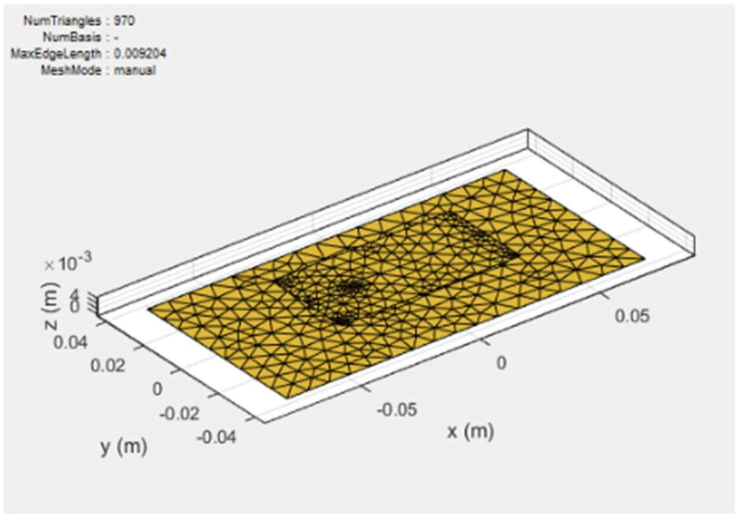
1. Antenna Electromagnetic Simulation

In this chapter we briefly describe the simulation techniques used in the rest of the work and provide an introduction and references for the two antenna types used as examples of implementation of the algorithm. The reader is referred to [2] for a detailed discussion of Matlab™ Antenna toolbox MoM implementation. The reader is encouraged to consult the references for a detailed exposition of the simulation techniques.

1.1. Simulation Techniques: Matlab™ Method of Moments.

1.1.1 Meshing

Also known as discretization of metals. It creates an array of triangles on the metal surface [2]. Shown in Figure 1-1.



```
NumTriangles : 970
NumBasis : -
MaxEdgeLength : 0.009204
MeshMode : manual
```

Figure 1-1 Matlab MoM meshing. Taken from [2].

1.1.2 Basis functions

Matlab uses the Rao-Wilton-Glisson basis functions on triangle pairs to calculate the surface currents. Shown in Figure 3-1. This is applied to the discretized metal surface [2].

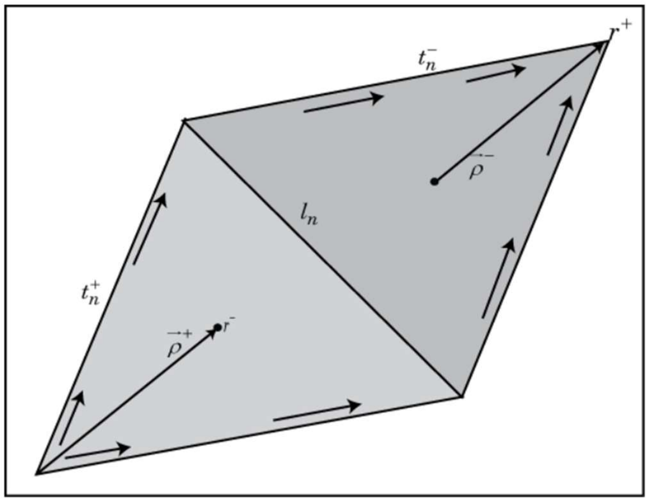


Figure 1-2 Basis functions applied to discretized metal surface. Taken from [2].

1.1.3 Interaction matrix

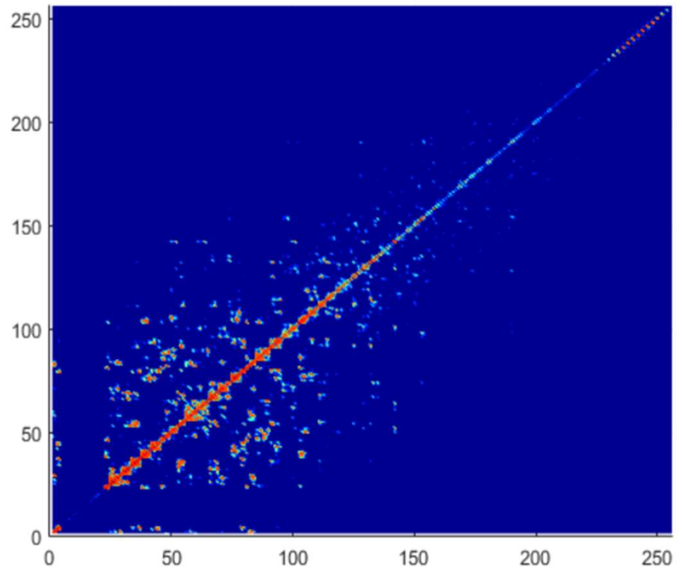


Figure 1-3 Typical interaction matrix for 256 basis functions. Taken from [2].

Next, the antenna is excited by 1V on the feeding edge. The free-space Green's function between all basis functions is calculated for all the triangles on the antenna surface. This creates a diagonally dominant matrix as shown in Figure 1-3 [2].

1.2. Simulation Techniques: HFSS 3D Finite Element analysis.

We use the HFSS 3D finite element modelling technique as the fine model. The finite element method technique divides a solid in thousands of tetrahedrons like the ones shown in the Figure 1-4 . The field equations, magnetic and electric fields are then solved for each element.

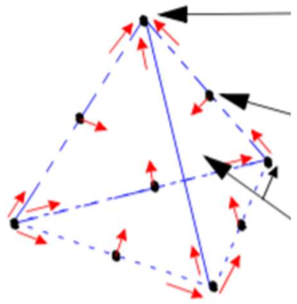


Figure 1-4 Tetrahedron for finite element analysis. Taken from [1]

1.3. Matlab™ to HFSS Driver.

When designing circuits using optimization techniques it is often desirable to use Matlab™ to drive other simulation software. This allows to leverage Matlab™ optimization functions and extensive mathematical libraries. In order to implement the algorithm, we drive Ansys™ EM desktop HFSS from Matlab.

The proposal is to create a generalized Matlab™-Python-HFSS™ driver based on python templates as shown in Figure 1-5.

- A Python template that is specific for the optimization objective is created. This template uses HFSS™ python instructions to modify HFSS™ parameters and run simulations. The parts that are not modified by the Matlab™ model are hardcoded in the template.

- The generalized Matlab™ driver receives from the optimization script a list of parameters and the line numbers where to write them in the template. Then it loads the template, modifies the requested lines and saves it.
- The generalized Matlab™ driver calls Ansys™ EM desktop executable file and loads the Python script generated from the template.
- HFSS™ executes the modifications to the model, runs the analysis and returns a model response in csv format to a specified directory. All the instructions are explicit in the python script generated from the template.
- The generalized Matlab™ driver waits for the HFSS routine to finish and imports the data generated in the csv file. It returns the model response to the optimization or model script that called it.

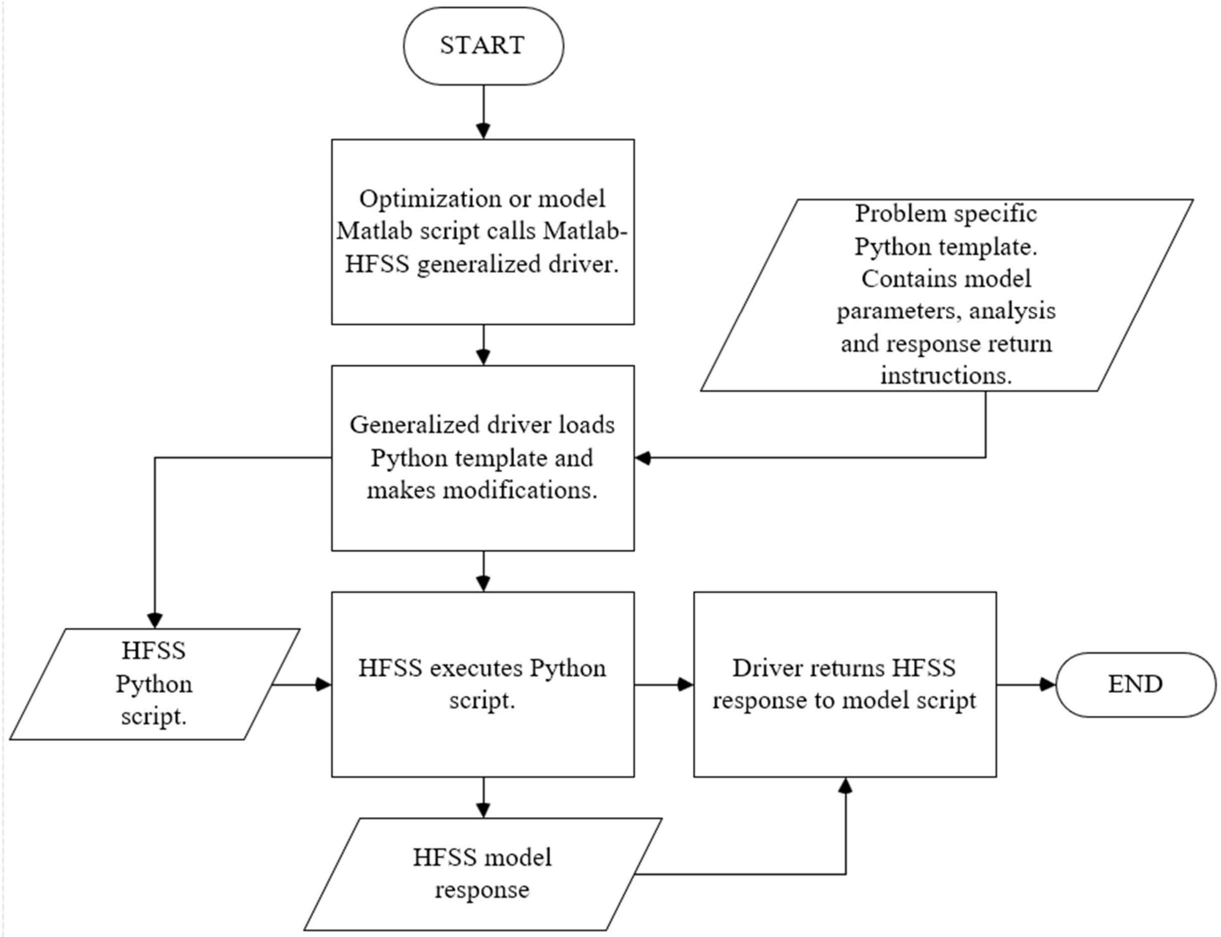

START
Optimization or model Matlab script calls Matlab-HFSS generalized driver.
Problem specific Python template. Contains model parameters, analysis and response return instructions.
Generalized driver loads Python template and makes modifications.
HFSS Python script.
HFSS executes Python script.
Driver returns HFSS response to model script
END
HFSS model response

Figure 1-5 Matlab-Python-HFSS generalized driver

1.4. Microstrip Patch Antenna.

The first microstrip antenna concepts date back to the 1950s with Deschamps array in 1953 being the first microstrip like antenna. However, at the time this was only an academic inquiry, it was until the aerospace industry provided power incentives for its adoption in the 1970s that its use became very popular [10]

Even though the theoretical base for the microstrip patch antenna analysis is that of the transmission line, the use of microstrip patch antennas exploded until the method of moments and computers became widely available [10].

Microstrip patch antennas are inexpensive to manufacture and have a very low profile. Nowadays microstrip antennas are extensively used for Bluetooth, GPS and WLAN [10].

For a very good first order approximation, using half a wavelength for the length of a microstrip patch antenna. This implies that a 300 MHz antenna is around 1 meter in length. For this reason, most of the applications are in the microwave range.

1.5. Dipole bowtie Antenna.

The bowtie antenna is a particular case of the family of biconical antennas, which can have 2D and 3D implementations. The bowtie antenna being a 2D triangular biconical antenna, usually etched on a PCB.

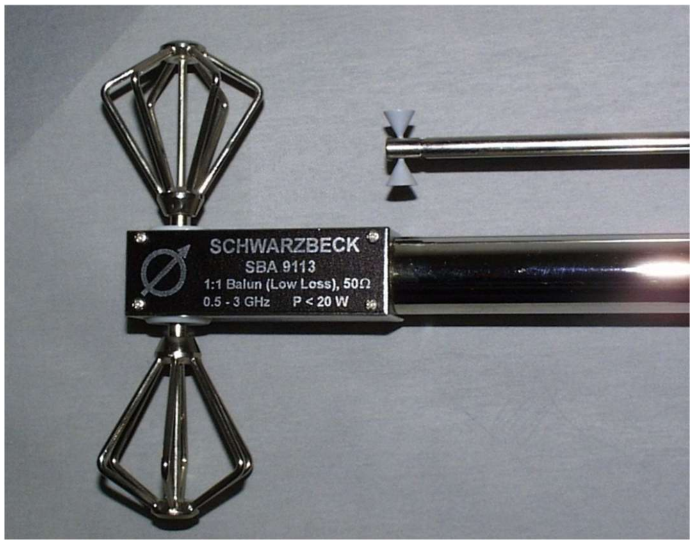


Figure 1-6 Examples of 3D biconical antennas. Taken from [22], CC license CC BY-SA 3.0.

Biconical antennas main advantage is their relatively wide bandwidth around the resonance frequency. Early versions of these antennas date back to 1987 by Sir Oliver Lodge [11].

As Kraus elaborates on chapter 8 of his classic book on antenna theory [11] “an infinite biconical antenna is analogous to an infinite uniform transmission line. The biconical antenna acts as a guide for a spherical wave in the same way that a uniform transmission line acts as a guide for a plane wave.” Which allows the analysis of the input impedance of the antenna using Maxwell’s equations to yield relatively simple closed form solutions. This in turn created a special interest in this antenna family before the powerful computer simulators were widely available.

The reader is referenced to Kraus [11] for an in-depth analysis on the subject.

2. Optimization Algorithms.

This chapter briefly describes the optimization algorithms used in the rest of the thesis dissertation. The reader is encouraged to consult the references for a detailed exposition of the algorithms.

2.1. Direct optimization.

2.1.1 Minimax.

The minimax formulation is used extensively in engineering optimization problems. Its main advantage consists in its straightforward ability to setup the optimization problem in terms of engineering specifications. Following the detailed exposition of Prof. Rayas in [12],we introduce the Minimax formulation for engineering circuit design and optimization:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \max\{\dots e_k(\boldsymbol{x}) \dots\} \qquad (2\text{-}1)$$

$$e_k = \begin{cases} R_k(\boldsymbol{x}) - S_{ub} & \text{for all } k \in \boldsymbol{I}^{ul} \\ S_{eq} - R_k(\boldsymbol{x}) & \text{for all } k \in \boldsymbol{I}^{eq} \\ R_k(\boldsymbol{x}) - S_{lb} & \text{for all } k \in \boldsymbol{I}^{lb} \end{cases} \qquad (2\text{-}2)$$

Where:

- R_k: is the model's response (simulation or calculation) at the kth index.
- S_{ub}: is the engineering specification at the upper bound.
- S_{lb}: is the engineering specification at the lower bound.
- S_{eq}: is any engineering specification that is not a bound but perhaps a desired characteristic of the response waveform, i.e. the value of the response at a resonance frequency.

The minimax formulation first compares the model responses to the engineering specification, then selects the maximum error as the objective function response. The objective function is then minimized using an optimization algorithm. This works uses the Matlab™ implementation of Nelder-Mead's minimization algorithm extensively. This dynamic then finds a solution for all the specifications, if the changes in the variables during the optimization favor one of the specifications too much, another specification will produce the maximum error and hence the algorithm will move in that direction.

In this work it was not necessary to normalize the error responses nature of the problem. However, the reader is referred to a more general formulation of the minimax problem in [12].

2.1.2 Nelder-Meade Algorithm.

The Nelder-Mead (NM) algorithm is a method used for unconstrained minimization of a vector valued function. It is based on heuristic transformations of a simplex; it replaces the vertex of the simplex that evaluates to the highest value of the objective function. NM belongs to the family of direct search optimization methods. The mathematical details of the algorithm are out of the scope of this work. The reader is referred to the original article by J.A. Nelder and R. Mead in [13], for a lucid and detailed explanation.

The NM is particularly well suited for both the minimization of the model's response in the minimax formulation and the parameter extraction stage during the space mapping algorithm. The main advantage of the NM method is that it does not require the computation of gradients. Gradients are computationally expensive. For the particular case of this work, the calculation of derivatives for the fine model of the antenna simulation, would result in a prohibitively expensive computation.

Due to its popularity all major mathematical packages include a heavily optimized implementation of the algorithm. This work uses the method extensively through Matlab™ function *fminsearch()*.

2.2. Input Space Mapping.

The space mapping algorithm is at the heart of this work, specifically the Broyden based input version of it. Hence, the method is described in detail in the following chapters and implemented in Matlab™, the code is available in appendices A-D.

The powerful idea behind space mapping is the creation of a map between a coarse, not so accurate but computationally effective, model and a fine, accurate but computationally expensive, model. This allows the designer to find the optimal input parameters for the fine model that produce a close enough approximation to an optimal response derived from the coarse model. The number of fine model evaluations using this approach is reduced significantly when compared to a direct optimization method.

The space mapping has come a long way since its inception in the nineties, the reader is referred to the excellent surveys by Bandler et al in [3] and Prof. Rayas in [14]. There are now multiple ways to implement the map, i.e. output space mapping, neural space mapping among others. This work implements the original Broyden based input space mapping.

3. Antenna Optimization Through Input Space Mapping Algorithm.

The core contribution of this work is an antenna optimization algorithm through input space mapping. We expand on the literature of antenna optimization through space mapping that started with Zhu et al seminal paper [4] in 2007. Notable contributions to the literature include: [5] [6] [7] [8]. The algorithm is summarized by Figure 3-1 and Figure 3-2. Each step is described in the following sections.

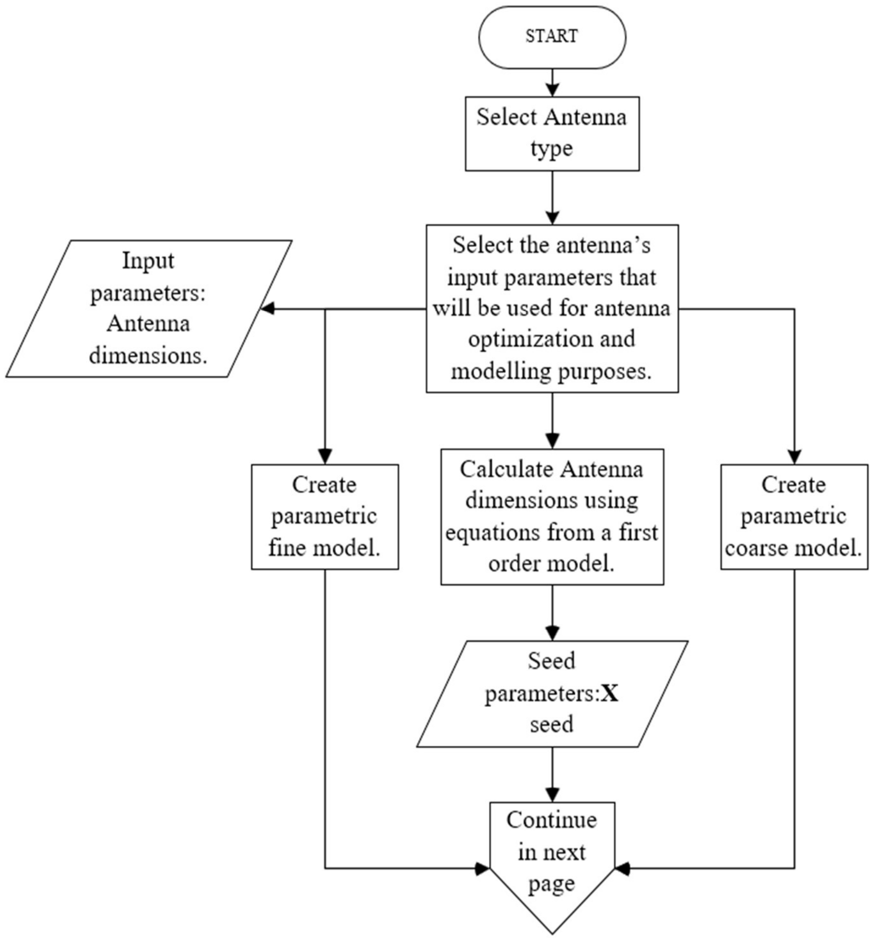

START
Select Antenna type
Select the antenna's input parameters that will be used for antenna optimization and modelling purposes.
Input parameters: Antenna dimensions.
Create parametric fine model.
Calculate Antenna dimensions using equations from a first order model.
Create parametric coarse model.
Seed parameters:X seed
Continue in next page

Figure 3-1 Antenna optimization through input space mapping algorithm 1 of 2.

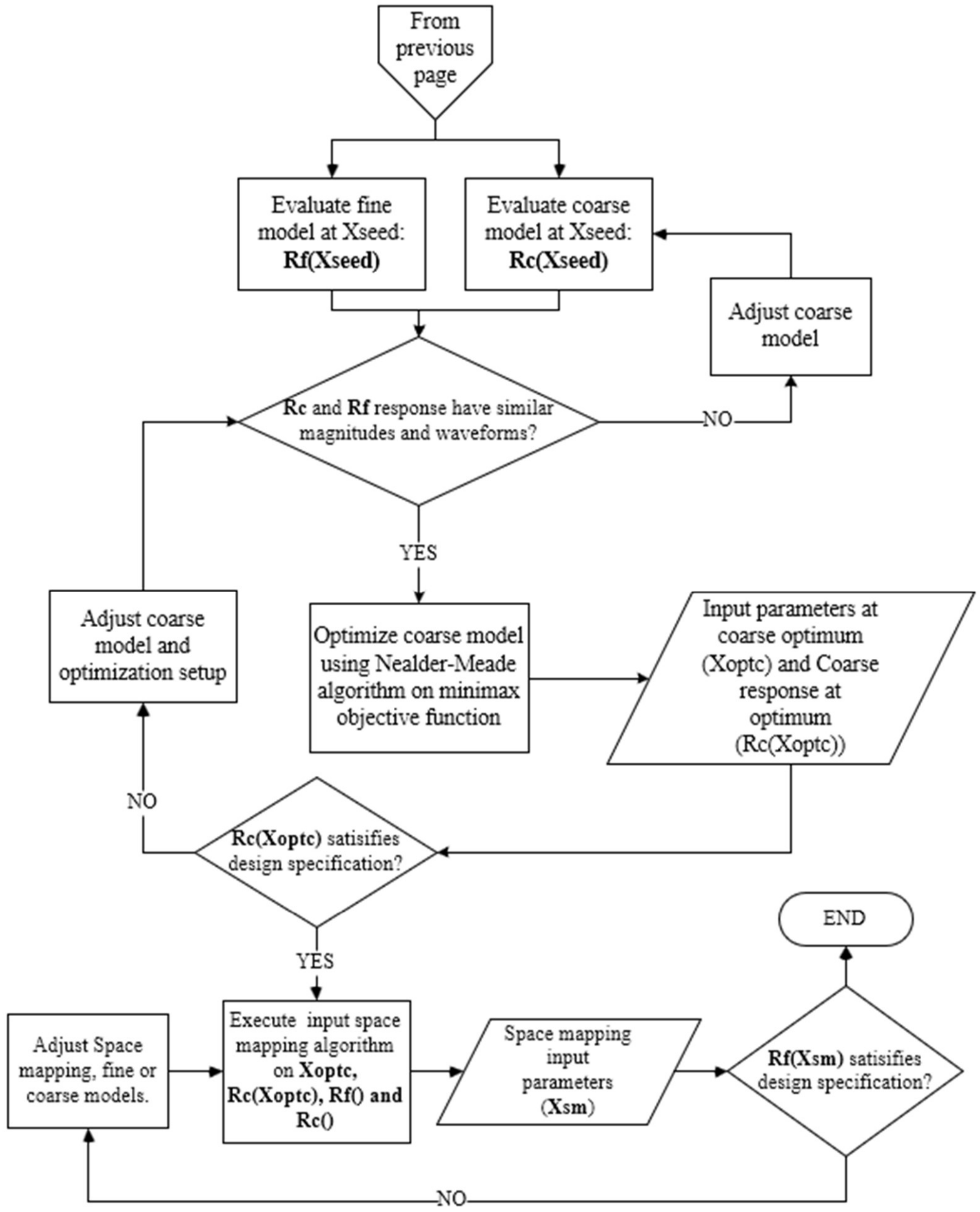

From previous page
Evaluate fine model at Xseed: Rf(Xseed)
Evaluate coarse model at Xseed: Rc(Xseed)
Adjust coarse model
Rc and Rf response have similar magnitudes and waveforms?
NO
YES
Adjust coarse model and optimization setup
Optimize coarse model using Nealder-Meade algorithm on minimax objective function
Input parameters at coarse optimum (Xoptc) and Coarse response at optimum (Rc(Xoptc))
NO
Rc(Xoptc) satisifies design specification?
YES
END
Adjust Space mapping, fine or coarse models.
Execute input space mapping algorithm on Xoptc, Rc(Xoptc), Rf() and Rc()
Space mapping input parameters (Xsm)
Rf(Xsm) satisifies design specification?
NO

Figure 3-2 Antenna optimization through input space mapping algorithm 2 of 2.

3.1. The algorithm

This section describes the antenna optimization through input space mapping in general and then proceed to apply it to two different antenna designs in chapters 4 and 5.

3.1.1 Antenna type

The first step is to have a target antenna type to design. This is driven by the engineering specification based on: size, gain, technology of implementation (wire, PCB, etc.), bandwidth, directivity and cost, among others. The focus of this work is on antenna optimization, hence the reader is referenced to the extensive literature on antennas for their theory of operation, for example [11] and [15],

Designing an antenna is understood to be an iterative process. If one antenna type fails to achieve the specifications, then a different antenna type is explored. The antenna optimization through input space mapping algorithm simplifies this process as most of the code written for one antenna type can be reused and adapted to try a different antenna.

3.1.2 Figure of merit

For the purpose of this work, the antenna's sole figure of merit is considered to be $|\boldsymbol{S_{11}}|$. Namely, the magnitude in the frequency band of interest of the S_{11} parameters, also known as reflection loss. We will also call this the response of the system formulated in equation (3-1).

The resonance frequency is the main optimization objective for the purposes of this work. The antenna reflection bands are a secondary objective. Even though the optimization problem can have some influence over them, they are for the most part predetermined by the antenna response waveform, a consequence of the antenna type chosen. Other antenna responses such as radiation pattern and directiveness are also considered a consequence of the antenna type chosen and will not be part of the optimization objective function.

$$\boldsymbol{R(x,\varphi) = |S_{11}(x)|} \qquad (3\text{-}1)$$

3.1.3 Antenna design parameters.

The best design parameters are the ones that have the largest effect on the figure of merit and that allow the smallest possible number of design parameters for the optimization algorithms.

That is, the marginal contribution of a parameter to the figure of merit must be larger than the cost (computational or mathematical) of adding that parameter to the model.

Let $\boldsymbol{R}(\boldsymbol{x_p})$ be the response of the model and $\boldsymbol{x_p}$ the set of all antenna and simulation parameters. The designer selects the disjoint sets $\boldsymbol{x}$ and $\boldsymbol{\varphi}$ such that $\boldsymbol{x_p} = \boldsymbol{x} \cup \boldsymbol{\varphi}$. Where $\boldsymbol{x} = (x_1, x_2, ..)$ become the design parameters, that is the inputs to optimization algorithms and $\boldsymbol{\varphi} = (\varphi_1, \varphi_2, ..)$ become exogenous or independent parameters that are necessary to obtain the model's response but are not acted upon by the optimization. We refer to the response of the model as $\boldsymbol{R}(\boldsymbol{x}, \boldsymbol{\varphi})$ which we shorten to $\boldsymbol{R}(\boldsymbol{x})$ for notation convenience, making implicit the effect of the $\boldsymbol{\varphi}$ exogenous parameters in the model.

The vector $\boldsymbol{\varphi}$ exogenous parameters can be constrained by other design considerations. A common example of this is when an antenna must be designed for a certain printed circuit board (PCB) stack-up. In such a case, parameters with large effects on the model response such as relative permittivity and loss tangent become independent or exogenous variables of the antenna model.

The fact that $\boldsymbol{\varphi}$ parameters are exogenous variables for the model does not mean the designer has no effect over them. For example, if the PCB stack-up is not a constrain and can be determined by the designer, the designer can perform a parameter exploration and choose a stack-up such that antenna gain, and dimensions are within a design envelope. Effectively choosing the relative permittivity and loss tangent as $\boldsymbol{\varphi}$ parameters. However, they are still not modified by the optimization problem.

Antenna design parameters are usually the antenna's geometric dimensions, as they tend to have the largest effects on the figure of merit. Not all dimensions will have large effects compared to the complexity that they introduce to the model. The designer is advised to do parameter explorations with the model to determine the best vector of design parameters $\boldsymbol{x}$ for the frequency bands of interest.

Finally, we provide the reader with the following design parameter recommendations for a successful algorithm implementation:

- Select the least possible design parameters $\boldsymbol{x}$. We recommend a parameter exploration on the antenna resonant frequency of interest and understand the contributions of each design parameter. Usually, more than a few design parameters create inadequate complexity in the model. We present two antenna design examples in which only a couple of design parameters are needed to successfully design an antenna using the algorithm.
- Design parameter selection is often an iterative process. It is better to start with a few and add more parameters only if strictly needed.
- Setting convenient values for the independent parameters $\boldsymbol{\varphi}$ is crucial for the success of the algorithm. A first order model of the antenna (discussed in detail in next section) is a great way to set these values. Again, this is an iterative process,

the reader is advised to reconsider and fine-tune the parameters for a successful algorithm implementation.

3.1.4 Fine model.

Fine models are relatively precise yet computationally expensive models, for which direct optimization is prohibitive, or at least inconvenient, for the designer. The fine model is assumed to be the most precise representation of the real response of the system available to the designer. We consider the antenna optimization successful when the fine response, denoted as $\boldsymbol{R_f}(\boldsymbol{x}, \boldsymbol{\varphi})$ or $\boldsymbol{R_f}(\boldsymbol{x})$ for short, satisfies the design specifications. We refer the reader to our exposition of antenna modelling techniques from chapter 1.

What represents a fine model depends on the context. Coarse and fine models can use the same modeling technique with just a mesh adjustment, such as in [4]. We use a different approach where the coarse model exploits a 2D MoM modeling technique from Matlab™. And the fine model uses 3D finite element analysis (ANSYS™ EM HFSS).

Fine models will most likely require drivers from other mathematical software that will execute the optimization routines and space mapping. The driver should be sufficiently general in its interface for reuse and easiness of adaptation to the iterations of the design, especially during the exploratory phase when antenna types and design parameters are determined. This work presents a generalized Matlab™ to ANSYS™ EM HFSS driver in appendix D.

3.1.5 Coarse model.

Coarse models should be computationally inexpensive yet precise enough to be able to be used in space mapping. Examples of coarse models for antenna optimization include closed form equations of different orders and coarse mesh MoM models.

Care must be taken so that the coarse response, $\boldsymbol{R_c}(\boldsymbol{x}, \boldsymbol{\varphi})$ or $\boldsymbol{R_c}(\boldsymbol{x})$ for short, has close enough magnitudes as the fine response $\boldsymbol{R_f}(\boldsymbol{x}, \boldsymbol{\varphi})$ in the frequency bands of interest. This is extremely important for the input space mapping part of the algorithm, specifically the parameter extraction.

We can exploit the exogenous parameters $\boldsymbol{\varphi}$ in order to adjust the magnitude of the response, that is have different exogenous parameters for the coarse and fine model $\boldsymbol{\varphi_c} \neq \boldsymbol{\varphi_f}$. This implies the coarse and fine models may have, for example, different PCB stack-ups. This is illustrated in section 5.4. The fine model's PCB stack-up parameters will represent the real system, whereas the coarse model exogenous parameters can be artificially set to obtain the desired response. This model discrepancy is completely acceptable, as the coarse model in this case is a mere mathematical artifice in the input space mapping algorithm and does not need to represent the real system.

The coarse model response $\boldsymbol{R_c}(\boldsymbol{x})$ waveform should be sufficiently similar to the fine model response $\boldsymbol{R_f}(\boldsymbol{x})$. When modelling antennas, this is very difficult if not impossible to achieve, especially at frequencies different from the resonance frequency. This represents a different challenge compared to the magnitude mismatch. For antenna optimization, the parameter extraction can be constrained to the regions or frequency bands where the waveform is sufficiently similar, provided the magnitudes of the responses are already close enough.

The coarse model has 2 main functions in the algorithm:

1. Produce the target response $\boldsymbol{R_c}(\boldsymbol{x_c^*}) = \boldsymbol{R_T}$ and the optimal design parameters for the coarse model $\boldsymbol{x_c^*}$..Details are presented in the following section.
2. Serve as the computationally inexpensive coarse model for the input space mapping algorithm.

Creating a coarse model that fulfils the aforementioned duties is a delicate balancing act. The designer should analyze and explore the coarse model to understand its sensitivities and limitations. An iterative process to find the right coarseness on the model is expected. In the author's experience, proposing such a model is the most challenging task of implementing the algorithm.

3.1.6 Generate input seed.

Once the fine and coarse models are in place, we need a seed for the design parameters $\boldsymbol{x}$. This can come from a first order model of the antenna type. The quality of the seed will determine a successful convergence of the optimization algorithm; care must be taken to get a high-quality seed. Starting with a random seed is not recommended.

There are first order models, usually closed form formulae, for most antenna types. The reader is referred to the classic books on the subject [11] [15]. Another option is to use the implementation of such equations on commercial software: Matlab™ Antenna toolbox [2] and ANSYS™ Electronics Desktop Antenna Toolkit [1] will provide an excellent seed.

Let $\boldsymbol{x_{seed}}$ be the corresponding design parameters obtained from the first order model. They will be used for the coarse model adjustment and as seed for optimization.

3.1.7 Compare responses and adjust coarse model.

Evaluate both the coarse and fine model and obtain their responses, $\boldsymbol{R_c}(\boldsymbol{x_{seed}}, \boldsymbol{\varphi})$ and, $\boldsymbol{R_f}(\boldsymbol{x_{seed}}, \boldsymbol{\varphi})$ respectively. Compare the magnitudes and waveforms of the responses and adjust the coarse model as per the following algorithm.

Let $\boldsymbol{R_c}(\boldsymbol{x_{seed}}, \boldsymbol{\varphi_c})$ be the new coarse model with response's magnitudes and waveform sufficiently similar to the fine model. Note the fine and coarse models' exogenous parameters may be different at this point, $\boldsymbol{\varphi_c} \neq \boldsymbol{\varphi_f}$. It is unlikely this step is not needed for antenna optimization through input space mapping. The authors consider this step of the algorithm the most important for the successful implementation of input space mapping.

During the literature survey no numerical methods to evaluate the degree of similarity between the magnitudes and waveforms of antenna responses were found. Visual inspection of a parameter sweep over the coarse model has given good results. Since this is iterated over the coarse model it is not computationally expensive.

There could be programmatic ways to determine de goodness of the coarse model that do not depend on visual inspection, for example the pth-norm between two vectors, which is an opportunity for future research and enhancement of this algorithm.

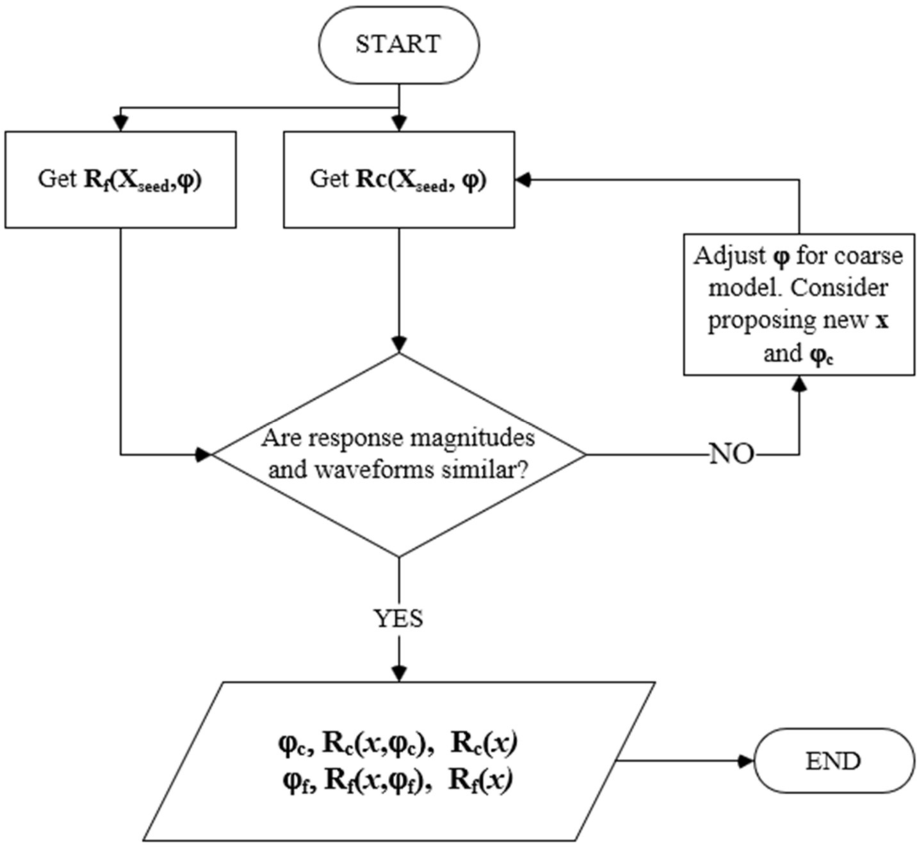

START
Get $\mathbf{R_f}(\mathbf{X}_{seed},\boldsymbol{\varphi})$
Get $\mathbf{Rc}(\mathbf{X}_{seed}, \boldsymbol{\varphi})$
Adjust $\boldsymbol{\varphi}$ for coarse model. Consider proposing new $\mathbf{x}$ and $\boldsymbol{\varphi}_c$
Are response magnitudes and waveforms similar?
NO
YES
$\boldsymbol{\varphi}_c, \mathbf{R}_c(x,\boldsymbol{\varphi}_c), \mathbf{R}_c(x)$
$\boldsymbol{\varphi}_f, \mathbf{R}_f(x,\boldsymbol{\varphi}_f), \mathbf{R}_f(x)$
END

Figure 3-3 Coarse model adjustment for input space mapping

3.1.8 Target response from coarse model optimization

A target response is required for parameter extraction on the space mapping algorithm. This is obtained from direct optimization of the coarse model using the minimax formulation in equation (3-2). Note that the constraint $\boldsymbol{x} \in \boldsymbol{\Phi}$ for the optimization problem is optional, but often necessary to get the target response $\boldsymbol{R_T}$.

Engineering specifications for the figure of merit are formulated in the objective function represented by equation (3-3). We propose three specifications:

1. **Lower reflection band:** a band lower or to the left of the reflection band where the response should be sufficiently large as to not be a resonance band.
2. **Resonance band:** where the response obtains its minimum value, that is the antenna radiates the most in this band since its reflection loss is at a minimum.
3. **Higher reflection band:** a band higher or to the right of the resonance band where the response should be sufficiently large as to not be a resonance band.

$$\boldsymbol{x_c^*} = \arg\min_{\boldsymbol{x}\in\boldsymbol{\Phi}} \max\{\dots e_k(\boldsymbol{x}) \dots\} \qquad (3\text{-}2)$$

$$e_k = \begin{cases} R_{c,k}(\boldsymbol{x}) - S_{reflect} & \text{for all } k \in \boldsymbol{I}^{ReflectL} \\ S_{resonate} - R_{c,k}(\boldsymbol{x}) & \text{for all } k \in \boldsymbol{I}^{Resonate} \\ R_{c,k}(\boldsymbol{x}) - S_{reflect} & \text{for all } k \in \boldsymbol{I}^{ReflectH} \end{cases} \qquad (3\text{-}3)$$

Where:

- $R_{c,k}$: is the coarse response at the kth index. Note this is a scalar value, compare to the coarse response vector $\boldsymbol{R_c}(\boldsymbol{x})$.
- $S_{reflect}$: is the specification value of the response for any of the reflect bands. Aim for a number close to 1, yet this is highly dependent on the response waveform of the antenna type chosen.
- $S_{resonate}$: is the specification value of the response for the resonate band. Aim for a number close to 0, or at least as low as possible. This will be heavily influenced by the exogenous parameters $\boldsymbol{\varphi}$. It will be an iterative process to find a reasonable value for this specification.
- $\boldsymbol{I}^{ReflectL}$: is the set of all indexes in the response vector that correspond to the lower reflection band.
- $\boldsymbol{I}^{Resonate}$: is the set of all indexes in the response vector that correspond to the resonate band.

- $\boldsymbol{I}^{ReflectH}$: is the set of all indexes in the response vector that correspond to the lower reflection band.
- $\boldsymbol{I}^{ReflectL}$, $\boldsymbol{I}^{Resonate}$ and $\boldsymbol{I}^{ReflectH}$ are disjoint subsets of the set of all indexes of the response vector $\boldsymbol{R_c}$.

The minimax formulation is optimized using Matlab™ implementation of the Nelder-Mead algorithm through its *fminsearch()* function. Evaluation of the optimal input parameters $\boldsymbol{x_c^*}$ on the coarse model produces the target response $\boldsymbol{R_T}$ as per equation (3-4) .

$$\boldsymbol{R_T} = \boldsymbol{R}_c(\boldsymbol{x_c^*}, \boldsymbol{\varphi_c}) \qquad (3\text{-}4)$$

Evaluation of the optimal input parameters in the fine model $(\boldsymbol{R}_f(\boldsymbol{x_c^*}, \boldsymbol{\varphi_f})$ produces a response that is different from the target response. This is represented by equation (3-5)

$$\left\|\boldsymbol{R}_c(\boldsymbol{x_c^*}, \boldsymbol{\varphi_c}) - \boldsymbol{R}_f(\boldsymbol{x_c^*}, \boldsymbol{\varphi_f})\right\| \gg 0 \qquad (3\text{-}5)$$

If this is not the case, it means both coarse and fine models are too similar. Either the coarse model is not coarse enough or the fine model is not fine enough. Hence, they must be adjusted. For the algorithm proposed in this work this is never a problem, as the coarse model is based on a 2D coarse mesh MoM simulation while the fine model is a 3D finite element simulation. However, this is an important consideration when both fine and coarse models use MoM simulations with different meshes, just as in reference [4].

Choosing the specifications and frequency bands is an iterative process that includes multiple evaluations of the minimax formulation. It is often easier to start with relaxed specifications, then tight them up until the optimization algorithm converges to an acceptable target response.

Specifications that are too tight will likely put an unnecessary burden on the optimization algorithm. Analogously, frequency bands should start with narrow bands: a thin band around the resonant frequency and narrow reflect bands located at the frequency cut-off of interest. Adjust accordingly in case multiple resonant frequencies are desired. See Figure 3-4 below:

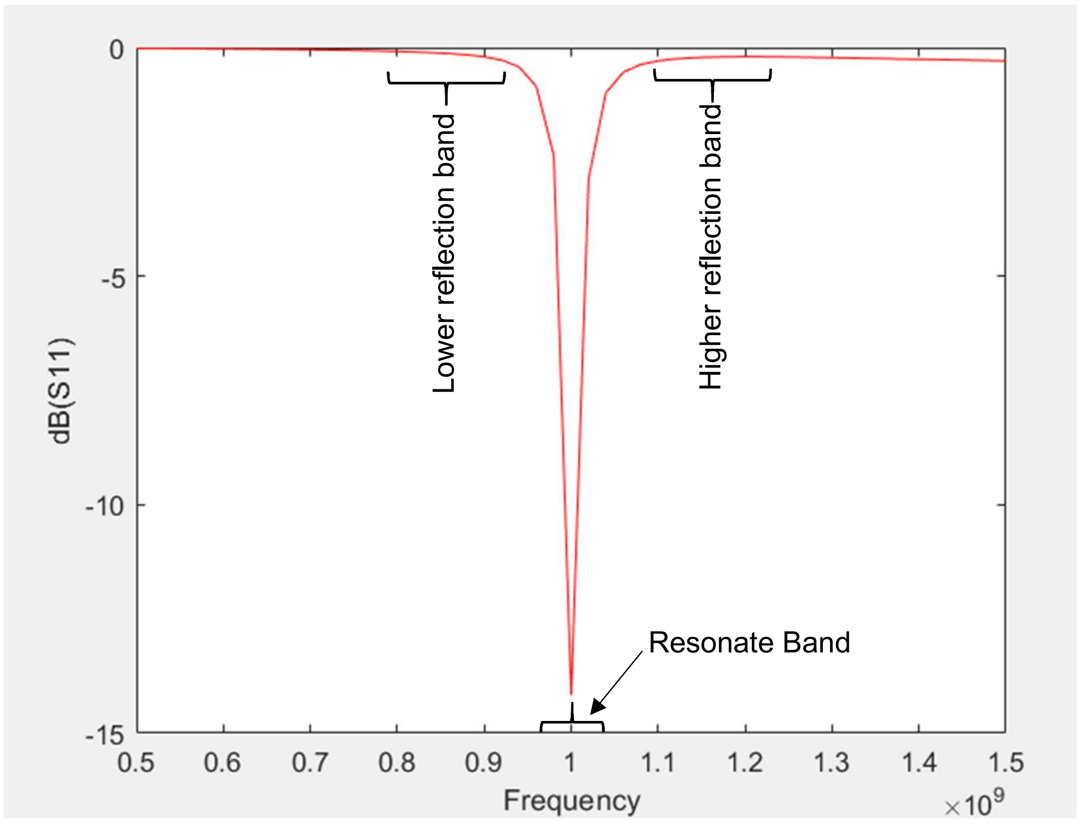


Figure 3-4 Narrow frequency bands set up for objective function of minimax formulation.

Several iterations are needed for the minimax formulation to yield a desired target response. The reader is advised to adjust optimization parameters, exogenous parameters of the coarse model, resonance and reflect specifications and the frequency bands of the objective function. The diagram in Figure 3-5 below provides an algorithm for this purpose.

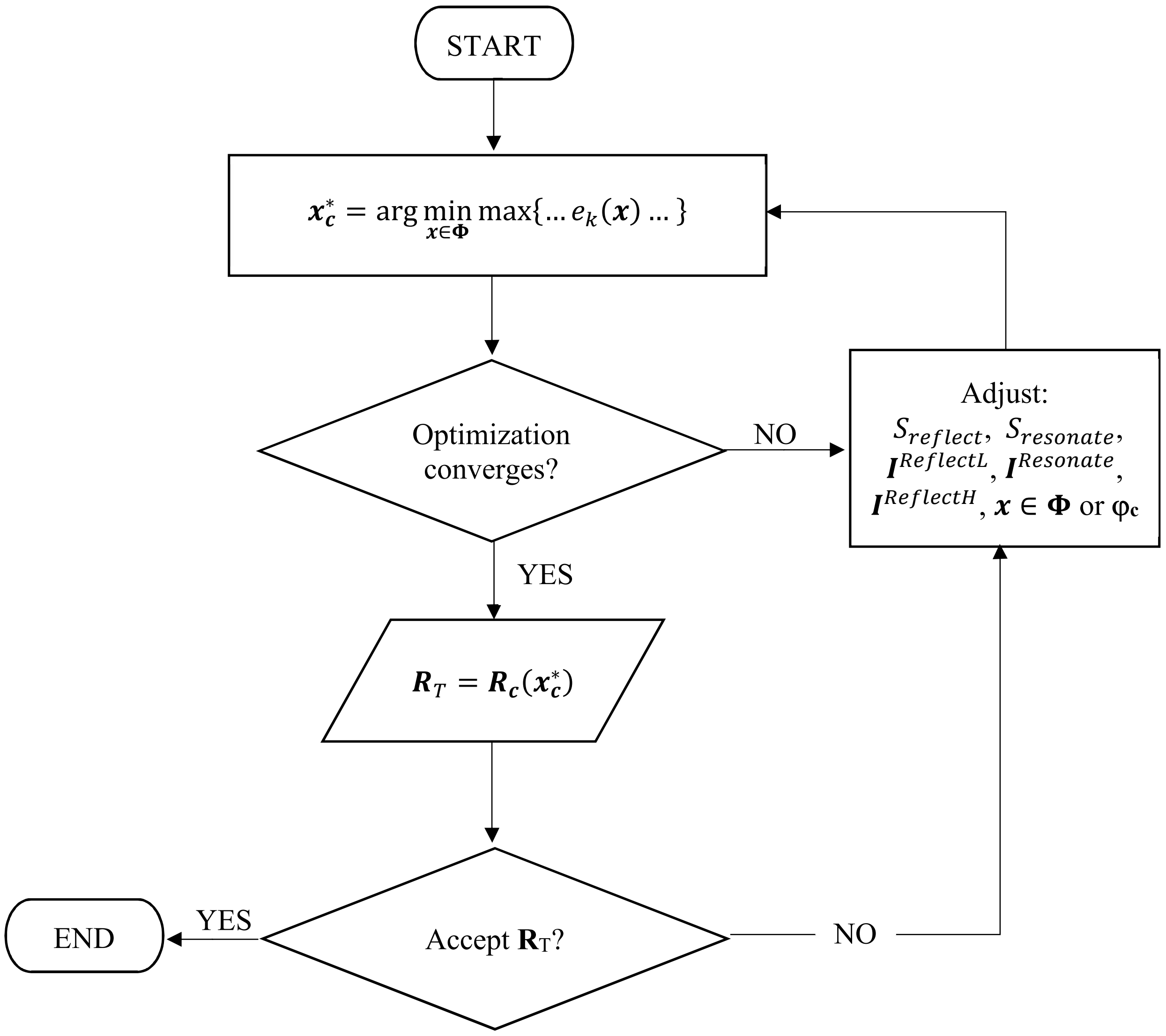

START
$x_c^* = \arg\min_{x\in\Phi} \max\{\dots e_k(x) \dots\}$
Optimization converges?
NO
YES
Adjust:
$S_{reflect}$, $S_{resonate}$, $I^{ReflectL}$, $I^{Resonate}$, $I^{ReflectH}$, $x \in \Phi$ or φ_c
$R_T = R_c(x_c^*)$
Accept R_T?
YES
NO
END

Figure 3-5 Minimax formulation Iterations to get a proper target response

3.1.9 Broyden based Input Space mapping.

Once a target response and suitable coarse and fine models have been developed, we proceed with the application of the Broyden based input space mapping (BISM), described in detail in section 2.2.

The algorithm will produce the space mapping input parameters $\boldsymbol{x}_{SM}^*$ and the fine model response $\boldsymbol{R}_f(\boldsymbol{x}_{SM}^*, \boldsymbol{\varphi}_f)$ which should be compliant with the engineering specifications of the design.

Otherwise, we need to adjust the models as per the previous sections until we get an acceptable fine model response.

The diagram in Figure 3-6 below applies the BISM algorithm to the antenna design case, following [14].

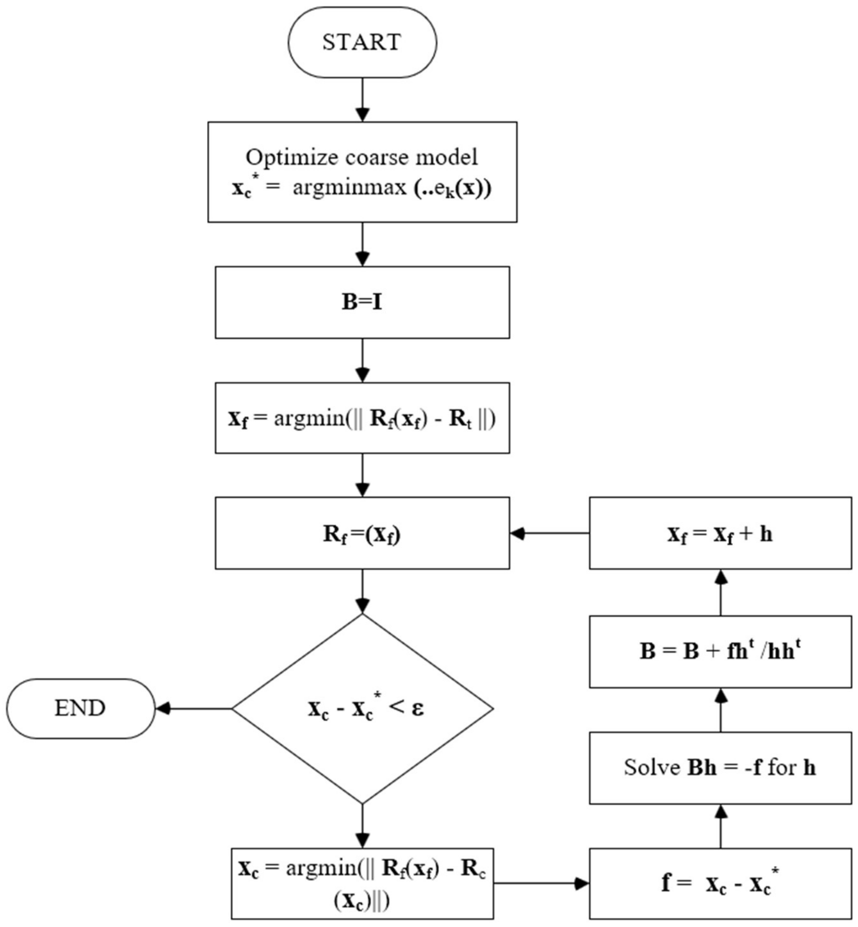

START
Optimize coarse model
$\mathbf{x}_c^* =$ argminmax (..$e_k(\mathbf{x})$)
$\mathbf{B}=\mathbf{I}$
$\mathbf{x_f} = \text{argmin}(\| \mathbf{R}_f(\mathbf{x}_f) - \mathbf{R}_t \|)$
$\mathbf{R_f}=(\mathbf{X}_f)$
$\mathbf{x}_c - \mathbf{x}_c^* < \varepsilon$
END
$\mathbf{x}_c = \text{argmin}(\| \mathbf{R}_f(\mathbf{x}_f) - \mathbf{R}_c(\mathbf{x}_c)\|)$
$\mathbf{f} = \mathbf{x}_c - \mathbf{x}_c^*$
Solve $\mathbf{Bh} = -\mathbf{f}$ for $\mathbf{h}$
$\mathbf{B} = \mathbf{B} + \mathbf{fh}^t/\mathbf{hh}^t$
$\mathbf{x_f} = \mathbf{x_f} + \mathbf{h}$

Figure 3-6 Broyden based input space mapping algorithm

The next step is to build an antenna prototype and correlate the simulations with laboratory measurements. Given that we are using a 3D finite element simulator as our fine model, a high correlation with laboratory measurements is expected.

This is the end of the algorithm; the antenna optimization and design has been completed.

4. Bowtie dipole antenna optimization through space mapping.

In this chapter the implementation of the antenna optimization through input space mapping algorithm applied to a dipole bowtie antenna printed on a PCB is presented. Refer to the antenna's characteristics discussed in section 1.5. The coarse and fine models are developed and a deep dive on the specifics needed for the successful implementation of the algorithm is provided.

The different characteristics of the antenna responses such as radiation pattern, 2D gain shape and our figure of merit waveform, are considered to be a consequence of the antenna type selection. Hence, these are not part of the optimization objective.

The objective is to design an antenna with resonance frequency at 1 GHz and reflection loss at the resonance frequency that is not greater than -13 dB.

4.1. Coarse model.

A coarse mesh Matlab™ MoM is used as the coarse model. The model is built as follows:

We start by creating a *bowtieTriangular* Matlab™ object from the Antenna Toolbox [2]. It that has the following parameters: length, flare angle, feed location, tilt and tilt axis.

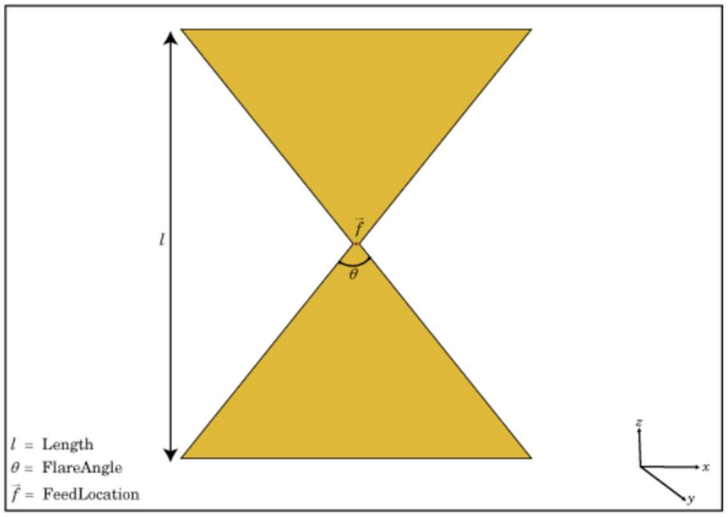


Figure 4-1 Geometric parameters for the bowtie antenna Matlab™ object. Image taken from [2].

Then we apply a coarse mesh MoM simulation to the *bowtieTriangular* object. We define the mesh parameter *m* in equation (4-1). It will enable us to fine-tune the coarseness of the model and hence balance its computational demands against its accuracy.

$$m = \frac{\lambda}{\eta} = \frac{c}{f_{max}\eta} \quad (4\text{-}1)$$

Where:

- o λ is the minimum wavelength of interest in the simulation.
- o c is the speed of light.
- o f_{max} is the maximum frequency of interest in the simulation.
- o η is an arbitrary parameter to modify the mesh edge length in terms of the maximum frequency of interest in the simulation.

We choose 2 geometric parameters as our input parameters; the rest are considered exogenous parameters. Namely, the coarse model for the bowtie antenna parameter becomes:

$$\boldsymbol{R_c}(\boldsymbol{x_c}, \boldsymbol{\varphi_c}) = |\boldsymbol{S_{11}}| \quad (4\text{-}2)$$

Where:

- o $\boldsymbol{x_c} = (x_1, x_2)$ is the vector of design parameters:
  - ▪ x_1 - Bowtie's length in meters
  - ▪ x_2 - Flare angle in radians.
  - ▪ See Figure 4-1 .
- o $\boldsymbol{\varphi_c} = (\varphi_1, \dots, \varphi_5)$ is the vector of exogenous parameters:
  - ▪ $\varphi_1 = 0$ - Tilt in degrees (no tilt).
  - ▪ $\varphi_2 = x$ - Tilt axis (x, y, z).
  - ▪ $\boldsymbol{\varphi}_3 = (0,0)$ - Feed location in cartesian coordinates in meters.
  - ▪ $\varphi_4 = 1.9986$ - Mesh parameter *m*.
  - ▪ $\varphi_5 = 101$ - Number of frequency points in the simulation.

Adequate values for φ_4 and φ_5 are determined through a parameter sweep. We get a coarse model that is both computationally inexpensive yet accurate at $\eta = 1x10^{-3}$ and 101 frequency points.

4.2. Input parameter seed.

Apply Matlab™ antenna toolbox *design()* function to the bowtie antenna object and get the seed in equation (4-3) then solve for a resonance frequency of 1GHz. This function implements a first order model for antenna design which is too coarse to use in our algorithm.

$$\boldsymbol{x_{seed}} = (0.0794, \frac{\pi}{2}) \qquad (4\text{-}3)$$

The coarse response at the input seed, $\boldsymbol{R_c}(\boldsymbol{x_{seed}}, \boldsymbol{\varphi_c})$, is shown in Figure 4-2. Note that the first order model produces a great seed for the coarse model optimization step. Even though the coarse model is not tuned for accuracy, it is still more accurate than the first order model implemented in the *design()* function. This is noted by the fact that the resonance frequency is not located at 1GHz when evaluated in the coarse model. The magnitude of the response complies with the design specifications.

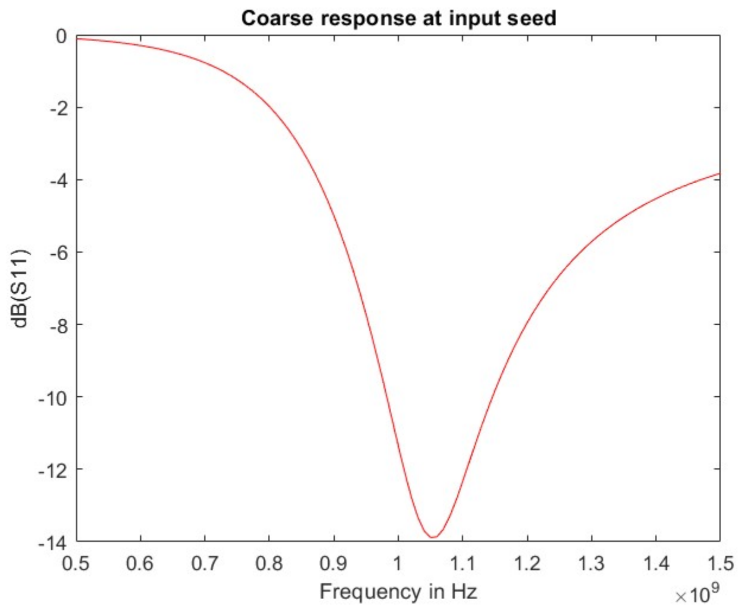


Figure 4-2 Coarse response at input seed. Note the resonance frequency is not located at 1 GHz

4.3. Fine model.

The fine model implementation of a dipole bowtie antenna in Ansys™ EM desktop 2019 is presented. We use the Antenna toolkit [1] to generate the antenna's geometric model. The toolkit uses 7 parameters to build the geometric model: substrate dimension X, substrate dimension Y, substrate thickness, port gap width, inner width, arm length and outer width. See Figure 4-3.

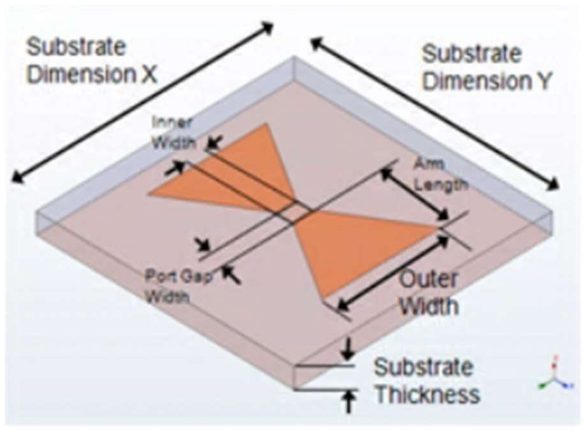


Figure 4-3 Dipole Bowtie antenna geometric parameters. Taken from ATK [1].

Two geometric dimensions are chosen as input parameters; the rest are considered exogenous parameters. The fine model for the bowtie antenna becomes:

$$\boldsymbol{R_f}(\boldsymbol{x_f}, \boldsymbol{\varphi_f}) = |\boldsymbol{S_{11}}| \qquad (4\text{-}4)$$

Where:

- $\boldsymbol{x_f} = (x_1, x_2)$ is the vector of design parameters:
  - x_1 - Arm length in meters
  - x_2 - Outer width in meters.
  - See Figure 4-3
- $\boldsymbol{\varphi_f} = (\varphi_1, \dots, \varphi_5\ )$ is the vector of exogenous parameters:
  - $\varphi_1 = 4x_2$ **-** Substrate dimension X in meters.
  - $\varphi_2 = 4x_1$ - Substrate dimension Y in meters.
  - $\varphi_3 = \ 0.016$ - Substrate thickness in meters.
  - $\varphi_4 = 0.0001$ - Port gap width in meters.
  - $\varphi_5 = 0.0001$ - Inner width in meters.
  - $\varphi_6 = 101$ - Number of frequency points in the simulation.
  - $\varphi_7 = 4.4$ - Substrate relative permittivity.
  - $\varphi_8 = 0.02$ - Substrate loss tangent.

Even though φ_1 and φ_2 are not constants determined exogenously, they are refreshed in every iteration of the fine model, they are not considered part of the input parameters. Their only purpose is to create a PCB substrate big enough to model the antenna during the simulation. They have no significant influence on the response of the antenna other than enabling the simulation to run.

The geometric model renders Figure 4-4.

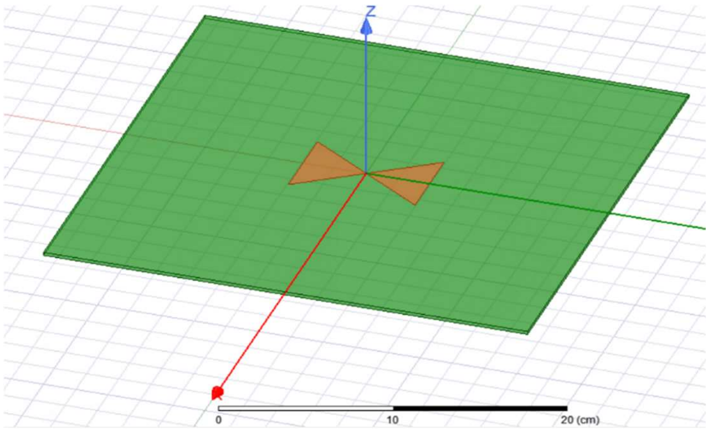


Figure 4-4 Bowtie antenna HFSS geometric Model

The fine response at the input seed, $\boldsymbol{R_f}(\boldsymbol{x_{seed}}, \boldsymbol{\varphi_f})$, is shown in Figure 4-5 below. Note that both response magnitude and waveform are sufficiently similar to the coarse model, which is a requisite for the rest of the algorithm to work. While optimizing the coarse model this ceased to be true. See section 4.4 for details on how this was addressed.

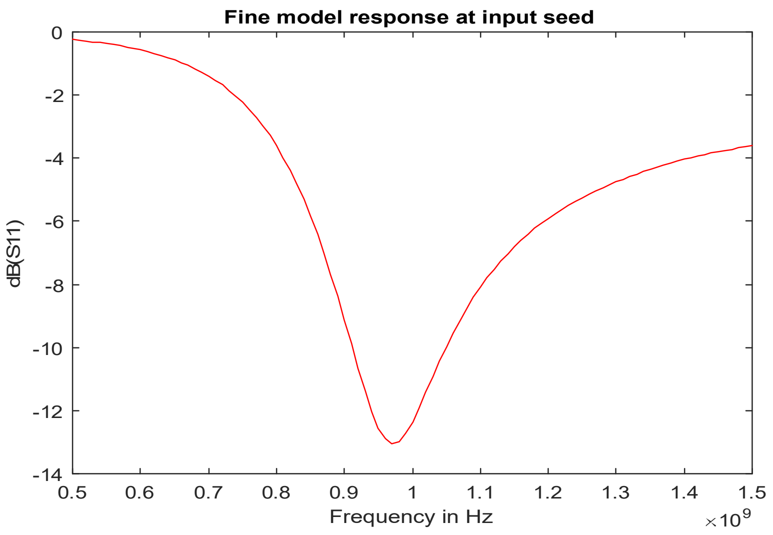


Figure 4-5 Fine model response at input seed. Note response magnitude and waveform.

As mentioned in the introduction, other responses from the model such as the antenna radiation pattern in Figure 4-6 and the shape of the 2D gain in Figure 4-7 are considered consequences of the antenna type selection and are not part of the optimization objectives.

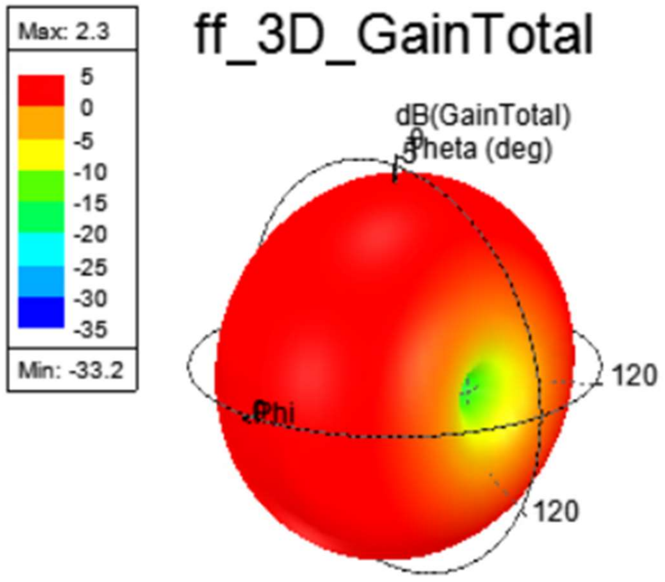


Figure 4-6 Bowtie dipole antenna 3D gain or radiation pattern

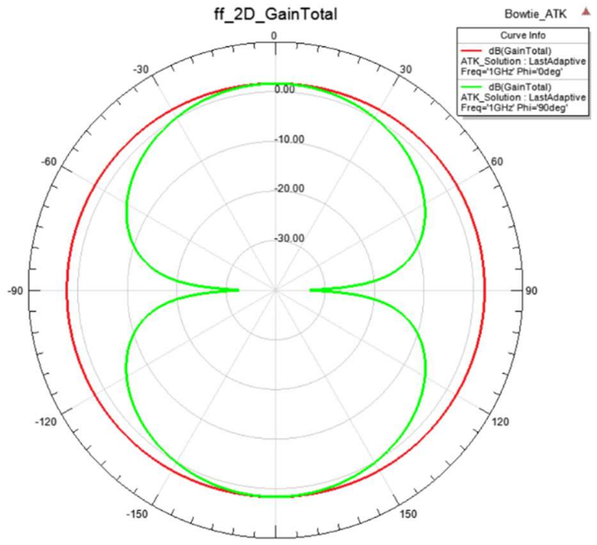


Figure 4-7 Bowtie dipole 2D gain plot

4.4. Fine tune coarse model.

It is imperative for the rest of the algorithm to work that the waveforms and magnitudes of the coarse and fine models are sufficiently similar. We recommend an iterative process in every step of the algorithm to confirm this is continues to be true.

In this case, the magnitude of the response of the coarse model matched the fine model when evaluated at the seed. However, after several optimization iterations it started to diverge significantly. The vector of exogenous parameters did not provide a good way to compensate this because the coarse response does not model substrate parameters such as relative permittivity or loss tangent.

After several parameter sweeps, we realized the major contributing factor was the flare angle magnitude. At values close to $\frac{\pi}{2}$., the magnitudes were similar; they diverged when the flare angle deviated from it.

The desired coarse model magnitude is guaranteed by introducing a box constraint to the optimization problem, formulated in equation (4-5). This constraint was implemented in the model response as opposed to the optimization algorithm. See code in appendix B.

$$87° < x_2 < 93° \qquad (4\text{-}5)$$

Figure 4-8 shows the similar magnitudes and waveforms required for successful implementation of the algorithm after implementation of the box constraint.

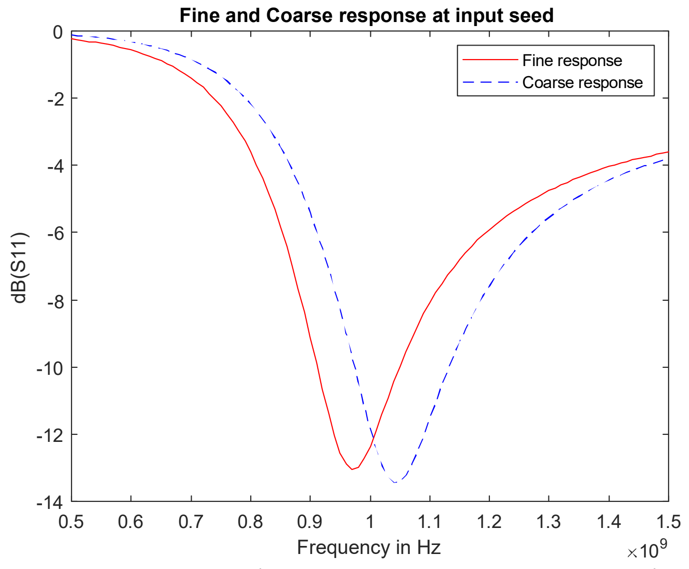


Figure 4-8 Fine and Coarse response at input seed. Note the magnitude and waveform similarities required for the successful implementation of the algorithm.

4.5. Coarse model optimization.

The target response $\boldsymbol{R_t}$ from the optimization of the coarse model is obtained. We propose the minimax formulation in equation (4-6) with the objective function in (4-7). As previously noted, a box constrain on the optimization problem is applied to guarantee the desired response magnitude. After several iterations, the specific formulation values follow.

We found two conditions that enabled the convergence of the optimization:

1. Encapsulate the *bowtieTriangular* Matlab™ object in order to convert parameter x_2 to radians from the original degrees. When in degrees the optimization was unable to converge.
2. Add the box constraint to the optimization shown in equation

Let the vector f_{sweep} be 101 frequencies, equally spaced between 0.5 GHz and 1.5 GHz, at which the simulation is executed. Let $\boldsymbol{I}^{sweep}$ be the set of all indices of f_{sweep} which correspond exactly with the indices of coarse response $\boldsymbol{R_c}(\boldsymbol{x_c}, \boldsymbol{\varphi_c})$. Such that, $\boldsymbol{I}^{ReflectL}$ and $\boldsymbol{I}^{Resonate}$ are disjoint subsets of $\boldsymbol{I}^{sweep}$. Then:

$$\boldsymbol{x_c^*} = \arg\min_{\boldsymbol{x}} \max\{\dots e_k(\boldsymbol{x}) \dots\} \qquad (4\text{-}6)$$
$$s.t.\, 87° < x_2 < 93°$$

$$e_k = \begin{cases} R_{c,k}(\boldsymbol{x}) - S_{reflect} & \text{for all } k \in \boldsymbol{I}^{ReflectL} \\ S_{resonate} - R_{c,k}(\boldsymbol{x}) & \text{for all } k \in \boldsymbol{I}^{Resonate} \end{cases} \qquad (4\text{-}7)$$

Where:

- $S_{reflect} = -3.1\ dB$
- $S_{resonate} = -9.12\ \ dB$
- $\boldsymbol{I}^{ReflectL} \subset \boldsymbol{I}^{sweep},\ f_k \in f_{sweep} \mid f_k \le 0.7\ GHz$
- $\boldsymbol{I}^{Resonate} \subset \boldsymbol{I}^{sweep} \mid f_k \in f_{sweep},\ 0.95\ GHz \le f_k \le 1.05\ GHz$:

Note how:

- The resonance band is thin, it contains only a few values around the design specification.
- $S_{reflect}$ & $S_{resonate}$ are loose values compared to the specification, the response at the resonance frequency should be -13 dB.
- We only used a reflect band to the left of the resonance frequency. The response is asymmetric around the resonance frequency; it has greater reflection losses at

frequencies lower than the resonant frequency. This modification allowed the optimization algorithm to converge to the specification value.

Optimization of the minimax formulation in equation (4-6) yields the optimal coarse response in Figure 4-9. It meets the design specifications: resonance frequency is at 1GHz with response lower than -13 dB. This is now the target response as formulated in equation (4-8).

$$\boldsymbol{R}_T = \boldsymbol{R}_c(\boldsymbol{x}_c^*) \tag{4-8}$$

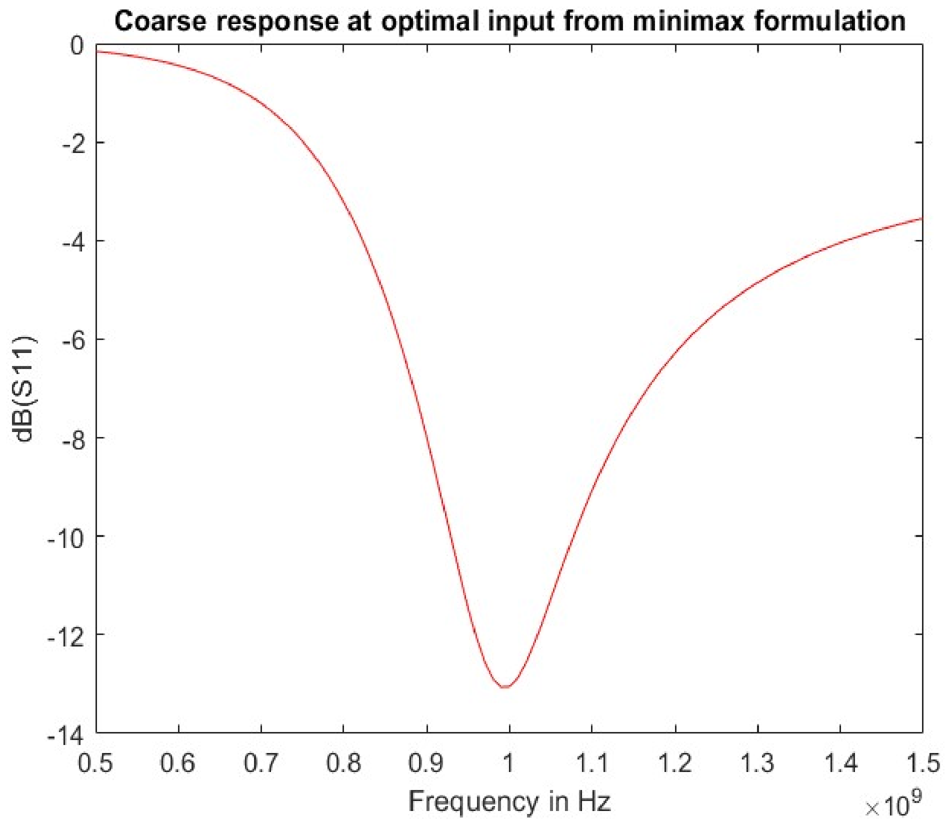


Figure 4-9 Coarse response at optimal input from minimax formulation. This is the target response.

Optimization time	389 seconds
Objective function evaluations	99
Iterations of the algorithm	46
Optimization result	$\boldsymbol{x}_c^* = (0.0824,1.6174)$
Objective function value	0.411
Output message	The current x satisfies the termination criteria using TolX of 1.000000e-05 and F(X) satisfies the convergence criteria TolFun of 1.000000e-04

Figure 4-10 shows the effect of the minimax formulation on the response, it moved the resonant frequency to the design specification of 1GHz.

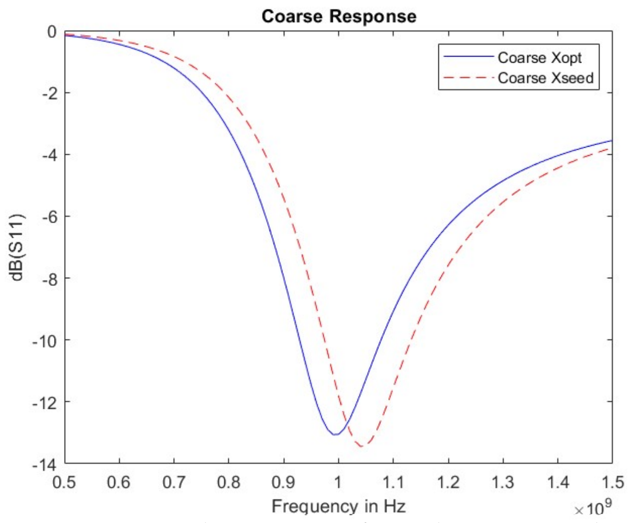


Figure 4-10 The minimax formulation moved the resonant frequency to the design specification of 1 GHz

Figure 4-11 shows the fine and target responses evaluated at the coarse optimum input $\boldsymbol{x}_c^*$. The space mapping algorithm will determine a fine response that complies with the design specification

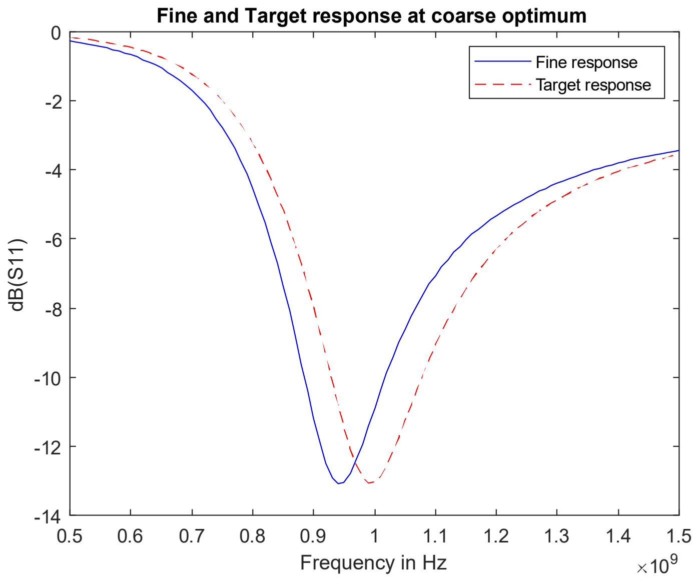


Figure 4-11 Fine response at coarse optimum and target response.

4.6. Broyden based input space mapping.

The coarse and fine models' inputs refer to fundamentally different geometric characteristics of the antenna. This requires a translation between inputs in equations (4-10) and (4-9). Equation (4-11) formulates the translation. It is implemented as a layer that encapsulates the fine model.

$$\boldsymbol{x_c} = (x_{c1}, x_{c2}) \qquad (4\text{-}9)$$

- is the vector of design parameters:
  - x_{c1}: Bowtie's length in meters
  - x_{c2}: Flare angle in radians.
  - See Figure 4-1 .

$$\boldsymbol{x_f} = (x_{f1}, x_{f2}) \qquad (4\text{-}10)$$

Where:

- x_{f1}: Arm length in meters
- x_{f2}: Outer width in meters.
- . See Figure 4-4

$$\boldsymbol{x_f} = (x_1, x_2) \qquad (4\text{-}11)$$

Where:

- $x_1 = \frac{x_{c1}}{2}$
- $x_2 = 2x_{c1}\cos\left(\frac{x_{c2}}{2}\right)$

A final consideration before the implementation of the space mapping algorithm is how to drive 2 different simulators from the same script. This is accomplished by a generalized Matlab™ to Ansys™ EM2019 HFSS driver. See details in appendix D.

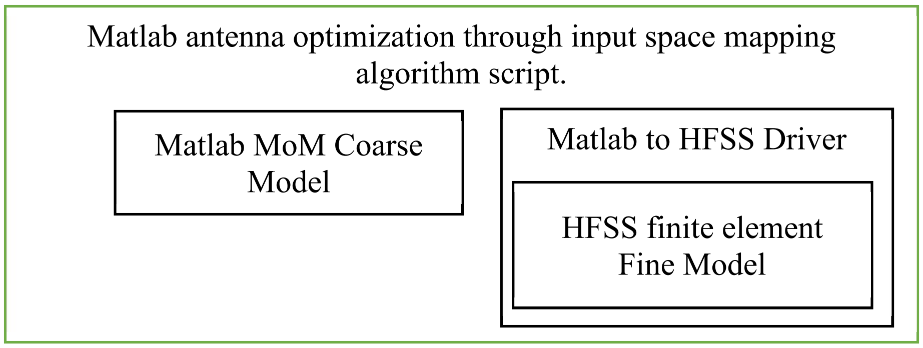

Matlab antenna optimization through input space mapping algorithm script.
Matlab MoM Coarse Model
Matlab to HFSS Driver
HFSS finite element Fine Model

Figure 4-12 Space mapping implementation in Matlab

Finally, the implementation of the Broyden based input space mapping algorithm as described in Figure 3-6 produces a fine response that complies with the design specification in only 4 iterations of the fine model.

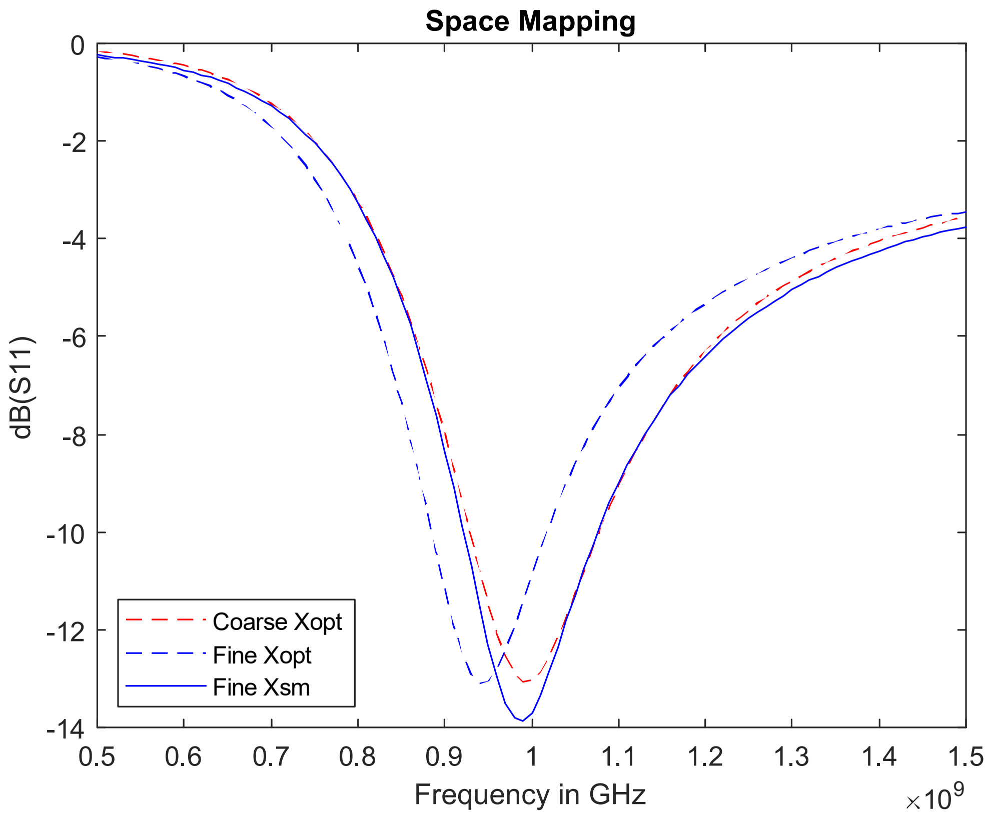


Figure 4-13 Fine response at space mapping optimal input that complies with design specification. Comparison with fine and coarse response at coarse optimal input.

Execution time	1,112 seconds
Coarse model evaluations	217
Fine model evaluations	4
	$\boldsymbol{x}_{SM}^{*} = (0.0785, 1.6174)$

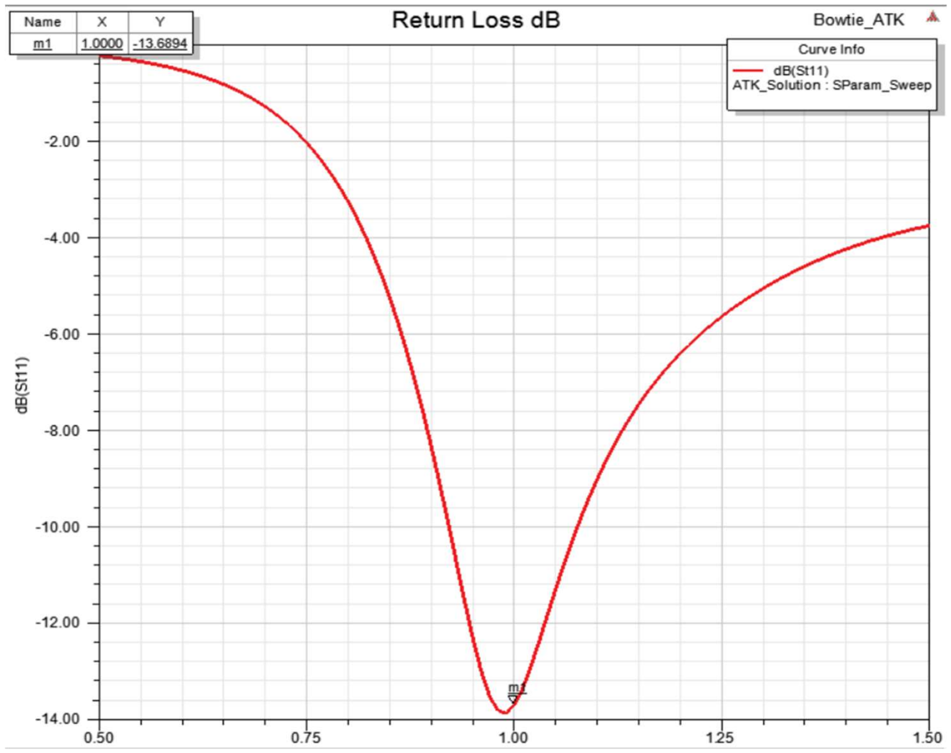


Figure 4-14 Bowtie dipole antenna S11 in dB plot for the 0.5GHz - 1.5GHz band. This is the fine response from HFSS.

5. Microstrip Patch Antenna Optimization Through Input Space Mapping.

The implementation of the antenna optimization through input space mapping algorithm applied to a microstrip patch antenna printed on a PCB is presented. Refer to the antenna's characteristics discussed in section 1.4. The development of the coarse and fine models is discussed and a deep dive on the specifics needed for the successful implementation of the algorithm is provided.

We consider different characteristics of the antenna responses such as radiation pattern, 2D gain shape and our figure of merit waveform, to be a consequence of the antenna type selection. Hence, these are not part of the optimization objective.

Our objective is to design an antenna with resonance frequency at 1 GHz and reflection loss at the resonance frequency that is not greater than -13 dB.

5.1. Coarse model.

A coarse mesh Matlab™ MoM is used as the coarse model. The model is built as follows:

We start by creating a *patchMicrostrip* Matlab™ object from the Antenna Toolbox [2]. It has the following parameters: length, width, height, ground plane length, ground plane width and feed location, see Figure 5-1 .

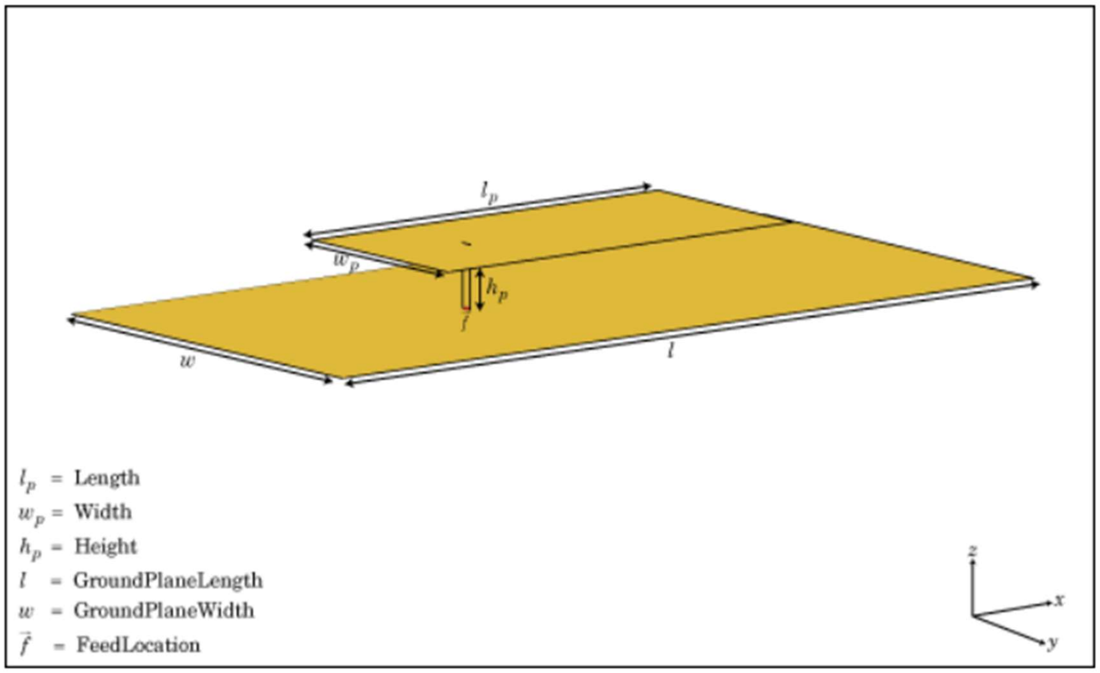


Figure 5-1 Coarse model parameters for patchMicrostrip object from antenna toolbox. Picture taken from [2].

Then we apply a coarse mesh MoM simulation to the *patchMicrostrip* object. We define the mesh parameter *m* in equation (5-1). It will enable us to fine tune the coarseness of the model and hence balance its computational demands against its accuracy.

$$m = \frac{\lambda}{\eta} = \frac{c}{f_{max}\eta} \tag{5-1}$$

Where:

- λ is the minimum wavelength of interest in the simulation.
- c is the speed of light.
- f_{max} is the maximum frequency of interest in the simulation.
- η is an arbitrary parameter to modify the mesh edge length in terms of the maximum frequency of interest in the simulation.

We choose 2 geometric parameters as our input parameters; the rest are considered exogenous parameters. The coarse model for the bowtie antenna parameter becomes:

$$\boldsymbol{R_c}(\boldsymbol{x_c}, \boldsymbol{\varphi_c}) = |\boldsymbol{S_{11}}| \tag{5-2}$$

Where:

- $\boldsymbol{x_c} = (x_1, x_2)$ is the vector of design parameters:
    - x_1- length in meters.
    - x_2 - width in meters.
    - See Figure 5-1.
- $\boldsymbol{\varphi_c} = (\varphi_1, \dots, \varphi_5\ )$ is the vector of exogenous parameters:
    - $\varphi_1 = 4$ - Height in millimeters.
    - $\varphi_2 = 2x_1$ - Ground plane length.
    - $\varphi_3 = 2x_2$ - Ground plane width.
    - $\boldsymbol{\varphi}_4 = (0,0)$ - Feed location in cartesian coordinates in meters.
    - $\varphi_5 = \ 1.9986$ - Mesh parameter *m.*
    - $\varphi_6 = 51$ - Number of frequency points in the simulation.
    - $\varphi_7 = 1$ - Substrate relative permittivity.

Adequate values for φ_5 and φ_6 are determined through a parameter sweep. We get a coarse model that is both computationally inexpensive yet accurate at $\eta = 1x10^{-3}$ and 51 frequency points. Simulation takes a proximately 10 seconds running on a personal computer.

5.2. Input parameter seed.

We apply Matlab™ antenna toolbox *design()* function to our microstrip patch antenna object and get the seed in equation (5-3) as we solve for a resonance frequency of 1GHz. This function implements a first order model for antenna design which is too coarse to use in our algorithm.

$$\boldsymbol{x}_{\boldsymbol{seed}} = (146{,}150) \tag{5-3}$$

The coarse response at the input seed, $\boldsymbol{R_c}(\boldsymbol{x_{seed}}, \boldsymbol{\varphi_c})$, is shown in Figure 5-2. Note that the first order model produces a great seed for our coarse model optimization step. Even though the coarse model is not tuned for accuracy, it is still more accurate than the first order model. This is noted by the fact that the resonance frequency is not located at 1GHz. The magnitude of the response complies with the design specifications.

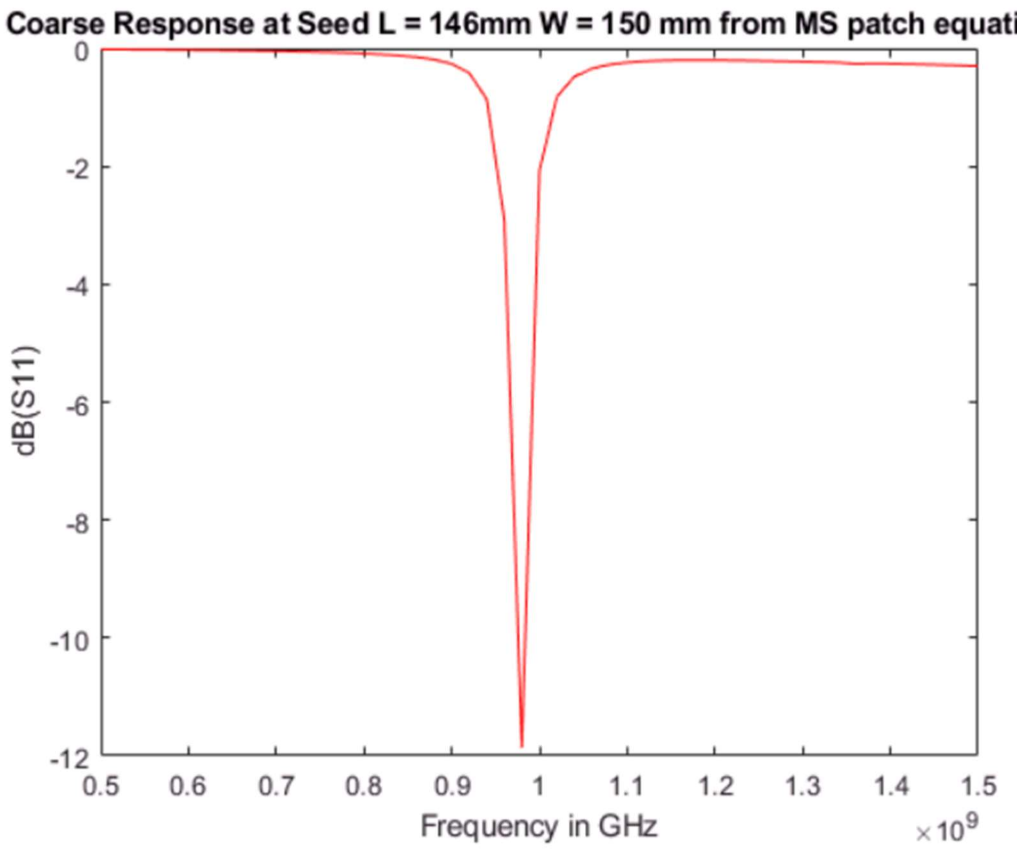


Figure 5-2 Coarse model evaluated at seed from first order model.

5.3. Fine model.

The fine model implementation of a dipole bowtie antenna in Ansys™ EM desktop 2019 is presented. The Ansys Antenna Toolkit [1] is used to generate the antenna geometric model. It uses 7 parameters to build the geometric model: patch dimension X, patch dimension Y, substrate dimension x, substrate dimension y, substrate thickness, feed width, inset gap, feed length and inset distance. See Figure 5-3

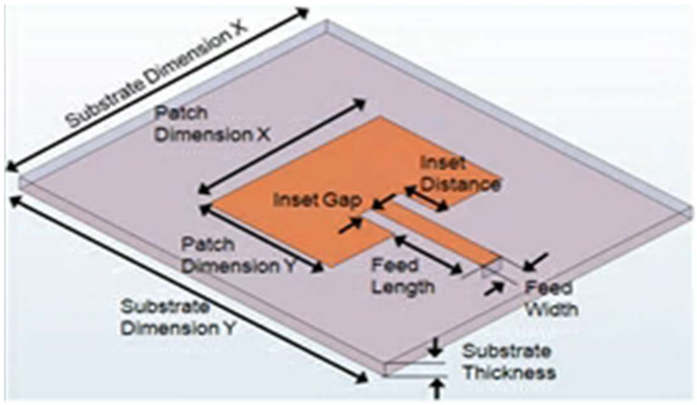


Figure 5-3 Microstrip patch antenna design parameters. Taken from [1].

Two geometric dimensions are chosen as the input parameters; the rest are considered exogenous parameters. Namely, the fine model for the microstrip patch antenna becomes:

$$\boldsymbol{R_f}(\boldsymbol{x_f}, \boldsymbol{\varphi_f}) = |\boldsymbol{S_{11}}| \qquad (5\text{-}4)$$

Where:

- $\boldsymbol{x_f} = (x_1, x_2)$ is the vector of design parameters:
  - x_1 - Patch dimension X in millimeters
  - x_2 - Patch dimension Y in millimeters.
  - See Figure 5-3.
- $\boldsymbol{\varphi_f} = (\varphi_1, \dots, \varphi_5\ )$ - is the vector of exogenous parameters:
  - $\varphi_1 = 4x_2$ - Substrate dimension X in meters.
  - $\varphi_2 = 4x_1$ - Substrate dimension Y in meters.
  - $\varphi_3 =\ 2$ - Substrate thickness in millimeters.
  - $\varphi_4 = 0.0001$ - Inset distance in meters.
  - $\varphi_5 = 0.0001$ - Inset gap in meters.
  - $\varphi_8 =\ 0.02$ - Feed width in meters
  - $\varphi_7 = 1$ - Substrate relative permittivity.
  - $\varphi_8 = 0.010$ - Feed length in meters.

- $\varphi_9 = 51$ - Number of frequency points in the simulation.

Even though φ_1 and φ_2 are not constants determined exogenously, they are refreshed in every iteration of the fine model, they are not considered part of the input parameters. Their only purpose is to create a PCB substrate large enough to model the antenna during the simulation. They have no significant influence on the response of the antenna other than enabling the simulation to run.

The geometric model renders Figure 5-4.

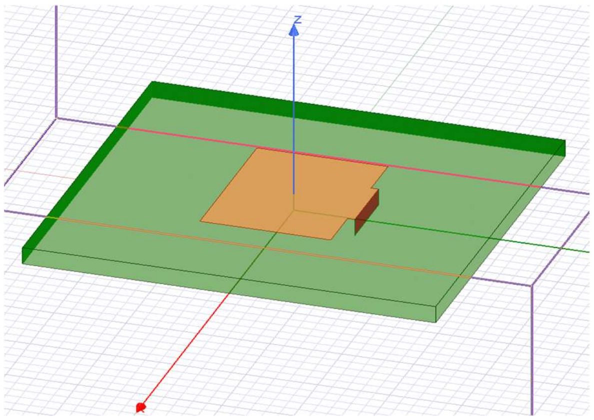


Figure 5-4 Microstrip patch antenna geometry model

The fine response at the input seed, $\boldsymbol{R_f}(\boldsymbol{x_{seed}}, \boldsymbol{\varphi_f})$, is shown in Figure 5-5. Note that both response magnitude and waveform are sufficiently similar to the coarse model after the coarse model was adjusted, which is a requisite for the rest of the algorithm to work. We adjusted the coarse model substrate to achieve this. See section 4.4 for details on how this was addressed.

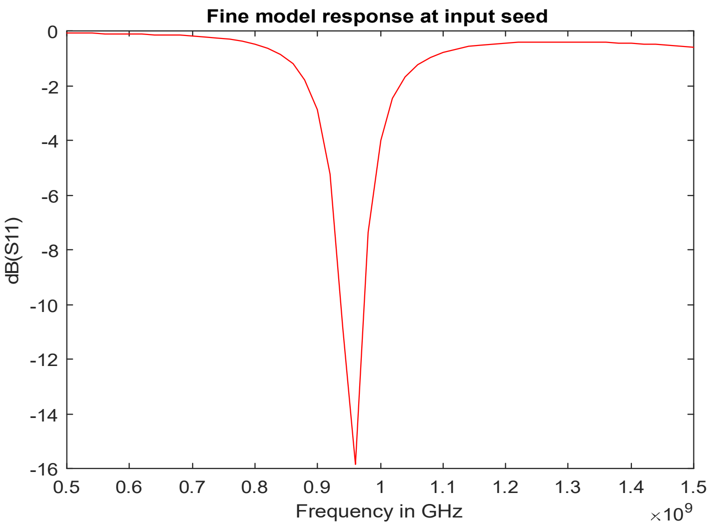


Figure 5-5 Fine model response at input seed.

As mentioned in the chapter's introduction, other responses from the model such as the antenna radiation pattern in Figure 5-6 and the shape of the 2D gain in Figure 5-7 are considered consequences of the antenna type selection and are not part of the optimization objectives.

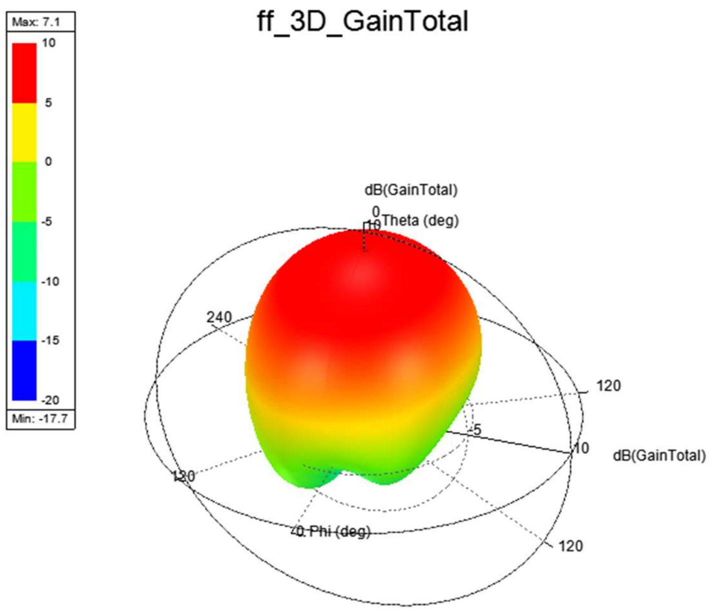


Figure 5-6 Microstrip patch antenna radiation pattern

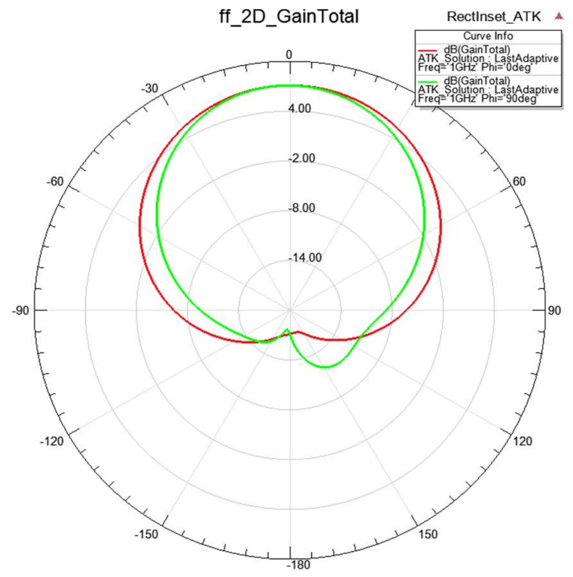


Figure 5-7 Microstrip patch antenna 2D

5.4. Fine tune coarse model.

It is imperative for the rest of the algorithm to work that the waveforms and magnitudes of the coarse and fine models are sufficiently similar. We recommend an iterative process in every step of the algorithm to confirm this continues to be true.

In this case, we had to adjust the coarse model substrate height to be twice that of the fine model to obtain a response with sufficiently similar magnitude. This was determined after several parameter sweeps. The relation is shown in equation (5-5).

$$\varphi_{c1} = 2\varphi_{f3} \tag{5-5}$$

In Figure 5-8 we show the similar magnitudes and waveforms required for successful implementation of the algorithm.

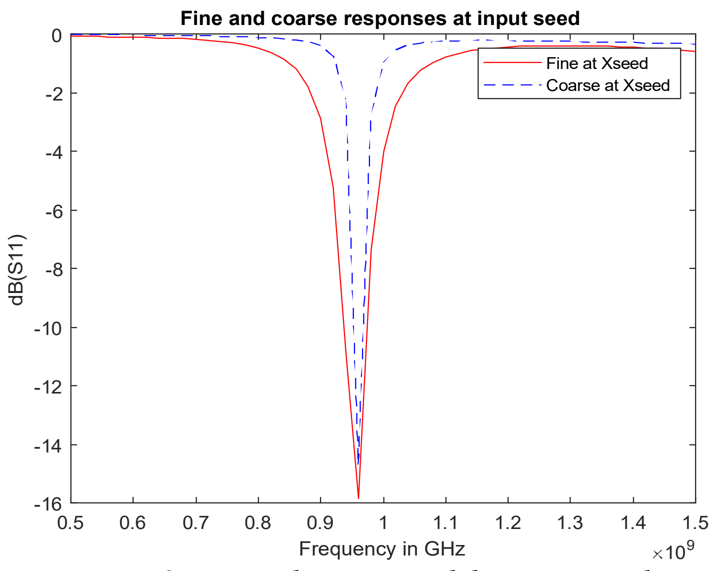


Figure 5-8 Fine and coarse model at input seed showing similar magnitudes and waveforms.

5.5. Coarse model optimization.

We obtain the target response $\boldsymbol{R_t}$ from the optimization of the coarse model. We propose the minimax formulation in equation (5-6) with the objective function in (5-7).. After several iterations the specific formulation values follow:

Let the vector f_{sweep} be the 51 frequencies, equally spaced between 0.5 GHz and 1.5 GHz, at which the simulation is executed. Let $\boldsymbol{I}^{sweep}$ be the set of all indices of f_{sweep}, which correspond exactly with the indices of coarse response $\boldsymbol{R_c}(\boldsymbol{x_c}, \boldsymbol{\varphi_c})$. Such that, $\boldsymbol{I}^{ReflectL}$, $\boldsymbol{I}^{Resonate}$ and $\boldsymbol{I}^{ReflectH}$ are disjoint subsets of $\boldsymbol{I}^{sweep}$. Then:

$$\boldsymbol{x_c^*} = \arg\min_{\boldsymbol{x}} \max\{\dots e_k(\boldsymbol{x}) \dots\} \qquad (5\text{-}6)$$

$$e_k = \begin{cases} R_{c,k}(\boldsymbol{x}) - S_{reflect} & \text{for all } k \in \boldsymbol{I}^{ReflectL} \\ S_{resonate} - R_{c,k}(\boldsymbol{x}) & \text{for all } k \in \boldsymbol{I}^{Resonate} \\ R_{c,k}(\boldsymbol{x}) - S_{reflect} & \text{for all } k \in \boldsymbol{I}^{ReflectH} \end{cases} \qquad (5\text{-}7)$$

Where:

- $S_{reflect} = -0.91\ dB$
- $S_{resonate} = -7.95\ \ dB$
- $\boldsymbol{I}^{ReflectL} \subset \boldsymbol{I}^{sweep},\ f_k \in f_{sweep} \mid f_k < 0.99\ GHz$
- $\boldsymbol{I}^{Resonate} \subset \boldsymbol{I}^{sweep} \mid f_k \in f_{sweep},\ 0.99\ GHz \le f_k \le 1.01\ GHz$:
- $\boldsymbol{I}^{ReflectL} \subset \boldsymbol{I}^{sweep},\ f_k \in f_{sweep} \mid f_k > 1.01\ GHz$
-

Note how:

- The resonance band is thin, it contains only a few values around the design specification.
- $S_{reflect}$ & $S_{resonate}$ are loose values compared to the specification, the response at the resonance frequency should be -13 dB.

Optimization of the minimax formulation in equation (5-6)(4-6) yields the optimal coarse response in Figure 5-9. It meets the design specifications: resonance frequency is at 1GHz with response lower than -13 dB. This is now the target response as formulated in equation (5-8).

$$\boldsymbol{R}_T = \boldsymbol{R}_c(\boldsymbol{x}_c^*) \qquad (5\text{-}8)$$

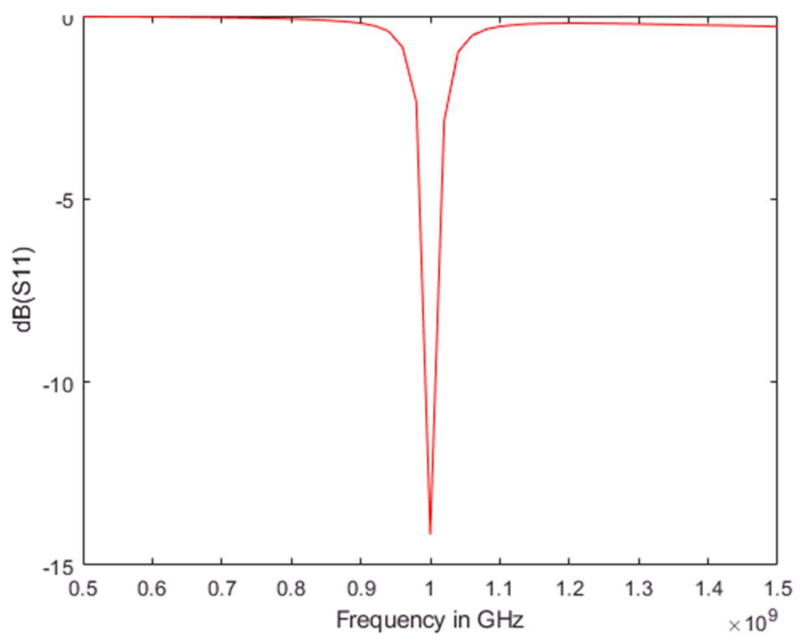


Figure 5-9 Coarse model response at optimum from minimax formulation.

Optimization time	694 seconds
Objective function evaluations	52
Iterations of the algorithm	20
Optimization result	$\boldsymbol{x}_c^* = (140.9,147.5)$
Objective function value	-0.0671
Output message	The current x satisfies the termination criteria using TolX of 1e-03 and F(X) satisfies the convergence criteria TolFun of 1e-04

Figure 5-10 Coarse model response at optimum from minimax formulation compared to seed. shows the effect of the minimax formulation on the response, it moved the resonant frequency to the design specification of 1GHz.

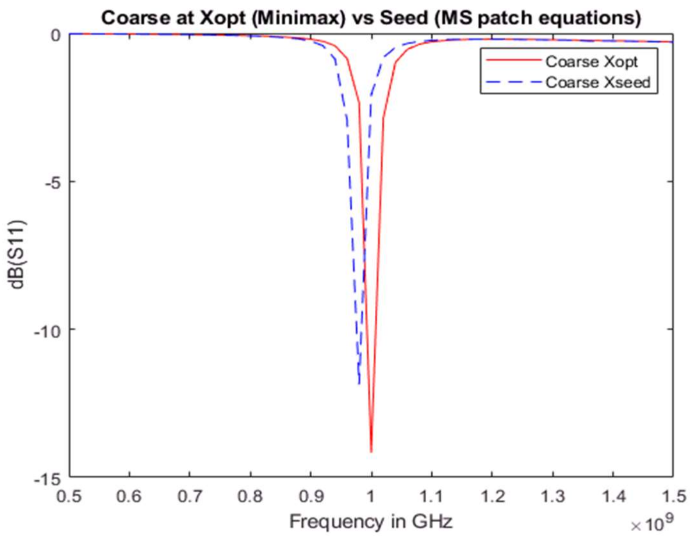


Figure 5-10 Coarse model response at optimum from minimax formulation compared to seed.

Figure 5-11 shows the fine and target responses evaluated at the coarse optimum input $\boldsymbol{x}_c^*$. The space mapping algorithm will determine a fine response that complies with the design specification

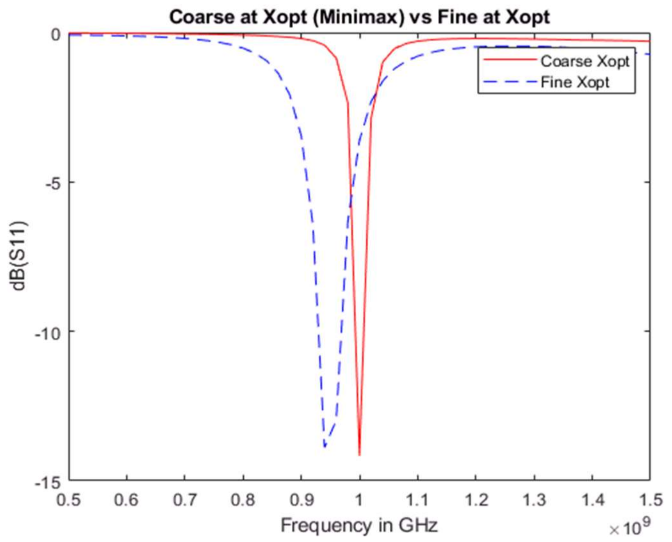
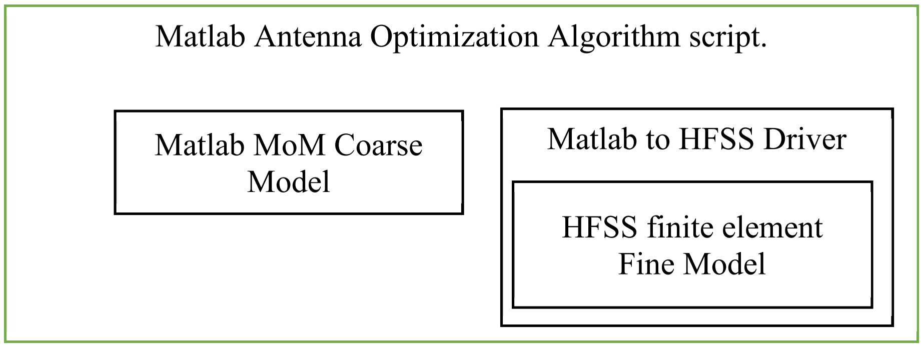


Figure 5-11 Fine and coarse models compared at optimum from minimax formulation.

5.6. Broyden based input space mapping.

A final consideration before the implementation of the space mapping algorithm is how to drive 2 different simulators from the same script. This is accomplished by a generalized Matlab™ to Ansys™ EM2019 HFSS driver, the code can be found in appendix D.


Matlab Antenna Optimization Algorithm script.
Matlab MoM Coarse Model
Matlab to HFSS Driver
HFSS finite element Fine Model

Finally, the implementation of the Broyden based input space mapping algorithm as described in Figure 3-6 produces a fine response that complies with the design specification in only 3 iterations of the fine model. All the responses are shown in Figure 5-12.

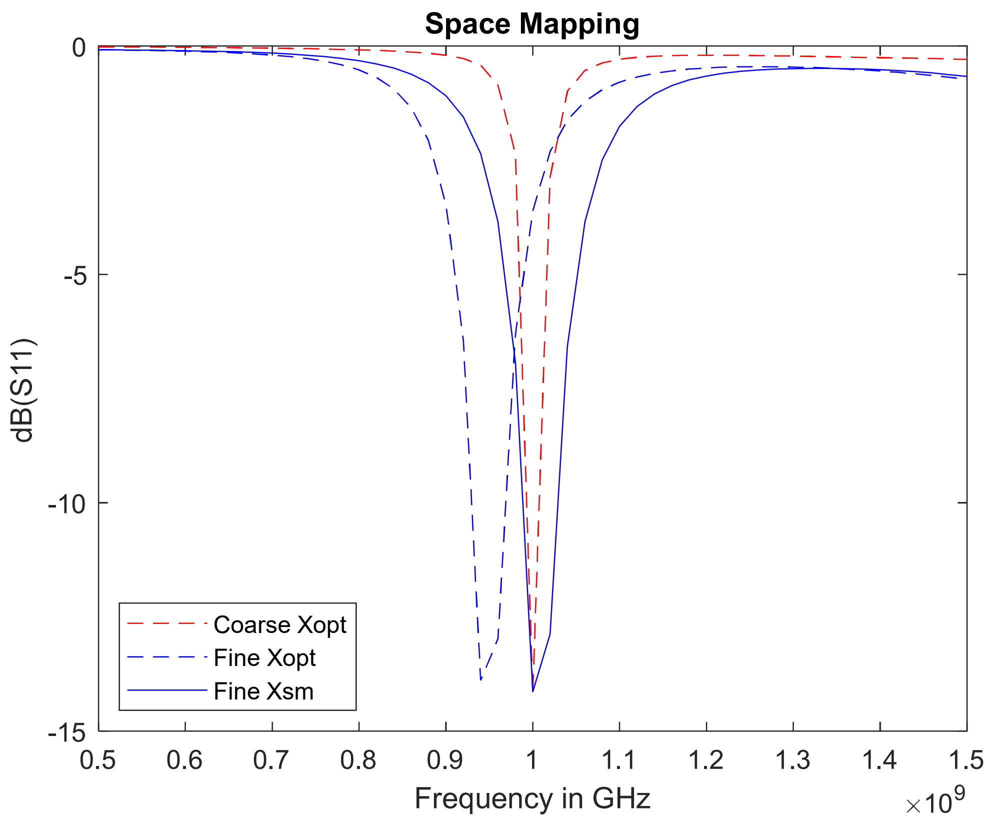


Figure 5-12 Space mapping implementation in Matlab

Execution time	1,112 seconds
Coarse model evaluations	135
Fine model evaluations	3
Space Mapping input	$\boldsymbol{x}_{SM}^{*} = (141.2,145.8)$

Conclusions.

This thesis has presented an innovative antenna optimization through Broyden based input space mapping algorithm. The algorithm is mathematically formulated and developed in detail for each step. We provide recommendations and learnings based on the implemented examples for the generalized algorithm.

We have shown that focus on creating an effective coarse model is the key to a straightforward implementation of the Broyden based input space mapping algorithm. We provide several recommendations on each step and criteria to move forward. The authors show a programmatic way to create a parameter sweep in order to find a coarse mesh that is effective for space mapping implementation.

Creating a target response that is sufficiently similar in magnitude and waveform to the fine response is fundamental to the successful implementation of the antenna optimization algorithm. The authors showed how a minimax formulation with loose engineering specifications and thin response bands is enough to obtain an adequate target response.

The authors consider that this work can be extended in the following ways:

- Create a response similarity criterion that is based on the p^{th}-norm of the difference of the coarse and fine responses. This work relies on parameter sweeps that can be time consuming and computationally expensive. The weakest point of the parameter sweep is that it relies on the expertise of the designer on antenna theory and simulation techniques. A p^{th}-norm approach will enable the use of optimization techniques and heuristics to fine tune the coarse model regardless of designers ability on antenna theory.
- Formulate and develop a version of the algorithm that implements neuro space mapping as in [3] [16]. This will necessarily provide new recommendations at each

step of the algorithm. It will introduce robustness to the formulation of the coarse response.

- Formulate and implement versions of the algorithm with different versions of the space mapping algorithm, such as output space mapping as in [4] or linear inverse input space mapping as in [14].
- Develop the application of the algorithm for all the antenna types available in Matlab™ and HFSS™ antenna toolboxes.
- Develop a Matlab™ driver to other commercial 3D and Multiphysics simulators such as Comsole™ for use with the antenna optimization algorithm.
- Develop and formulate similar specific algorithms for other radio frequency circuits such as filters or impedance couplings.

Appendix.

A. MICROSTRIP PATCH ANTENNA CODE.

Matlab™ and Python™ code used to optimize Microstrip Patch Antenna.

- **Matlab™ MoM Coarse Model**

```
function S11 = PatchCoarse(x)
% Usage: S11 = PatchCoarse(x)
% x: array containing Patch length in meters and patch width in meters.
% x = [Patch Length in m, Patch Width in meters]
% S11: returns abs() of S11 parameters of Microstrip patch Antenna for fSweep
frequency values
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-05-17
%% Create patch antenna using patchL and patchW
    patchL = x(1);%m
    patchW = x(2);%m
    patchAntenna = patchMicrostrip;
    patchAntenna.Length = patchL;
    patchAntenna.Width = patchW;
    patchAntenna.Height = 4e-3;
    patchAntenna.GroundPlaneLength = 2*patchL;
    patchAntenna.GroundPlaneWidth = 2*patchW;

%adjust mesh to create a coarse model
    fSweep = 500e6:20e6:1.5e9;
    fmax = max(fSweep);
    n=1e-3;
    lambda = physconst('LightSpeed')/fmax;
    mesh (patchAntenna,'MaxEdgeLength',   lambda/n);

%% Calculate S11
    S = sparameters(patchAntenna,fSweep);
    S11 = abs(S.Parameters(:));

end
```

- **Minimax Objective Function**

```
function MaxError = OF_patchCoarseOptim(x)
% Usage: S11 = OF_bowtieCoarseOptim(x)
% x: array containing Patch length in meters and  patch width in radians.
% x = [Patch Length in m, Patch Width in meters]
% Objective function for Minimax optimization program. Retuns the maximum
% error as per criteria below.
% Requires PatchCoarse()
% Maxerror returns the max of e1 and e2.
% e1 contains the error for the resonance band.
% e2 contains the error for the reject band.
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-05-17

% Get coarse reponse
S11 = PatchCoarse(x);

%% Calculate error functions
%Problem set up
S11resonate = 0.4;%aprox -6dB
S11reflect = 0.9;
%resonate band freq
fresonateL = 0.99e+9;
fresonateH = 1.01e+9;
%reflect band freq
freflectH = 1.1e9;
freflectL = 0.9e9;

e1 = []; %initialize error functions
e2 = []; %initialize error functions

FP = length(S11);
f = 500e6:20e6:1.5e9;
for k = 1:FP
    if (f(k) >= fresonateL) && (f(k) <= fresonateH)
        e1n = S11(k) - S11resonate;
        e1 = [e1 e1n];
    elseif (f(k) >= freflectH) || (f(k) <= freflectL)
        e2n = S11reflect - S11(k);
        e2 = [e2 e2n];
    end
end

% Obj fn value
e = [e1 e2];
MaxError = max(e);

end
```

- **Nelder-Mead Optimization of Minimax objective function**

```
%% optimize coarse model
tic
%define constants for optim algo
    epsf = 1e-4;
    epsx = 10e-4;
    epsg = 1e-4;
    MaxIter = 1e3;
    MaxfunEval = .5e3;

    algo = 'Nelder-Mead';
    objfun ='OF_patchCoarseOptim' ;
    x = [144e-3 150e-3]; % from microstrip patch antenna calc
%'PlotFcns',@optimplotfval
    options = optimset('MaxFunEvals',MaxfunEval,'MaxIter',Max-
Iter,'TolX',epsx,'PlotFcns','optimplotfval');
     [Xopt,FunVal,EF,output] = fminsearch(objfun,x,options);
toc
```

- **Fine model**

```
function [Rf] = PatchFine(x)
% Usage: S11 = PatchFine(x)
% x: array containing Patch length in meters and  patch width in meters.
% x = [Patch Length in m, Patch Width in meters]
% % S11: returns abs() of S11 parameters of Microstrip Patch Antenna for
fSweep
% frequency values. Uses MATLAB2HFSS() driver.
% Requires that HFSSpyScriptTemplate has a valid path to an HFSS script
% template.
% Requires that HFSSPath contains  a valid path to ansysedt.exe.
% Requires valid HFSS parametrized design referenced in HFSSpyScriptTemplate
% Requires HFSS 19 or later
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-05-17

HFSSpyScriptTemplate = 'HFSSscripttemplate.py';
       HFSSPath = '"C:\Program Files\AnsysEM\AnsysEM19.1\Win64\an-
sysedt.exe"';
       x = x * 1e3;
       %convert x to strings for MATLAB driver

            xd = {strcat('"',num2str(x(1)),'mm','"'),... patchL
                  strcat('"',num2str(x(2)),'mm','"'),... patchW
               };
        %param line location to modify template
        %Lines =[SubSizeY=21,SubSizeX=39, FlareLegth=57]
        lines =[21 39];

        %Simulate using MATLAB2HFSS driver
        FineR = MATLAB2HFSS(xd,lines,HFSSPath,HFSSpyScriptTemplate);
        Rf = FineR(:,2);
end
```

- **Python template for HFSS driver**

```
import ScriptEnv
ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")
oDesktop.RestoreWindow()
oDesktop.OpenProject("C:\Users\emichel\Desktop\personal\mde\CircOpt\Final\HFSS\Patch\Patch.aedt")
oProject = oDesktop.SetActiveProject("Patch")
oDesign = oProject.SetActiveDesign("HFSSDesign1")
oDesign.ChangeProperty(
	[
		"NAME:AllTabs",
		[
			"NAME:LocalVariableTab",
			[
				"NAME:PropServers",
				"LocalVariables"
			],
			[
				"NAME:ChangedProps",
				[
					"NAME:patchL",
					"Value:="			, "140.9337mm"
				]
			]
		]
	])
oDesign.ChangeProperty(
	[
		"NAME:AllTabs",
		[
			"NAME:LocalVariableTab",
			[
				"NAME:PropServers",
				"LocalVariables"
			],
			[
				"NAME:ChangedProps",
				[
					"NAME:patchW",
					"Value:="			, "140.4739mm"
				]
			]
		]
	])
oProject.Save()
oDesign.Analyze("1GHz")
oModule = oDesign.GetModule("ReportSetup")
oModule.ExportToFile("S Parameter Plot 1",
"C:/Users/emichel/Desktop/personal/mde/CircOpt/Final/HFSSoutput.csv")
oProject.Save()
```

- **Input Space Mapping**

```
% Usage: Patch Antenna optimization through input space
% mapping.
% % Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-05-17
 %% PatchAntenna
clear;
clc;
format shorteng;
%%
%coarse opt
load('final')
Xoptc = Xopt';%degrees **must be col vector
%Initialize B
B = eye(length(Xoptc));
%initialize Xf
Xf = Xoptc;
Xccheck = Xoptc;
%initialize k
k = 0;
kmax = length(Xf);
%initialize EF
EF = 0;
%initialize Rt
Rtarget = PatchCoarse(Xoptc);
Rtarget = 20*log10(Rtarget);
Rf = PatchFine(Xf);
Rfinit = 20*log10(Rf);
Xc = SPE('PatchCoarse',Rf,Xf);
f = Xc - Xoptc;
EF = SMstop(f);
%%
    while (EF == 0 && k <= kmax)
        h = -B\f;
        Xf = Xf+h;
        Rf = PatchFine(Xf);
        Xc = SPE('PatchCoarse',Rf,Xf);
        f = Xc - Xoptc;
        EF = SMstop(f);
        B = B + (f * h')/(h' * h);
        k=k+1;
        Xccheck = [Xccheck Xc];
    end
    %%

    RfSM = 20*log10(Rf);
        %%
    plot(fSweep,Rtarget,'r--',fSweep,Rfinit,'b--',fSweep,RfSM,'b');
     xlabel('Frequency in GHz')
        ylabel('dB(S11)')
        title('Space Mapping')
    legend('Coarse Xopt','Fine Xopt','Fine Xsm','Location','southwest');
```

B. DIPOLE BOWTIE ANTENNA CODE.

- **Matlab™ MoM Coarse Model for Minimax optimization (constrained)**

```
function S11 = BowtieCoarse(x)
% Usage: S11 = BowtieCoarse(x)
% x: array containing Bowtie length in meters and  Flare Angle in radians.
% x = [Bowtie Length in m, FlareAnge in radians]
% BowtieCoarse introduces a contraint on angle values to help optmization
% with Nealder mead algorithm, compare to BowtieCoarseSM that has no such
% contraint.
% S11: returns abs() of S11 parameters of bowtie Antenna for fSweep frequency
values
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-10-27

%% Create Bowtie antenna using bowtieL and bowtieAng
    %x = [0.0794 pi/4];%use to test function
    bowtieL = x(1);%m
    bowtieAng = rad2deg(x(2));%convert Flare angle from radians to degrees

   %constrain Flare angle to be between 5 and 175 deg
   if(bowtieAng < 87)
      bowtieAng = 87;
   elseif (bowtieAng > 93)
      bowtieAng = 93;
   end

   %% Create antenna object

    bowtieAntenna = bowtieTriangular;
    bowtieAntenna.Length = bowtieL;
    bowtieAntenna.FlareAngle = bowtieAng;

%adjust mesh to create a coarse model
    fSweep = 500e6:10e6:1.5e9;
    fmax = max(fSweep);
    n=100e-3;
    lambda = physconst('LightSpeed')/fmax;
    %mesh (bowtieAntenna,'MaxEdgeLength',   lambda/n);

%% Calculate S11
    S = sparameters(bowtieAntenna,fSweep);
    S11 = abs(S.Parameters(:));

end
```

- **Matlab™ MoM Coarse Model for Space Mapping (unconstrained)**

```
function S11 = BowtieCoarseSM(x)
% Usage: S11 = BowtieCoarse(x)
% x: array containing Bowtie length in meters and  Flare Angle in radians.
% x = [Bowtie Length in m, FlareAnge in radians]
% BowtieCoarseSM has no constraints for Flare Angle, compare to
% BowtieCoarse that has Flare angle constraints.
% S11: returns abs() of S11 parameters of bowtie Antenna for fSweep frequency
values
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-10-27

%% Create Bowtie antenna using bowtieL and bowtieAng
   % x = [0.0794 pi/4];%use to test function
    bowtieL = x(1);%m
    bowtieAng = rad2deg(x(2));%convert Flare angle from radians to degrees

   %make sure Flare angle is between 5 and 175 deg
   if(bowtieAng < 5)
       bowtieAng = 87;
   elseif (bowtieAng > 170)
       bowtieAng = 170;
   end

   %% Create antenna object

    bowtieAntenna = bowtieTriangular;
    bowtieAntenna.Length = bowtieL;
    bowtieAntenna.FlareAngle = bowtieAng;

%adjust mesh to create a coarse model
    %fSweep = 0.9e9:5e6:1.1e9;
    fSweep = 500e6:10e6:1.5e9;
    fmax = max(fSweep);
    n=100e-3;
    lambda = physconst('LightSpeed')/fmax;
    %mesh (bowtieAntenna,'MaxEdgeLength',   lambda/n);

%% Calculate S11
    S = sparameters(bowtieAntenna,fSweep);
    S11 = abs(S.Parameters(:));

end
```

- **Minimax Objective Function**

```
function MaxError = OF_bowtieCoarseOptim(x)
% Usage: S11 = OF_bowtieCoarseOptim(x)
% x: array containing Bowtie length in meters and  Flare Angle in radians.
% x = [Bowtie Length in m, FlareAnge in radians]
% Objective function for Minimax optimization program. Retuns the maximum
% error as per criteria below.
% Requires BowtieCoarse()
% Maxerror returns the max of e1 and e2.
% e1 contains the error for the resonance band.
% e2 contains the error for the reject band.
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-10-27

%% Get S11 parameter from Bowtie Coarse model
S11 = BowtieCoarse(x);



%% Calculate error functions
%Problem set up
S11resonate = 0.35;%aprox -6dB
S11reflect = 0.7;
%resonate band freq
fresonateL = 0.95e+9;
fresonateH = 1.05e+9;
%reflect band freq
freflectH = 0.7e9;
freflectL = 0.51e9;

e1 = []; %initialize error functions
e2 = []; %initialize error functions

FP = length(S11);
f = 500e6:10e6:1.5e9;
for k = 1:FP
    if (f(k) >= fresonateL) && (f(k) <= fresonateH)
        e1n = S11(k) - S11resonate;
        e1 = [e1 e1n];
    elseif (f(k) >= freflectH) || (f(k) <= freflectL)
        e2n = S11reflect - S11(k);
        e2 = [e2 e2n];
    end
end

% Obj fn value
e = [e1 e2];
MaxError = max(e);

end
```

- **Nelder-Mead Optimization of Minimax objective function**

```
%% optimize coarse model
tic
%define constants for optim algo
    epsf = 1e-5;
    epsx = 1e-5;
    epsg = 1e-5;
    MaxIter = 1e3;
    MaxfunEval = 1e3;
%
    objfun ='OF_bowtieCoarseOptim' ;


    Xseed = [0.0794 pi/2]; % from matlab antenna design tool
%'PlotFcns',@optimplotfval
    options = optimset('MaxFunEvals',MaxfunEval,'MaxIter',Max-
Iter,'TolX',epsx);
     [Xopt,FunVal,EF,output] = fminsearch(objfun,Xseed,options);
     output.message
  toc
```

- Seed for Nelder mead optimization.

```
      % Seed
         design(bowtieTriangular,1e9)

    ans =

      bowtieTriangular with properties:

            Length: 0.0794
        FlareAngle: 90
              Tilt: 0
          TiltAxis: [1 0 0]
              Load: [1×1 lumpedElement]
```

- **Fine model**

```
function [Rf] = BowtieFine(x)
% Usage: S11 = BowtieFine(x)
% x: array containing Bowtie length in meters and  Flare Angle in radians.
% x = [Bowtie Length in m, FlareAnge in radians]
% % S11: returns abs() of S11 parameters of bowtie Antenna for fSweep
% frequency values. Uses MATLAB2HFSS() driver.
% Make sure HFSSpyScriptTemplate has a valid path to an HFSS script
% template.
% Make sure HFSSPath contains  a valid path to ansysedt.exe.
% Requires HFSS 19 or later
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-10-27
      %%
       %x = [0.08 90]; %[BowtieLength FlareAngle] used to test module

       HFSSpyScriptTemplate = 'HFSSscripttemplate.py';
       HFSSPath = '"C:\Program Files\AnsysEM\AnsysEM19.3\Win64\an-
sysedt.exe"';

       % convert Bowtie L from coarse model to Arm Length for fine model.
       x(1) = x(1)/2;
       % convert flare angle from coarse model to outer width for fine model
       x(2) = 2 * x(1) * cos(x(2)/2); % meters
       % round to 4 digits (tenths of milimiters)
       x = round(x,4);
       %convert x to strings for MATLAB driver

            xd = {strcat('"',num2str(x(1)),'meter','"'),... patchL
                  strcat('"',num2str(x(2)),'meter','"'),... patchW
               };
        %param line location to modify template
        %Lines =[SubSizeY=21,SubSizeX=39, FlareLegth=57]
        lines =[21 39];

        %% Simulate using MATLAB2HFSS driver
        FineR = MATLAB2HFSS(xd,lines,HFSSPath,HFSSpyScriptTemplate);
        Rf = FineR(:,2);%return mag(S11) in column vector
end
```

- **Python template for HFSS driver**

```
import ScriptEnv
ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")
oDesktop.RestoreWindow()
oDesktop.OpenProject("C:/Users/emichel/Desktop/HFSS/tesis/Bowtie/Bowtie_ATK1.aedt")
oProject = oDesktop.SetActiveProject("Bowtie_ATK1")
oDesign = oProject.SetActiveDesign("Bowtie_ATK")
oDesign.ChangeProperty(
        [
                "NAME:AllTabs",
                [
                        "NAME:LocalVariableTab",
                        [
                                "NAME:PropServers",
                                "LocalVariables"
                        ],
                        [
                                "NAME:ChangedProps",
                                [
                                        "NAME:Arm_Length",
                                        "Value:="                       , "0.0392meter"
                                ]
                        ]
                ]
        ])
oDesign.ChangeProperty(
        [
                "NAME:AllTabs",
                [
                        "NAME:LocalVariableTab",
                        [
                                "NAME:PropServers",
                                "LocalVariables"
                        ],
                        [
                                "NAME:ChangedProps",
                                [
                                        "NAME:Outer_Width",
                                        "Value:="                       , "0.0542meter"
                                ]
                        ]
                ]
        ])
oProject.Save()
oDesign.Analyze("ATK_Solution")
oModule = oDesign.GetModule("ReportSetup")
oModule.ExportToFile("Return Loss", "C:/Users/emichel/Desktop/Bowtie/HFSSoutput.csv")
oProject.Save()
```

- **Input Space Mapping**

```
%%test space mapping algo
%% Bowtie Antenna
clear;
clc;
format shorteng;
%%
%coarse opt
load('Bowtie')
Xoptc = Xopt';%degrees **must be col vector
%Initialize B
B = eye(length(Xoptc));
%initialize Xf
Xf = Xoptc;
Xccheck = Xoptc;
%initialize k
k = 0;
kmax = 4;%length(Xf);
%initialize EF
EF = 0;
%initialize Rt
Rtarget = BowtieCoarseSM(Xoptc);
Rtarget = 20*log10(Rtarget);
Rf = BowtieFine(Xf);
Rfinit = 20*log10(Rf);
Xc = SPE('BowtieCoarseSM',Rf,Xf);
f = Xc - Xoptc;
EF = SMstop(f);
%%
    while ( EF == 0 && k <= kmax)%
        h = -B\f;
        Xf = Xf+h;
        Rf = BowtieFine(Xf);
        Xc = SPE('BowtieCoarseSM',Rf,Xf);
        f = Xc - Xoptc;
        EF = SMstop(f);
        B = B + (f * h')/(h' * h);
        k=k+1;
        Xccheck = [Xccheck Xc];
    end
    %%

    RfSM = 20*log10(Rf);
        %%
    plot(fSweep,Rtarget,'r--',fSweep,Rfinit,'b--',fSweep,RfSM,'b');
     xlabel('Frequency in GHz')
        ylabel('dB(S11)')
        title('Space Mapping')
    legend('Coarse Xopt','Fine Xopt','Fine Xsm','Location','southwest');
```

C. SPACE MAPPING ANCILLIARY CODE.

Matlab™ code needed by space mapping implementation in appendixes A and B.

- **Parameter extraction**

```
function xspe = SPE(R,Rt,x0)
% Usage:  xspe = SPE(R,Rt,x0
% Statistical (optional, comment/uncomment lines 20 to 35) parameter extrac-
tion
% R: Response from Space mapping.
% Rt: Target response
% requires that R and Rt are the same array size,
% x0: seed for optmization program.
% xspe: returns parameter extraction input.
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-05-17

    %initialize parameters for optimization
    epsx = 1e-3;
    MaxIter = 1e3;
    MaxfunEval = 3;
    options = optimset('MaxFunEvals',MaxfunEval,'MaxIter',Max-
Iter,'TolX',epsx);

    %create objective function for param extraction
     PE = @(x)norm((feval(R,x)- Rt));
     xspe = fminsearch(PE,x0,options);

    %initialize parameters for statistical parameter extraction control loop
    epsPE = 1e-2;%Acceptable PE error
    Na = 2;%max attempts to escape poor local min
    alpha = 1e-4;%set 0<alpha<1
    x0sz = size(x0);
    r = 2.*rand(x0sz) - 1;%ranom vector with values between -1 and 1
    k = 0;

    %Check if it is trapped in a local min

    while ((norm((feval(R,xspe)- Rt),Inf) / norm(Rt/Inf)) > epsPE) && (k <
Na)

        x0 = x0 .* (ones(x0sz) + alpha.*r);
        xspe = fminsearch(PE,x0,options);
        k=k+1;
    end

end
```

- **Stop criteria**

```
function EF = SMstop (f)
% Usage:  EF = SMstop (f)
% Stop criteria for input at iteration for coarse model is sufficiently
% similar to target input (from coarse optimization).
% f: array from Space mapping algorithm containing the difference between
% the input at the iteration and Xoptc.
% EF: returns exit function
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-05-17
    EF = 0;
    epsf = 1e-4;
    if(norm(f) < epsf)
        EF = 1;
    end

end
```

D. MATLAB™ TO HFSS DRIVER.

Generalized MATLAB™ to HFSS driver used by both Antenna implementations. A template for each specific problem is required.

```
function [Response] = MATLAB2HFSS(x,lines,HFSSPath,HFSSpyScriptTemplate)
% Usage: [Response] = MATLAB2APLAC(x,lines,APLACPath,templateName)
% x: cell array contraining strings as elements to pass parameters to
template
% lines: integer array containg line numbers to pass parameters,
% make sure it is the same length as x
% HFSSpath: path to HFSS exe (ansysedt.exe). Requires HFSS 19 or later
% HFSSpyScriptTemplate:name of python script, use .py file.
% Response: reads  and returns contents of HFSSoutput.csv
% Author: Eduardo Michel md710644@iteso.mx
% Last Update: 2019-05-25

%% hfss commnad line mode
    %hfss python script
    lenx = length(x);

    %% define python script
    %read template and store in cell array, template must be in same
directory
    hfssstr = fileread(HFSSpyScriptTemplate);
    hfssf = regexp(hfssstr, '\n', 'split');

    %% Write parameters on file using x and target lines
    for k = 1:lenx
        hfssf{lines(k)} = strcat(hfssf{lines(k)},32,x{k},32,13); %adding
ASCII chars 32 = space 13 = CR
    end
    %% Write file for hfss python script

    fid = fopen('HFSSpyScript.py','w');
    fprintf(fid,'%s\n', hfssf{:});
    fclose(fid);

    %% Run HFSS
        system(strcat(HFSSPath,32,'-RunScriptAndExit',32,'HFSSpyScript.py'));
        %Read HFSS output
        Response = csvread('HFSSoutput.csv',1,0);
        %clean up HFSS output file
         delete HFSSoutput.csv

end
```

6. Bibliography

[1] Ansys Inc., "Ansys Electroncis Desktop and HFSS Antenna Toolkit 2019," Canonsburg, PA, 2019.

[2] Mathworks, "Matlab Antenna Design Toolbox," [Online]. Available: https://www.mathworks.com/help/antenna/gs/antenna-modeling-and-analysis.html. [Accessed 09 05 2019].

[3] J. W. Bandler, Q. S. Cheng, S. A. Dakroury, A. S. Mohamed, M. H. Bakr, K. Madsen, Søndergaard and Jacob, "Space Mapping: The State of the Art," *IEEE Transactions On Microwave Theory And Techniques,* vol. 52, no. 1, pp. 337-361, 2004.

[4] J. Zhu, J. W. Bandler, N. K. Nikolova and S. Koziel, "Antenna Optimization Through Space Mapping," *EEE Transactions On Antennas and Propagation,* vol. 55, no. 3, pp. 651-658, 2007.

[5] M. Fernández-Pantoja, P. Meincke and A. Rubio-Bretones, "A Hybrid Genetic-Algorithm Space-Mapping Tool for the Optimization of Antennas," *IEEE Transactions On Antennas and Propagation,* vol. 55, no. 3, pp. 77-781, 2007.

[6] S. Koziel, S. Ogurtsov and M. H. Bakr, "Antenna Modeling Using Space-Mapping Corrected Cauchy-Approximation Surrogates," in *IEEE International Symposium on Antennas and Propagation (APSURSI)*, Spokane, WA, 2011.

[7] S. Koziel, "On Space Mapping Optimization with Coarsely-Discretized EM Coarse Models," in *IEEE MTT-S International Microwave Symposium*, Baltimore, MD, 2011.

[8] S. Tu, Q. S. Cheng, J. W. Bandler and N. K. Nikolova, "Space Mapping Design Exploiting Library Antenna Models," in *Proceedings of the 2012 IEEE International Symposium on Antennas and Propagation*, Chicago, IL, 2012.

[9] S. Tu, Q. S. Cheng, Y. Zhang, J. W. Bandler and N. K. Nikolova, "Space Mapping Optimization of Handset Antennas Exploiting Thin-Wire Models," *EEE Transactions On Antennas and Propagation,* vol. 61, no. 7, pp. 3797-3807, 2013.

[10] R. Bancroft, Microstrip and Printed Antenna Design, Atlanta, GA: Noble Publishing, 2004.

[11] J. D. Kraus, Antennas, New York: McGraw-Hill, 1988.

[12] E. Rayas, "Optimization-Based Modeling and Design of Electronic Circuits," 01 01 2019. [Online]. Available: https://desi.iteso.mx/erayas/cir_opt.htm. [Accessed 10 12 2019].

[13] R. M. J.A. Nelder, "A Simplex Method for Function Minimization," *The Computer Journal,* vol. 7, no. 4, pp. 308-313, 1965.

[14] J. E. Rayas-Sánchez, "Power in simplicity with ASM: tracing the aggressive space mapping algorithm over two decades of development and engineering applications," *IEEE Microwave Magazine,* vol. 17, no. 4, pp. 64-76, 2016.

[15] C. Balanis, Antenna theory, Analysis and Design, John Wiley & Sons, 2016.

[16] J. E. Rayas-Sánchez, *Neural Space Mapping Methods for Modeling and Design of Microwave Circuits,* Hamilton, Canada L8S 4K1: McMaster University, 2001.

[17] R. B. Waterhouse, Microstrip Patch Antenas : A Designer's Guide, Boston: Kluwer, 2003.

[18] J. E. Rayas-Sánchez, "EM-based optimization of microwave circuits using artificial neural networks: the state of the art," *IEEE Transactions on Microwave Theory and Techniques,* vol. 52, pp. 420-435, 2004.

[19] J. W. Bandler, M. A. Ismail, J. E. Rayas-Sánchez and Q.-J. Zhang, "Neuromodeling of Microwave Circuits Exploiting Space-Mapping Technology," *IEEE Transactions On Microwave Theory And Techniques,* vol. 47, no. 12, pp. 2417-2427, 1999.

[20] J. E. Rayas-Sánchez, F. Lara-Rojo and E. Martínez-Guerrero, "A Linear Inverse Space-Mapping (LISM) Algorithm to Design Linear and Nonlinear RF and Microwave Circuits," *IEEE Transactions On Microwave Theory And Techniques,* vol. 53, no. 3, pp. 960-968, 2005.

[21] The MathWorks, Inc., "MATLAB and Antenna toolbox Release 2019b," Natick, Massachusetts, 2019.

[22] Wikimedia.org, "Biconical Antenna," [Online]. Available: https://en.wikipedia.org/wiki/Biconical_antenna#/media/File:SBA_9113_(side_view).jpg. [Accessed 08 12 2019].

Index