UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

DESIGN AND OPTIMIZATION OF BROADBAND MATCHING NETWORKS FOR WIDELY STEERABLE PHASED ARRAY RADAR SYSTEMS

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE

By

PATRICK O'CONNOR-LYNCH
Norman, Oklahoma
2022

DESIGN AND OPTIMIZATION OF BROADBAND MATCHING NETWORKS
FOR WIDELY STEERABLE PHASED ARRAY RADAR SYSTEMS

A THESIS APPROVED FOR THE
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

BY THE COMMITTEE CONSISTING OF

Acknowledgments

My time at the University of Oklahoma has been filled with many inspiring and wonderful people. The first I would like to thank is my advisor and chair, Dr. Hjalti Sigmarsson. Throughout the past few years, he has contributed an abundance of wisdom and encouragement. He has provided me with the framework of what an engineer ought to be. I would also like to thank Dr. Jay McDaniel, another instrumental figure in my education. His humorous and infectious method of teaching affirmed my interest in RF engineering. Dr. Caleb Fulton taught me much of my practical knowledge of fabrication. His unusual methods have greatly improved my skills as an engineer. I would also like to thank Dr. Jessica Ruyle. I am indebted to her for introducing me to the Advanced Radar Research Center (ARRC) and stoking my interest in research. All of these professors have made me the engineer that I am today and their lessons will carry on with me throughout the rest of my career.

Many fellow students have enhanced my education and research, as well. I would like to thank Eric Wells for helping me with the fabrication of my boards; he is a great friend inside and outside the lab. Jonathan Knowles helped provided much academic and emotional support, as well as chicken. Clayton Blosser has been with me throughout most of my research career, as both a mentor and a colleague. His expertise in RF engineering is remarkable and I am glad to have had him on my side. I would also like to thank Collin Smith. He fabricated and tested many of my boards, while I was busy writing. These people have all helped in my research endeavors and contributed to the collaborative environment at the ARRC.

Outside of academics, others helped me get through my time in college. Nicholas Fox has been a great friend and kept me sane when times were tough. Michael Johnson and Putri Pradana have provided constant support. Mohammad Alwahdanee was a fountain of knowledge for my entire undergraduate career and a close mate. Mick Christensen was a tremendous friend, roommate, and optimization whiz. Reece Witcher helped me unwind and keep an optimistic outlook on life.

Finally, I would like to thank my family. My parents, Jennifer O'Connor-Lynch and Larry Lynch, were tremendously supportive of my education. Brendon O'Connor-Lynch was always there to bounce ideas off of and talk to when I needed him. Colemon O'Connor-Lynch was a constant advocate for Colemon day to support me despite the distance between us. I must also acknowledge the ones that oversaw the entirety of my thesis writing and have been a tremendous source of joy in my life, Pimm and Moscato. All of these people are a major part of my life and made me the person that I am.

This work was supported by NASA Grant/Cooperative Agreement Number: 80NSSC18M0032 P00002. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of NASA.

Table of Contents

Acknowledgments **iv**

Table of Contents **vi**

List of Tables **ix**

List of Figures **x**

Abstract **xiii**

1 Introduction **1**

1.1 Motivation 1

1.2 Tunable Matching Networks 4

1.3 Research Objectives 6

1.4 Thesis Outline 6

2 Theory of Matching Networks **8**

2.1 Introduction 8

2.2 Impedance Mismatches 9

2.3 Impedance Matching 10

2.4 Topologies of Matching Networks 13

2.5 Bode-Fano Limit 15

2.6 Filters and Matching Networks 17

2.7 Summary 18

3 Introduction to Optimization 19

3.1 History of Optimization 19

3.2 Linear Programming 20

3.3 Nonlinear Programming 21

3.4 Optimization in Simulators 22

3.5 Summary 23

4 Static Matching Network 24

4.1 Introduction 24

4.2 Design of Static Network 25

4.3 Optimization Model 26

4.3.1 Decision Variables and Objective Function 27

4.3.2 Model Constraints and Parameters 28

4.4 Implementation in MATLAB 28

4.4.1 'Fmincon' optimizer 29

4.4.2 'Genetic algorithm' optimizer 30

4.5 Application to EcoSAR 30

4.5.1 Simulated Network 31

4.5.2 Fabricated Network 35

4.6 Effects of Component Tolerances 39

4.7 Summary 41

5 Tunable Matching Network 42

5.1 Introduction 42

5.2 Design of Tunable Network 42

5.3 Optimization Model 44

5.3.1 Decision Variables and Objective Function 45

5.3.2 Model Constraints and Parameters 46

5.4 Implementation in MATLAB 46

5.4.1 Inner optimizer 47
5.4.2 Outer optimizer 47
5.5 Application to EcoSAR 48
5.5.1 Simulated Network 48
5.5.2 Fabricated Network 53
5.6 Summary 57

6 Future Work and Conclusion 58
6.1 Conclusion 58
6.2 Future Work 59

A MATLAB Code 65
A.1 Static Matching Network Program 65
A.2 Tunable Matching Network Program 73
A.3 Excel Document Format 85

B Matching Network Fabrication Process 86
B.1 Detailed Fabrication Process 86
B.1.1 Board preparation 87
B.1.2 Electroplating 87
B.1.3 Board milling 87
B.1.4 Assemble board 87

List of Tables

4.1 Static Program Parameters 28
4.2 Simulated Component Values 33
4.3 Effect of Filter Order on Performance 33
4.4 Fabricated Component Values 35

5.1 Tunable Program Parameters 46

List of Figures

1.1 Visualization of the EcoSAR system characterizing the terrain of different environments, from [3] 2
1.2 Performance variation of a single element within the EcoSAR array, each trace represents a scan angle, data courtesy of NASA Goddard 3
1.3 Impedance matching of an antenna to a source, adapted from [7] . . 3
1.4 Tunable matching network for RF power amplifier, from [8] 4
1.5 Reconfigurable matching network for phased array, from [9] 5
1.6 Tunable-varactor matching network, adapted from [10] 5

2.1 Impedance Mismatch and Reflection, adapted from [14] 9
2.2 The Smith Chart, from [16] 10
2.3 Matching Network Example 12
2.4 Lumped Element Matching Network Topologies, adapted from [14] 14
2.5 Illustrated Relationship Between Bandwidth and Quality of Impedance Match, from [17] 16
2.6 Non-Foster Impedance Matching, from [20] 17

3.1 The optimization cockpit within Keysight's ADS program, from [30] 22

4.1 Proposed approach to generate optimized static matching network . 25
4.2 Bandpass filter prototype 26
4.3 Incorporation of a matching network array for the EcoSAR system . 31

4.4 Performance evaluation of EcoSAR data (a) S_{11} (b) optimizer objective function evaluation 32

4.5 Performance evaluation of simulated network (a) S_{11} (b) optimizer objective function evaluation 34

4.6 Photograph of the fabricated static matching network 36

4.7 Filter responses of the simulated matching network and fabricated matching network, showing similar performance. 36

4.8 Performance evaluation of fabricated network (a) S_{11} (b) optimizer objective function evaluation 37

4.9 Phase shift measured from five different fabrications runs 38

4.10 S-parameters of five different fabricated boards using the same fabrication process and components (a) S_{11}, (b) S_{22}, (c) S_{21} 40

5.1 Proposed approach to generate optimized tunable matching network 43

5.2 Tunable filter prototype, adapted from [10] 43

5.3 Overlapping frequency bins with 50 MHz bandwidth 44

5.4 Performance evaluation of EcoSAR data (a) S_{11} (b) optimizer objective function evaluation 49

5.5 Performance evaluation of simulated network with frequency bin of 20 MHz (a) S_{11} (b) optimizer objective function evaluation 51

5.6 Performance evaluation of simulated network with frequency bin of 10 MHz (a) S_{11} (b) optimizer objective function evaluation 52

5.7 Schematic of the tunable matching network 53

5.8 Photograph of the fabricated tunable matching network 54

5.9 Return loss of 5 dB improved to 5 MHz of 20 dB match 54

5.10 Return loss of 6 dB improved to 6 MHz of 20 dB match 55

5.11 Return loss of 4 dB improved to over 7 MHz of at least 10 dB match 55

5.12 Return loss of 17 dB improved to 25 dB match 56

A.1 S-parameter format in excel sheet used for code 85

Abstract

The use of phased arrays have become prevalent in radar systems for military and weather applications due to their planar configuration and ability of electronic steering. However, mutual coupling between surrounding antenna elements causes degradation in performance at wide scan angles and broad bandwidth. This mutual coupling creates impedance mismatches at the element-level. A matching network can be used to correct this, but broadband matching networks that consider scan angle have not been explored.

This work introduces a novel optimization method for designing matching networks. This method seeks to improve performance for wide-scanning broadband phased arrays, especially NASA's Ecological Synthetic Aperture Radar (EcoSAR). The fabricated static matching network achieves a 78% reduction in the optimizer objective function and provides a 10 dB match for the majority of the scanning range of -40° to 40° at a fractional bandwidth of 28.7%. This method is also used to design a tunable matching network that achieves a measured 99.95% reduction in the optimizer objective function and provides a 15 dB match across frequency and scanning range for the EcoSAR array using frequency bins. This novel design method shows great promise for improving performance for future wide-scanning broadband phased array systems.

Chapter 1

Introduction

1.1 Motivation

Radar systems are vital for both the defense and meteorology sectors and have seen vast improvements over the past decades. One such improvement is the phased array radar, which allows for electronic beam-steering and a planar antenna geometry. These advantages allow them to be easily outfitted on any vehicle and do not consume as much space as conventional radar systems. While providing many benefits, the phased array suffers degraded performance when scanning off-broadside. This limitation hinders a wide scanning range for many systems and arises from the mutual coupling of surrounding antenna elements causing changes in the electromagnetic fields of each element [1], [2]. These changes correlate to an impedance mismatch, which does not allow for maximum power transfer from the antenna to the radar electronics. To correct impedance mismatches, a matching network is required. The impedance mismatch resulting from the increasing scan angle of the phased array varies tremendously. Impedance matching to a single impedance is simple, but designing a network to match to a range of impedances adds great complexity.

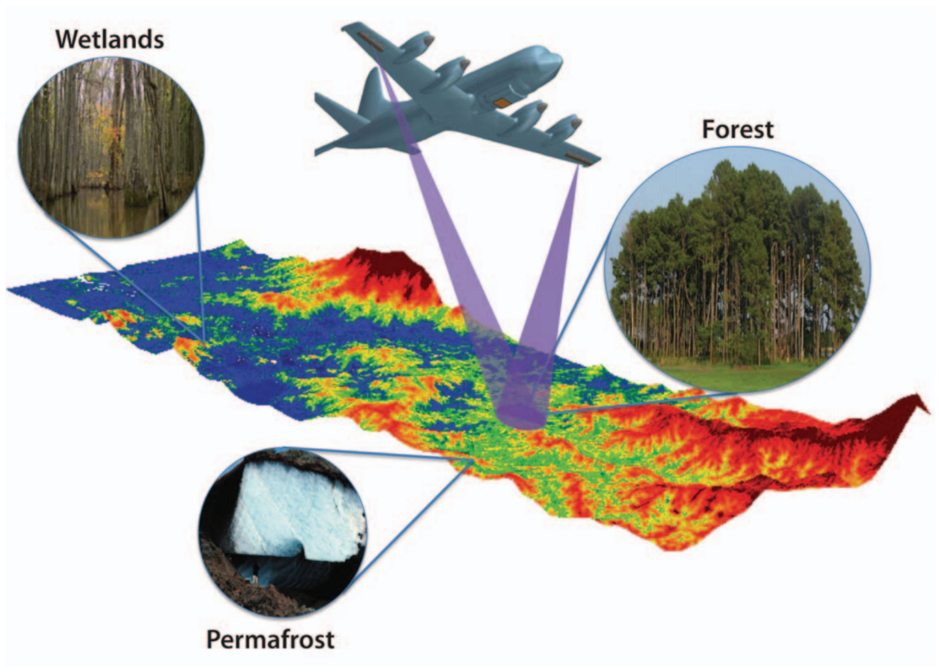


Figure 1.1: Visualization of the EcoSAR system characterizing the terrain of different environments, from [3]

NASA's Ecological Synthetic Aperture Radar (EcoSAR), depicted in Fig. 1.1, is an instrument being developed by NASA to take various measurements of an ecosystem to characterize them and monitor disturbances. The system is capable of measuring canopy height, woody biomass, ice sheet thickness, and permafrost depth, as well as mapping forest and polar ecosystems [3]–[6]. These measurements are important in determining an ecosystem's health as impacted by climate change or deforestation and evaluating other environmental processes. The EcoSAR uses two wide-band antenna arrays operating at a center frequency of 435 MHz with 125 MHz bandwidth. As it uses a phased array configuration, the EcoSAR array suffers from impedance mismatches resulting from scanning off-broadside, as discussed earlier. Fig. 1.2 shows the variation in the performance of the first antenna element of the array. Each trace of the plot represents the return loss of the element at each scan angle from -40° to 40°, resulting in 81 traces. This element experienced the worst performance as it can be seen that the return loss in the passband goes lower than 5 dB at certain scan angles and frequencies. This limits the usable scanning range. To increase the performance of the antennas, a matching network can be added, such as in Fig. 1.3.

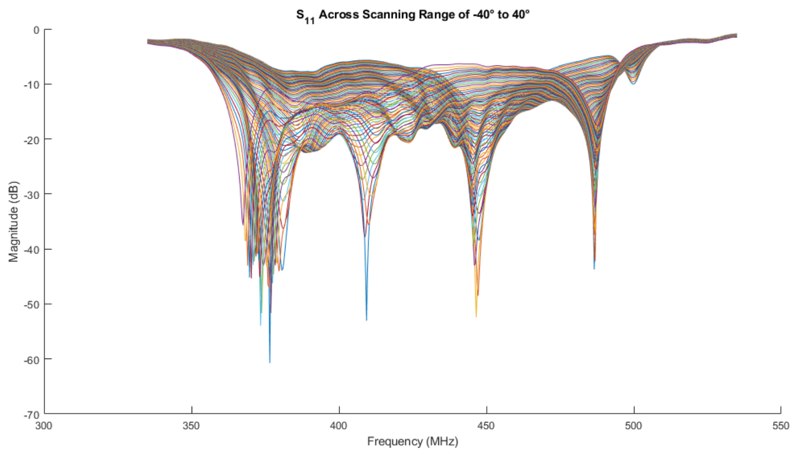


Figure 1.2: Performance variation of a single element within the EcoSAR array, each trace represents a scan angle, data courtesy of NASA Goddard

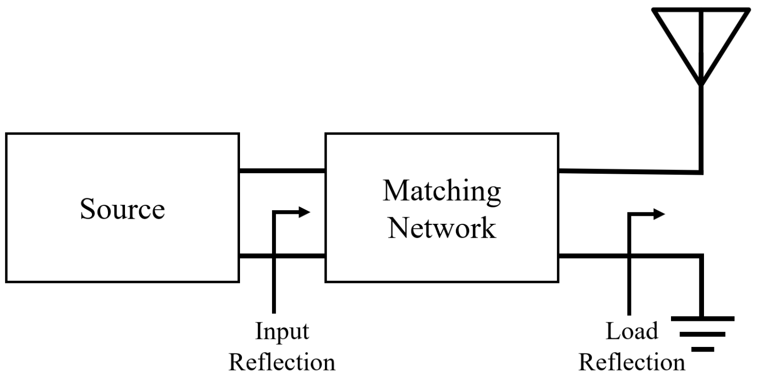


Figure 1.3: Impedance matching of an antenna to a source, adapted from [7]

1.2 Tunable Matching Networks

The concept of tunable filters and matching networks is not new. They can be found in many applications including filter banks in mobile phones. Many research groups have looked into tunable matching networks [8]–[12]. One paper presented a tunable matching network to match the input and output impedance of an RF power amplifier [8]. The tunable configuration used four switches to strategically connect various capacitors and a varactor for additional fine-tuning, seen in Fig. 1.4. This provides high tunability for impedance matching between the 16 switch states and the varactor's tuning range.

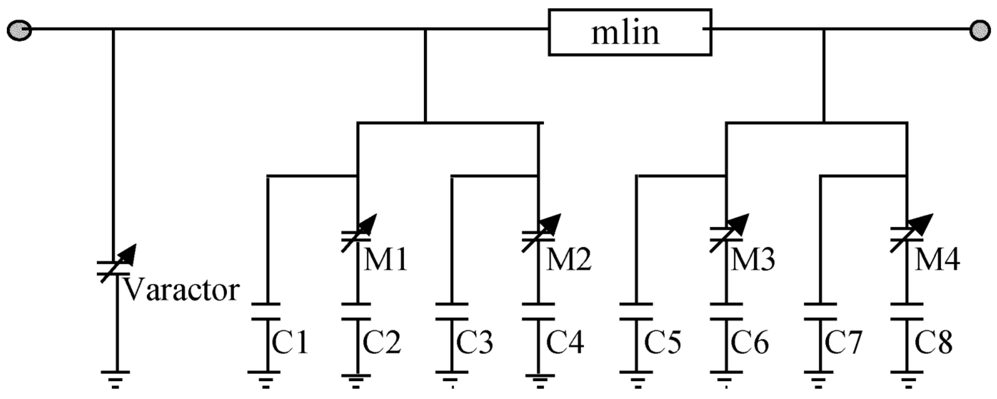


Figure 1.4: Tunable matching network for RF power amplifier, from [8]

Another tunable matching network can be seen in a proceeding from a research group at Baylor University [9]. Instead of switching between discrete capacitors, switching is done between radial stubs of varying sizes, seen in Fig. 1.5. This lends itself towards high-frequency designs, as lumped components may self-resonate well before the operating frequency of higher frequency systems. This design operates at 3.55 GHz and uses six switches and thus has 64 switch states, an impressive range of tunability. A distributed network design can be more easily fabricated and suffer less from tolerances associated with fabrication as a lumped element design.

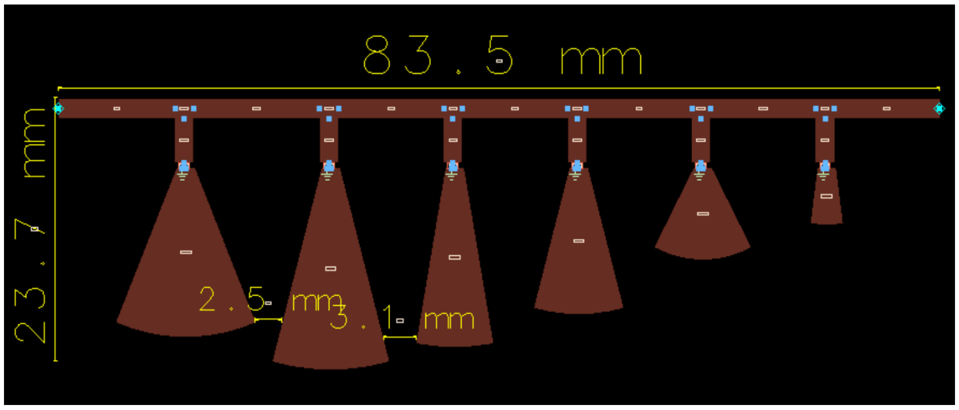


Figure 1.5: Reconfigurable matching network for phased array, from [9]

The previous two designs used switches to select between several impedances for matching. Fig. 1.6 shows a matching network that uses three varactors and an inductor to tune to a wide range of impedances for a 1.3 GHz system [10]. This design allows for much finer tuning as varactors allow for continuous tuning, rather than digital tuning from switches. Instead of having a finite number of switch states, the varactor network has an almost infinite number of tuning states.

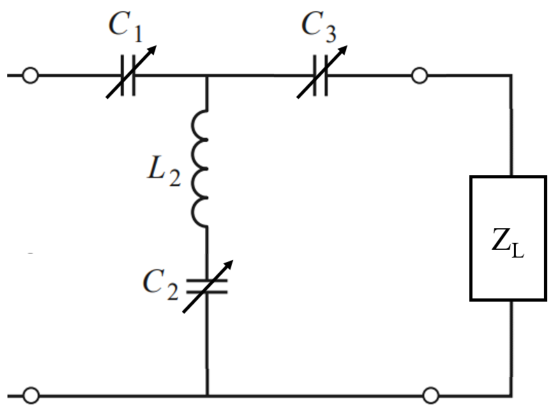


Figure 1.6: Tunable-varactor matching network, adapted from [10]

1.3 Research Objectives

Tunable matching networks for phased array systems have been explored, however, using a static matching network to optimally impedance match the a range of impedances associated with the scan angles of a phased array has not, to the author's knowledge. Thus, a novel design method for designing a static matching network is presented. This design method is then extended to a tunable varactor network to optimally tune a phased array system and achieve better performance over the static network. Through implementing optimization algorithms, a broadband matching network can be designed to improve wide-scanning phased array systems.

1.4 Thesis Outline

Chapter 2 reviews the theory of matching networks. The need for matching networks, followed by considerations for impedance matching, is discussed. The limitations of matching networks and their filtering properties are considered.

Chapter 3 provides an introduction to optimization. It explores the necessity of mathematical programming for complex problems and how these problems are set up for computer algorithms to solve. The incorporation of optimization functions in popular RF circuit and structure simulators is reviewed.

Chapter 4 applies the ideas presented in the previous two chapters toward a novel design process for a static matching network using optimization. A step-by-step method is outlined for designing a matching network for NASA's EcoSAR array. The designed static network is simulated and fabricated to verify the overall design process.

Chapter 5 extends the design method of the previous chapter to a tunable matching network configuration. The tunable method is outlined in a similar fashion. A simulated and fabricated network is presented to verify the design process.

Chapter 6 concludes this thesis by providing a summary of the research presented in this work. It then discusses possible future routes to continue this work.

Chapter 2

Theory of Matching Networks

2.1 Introduction

Impedance matching is a concept ubiquitous in the field of microwave engineering. For AC circuits operating in the kilohertz range, impedance matching is not often discussed, since losses at these frequencies are less exaggerated than those at RF. As the operating frequencies of systems increase to attain larger bandwidths and faster clock speeds, the assumptions made in conventional circuit theory, which lead to simplified circuit models, degrade critically. The use of Maxwell's equations to evaluate RF circuits becomes necessary and losses arising from the non-idealities of realizable components must be considered. Many applications at higher frequencies handle small signals, such as is the case for receivers, thus the utmost care must be taken to minimize losses [13]. The practice of impedance matching is used to maximize power transfer and prevent further losses. While impedance matching to a single load has been well defined in many texts [13]–[15], matching to several loads with a single circuit is a newer concept. In this text, optimization algorithms are introduced into the design process to tackle this problem. Before delving into the inner workings of optimization, the theory surrounding matching networks will be explored.

2.2 Impedance Mismatches

Before considering impedance mismatches, it is prudent to introduce the concept of the reflection coefficient, as it will be used as a metric for performance throughout this text. The reflection coefficient (Γ) describes the ratio of voltages of an incident wave (V_o^+) and its reflection (V_o^-). In the ideal case of power transfer, there will be no reflection and Γ will be equal to zero. However, when an impedance mismatch is introduced, a reflection is created and Γ takes on a value between zero and one, in a passive circuit, seen in Fig. 2.1. The reflection coefficient is mathematically defined as

$$\Gamma = \frac{V_o^-}{V_o^+} = \frac{Z_L - Z_0}{Z_L + Z_0} \,, \tag{2.1}$$

where Z_L represents the load impedance and Z_0 represents the source impedance. As can be seen from (2.1), reflection is at a minimum when the source and load impedances are equal. The further away these values become, the large the amplitude of the reflected voltage wave becomes.

Impedance mismatches in a system are unavoidable. Reactive components are intentionally used for many applications, for example, RF chokes and bypass ca-

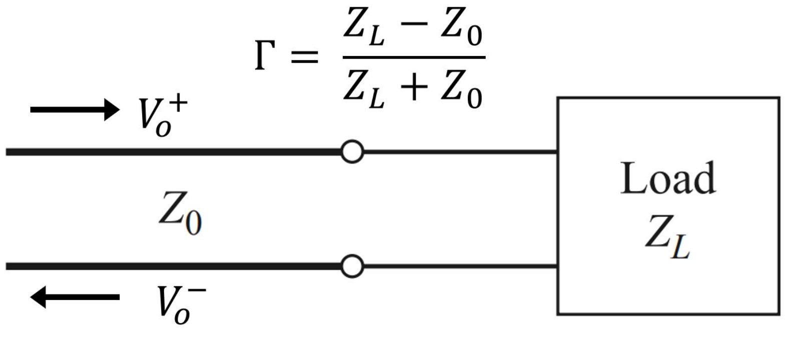


Figure 2.1: Impedance Mismatch and Reflection, adapted from [14]

pacitors are used commonly in biasing networks to separate RF and DC signals. At RF frequencies, reactive elements can arise unintentionally, as well; the leads of a component package exhibit an inherently inductive quality. The presence of reactive and resistive elements throughout a system causes impedance mismatches and requires correction by the use of matching networks.

2.3 Impedance Matching

The best way to visualize impedances and reflection coefficients is on the Smith chart, depicted in Fig. 2.2. The center of the chart corresponds to the system impedance, usually 50 Ω. There are constant resistance and reactance circles throughout the chart. For impedance matching, these are quite useful; for example, when

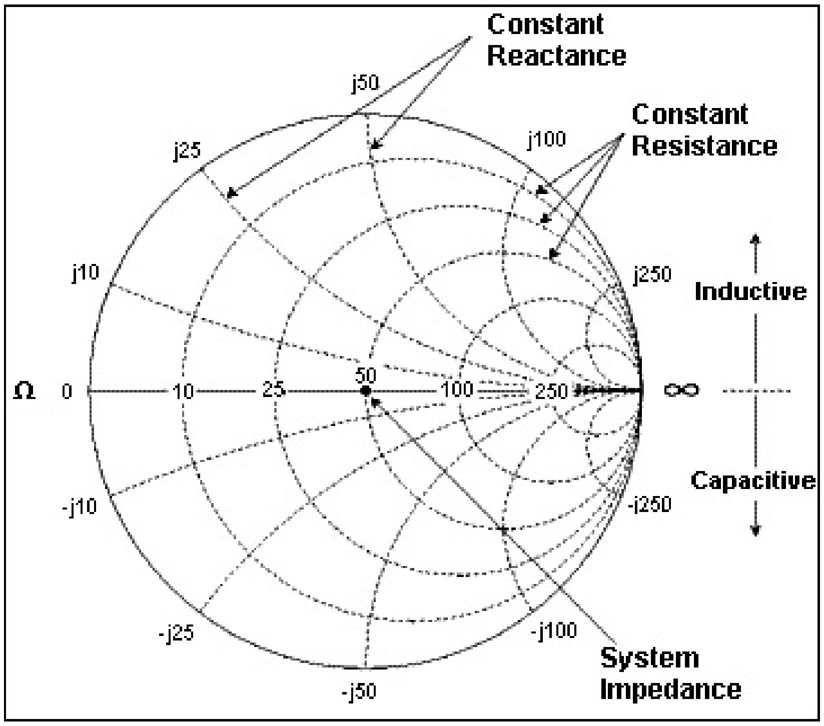
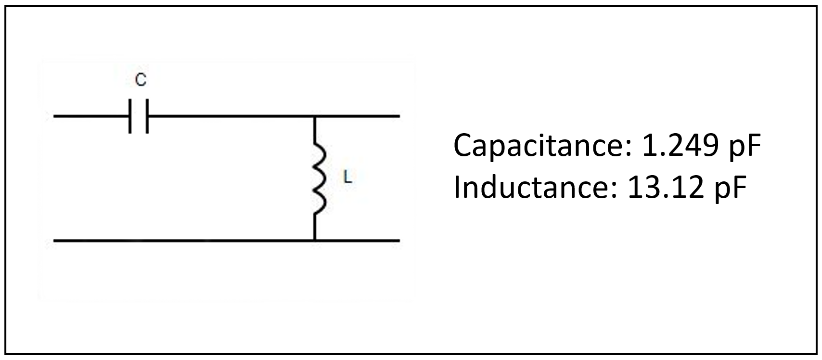


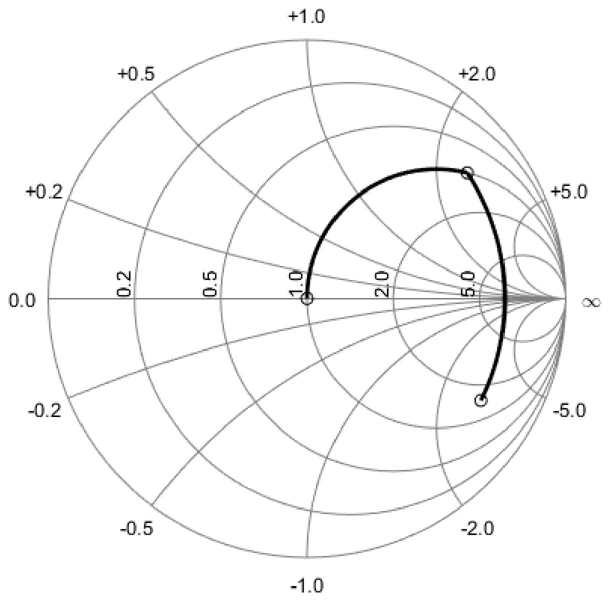
Figure 2.2: The Smith Chart, from [16]

adding a series reactive component, say an inductor, the impedance will travel clockwise along the constant resistance circle. In this way, it is easy to see how a certain impedance can be manipulated by adding circuit elements. On the chart, Γ is determined by how far away an impedance is from the center of the chart. The center of the chart corresponds to a Γ of 0 and the edge of the chart corresponds to a Γ of 1.

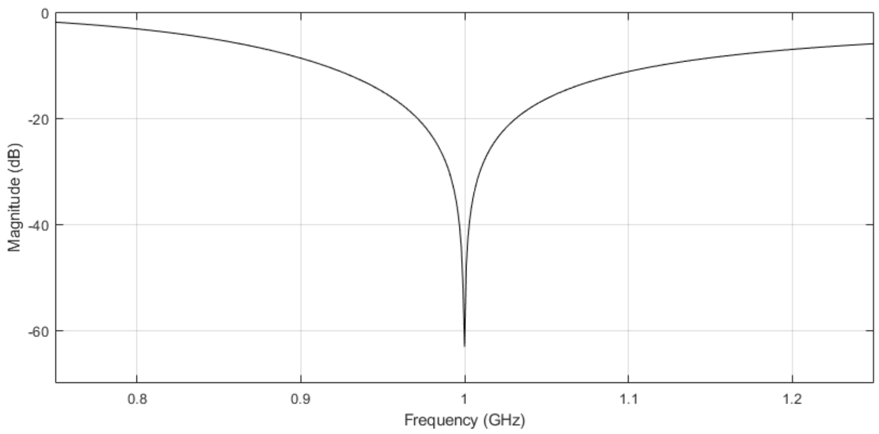
For example, Fig. 2.3(a) shows a matching network circuit that matches a load impedance of $75-j150\ \Omega$ to 50 Ω at 1 GHz. Fig. 2.3(b) shows the matching path on the Smith chart. From the load, the shunt inductor travels along a constant conductance circle to Z_1 of $50.0168 + j127.4936\ \Omega$ and the series capacitor travels along a constant resistance circle to 50 Ω. Fig. 2.3(c) shows the bandwidth of the match, about 200 MHz of 10 dB match. Note that S_{11} corresponds to Γ on the logarithmic scale. The circuit in this example uses an L-network, however there exist different topologies of matching networks with each having its own characteristics.



(a) L-Topology Circuit



(b) Matching Path on Smith Chart



(c) Impedance Match over Frequency

Figure 2.3: Matching Network Example

2.4 Topologies of Matching Networks

There are many circuit topologies of matching networks and reasons to use one over another. The three topologies that will be discussed are:

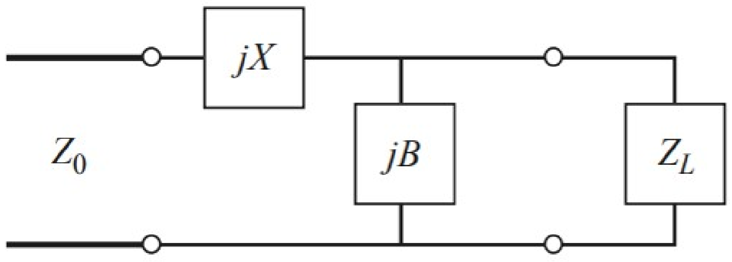
1. L-network
2. Π-network
3. T-network

The simplest topology is the L-network, found in Fig. 2.4(a)-(b), which was used in the example of the previous section. Analytical solutions for the required impedance and admittance of this network can be found in [14]. While the L-network uses the least number of components, the downside is that the quality-factor (Q) cannot be set by the designer, since the two components do not provide enough degrees of freedom [15].

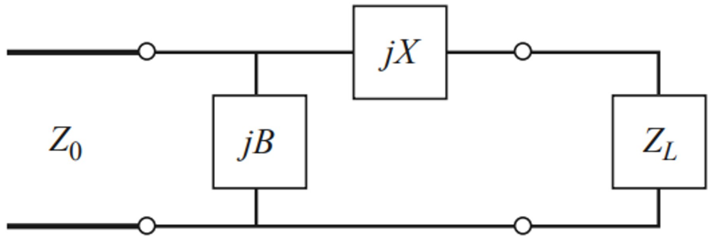
The Π- and T-networks, found in Fig. 2.4(c)-(d), can be thought of as a cascade of two L-networks, where the middle two elements are combined either in series or parallel, respectively. Why do this in the first place if we can match with just an L-network? Dr. Thomas Lee answers,

> "Now, it may feel suspiciously like a government-works project to use one L-section to go down [the Smith chart] and then another to go back up. However, we have gained an important additional degree of freedom... this allows us to achieve much higher Q than is generally available from an L-match [15]."

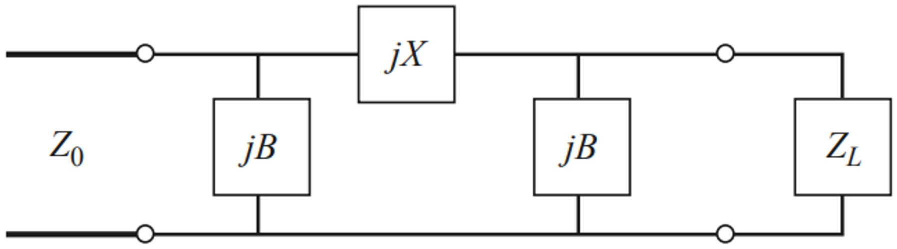
This is an important property of using a network with more than two components; a higher quality match can be attained, however at the expense of bandwidth.



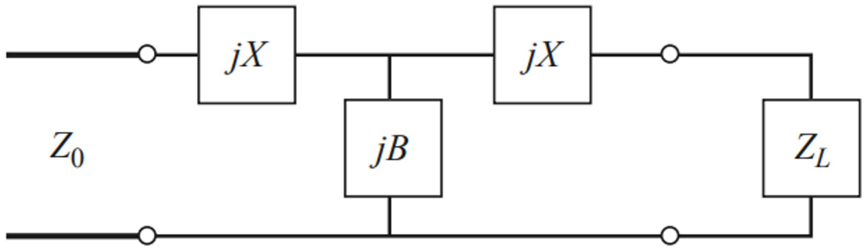
(a) L-network for z_L inside 50+jx circle



(b) L-network for z_L outside 1+jx circle



(c) Π-network



(d) T-network

Figure 2.4: Lumped Element Matching Network Topologies, adapted from [14]

Only networks composed of up to three components are shown in this chapter, but more components can always be used. For matching to a single impedance, this is unnecessary, but for complex matching, higher-order networks will later show to provide better performance, albeit with diminishing returns [17].

While the topologies in Fig. 2.4 only include generic impedances and admittances, they can be implemented in any number of ways. When using lumped components, inductors and capacitors can be populated in any position of the network. This has filtering implications which will be discussed later. Additionally, at higher frequencies, the values required to use lumped components become non-realizable while wavelengths become practical lengths, thus, using distributed components, such as stubs and lines, become common practice.

2.5 Bode-Fano Limit

It must be stated that there is a limit to the quality of an impedance match over a given bandwidth. Hendrik Bode first explored the limitation of a match for an RC load in his text *Network Analysis and Feedback Amplifier Design* [18]. He derived that limitation to be

$$\int_{\omega_1}^{\omega_2} \ln\left|\frac{1}{\Gamma}\right| d\omega \leq \frac{\pi}{RC} \,. \tag{2.2}$$

Given the case that Γ is at a constant Γ_{max} value in the given bandwidth and unity outside then (2.2) simplifies to

$$\ln\left|\frac{1}{\Gamma_{max}}\right| \leq \frac{\pi}{(\omega_2 - \omega_1)RC} \,. \tag{2.3}$$

This relationship shows that the quality of the match depends on the bandwidth and the particular resistance and capacitor values being matched, but only for an

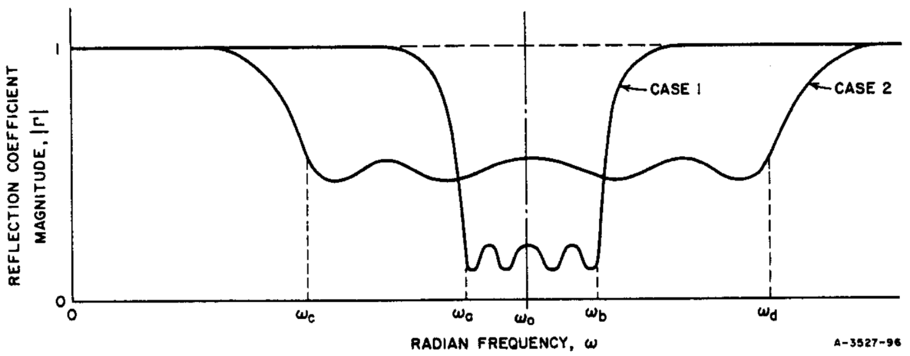


Figure 2.5: Illustrated Relationship Between Bandwidth and Quality of Impedance Match, from [17]

RC load. Robert Fano took Bode's work a step further and showed this relationship to be true for any arbitrary impedance [19]. The relationship between bandwidth and the quality of impedance match is highlighted in Fig. 2.5. Fano also outlined a method for designing a broadband matching network in [19]. However, this method requires that the load be characterized as a finite number of linear passive elements and becomes mathematically complex as the number of elements increases.

There has been research in recent years focused on breaking the Bode-Fano limit. These efforts center around using non-Foster elements, circuits that realize negative capacitance or inductance values, which do not adhere to Foster's reactance theorem [20], [21]. This allows for broadband matching by canceling out the reactive components of the load, rather than creating a resonance which is often narrow-band, illustrated in Fig. 2.6. While this method has its merits, there are serious drawbacks, depending on the application. Non-Foster circuits require transistors to implement which "results in very sensitive stability issues, ... generate noise which can degrade performance in certain small signal applications, ... and also have nonlinearity issues, which become evident in high power applications" [21]. This makes them less appealing for use in high-power radar applications. Thus, non-Foster matching will not be explored in this research as the end goal is a matching network for use in a radar system.

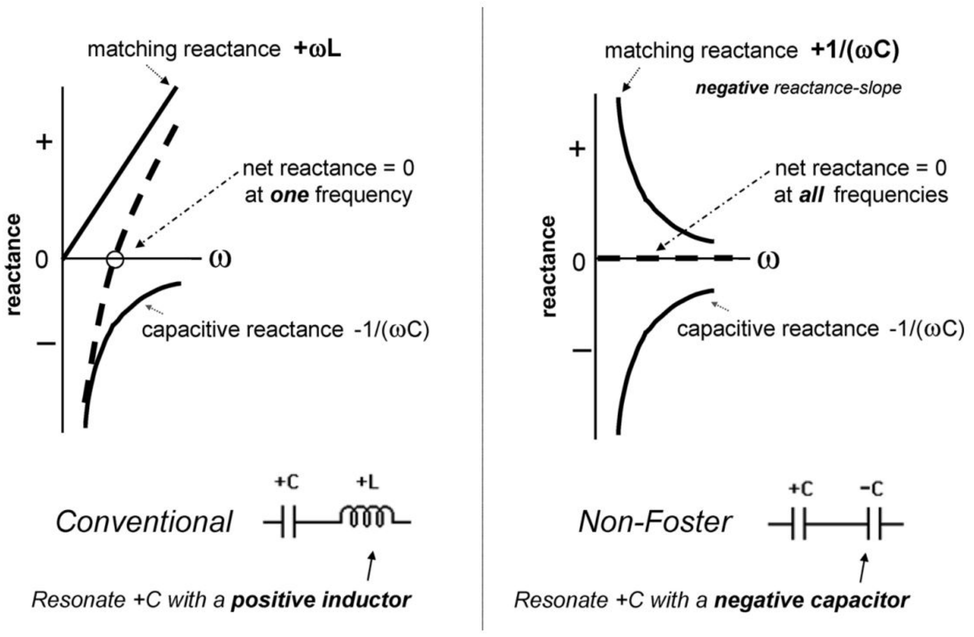

matching reactance +ωL
+
net reactance = 0 at one frequency
reactance
0
ω
–
capacitive reactance -1/(ωC)
+C
+L
Conventional
Resonate +C with a positive inductor
matching reactance +1/(ωC)
negative reactance-slope
+
net reactance = 0 at all frequencies
reactance
0
ω
–
capacitive reactance -1/(ωC)
+C
-C
Non-Foster
Resonate +C with a negative capacitor

Figure 2.6: Non-Foster Impedance Matching, from [20]

2.6 Filters and Matching Networks

When designing matching networks, the decision between choosing an inductor or capacitor can be an important one. Oftentimes, matching networks will serve dual purposes. Intrinsically, inductors and capacitors exhibit filtering qualities. This can be used to the advantage of the designer. Impedance matching to the input and output of an RF amplifier requires caution as the amplifier requires both RF and DC signals to operate. The matching network can be used to keep the RF and DC signals separate, but it must not filter out the RF signal from the input and outputs of the amplifier. Thus, RF designers must be careful when selecting the topology of the matching network.

2.7 Summary

Matching networks are required in RF systems to prevent impedance mismatches that may cause significant losses. Impedance matching on the Smith chart provides a convenient way of visualizing an impedance and its associated reflection. Different configurations of matching networks allow for different quality-factors and filtering properties. The Bode-Fano limit defines the trade-off between the quality and bandwidth of an impedance match. With the theory of matching networks reviewed, the basics of optimization will next be discussed with the intention of later applying it to the design of a matching network.

Chapter 3

Introduction to Optimization

Later chapters will introduce a method for designing matching networks that revolves around the use of optimization to determine circuit values. To better understand the process, a brief introduction to optimization is provided in this chapter. The history of optimization is discussed. Linear problems are considered, followed by nonlinear problems – the nature of the matching network problem. A discussion of optimizers used in electromagnetic and circuit solvers then follows.

3.1 History of Optimization

Finding a minimum to a function is rudimentary by today's standards, however, this was one of the first steps made in the field of optimization several centuries ago. In the 17th century, Pierre de Fermat proposed finding the local minimum or maximum of a function by setting the derivative of the function to zero [22]. However, finding the minimum of a constrained multi-variable problem is more involved and required the introduction of linear programming to solve. In 1947, George Dantzig released his book, *Linear Programming*, which paved the way for mathematical programming [23]. In his text, Dantzig outlined the simplex method, the most common linear programming algorithm today. According to Dantzig, computer technology was partially pursued by the Pentagon to implement his linear program. Linear programming proved to be a powerful tool in optimization, but many of the world's problems are not of a linear nature.

Nonlinear programming was the natural next step in mathematical programming. It first appeared in a paper by H.W. Kuhn and A.W. Tucker, that built upon the work of William Karush [24]. This paper introduced the Karush-Kuhn-Tucker conditions that allowed for inequality constraints, in addition to the equality constraints of linear programming. From there, according to the *History of Optimization*, G. Zoutendijk, J.B. Rosen, P. Wolfe, M.J.D. Powell, and others defined further algorithms for nonlinear programming [25]. These algorithms are the foundation of nonlinear programming as it is today. With the history of optimization explored, an overview of linear and nonlinear programming follows.

3.2 Linear Programming

Linear algebra provides a framework to solve linear problems. Systems of linear equations are best solved using matrices and performing various operations on that matrix to arrive at a solution [26]. Adding constraints to linear problems adds complexity which can make the problem difficult to solve by hand. Thus, linear programming is used to solve complex linear problems. A linear problem is defined by any objective function, $f(x)$, linear inequality constraints, A and b, linear equality constraints, Aeq and beq, and lower and upper variable bounds, lb and ub [27].

The problem takes the form

$$\min_x f(x) \text{ such that } \begin{cases} A \cdot x \leq b \\ Aeq \cdot x = beq \\ lb \leq x \leq ub \,. \end{cases}$$

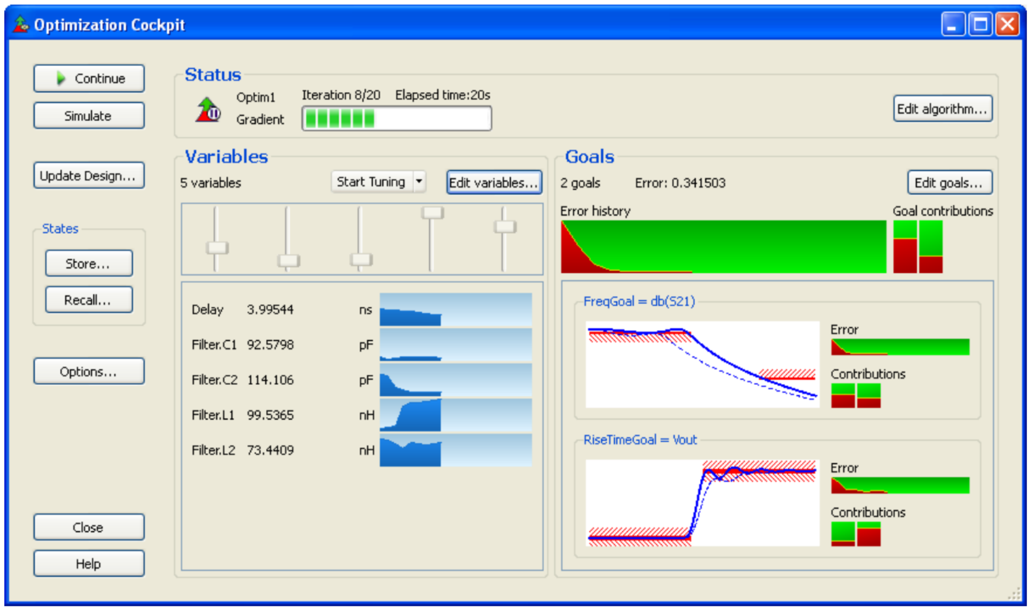
With this framework defined, the program then uses an algorithm, such as the simplex method, to determine the optimal solution to the linear problem, if one is feasible.

3.3 Nonlinear Programming

Nonlinear problems are defined in a similar way as linear problems, with the exception that the objective function takes a nonlinear form and the problem may have nonlinear constraints in addition to linear constraints. However, while it is clear whether a linear problem has an optimal solution, it is not for nonlinear problems. A nonlinear program may optimize the solution until it meets a certain stopping criterion. The algorithms used between linear and nonlinear programs may also differ. As stated before, one of the most common linear algorithms is the simplex method, which is an iterative process that improves the solution for a finite number of iterations [28]. The most common nonlinear algorithm is the interior point method, outlined in [29]. This method does not require a gradient to be provided, unlike other algorithms, making it less complicated to implement.

3.4 Optimization in Simulators

Optimization can be a useful tool for RF design. It can aid a designer in increasing performance of a circuit or structure to meet specific requirements. Keysights's Advanced Design System (ADS) is a software program that simulates high frequency circuits and has optimization capabilities. It has a dozen built-in optimizers from which to choose [30]. Ansys' High Frequency Structure Simulator (HFSS) is a popular electromagnetic full-wave simulator that allows for optimization to be integrated into a simulation [31]. Both of these simulators allow the user to define multiple, custom objective functions which would require nonlinear programming. The incorporation of optimization in at least two popular RF design programs shows optimization can be an important part of the design process.



Optim1 Gradient Iteration 8/20 Elapsed time:20s

5 variables

```
Delay      3.99544   ns
Filter.C1  92.5798   pF
Filter.C2  114.106   pF
Filter.L1  99.5365   nH
Filter.L2  73.4409   nH
```

2 goals Error: 0.341503

Figure 3.1: The optimization cockpit within Keysight's ADS program, from [30]

3.5 Summary

Optimization is a powerful tool for solving the complicated problems of today. It has its roots several centuries ago and has since proliferated with the advent of computers. Linear programming was the first step in using computers to solve problems. Nonlinear programming soon followed, allowing for the solving of more constrained and complex problem structures. Optimization has been incorporated into many popular RF design programs as a way to fine-tune designs to meet certain specifications. In the next chapter, it will be applied to the design process of a matching network for phased array systems.

Chapter 4

Static Matching Network

4.1 Introduction

With ample background of matching networks from Chapter 2 and optimization from Chapter 3, these concepts will now be applied to the generation of a static matching network. This network is designed to correct impedance mismatches resulting from the mutual coupling of phased arrays scanning off broadside. In this chapter, a prototype matching network is designed. The performance of this network is assessed and tuned to create a single optimal network that improves performance for all scan angles. This is achieved by importing a data file containing the S-parameters of a phased array antenna element for each frequency in the band of operation and each scan angle and optimizing a matching network using this data. The proposed approach is outlined in Fig. 4.1. The model is implemented in MATLAB, the resulting network simulated in Keysight's Advanced Design System (ADS), and then fabricated to verify the entire process.

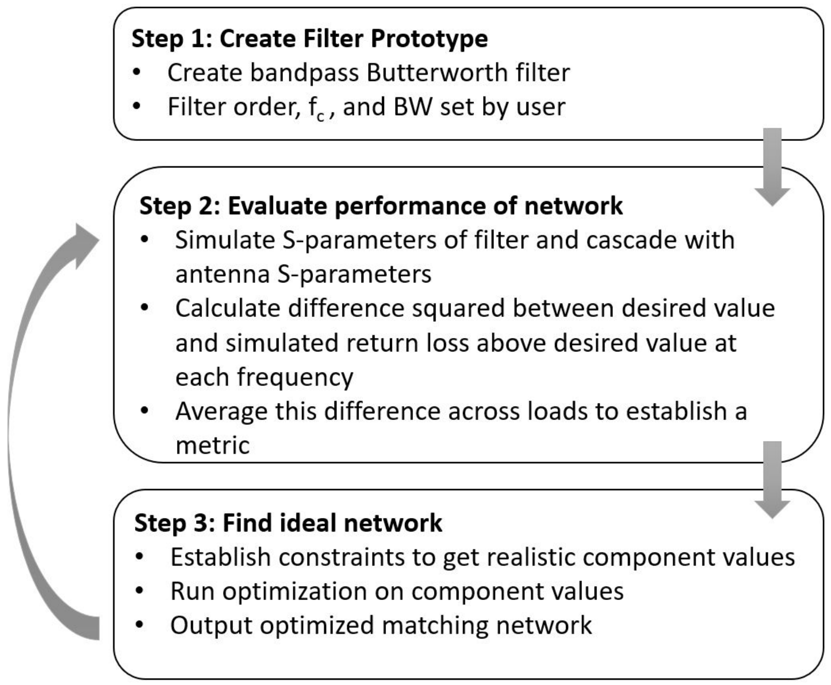

Step 1: Create Filter Prototype
• Create bandpass Butterworth filter
• Filter order, f_c, and BW set by user
Step 2: Evaluate performance of network
• Simulate S-parameters of filter and cascade with antenna S-parameters
• Calculate difference squared between desired value and simulated return loss above desired value at each frequency
• Average this difference across loads to establish a metric
Step 3: Find ideal network
• Establish constraints to get realistic component values
• Run optimization on component values
• Output optimized matching network

Figure 4.1: Proposed approach to generate optimized static matching network

4.2 Design of Static Network

For high-frequency applications, it is important to consider the filtering properties of any circuit. In the case of a matching network, it should not impede the frequency of operation. Thus, a filter prototype will be used for the initial iteration of the matching network circuit. A bandpass or bandstop filter utilizes more components, per filter order, providing more degrees of freedom for impedance matching. A bandpass filter is selected, as no DC signals will propagate through the network and the operating frequency band can be included in the passband of the filter, making it suitable for a radar system. The topology of this network is shown in Fig. 4.2. The order of the filter is set by the user. A Butterworth prototype filter is recommended as the optimizer converges faster for a Butterworth filter over a Chebyshev.

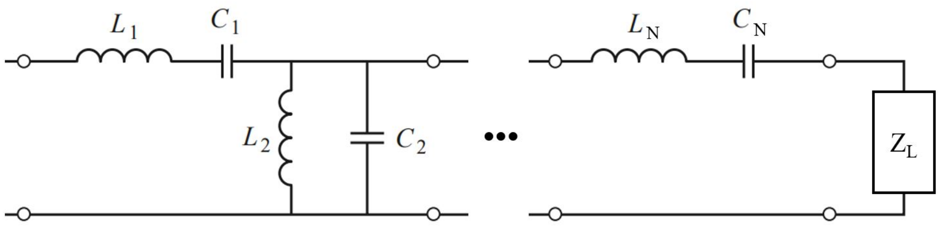


Figure 4.2: Bandpass filter prototype

The initial component values of the network can be calculated, from [14], as

$$\text{For odd components:} \quad L_n = \frac{g_n Z_0}{\omega_0 \Delta} \tag{4.1}$$

$$C_n = \frac{\Delta}{\omega_0 g_n Z_0} \tag{4.2}$$

$$\text{For even components:} \quad L_n = \frac{\Delta Z_0}{\omega_0 g_n} \tag{4.3}$$

$$C_n = \frac{g_n}{\omega_0 \Delta Z_0}, \tag{4.4}$$

where g_n is found in Table 8.5 of [14], $\Delta = \frac{\omega_2 - \omega_1}{\omega_0}$, ω_0 is the center frequency, ω_1 is the lower cutoff frequency, ω_2 is the upper cutoff frequency, and Z_0 is the system impedance.

4.3 Optimization Model

In this section, the optimization model for the static matching network will be described in detail. The decision variables, objective function, model constraints, and parameters for this problem follow. With the entire model outlined, it can be readily implemented using any programming language.

4.3.1 Decision Variables and Objective Function

The optimization of this matching network can be set up as a nonlinear programming problem [28]. The network is composed of an N number of inductors and capacitors, labeled by n = 1, . . . , N. The inductors and capacitors make up a set that will be denoted as L and C, respectively. The decision variables are Λ_n, corresponding to the inductance of inductor n in L, and ς_n, corresponding to the capacitance of capacitor n in C. The frequency range of interest has K number of frequencies, indexed by F = 1, . . . , K, where each index of F has a corresponding frequency. The user imports M number of data files corresponding to each scan angle, indexed by D = 1, . . . , M. The filter response of the matching network is cascaded with the return loss from the imported data, resulting in the overall return loss of the system, $|\Gamma_{DF}|$, which is unique for each frequency index, F, and data index, D. The user sets a targeted return loss, G. Note that both $|\Gamma_{DF}|$ and G are expressed in dB. The objective of the optimizer is to minimize the mean difference squared between G and $|\Gamma_{DF}|$ for every $|\Gamma_{DF}| < G$. This is mathematically expressed as

$$minimize: \quad \frac{\sum\limits_{D=1}^{M} \sum\limits_{F=1}^{K} (|\Gamma_{DF}| - G)^2}{KM}, \tag{4.5}$$

for every $|\Gamma_{DF}| < G$.

4.3.2 Model Constraints and Parameters

Not every arbitrary value of inductance and capacitance is realizable. As such, the model must have the following constraints:

$$\Lambda_{min} \leq \Lambda_n \leq \Lambda_{max} \tag{4.6}$$

$$\varsigma_{min} \leq \varsigma_n \leq \varsigma_{max} \tag{4.7}$$

where Λ_{min}, Λ_{max}, ς_{min}, and ς_{max} are set by the user.

The optimization program includes many parameters discussed throughout the course of this chapter. Table 4.1 lists each parameter and its associated value used in the program.

Table 4.1: Static Program Parameters

Parameter	**Value**
f_c	435 MHz
BW for filter	200 MHz
BW of interest	125 MHz
G	10 dB
Λ_{min}	1.65 nH
Λ_{max}	47 nH
ς_{min}	1 pF
ς_{max}	40 pF

4.4 Implementation in MATLAB

MATLAB was selected as the program of choice because it contains native functionality for both RF circuits and optimization. The Optimization and Global Optimization Toolboxes provide access to many optimizers and algorithms. For non-linear programming, 'fminsearch' is an appropriate function to minimize an unconstrained multi-variable objective function. However, to specify certain constraints, 'fmincon' is used. Additionally, to optimize between specified values, the 'genetic algorithm' can be used. These two optimizers and their applications to the design of matching networks will be discussed in this section.

4.4.1 'Fmincon' optimizer

The optimization model outlined in Sec. 4.3 can be implemented using 'fmincon' quite readily. From [32], the 'fmincon' function is set up to

$$\min_x f(x) \text{ such that } \begin{cases} c(x) \le 0 \\ ceq(x) = 0 \\ A \cdot x \le b \\ Aeq \cdot x = beq \\ lb \le x \le ub\ . \end{cases}$$

Let $f(x)$ be the objective function from (4.5). $c(x)$ and $ceq(x)$ are non-linear constraints that will not be used. A, Aeq, b, and beq are linear constraints that will, likewise, not be used. lb and ub define the bounds of the variables which will take the min and max values of Λ and ς from Table 4.1. In the options of 'fmincon', the step tolerance is set to be 1×10^{-3} so that the optimizer will not concern itself with optimizing to an excessive number of decimal places. With this setup, the matching network design is ready to be optimized.

4.4.2 'Genetic algorithm' optimizer

There is a drawback to using 'fmincon'; there is no guarantee that the inductance and capacitance values that the solver converges to are realizable. To guarantee that, the 'genetic algorithm' must be used. This function provides the option to require decision variables to take only integer values. A matrix of realizable values can be specified, and then, the optimizer can optimize the indices of this matrix.

The 'genetic algorithm' operates by creating mutations from an initial population to find a global minimum of the objective function [33]. To speed up convergence, an initial population can be set by the user. The values determined from the 'fmincon' function are interpolated to the nearest values defined in the matrix of realizable values. Populations are then created from values above and below those of the solution founded by 'fmincon'. The result from the genetic algorithm is an optimized matching network with values that the user has readily available.

4.5 Application to EcoSAR

To demonstrate the impact of the design method, a matching network is designed and simulated using data from NASA's Ecological Synthetic Aperture Radar (EcoSAR) [3], [4]. Fig. 4.3 shows a graphical representation of an array of matching networks for each active element in the 10×2 EcoSAR antenna array, where eight of the columns will be connected to a matching network for both the horizontal and vertical polarizations, resulting in sixteen matching networks. A good impedance match is desired for a bandwidth of 125 MHz and a scanning range of -40° to 40°.

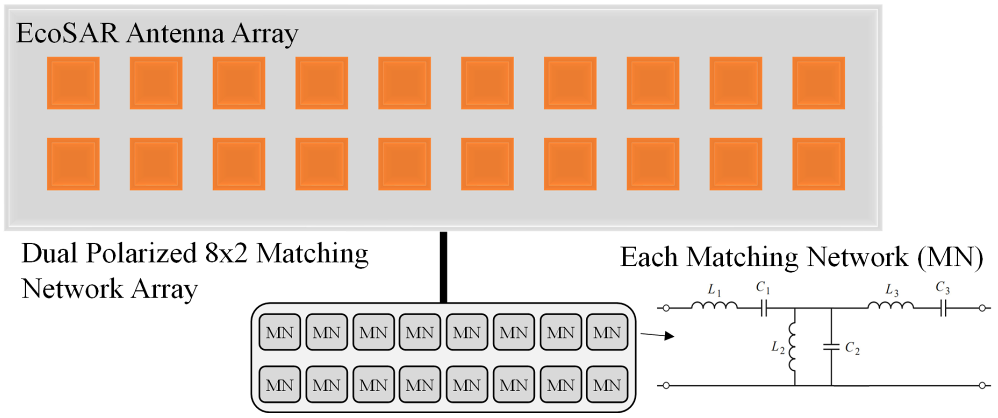
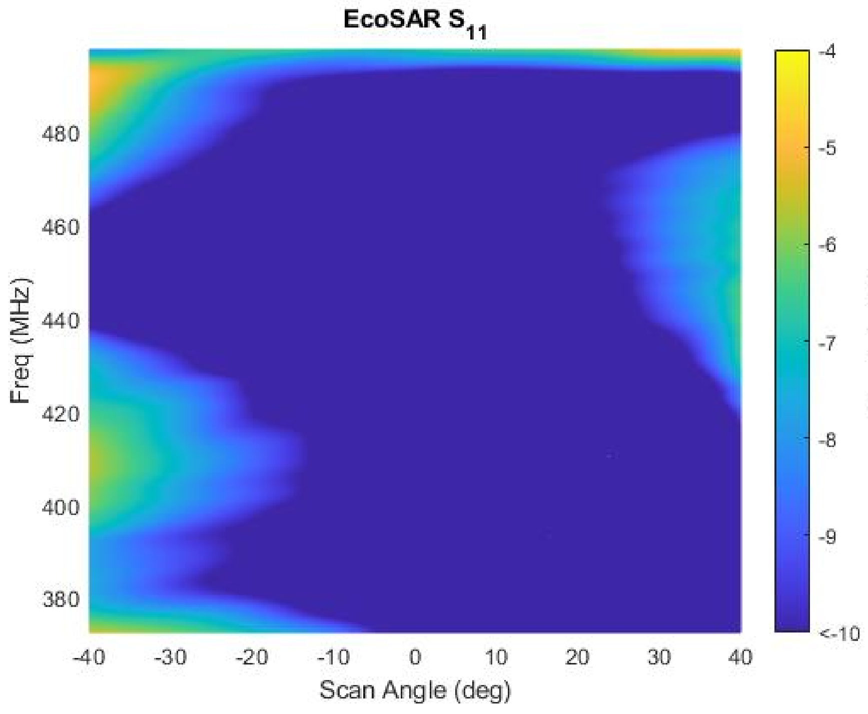

EcoSAR Antenna Array
Dual Polarized 8x2 Matching Network Array
Each Matching Network (MN)
MN
L_1
C_1
L_2
C_2
L_3
C_3

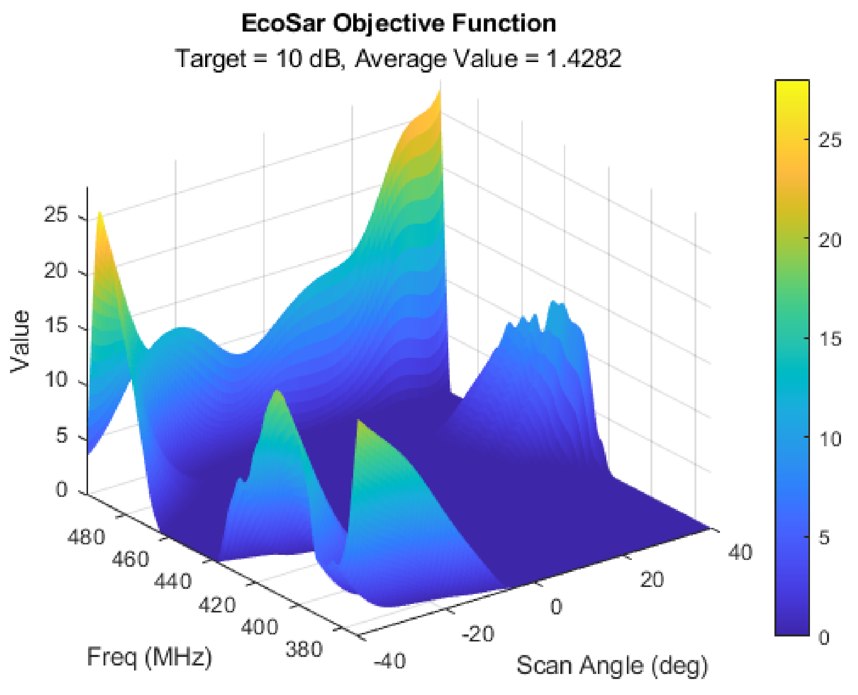
Figure 4.3: Incorporation of a matching network array for the EcoSAR system

4.5.1 Simulated Network

A matching network was designed and simulated to improve the performance of an antenna element from the EcoSAR. Looking at the data from one such element, Fig. 4.4(a) shows that the return loss at the edges of the scanning range gets lower than 5 dB. This is unacceptable for most radar applications, if those scan angles wish to be used. The optimizer program evaluates an average objective function value from (4.5) of 1.4282 for the initial EcoSAR data, displayed in Fig. 4.4(b). A matching network that achieves 10 dB across all scan angles and 125 MHz of bandwidth is desired. Table 4.2 shows the values calculated at every stage of the program. The values from this table correspond to those of the third-order filter shown in Fig. 4.2. The second solver uses inductor values from Coilcraft's air-core series, while the capacitor values are from a range of 0.1 pF to 40 pF at 0.1 pF increments. Table 4.3 shows the average objective function as a function of filter order, illustrating diminishing returns past a third-order. Interestingly, the fourth-order does not perform better than the third-order. This could be because the optimizer reaches a local minimum that does not correspond to the global minimum.



(a)



(b)

Figure 4.4: Performance evaluation of EcoSAR data (a) S_{11} (b) optimizer objective function evaluation

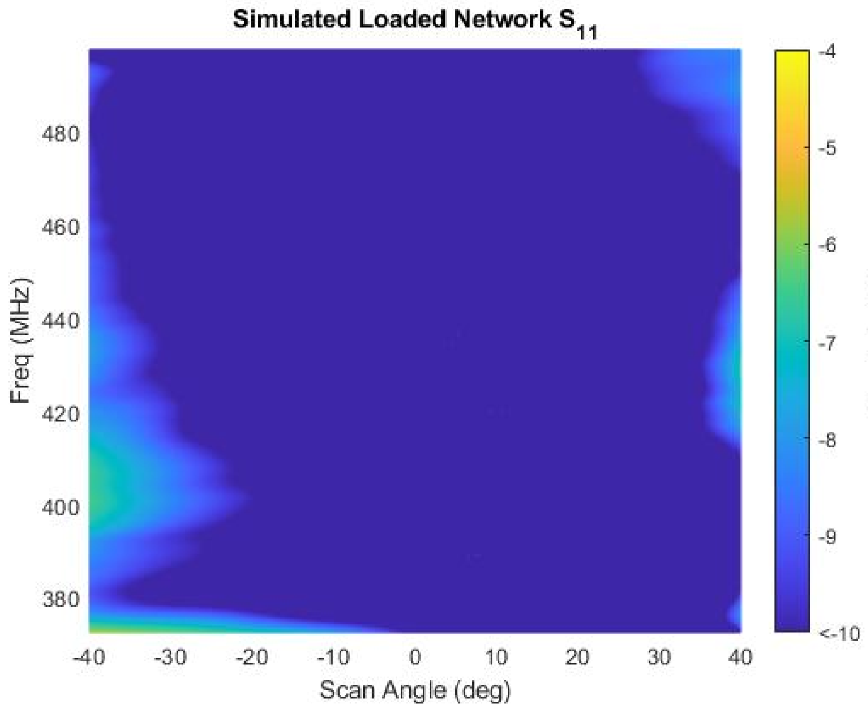
Table 4.2: Simulated Component Values

Values	Prototype	'Fmincon'	'Genetic'
Λ_1 (nH)	39.789	35.138	36.0
Λ_2 (nH)	4.440	6.219	6.0
Λ_3 (nH)	39.789	2.136	1.65
ς_1 (pF)	3.552	3.799	3.7
ς_2 (pF)	31.831	20.679	21.4
ς_3 (pF)	3.552	14.741	15.2

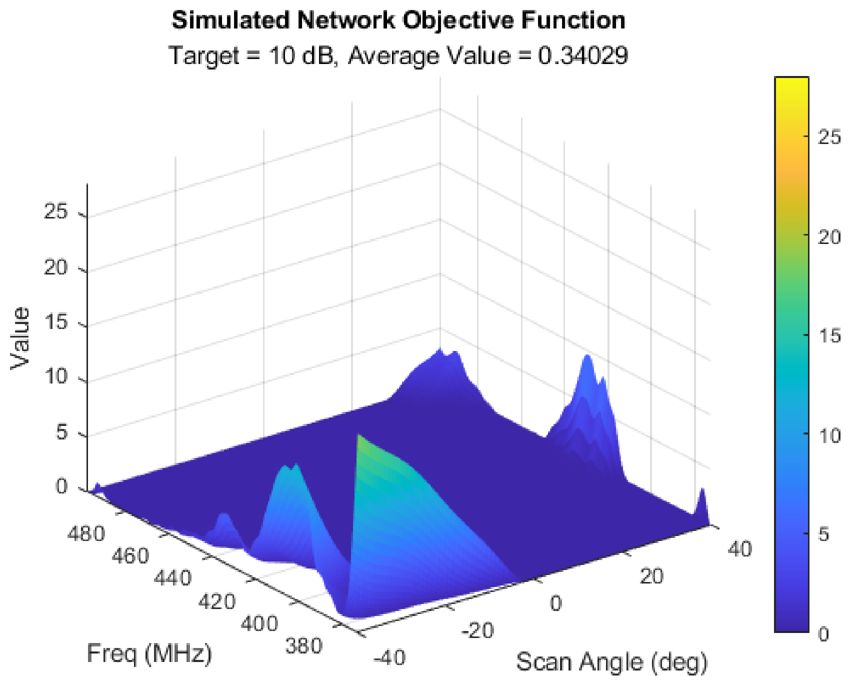
Table 4.3: Effect of Filter Order on Performance

Filter Order (N)	Average Objective Function Value
1	1.0434
2	0.4823
3	0.3407
4	0.3727
5	0.2953
6	0.2657
7	0.2593

The matching network computed by the program was cascaded with the EcoSAR array data. Fig 4.5(a) shows the performance of this network. Return loss is much improved, with a 10 dB match being achieved for a majority of frequencies. With the simulated network, the average objective function value drops to 0.34029, seen in Fig. 4.5(b), a 76.1% decrease from the initial EcoSAR data. This is a great improvement; however, the objective function could not reach zero. This is not a result of the optimizer being unable to generate the most effective matching network, but rather shows the limitations of impedance matching due to the Bode-Fano limit [19], discussed in Sec. 2.5. Additionally, the load across frequency and scan angle varies drastically, and therefore a static matching network will inherently be limited in performance. Regardless of this inherent limitation, the optimized matching network shows great improvement compared to the original EcoSAR data.



(a)



(b)

Figure 4.5: Performance evaluation of simulated network (a) S_{11} (b) optimizer objective function evaluation

4.5.2 Fabricated Network

The network from the previous subsection was fabricated on Rogers 4350 (ϵ_r = 3.48, tan(δ) = 0.0031) with 1.524 mm thickness. The board was milled using the LPKF ProtoMat S104 and electroplated. Coilcraft's air-core inductors and Murata's GJM capacitors were used. The matching network was simulated in ADS using S2P files provided by the manufacturers. Many component values required tuning to account for non-idealities of components and additional reactance of lines. These values are shown in Table 4.4. The fabricated matching network is shown in Fig. 4.6. Details of the fabrication process can be found in Appendix B.

The filter response of the fabricated network shows agreement with the ADS simulation, seen in Fig 4.7. The insertion loss of the fabricated matching network is below 0.5 dB for most of the 125 MHz bandwidth, indicating minimal losses. When cascaded with the antenna data of EcoSAR, the fabricated matching network achieves similarly improved return loss as the simulated network, as seen in Fig. 4.8(a). The average objective function value resulting from the fabricated network is 0.31131, seen in Fig. 4.8(b), a 78.2% decrease from the initial EcoSAR data. The fabricated network outperformed the simulated network, a possible effect of the additional resistance from the transmission lines and components and fabrication tolerances which was not simulated by the optimizer.

Table 4.4: Fabricated Component Values

Values	Simulated	Fabricated
Λ_1 (nH)	36	33
Λ_2 (nH)	6	5.5
Λ_3 (nH)	1.65	1.65
ς_1 (pF)	3.7	3.7
ς_2 (pF)	21.4	16.3
ς_3 (pF)	15.2	15

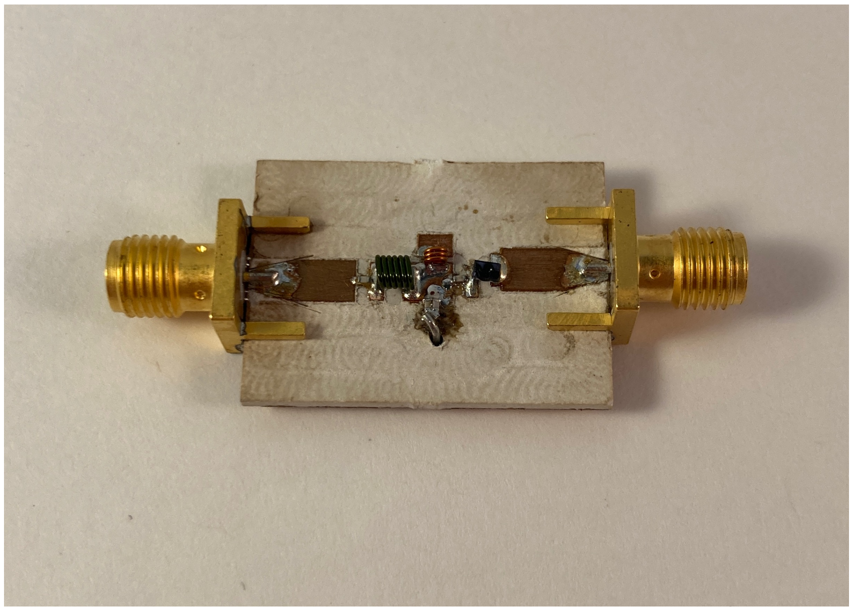


Figure 4.6: Photograph of the fabricated static matching network

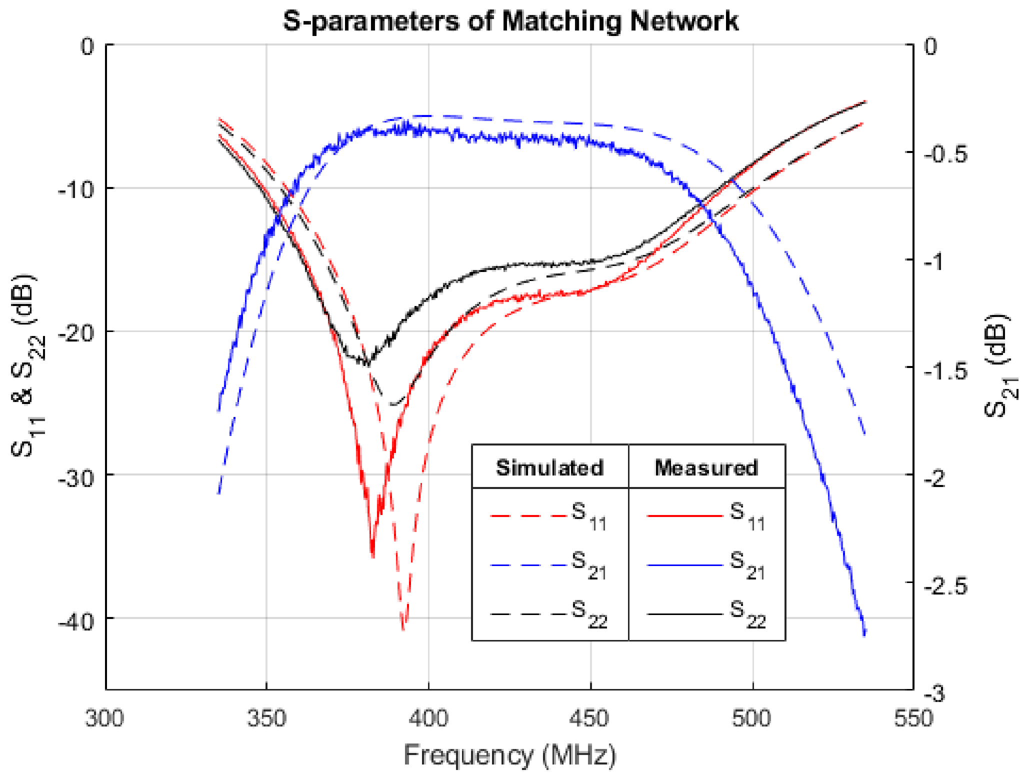
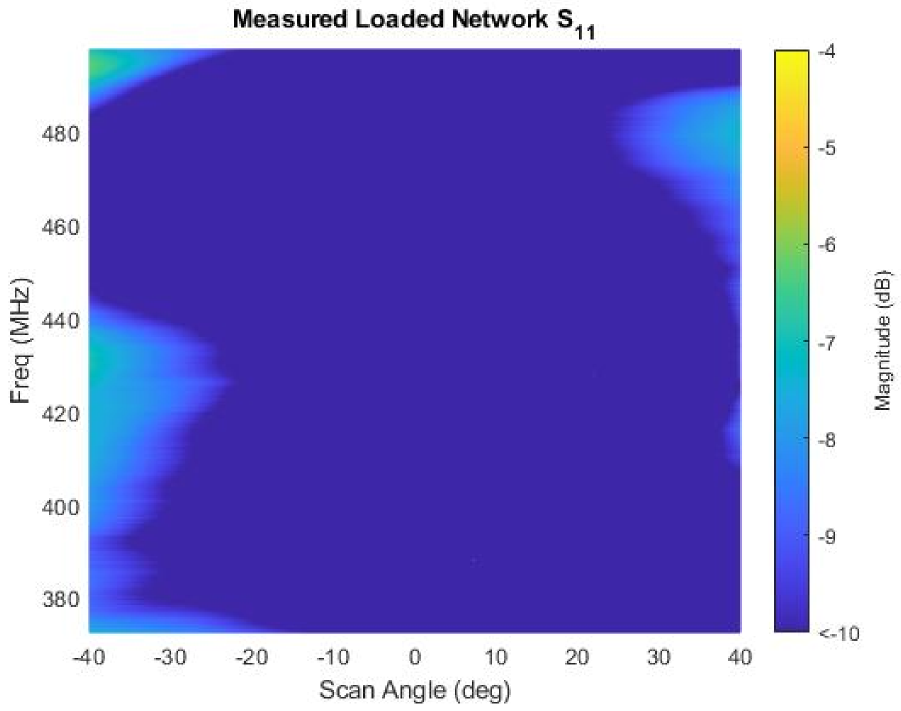
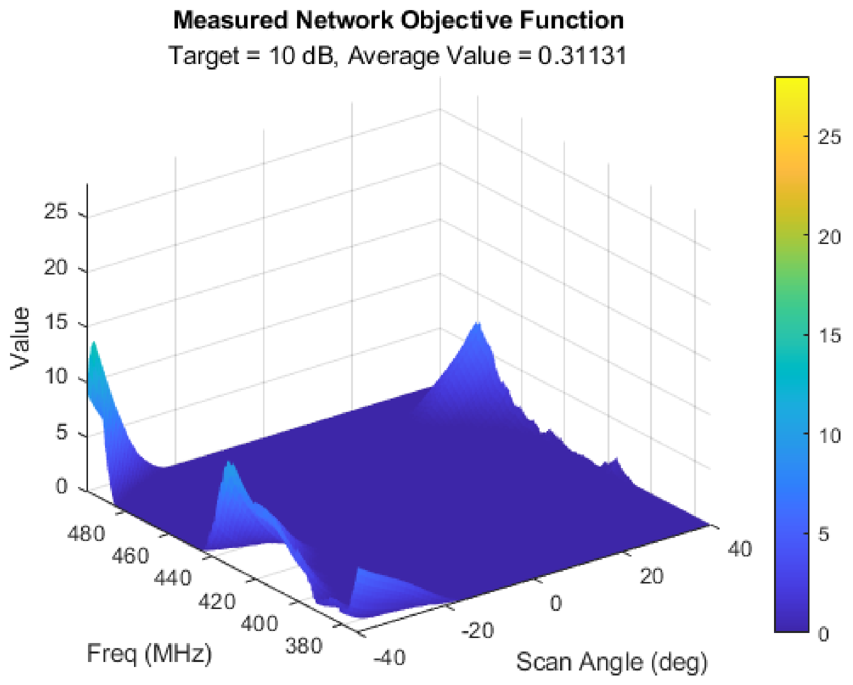


Figure 4.7: Filter responses of the simulated matching network and fabricated matching network, showing similar performance.



(a)



(b)

Figure 4.8: Performance evaluation of fabricated network (a) S_{11} (b) optimizer objective function evaluation

The matching network sufficiently matches to a 28.7% fractional bandwidth, showing impressive broadband matching capabilities. There is a frequency shift between the simulated network and the fabricated network, observed in Fig. 4.7. This 10 MHz shift is to be expected from component tolerances and can impact the performance of the network. However, the improved return loss in Fig. 4.8a verifies that a 2% shift did not impact the improved performance from this design method.

Phase is an important metric within phased array systems. Thus, additional phase added by a matching network must be considered. The phase of the matching network can be well characterized by measurements. Fig. 4.9 shows that the variance of phase shift across five different boards is minimal. Thus, the additional phase can be accurately accounted for by the radar system.

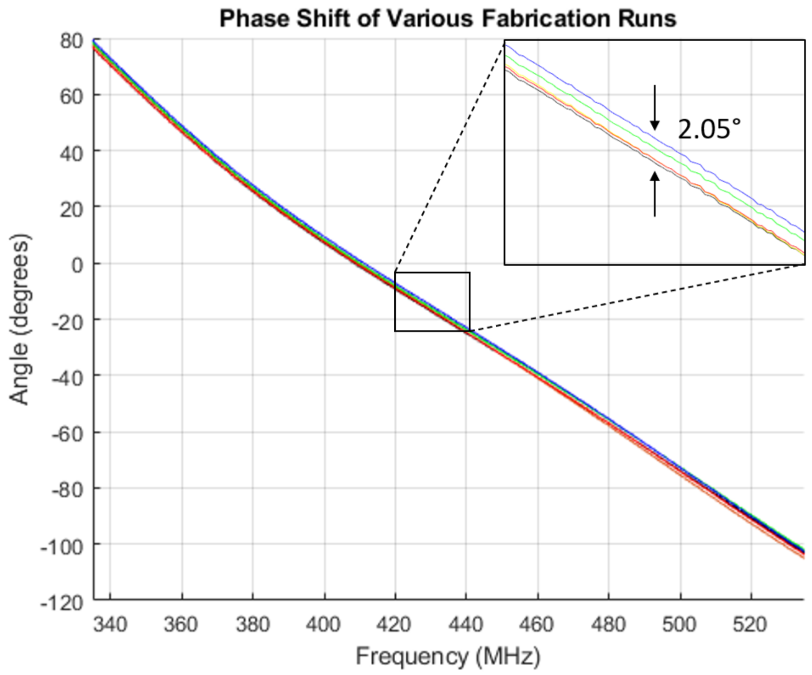
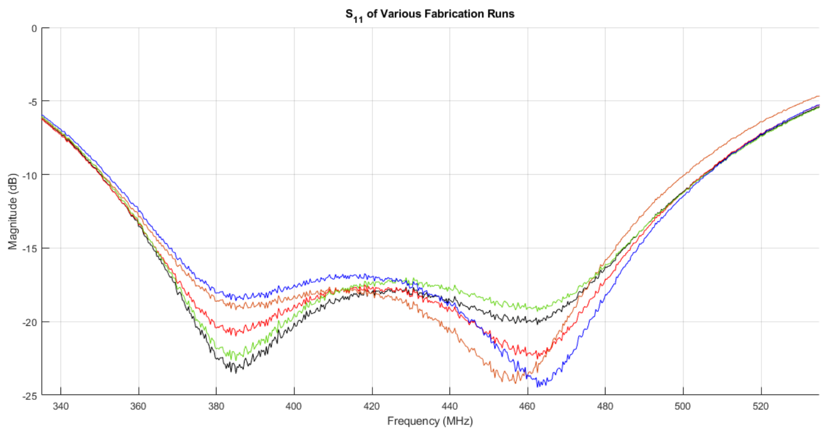


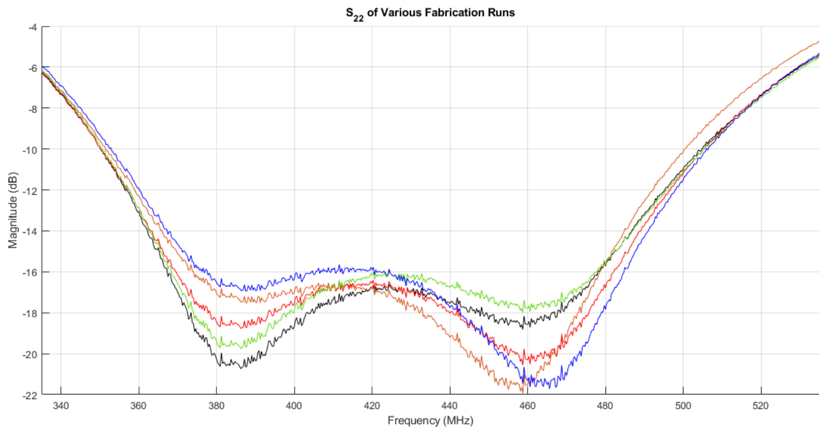
Figure 4.9: Phase shift measured from five different fabrications runs

4.6 Effects of Component Tolerances

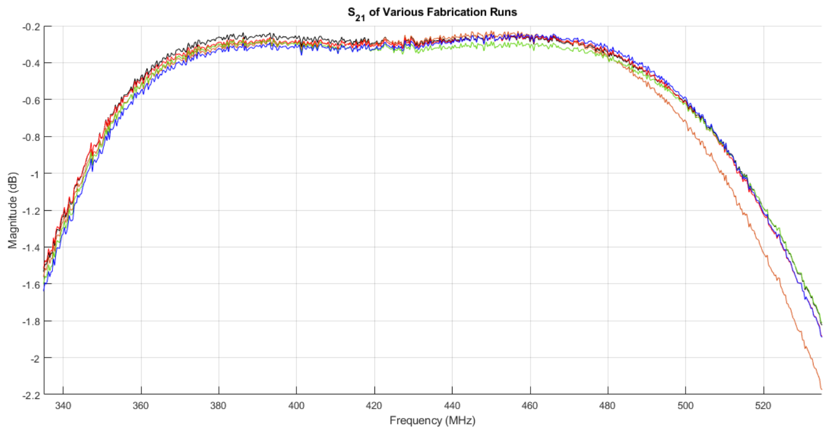
Allowing an optimization program to design a filter has interesting consequences. In conventional filter theory, each resonator of a filter resonates at the same frequency to achieve a higher quality-factor. For a broadband matching network, a high quality-factor is not necessarily desired as it trades off with bandwidth. As such, there is no requirement for each resonator to resonate at the same frequency. Considering the fabricated component values from Table 4.4 and the equation for the resonant frequency of a resonator, $f_r = \frac{1}{2\pi\sqrt{LC}}$, it can be seen that the first, second, and third resonators have resonant frequencies of 455 MHz, 532 MHz, and 1.01 GHz, respectively. While the first two resonators have similar resonant frequencies, the third resonator does not. With three different resonant frequencies, component tolerances for this filter design become more impactful. Fig. 4.10 shows five different filters, fabricated with the same fabrication process and components. Note that the overall shape of the filter responses is similar with slight shifts in magnitude and frequency between the different boards. For filtering, this 1% frequency shift is within tolerance but for broadband matching, this can cause slight performance degradation. The objective function evaluation of these boards differs by 0.2 from best to worst. Despite the sensitivity to fabrication tolerances, all of the fabricated boards improved the impedance match for the EcoSAR antenna element.



(a)



(b)



(c)

Figure 4.10: S-parameters of five different fabricated boards using the same fabrication process and components (a) S_{11}, (b) S_{22}, (c) S_{21}

4.7 Summary

This optimization design method is shown to greatly improve performance and provides user flexibility in the designing process. A simulated matching network, designed using measured data from the EcoSAR array, demonstrated a 76% increase in performance based on the optimizer metrics compared to the original measured data. A fabricated network validates this method, achieving a 78% reduction in the optimizer objective and providing a 10 dB match for the majority of the scanning range of -40° to 40° at a fractional bandwidth of 28.7%. Incorporating an array of these broadband matching networks would greatly improve the performance of wide-scanning phased array antennas. However, the limitation of a static matching network can be seen. A tunable network, while adding additional complexity, would improve performance. Varactor diodes can be used to tune the matching network to impedance match to a desired frequency and scan angle. This approach is presented in the next chapter.

Chapter 5

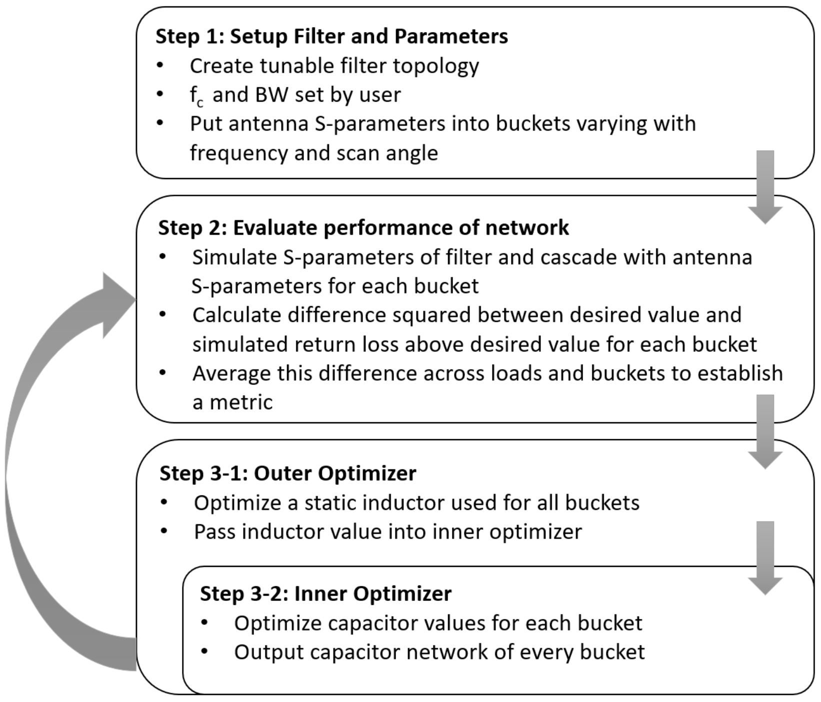
Tunable Matching Network

5.1 Introduction

Oftentimes, a radar system may not use the entire operating bandwidth of the system, but rather a subset. Thus, a matching network that corresponds to that subset need only be used. A tunable matching network can be implemented corresponding to the scan angle and desired bandwidth to achieve higher return loss. In this way, the matching network does not need to impedance match to the entire scanning range and bandwidth simultaneously, as with the approach from the previous chapter. A high-quality match can be achieved using a tunable matching network regardless of scan angle and frequency. The proposed approach is outlined in Fig. 5.1. In this chapter, the same design steps are taken as in Chapter 4: a prototype filter is created and then optimized to the data, the resulting network is simulated using ADS and then fabricated to verify the process.

5.2 Design of Tunable Network

Due to the tunability of varactors, the matching network does not need to utilize as many components as the network of the previous chapter. Rather the topol-


Step 1: Setup Filter and Parameters
- Create tunable filter topology
- f_c and BW set by user
- Put antenna S-parameters into buckets varying with frequency and scan angle

Step 2: Evaluate performance of network
- Simulate S-parameters of filter and cascade with antenna S-parameters for each bucket
- Calculate difference squared between desired value and simulated return loss above desired value for each bucket
- Average this difference across loads and buckets to establish a metric

Step 3-1: Outer Optimizer
- Optimize a static inductor used for all buckets
- Pass inductor value into inner optimizer

Step 3-2: Inner Optimizer
- Optimize capacitor values for each bucket
- Output capacitor network of every bucket

Figure 5.1: Proposed approach to generate optimized tunable matching network

ogy from [10], shown in Fig. 5.2, is used due to the ease of implementation and demonstrated results from that paper. This filter topology is a high-pass filter with a transmission zero resulting from the resonator in the middle.

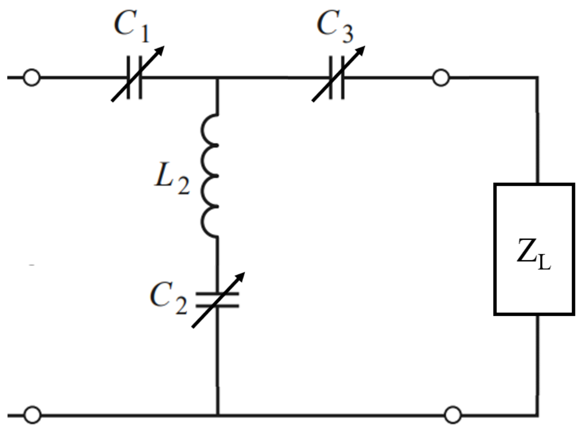


Figure 5.2: Tunable filter prototype, adapted from [10]

The initial component values of the network can be calculated, from [14], as

$$\text{For series capacitors:}\quad C_n = \frac{1}{\omega_0^2 g_n Z_0} \tag{5.1}$$

$$\text{For shunt components:}\quad L_n = \frac{Z_0}{\omega_0 g_n \Delta} \tag{5.2}$$

$$C_n = \frac{g_n \Delta}{\omega_0 Z_0}, \tag{5.3}$$

where g_n is found in Table 8.5 of [14], $\Delta = \frac{\omega_2 - \omega_1}{\omega_0}$, ω_0 is the center frequency, ω_1 is the lower cutoff frequency, ω_2 is the upper cutoff frequency, and Z_0 is the system impedance.

5.3 Optimization Model

In this section, the optimization model for the tunable matching network will be outlined in the same fashion as the static network. The main difference between the optimization model for the tunable matching network compared to the static network is that the frequency ranges are separated into overlapping bins. The overlap ensures good performance at the boundaries of the bins. This is illustrated in Fig. 5.3 for an example range of 372.5 MHz to 497.5 MHz with a bin bandwidth of 50 MHz, comprising six bins.

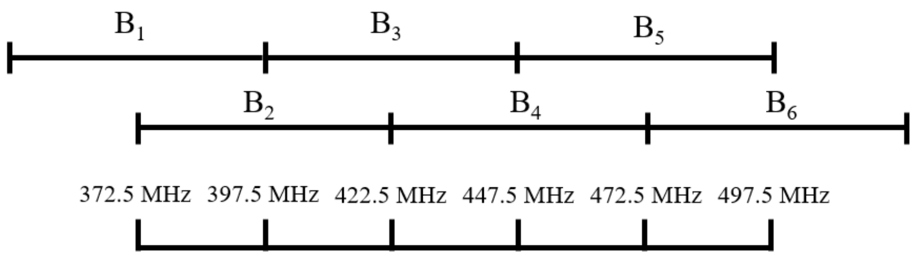


Figure 5.3: Overlapping frequency bins with 50 MHz bandwidth

5.3.1 Decision Variables and Objective Function

The network is composed of a single inductor and three variable capacitors, labeled by n = 1 ,2 , 3. The inductor and capacitors make up a set that will be denoted as L and C, respectively. The decision variables are Λ, corresponding to the inductance of the single inductor, and ς_n, corresponding to the capacitance of varactor n in C. The frequency range is broken up into P number of bins, where each bin is indexed by B = 1, . . . , P. Each frequency bin has K number of frequencies, indexed by F_B = 1, . . . , K, where each index of F_B has a corresponding frequency and varies depending on each bin. The capacitance of each varactor can vary for each bin, but the inductor keeps a constant value for all bins. The user, then, imports M number of data files corresponding to each scan angle, indexed by D = 1 , . . . , M. The filter response of the matching network for each bin is cascaded with the return loss from the imported data, resulting in the overall return loss for that bin, $|\Gamma_{DF_B}|$, which is unique for each frequency index of the bin, F_B, and data index, D. The user sets a targeted return loss, G. Note that both $|\Gamma_{DF_B}|$ and G are expressed in dB. The objective of the optimizer is to minimize the mean difference squared between G and $|\Gamma_{DF_B}|$ for every $|\Gamma_{DF_B}| < G$. This is mathematically expressed as

$$minimize: \quad \frac{\sum\limits_{B=1}^{P}\sum\limits_{D=1}^{M}\frac{\sum\limits_{F_B=1}^{K}(|\Gamma_{DF_B}|-G)^2}{K}}{PM}, \tag{5.4}$$

for every $|\Gamma_{DF_B}| < G$.

5.3.2 Model Constraints and Parameters

Not every arbitrary value of inductance and capacitance is realizable. As such, the model must have the following constraints:

$$\Lambda_{min} \leq \Lambda_n \leq \Lambda_{max} \tag{5.5}$$

$$\varsigma_{min} \leq \varsigma_n \leq \varsigma_{max} \tag{5.6}$$

where Λ_{min}, Λ_{max}, ς_{min}, and ς_{max} are set by the user. ς_{min} and ς_{max} are set to correspond to the upper and lower range of the varactors used.

The optimization program includes many parameters discussed throughout the course of this chapter. Table 5.1 lists each parameter and its associated value used in the program.

Table 5.1: Tunable Program Parameters

Parameter	**Value**
f_c	200 MHz
BW for filter	100 MHz
BW of bins	10/20 MHz
G	15 dB
Λ_{min}	1.65 nH
Λ_{max}	47 nH
ς_{min}	0.71 pF
ς_{max}	22.47 pF

5.4 Implementation in MATLAB

The tunable network optimization uses a different approach than that of the previous chapter. The tunable optimization program uses nested optimizers, compared to the static optimization program which used two independent optimizers. The inner optimizer tunes the values of the varactors for each frequency bin and for each scan angle. It uses the 'fmincon' optimizer since varactors can achieve precise values. The outer optimizer tunes the value of the single inductor and uses the 'genetic algorithm' optimizer since the inductors must be discrete values. These two optimizers will be discussed in this section.

5.4.1 Inner optimizer

The inner optimizer uses the same setup from Sec. 4.4.1. It produces an optimized matching network for each frequency bin and scan angle. This simulates the effect of varactors as the capacitance of the varactors can be changed to match the tuned capacitance values found by the inner optimizer. The inductance value is kept constant across each frequency bin and scan angle. For the specific optimizer settings used, refer to Sec. 4.4.1.

5.4.2 Outer optimizer

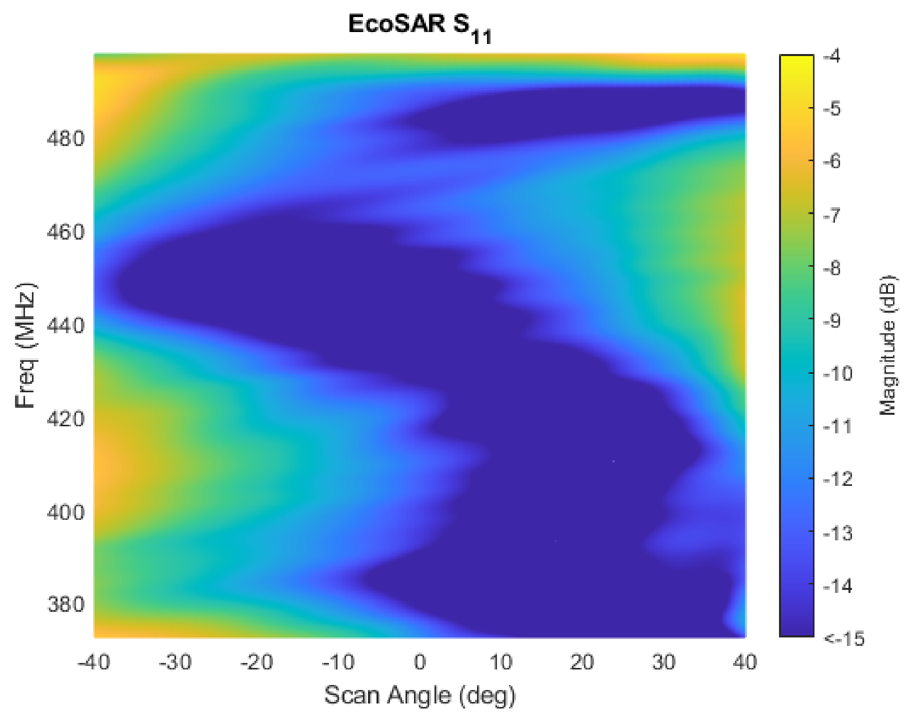
The outer optimizer tunes the inductor of the matching network to find the optimal value. It takes the resulting objective function value from each frequency bin and scan angle found by the inner optimizer and determines the best values from the overlap in the frequency bins. These overlaps in the frequency bins exist to improve optimizer performance at the edges of the bins. It was found that using a model that did not include overlaps ran into boundary issues, in which the optimizer would have trouble impedance matching at the edges of each bin. Thus, overlaps are included and the outer optimizer selects the capacitor network that performs the best from the two frequency bins. The outer optimizer uses the 'genetic algorithm' optimizer, refer to Sec. 4.4.2 for specifics on how this algorithm was used.

5.5 Application to EcoSAR

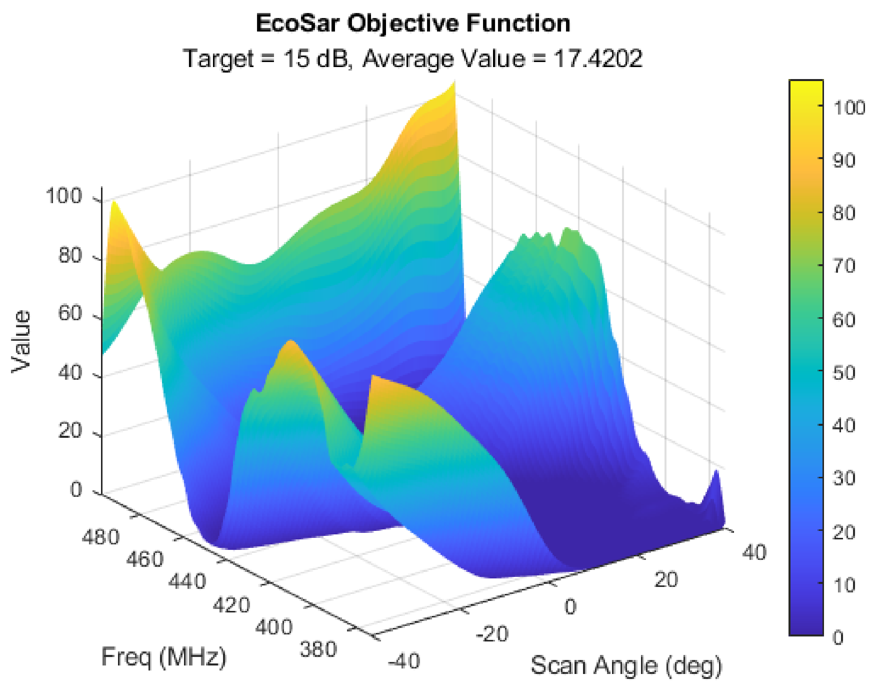
Like the static matching network, the end goal of the tunable matching network is improving the return loss for the EcoSAR antenna array. The tunable matching network would be configured in the radar system the same way as described in Sec. 4.5 with additional circuitry to control the biasing of each varactor. The EcoSAR has a programmable bandwidth over a 200 MHz range with the the smallest operational bandwidth of 6 MHz [4]. The desired performance is at least 15 dB of return loss across 125 MHz bandwidth, divided into 10-20 MHz bins. The desired return loss was increased from the desired 10 dB of the static matching network due to the improved performance possible from a tunable network.

5.5.1 Simulated Network

A tunable matching network was designed and simulated using data from the EcoSAR array. Looking at the same data from Sec. 4.5.1, Fig. 5.4(a) shows that most of the data does not achieve 15 dB of return loss. The optimizer program evaluates an average objective function value of 17.4202 for the initial EcoSAR data relative to 15 dB, displayed in Fig. 5.4(b). This evaluation shows that the element is not achieving high performance. The tunable optimizer program was run using this data. The capacitor values are bounded by the minimum and maximum capacitance values of Skyworks' SMV1265-040LF varactor, corresponding to 0.71 pF and 22.47 pF, respectively. The inductor values are taken from Coilcraft's air-core series.



(a)

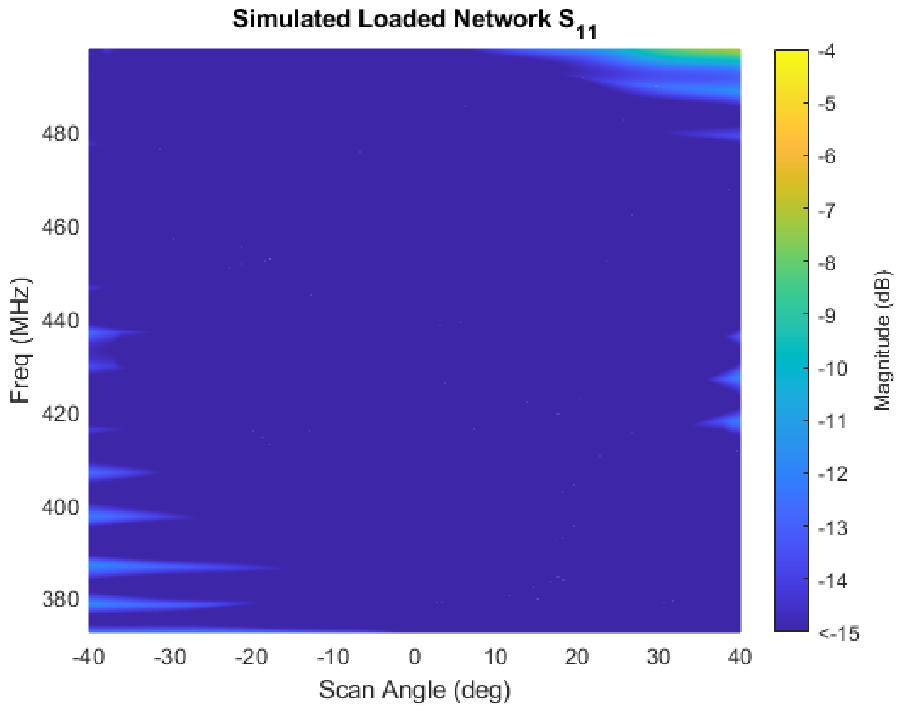


(b)

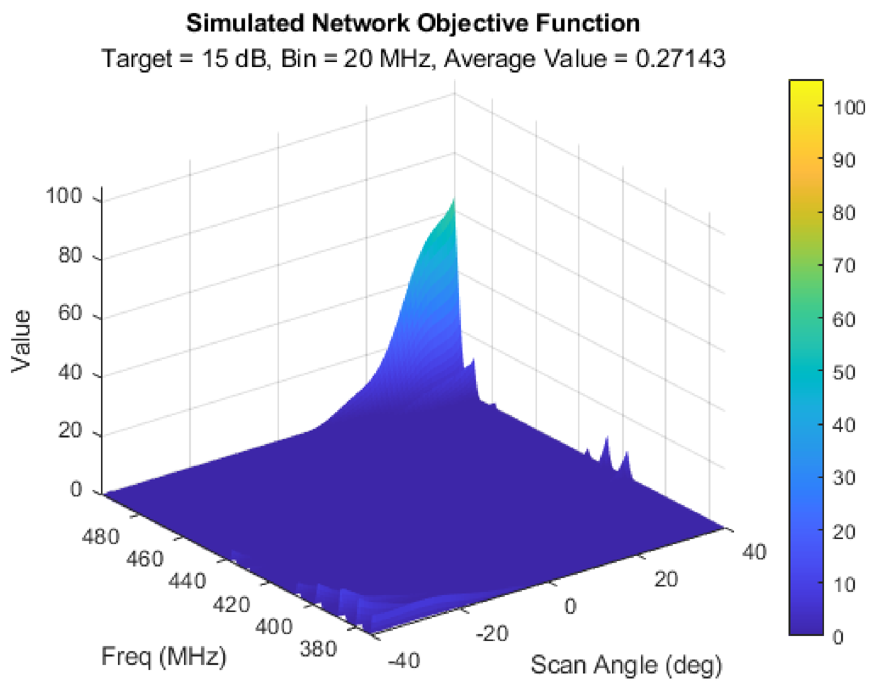
Figure 5.4: Performance evaluation of EcoSAR data (a) S_{11} (b) optimizer objective function evaluation

The frequency bin was initially set to 20 MHz, thus a 10 MHz tunable bandwidth is achieved when accounting for the bin overlap. The matching network outputted by the program was cascaded with the EcoSAR array data. Fig 5.5(a) shows the performance of this network. Return loss is improved, with a 15 dB match being achieved for almost the entirety of the frequencies and scan angles of interest. The average objective function value with the simulated network drops to 0.27143, seen in Fig. 5.5(b), a 98.44% decrease from the initial EcoSAR data. This shows that the tunable matching network can tune to almost the entire frequency and scanning range. The poorest performance is observed at 500 MHz at a scan angle of 40°. Since the optimization program is customizable, the frequency bin bandwidth can be reduced to further improve the impedance match.

The frequency bin is reduced from 20 MHz to 10 MHz to improve performance. Fig 5.6(a) shows the performance of this network. Return loss is improved further. Using a 10 MHz frequency bin, the average objective function value drops to 0.0088786, seen in Fig. 5.6(b), a 99.95% decrease from the initial EcoSAR data. This shows that the tunable matching network can tune to almost any impedance for this antenna element. This tunable approach, thus, pushes past the limitations of the static matching network described in Sec. 4.5.1. Adhering to the Bode-Fano limit, the bandwidth of interest is reduced to achieve a higher quality match.

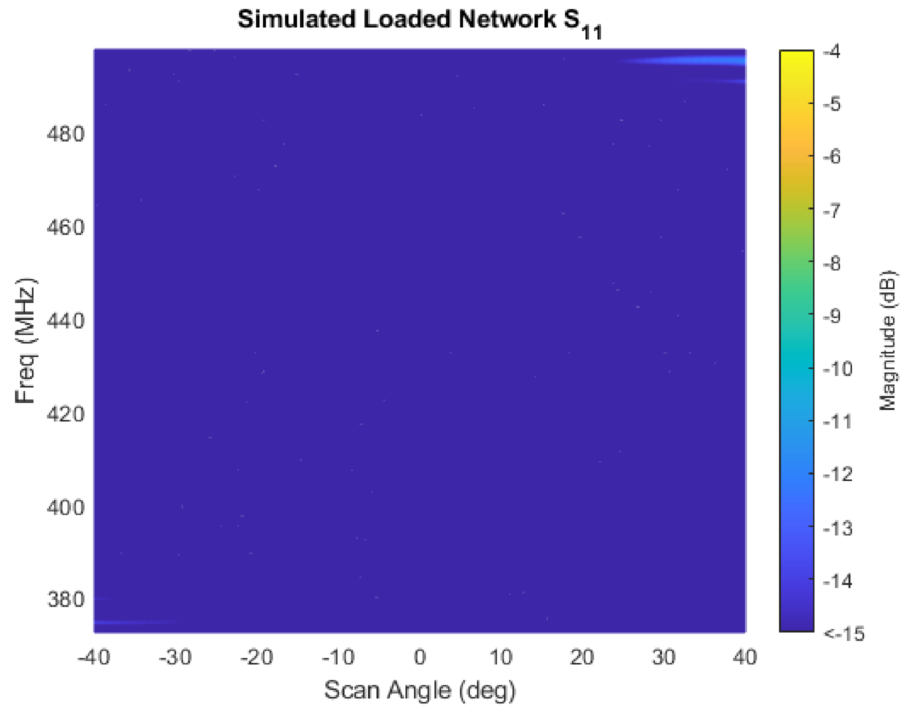


(a)

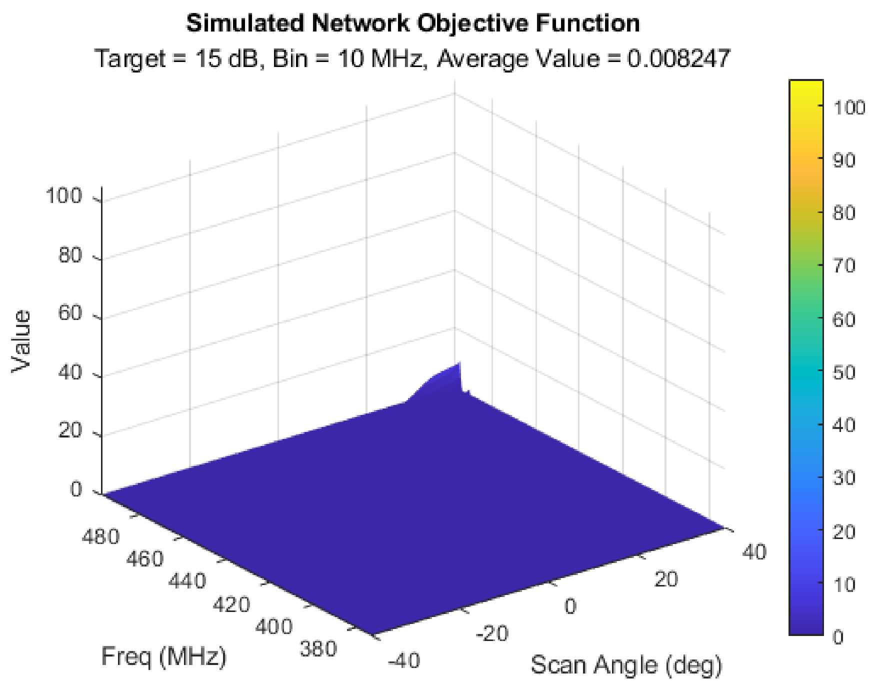


(b)

Figure 5.5: Performance evaluation of simulated network with frequency bin of 20 MHz (a) S_{11} (b) optimizer objective function evaluation



(a)



(b)

Figure 5.6: Performance evaluation of simulated network with frequency bin of 10 MHz (a) S_{11} (b) optimizer objective function evaluation

5.5.2 Fabricated Network

The network from the previous subsection was fabricated on Rogers 4350 (ϵ_r = 3.48, tan(δ) = 0.0031) with 1.524 mm thickness. The board was milled using the LPKF ProtoMat S104 and electroplated. Coilcraft's air-core inductors and Skyworks' SMV1265-040LF varactors were used. The matching network was simulated in Keysight ADS using S2P files provided by Coilcraft and a diode model provided by Skyworks. Fig. 5.7 shows the schematic of the tunable matching network. Bias lines and RF chokes were added to operate the varactors. Tuning was performed to achieve agreement between the ideal capacitances of the MATLAB program, the non-ideal ADS simulation of the board, and the fabricated board, seen in Fig. 5.8. Due to the massive amount of data points required to characterize the fabricated board, only a selection of impedance-matched data points are shown in this section.

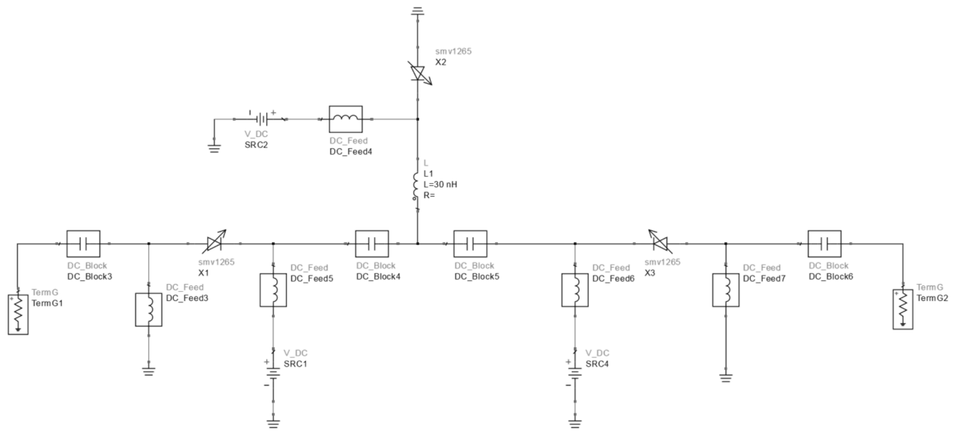


Figure 5.7: Schematic of the tunable matching network

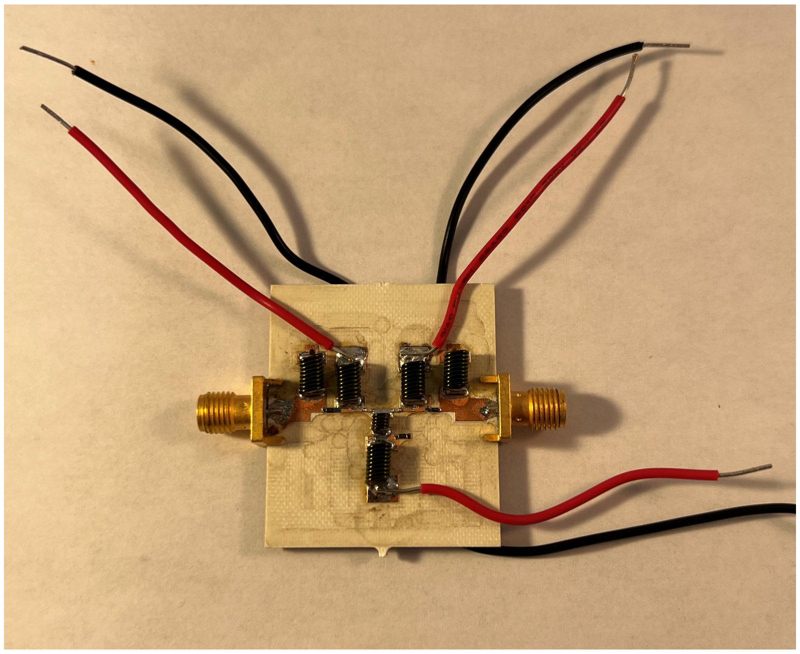


Figure 5.8: Photograph of the fabricated tunable matching network

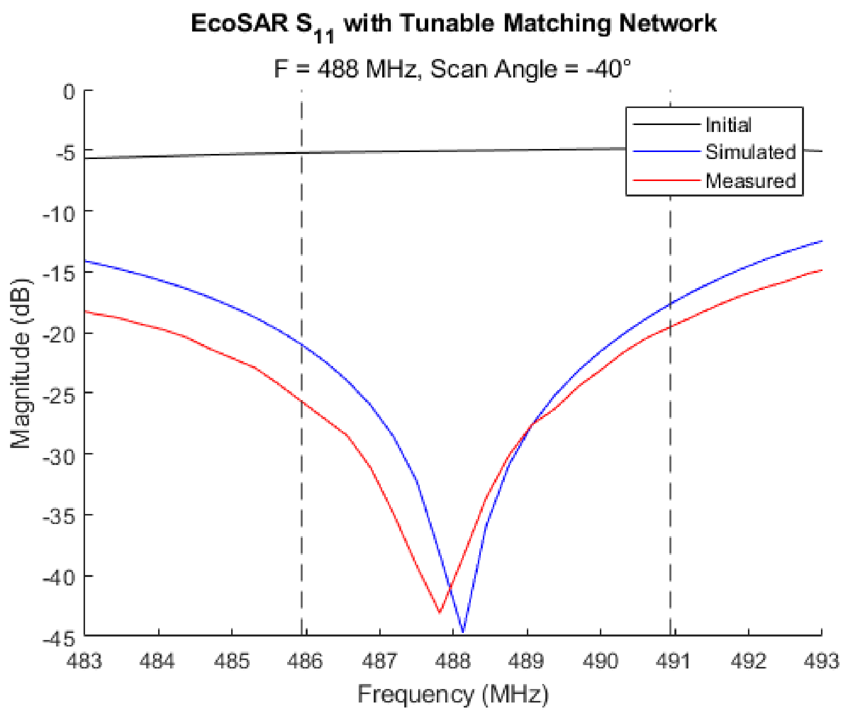


Figure 5.9: Return loss of 5 dB improved to 5 MHz of 20 dB match

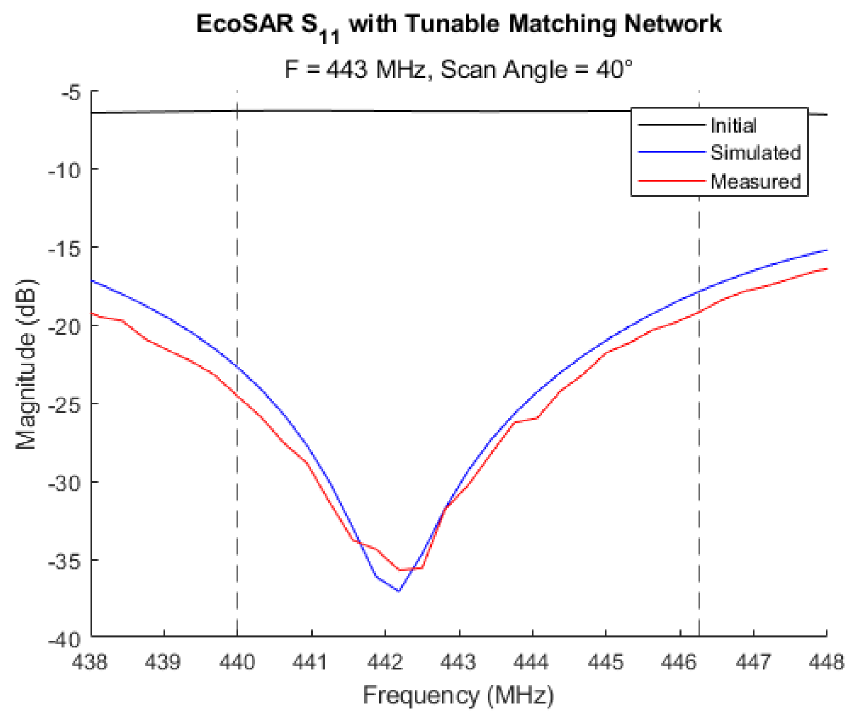


Figure 5.10: Return loss of 6 dB improved to 6 MHz of 20 dB match

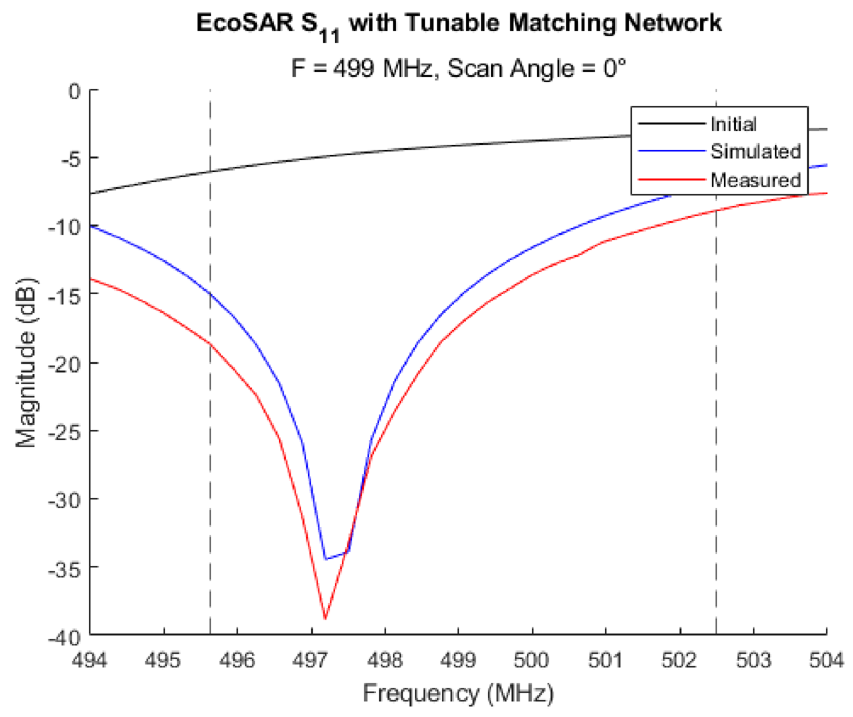


Figure 5.11: Return loss of 4 dB improved to over 7 MHz of at least 10 dB match

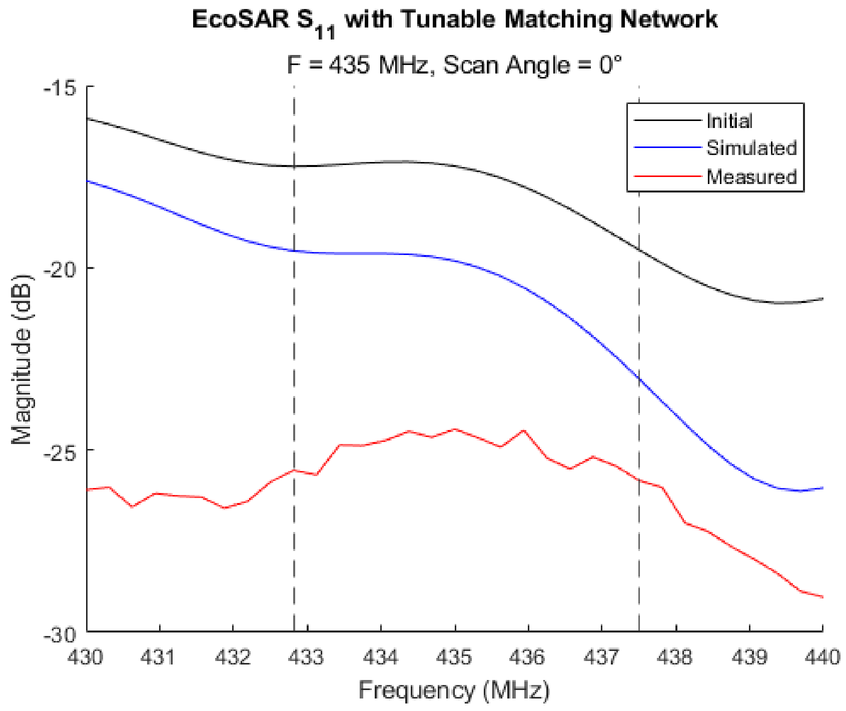


Figure 5.12: Return loss of 17 dB improved to 25 dB match

The filter response of the tunable network is tuned and then cascaded with the EcoSAR data for various points within the frequency and scanning range. The fabricated tunable matching network performs well, exceeding the simulations at times, as seen in Figs. 5.9 - 5.12. Note that the dashed lines in these figures represent the range within the frequency bin for which the program guarantees the best performance. Fig. 5.9 shows an initial return loss of 5 dB improved to 5 MHz of 20 dB match. Fig. 5.10 shows an initial return loss of 6 dB improved to 6 MHz of 20 dB match. Fig. 5.11 shows an initial return loss of 4 dB improved to over 7 MHz of at least 10 dB match. Fig. 5.12 shows an already matched initial return loss of 17 dB improved to 25 dB. The worst matched data points from the EcoSAR see drastic improvements. Additionally, the tunable network is not shown to degrade performance where the array is already matched well, as seen in Fig. 5.12.

5.6 Summary

Pairing the optimization design process with a tunable matching network topology increases the return loss of the EcoSAR array significantly. An impressive 99.95 % improvement in the objective function shows that an impedance match can be made despite the mutual coupling resulting from wide-scanning arrays. The tunability of the network allows the user to control the bandwidth of interest and, thus, the quality of the impedance match as described by the Bode-Fano limit, reviewed in Sec. 2.5. This configuration does not suffer from the problems associated with component tolerances, discussed in Sec. 4.6, since these tolerances can be tuned out. The tunable matching network presented in this work shows great promise in improving the performance of wide-scanning phased array antennas.

Chapter 6

Future Work and Conclusion

6.1 Conclusion

As phased array antennas continue to be used for scientific and military applications, wider scanning and broader band systems will be desired. However, these systems suffer from impedance mismatch and degraded return loss resulting from mutual coupling arising from the configuration of the antenna array. While the performance of these systems may be adequate without alteration, these limitations may hinder pushing the scanning range and bandwidth of future systems. Thus, a matching network that addresses these issues was introduced in this body of work by utilizing computer optimization, a powerful tool that can address problems which conventional theory cannot presently.

An optimized static matching network was designed by importing measured data from a phased array element. Through this method, great improvement in return loss was achieved. When this method was implemented for an array element for NASA's EcoSAR array, it demonstrated a 78% reduction in the optimizer objective and provided a 10 dB match for the majority of the scanning range of -40° to 40° at a fractional bandwidth of 28.7%. However, there is an inherent limitation to the degree that this network can match given a wide scanning range and broad bandwidth. Thus, a tunable matching network was explored to provide greater matching performance.

A tunable matching network was designed to impedance match to a subset of bandwidth to realize higher return loss. This utilizes the Bode-Fano limit, achieving a higher quality impedance match through reducing the bandwidth of interest. Since a radar system may not use the entirety of its bandwidth simultaneously, a tunable network can impedance match the frequency range that the system is using. This tunable network demonstrated a 99.95% reduction in the optimizer objective function, providing a 15 dB match for virtually the entire scanning and frequency range. Implementing an array of these optimized tunable matching networks will improve the performance of phased array systems and allow for wider scanning ranges and broader bandwidths for future systems.

6.2 Future Work

The novelty of this research is the design method presented. Thus, multiple routes may be taken in expanding upon this research: either improving the underlying optimization, applying this method to other filter topologies, or providing automatic tuning functionality. It would be of interest to improve the optimization engine presented in this work. MATLAB was used as the program of choice due to familiarity and impressive RF functions, however, other programs may provide more efficient and effective optimization capabilities. Alternatively, the design method used could be applied to other filter topologies. A lumped element filter was used due to the relatively low operating frequency of the EcoSAR system. However, for systems built for higher frequencies, the code could be adapted to optimize tunable distributed filters. In its present stage, the tunable matching network uses an open-loop controller. Designing a closed-loop controller to integrate into the radar system is envisioned for the tunable matching network. It was also found that bias voltages could be tuned by hand to improve performance beyond that of the optimized values. It would be of interest to design a feedback loop for the network to automatically adjust the bias voltages to improve performance beyond that of the optimizer.

References

[1] M. I. Skolnik, *Introduction to Radar Systems*, 3rd Edition. McGraw-Hill, 1980.

[2] M. I. Skolnik, *Radar Handbook*, 3rd Edition. McGraw-Hill, 2008.

[3] T. Fatoyinbo, R. F. Rincon, G. Sun, and K. J. Ranson, "EcoSAR: A P-band digital beamforming polarimetric interferometric SAR instrument to measure ecosystem structure and biomass", in *2011 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, 2011, pp. 1524–1527.

[4] M. L. Perrine, R. Rincon, T. Fatoyinbo, *et al.*, "Development of the RF electronics unit for NASA's ecological synthetic aperture radar", in *2013 IEEE International Symposium on Phased Array Systems and Technology*, Oct. 2013, pp. 191–197.

[5] R. F. Rincon, T. Fatoyinbo, G. Sun, *et al.*, "The EcoSAR P-band Synthetic Aperture Radar", in *2011 IEEE International Geoscience and Remote Sensing Symposium*, 2011, pp. 1512–1515.

[6] C. F. du Toit, M. Deshpande, and R. F. Rincon, "Advanced antenna design for NASA's EcoSAR instrument", in *2016 IEEE International Symposium on Phased Array Systems and Technology (PAST)*, 2016, pp. 1–7.

[7] *Design Broadband Matching Networks for Antennas*. [Online]. Available: https://www.mathworks.com/help/rf/ug/designing-broadband-matching-networks-part-1-antenna.html.

[8] D. Qiao, R. Molfino, S. Lardizabal, B. Pillans, P. Asbeck, and G. Jerinic, "An intelligently controlled RF power amplifier with a reconfigurable MEMS-varactor tuner", *IEEE Transactions on Microwave Theory and Techniques*, vol. 53, no. 3, pp. 1089–1095, 2005.

[9] P. Rodriguez-Garcia, J. Sifri, C. Calabrese, C. Baylis, and R. J. Marks, "Range Improvement in Single-Beam Phased Array Radars by Amplifier Impedance Tuning", in *2021 IEEE Texas Symposium on Wireless and Microwave Circuits and Systems (WMCS)*, 2021, pp. 1–6.

[10] Z. Hays, C. Kappelmann, S. Rezayat, *et al.*, "Real-time amplifier optimization algorithm for adaptive radio using a tunable-varactor matching network", in *2017 IEEE Radio and Wireless Symposium (RWS)*, 2017, pp. 215–217.

[11] J.-S. Fu and A. Mortazawi, "Improving Power Amplifier Efficiency and Linearity Using a Dynamically Controlled Tunable Matching Network", *IEEE Transactions on Microwave Theory and Techniques*, vol. 56, no. 12, pp. 3239–3244, 2008.

[12] H. M. Nemati, C. Fager, U. Gustavsson, R. Jos, and H. Zirath, "Design of Varactor-Based Tunable Matching Networks for Dynamic Load Modulation of High Power Amplifiers", *IEEE Transactions on Microwave Theory and Techniques*, vol. 57, no. 5, pp. 1110–1118, 2009.

[13] C. Bowick, *RF Circuit Design*, 2nd Edition. Elsevier/Newnes, 2008.

[14] D. Pozar, *Microwave Engineering*, 4th edition. Wiley, 2012.

[15] T. H. Lee, *Planar Microwave Engineering*. Cambridge University Press, 2004.

[16] B. Schweber. "Impedance matching and the Smith Chart". (2017), [Online]. Available: https://www.analogictips.com/impedance-matching-smith-chart-part-2/.

[17] G. Matthaei, L. Young, and E. Jones, *Microwave Filters, Impedance-matching Networks, and Coupling Structures*. Artech House Books, 1980.

[18] H. Bode, *Network Analysis and Feedback Amplifier Design*, ser. Bell Telephone Laboratories series. D. Van Nostrand Company, Incorporated, 1945.

[19] R. Fano, *Theoretical Limitations on the Broadband Matching of Arbitrary Impedances*. Massachusetts Institute of Technology, Research Laboratory of Electronics, 1947.

[20] S. E. Sussman-Fort and R. M. Rudish, "Non-Foster Impedance Matching of Electrically-Small Antennas", *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 8, pp. 2230–2241, 2009.

[21] M. M. Jacob, “Non-Foster Circuits for High Performance Antennas: Advantages and Practical Limitations”, Ph.D. dissertation, University of California, San Diego, 2016.

[22] P. de Fermat, *Abhandlungen über Maxima und Minima (1629)*, ser. Ostwalds Klassiker der exakten Wissenschaften. Akademische verlagsgesellschaft m. b. h., 1934, M. Miller, reprint from original.

[23] M. N. T. George B. Dantzig, *Linear programming*, eng, ser. Springer series in operations research. New York: Springer, 1997.

[24] H. W. Kuhn and A. W. Tucker, “Nonlinear programming”, in *Traces and emergence of nonlinear programming*, Springer, 2014, pp. 247–258.

[25] D.-Z. Du, P. M. Pardalos, and W. Wu, “History of optimizationhistory of optimization”, in *Encyclopedia of Optimization*, C. A. Floudas and P. M. Pardalos, Eds. Boston, MA: Springer US, 2009, pp. 1538–1542.

[26] B. Kolman and D. Hill, *Elementary Linear Algebra with Applications*, 9th edition, ser. Featured Titles for Linear Algebra. Pearson Prentice Hall, 2008.

[27] *Linprog*. [Online]. Available: https://www.mathworks.com/help/optim/ug/linprog.html.

[28] S. Bradley, A. Hax, A. Hax, and T. Magnanti, *Applied Mathematical Programming*. Addison-Wesley Publishing Company, 1977.

[29] R. H. Byrd, J. C. Gilbert, and J. Nocedal, “A trust region method based on interior point techniques for nonlinear programming”, *Mathematical programming*, vol. 89, no. 1, pp. 149–185, 2000.

[30] *Tuning, Optimization, and Statistical Design*. [Online]. Available: http://edadownload.software.keysight.com/eedl/ads/2011/pdf/optstat.pdf.

[31] *Lecture 6: Introduction to Optimetrics*. [Online]. Available: http://www.ece.uprm.edu/~rafaelr/inel6068/HFSS/HFSS_Antenna_v2015_v1/lectures_trainee/ANSYS_HFSS_Antenna_L06_0_Optimetrics.pdf.

[32] *Fmincon*. [Online]. Available: https://www.mathworks.com/help/optim/ug/fmincon.html.

[33] *How the Genetic Algorithm Works*. [Online]. Available: https://www.mathworks.com/help/gads/how-the-genetic-algorithm-works.html.

Appendix A

MATLAB Code

A.1 Static Matching Network Program

```
%% Description
% Finds optimized component values for static matching network
% Required information:
%   User variables: outlined in code
%   S-parameters: this code takes an excel sheet to
%                 extract s-parameter data, format is
%                 shown in Appendix A.3
%% Initialization & Variables
clc;
clear all;
close all;

% User variables
fc = 435e6;                    % Center freq (Hz)
BW_design = 200e6;             % BW for filter (Hz)
BW_interest = 125e6;           % BW of interest (Hz)
Zs = 50;                       % Source impedance (ohm)
Z0 = 50;                       % System impedance (ohm)
N = 3;                         % Filter order
Goal = -10;                    % Goal S11 (dB)
L_lb = 1.8;                    % Lower bound of inductance
L_ub = 47;                     % Upper bound of inductance
C_lb = 1;                      % Lower bound of capacitance
C_ub = 40;                     % Upper bound of capacitance

% Frequency calculations
fLower = fc - (BW_design/2);
fUpper = fc + (BW_design/2);
nfreq = (fUpper-fLower)/0.3125*10^-6+1;
freq = linspace(fLower,fUpper,nfreq);
w = 2*pi*freq;
wU = 2*pi*fUpper;
wL = 2*pi*fLower;
```

```
w0 = sqrt(wL*wU);

% Take excel file and extract s-parameters
[file,path] = uigetfile('*.xlsx');
[freq_data, s11_data] = sweep_excel_to_s1p_func(file);

% Define optimization freq range
freq_lower_interest = fc-0.5*BW_interest;
freq_upper_interest = fc+0.5*BW_interest;
lower_interest = find(freq_data==freq_lower_interest);
upper_interest = find(freq_data==freq_upper_interest);
freq_interest = freq_data(lower_interest:upper_interest);

% Find indexes for optimized freq bounds
[rowLower,colLower] = find(freq_data==freq_lower_interest);
[rowUpper,colUpper] = find(freq_data==freq_upper_interest);

% Grab data for optimized freq range
for i = 1:81
    % Select s11 data that is within passband
    s11_interest{i} = s11_data{i}(rowLower:rowUpper);

    % Load impedance of interest
    Zl_interest{i} = Z0.*(1+s11_interest{i})./...
        (1-s11_interest{i});

    % Load impedance of all data
    Zl_data{i} = Z0.*(1+s11_data{i})./(1-s11_data{i});
end
%% Calculate LMS of initial network
for i = 1:81
    db_s11_data{i} = db(abs(s11_data{i}));
    for k = 1:length(db_s11_data{i})
        if db_s11_data{i}(k) ≤ Goal
           LMS_initial{i}(k) = 0;
        else
           LMS_initial{i}(k) = (abs(db_s11_data{i}(k)...
               - Goal))^2;
        end
    end
    LMS_initial_interest{i} = LMS_initial{i}...
        (rowLower:rowUpper);
end
%% Filter prototype
% Filter design
% Butterworth coefficients
LCproto = butter_g(N);

% Initialize Lvals and Cvals matrices
Lvals = zeros(1,N);
Cvals = zeros(1,N);
```

```

% Series L
Lvals(1:2:end) = LCproto(1:2:end).*Zs./(wU-wL);
% Series C
Cvals(1:2:end) = (wU-wL)./(Zs.*(w0^2).*LCproto(1:2:end))
% Shunt L
Lvals(2:2:end) = ((wU-wL)*Zs)./((w0^2).*LCproto(2:2:end));
% Shunt C
Cvals(2:2:end) = LCproto(2:2:end)./((wU-wL).*Zs);


% Create the matching network
matchingNW_opt1 = lcladder('bandpasstee',Lvals,Cvals);

% Save initial values for comparison
L_initial = Lvals;
C_initial = Cvals;
%% Fmincon Optimizer
% Set options for optimizer,
% number of iterations and tolerance
niter = 1000;
options = optimoptions(@fmincon,'Display','iter',...
    'MaxIter',niter,'PlotFcn','optimplotfval',...
    'StepTolerance', 1e-3);

% Increase sensitivity of optimizer
% by changing order of magnitude
L_Optimized_not_nH = Lvals * 10^9;
C_Optimized_not_pF = Cvals * 10^12;

% Define constraints for variables
L_lb_opt1 = ones(1,N)*L_lb;
L_ub_opt1 = ones(1,N) * L_ub;
C_lb_opt1 = ones(1,N)*C_lb;
C_ub_opt1 = ones(1,N) * C_ub;
lb_opt1 = [L_lb_opt1 C_lb_opt1];
ub_opt1 = [L_ub_opt1 C_ub_opt1];
LC_scaled = [L_Optimized_not_nH C_Optimized_not_pF];

% Set constraints to null
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];

% Run optimizer
[LC_opt1, LMS_opt1_avg] = ...
    fmincon(@(LC_scaled)Inductor_and_Capacitor_Least_Sq_EF...
    (matchingNW_opt1,LC_scaled,N,freq_interest,Zl_interest,...
    Z0,Goal), LC_scaled,A,b,Aeq,beq,lb_opt1,ub_opt1,...
    nonlcon,options);

% Update inductor values
matchingNW_opt1.Inductances = LC_opt1(1:N) * 1e-9;
matchingNW_opt1.Capacitances = LC_opt1(N+1:2*N) * 1e-12;

% Simulate optimized network
S = sparameters(matchingNW_opt1,freq_data,Z0);
%% Genetic algorithm
% Create new network for second optimizer
matchingNW_opt2 = lcladder('bandpasstee',LC_opt1(1:N)...
    * 1e-9,LC_opt1(N+1:2*N) * 1e-12);

% Coilcraft inductors
Lvals_0906_1606 = [1.65 2.55 3.85 5.4 5.6 7.15...
    8.8 9.85 12.55];
Lvals_0806SQ_0807SQ_0908SQ = [5.5 6 6.9 8.1 8.9...
    10.2 11.2 12.1 12.3 13.7 14.7 15.7 16.6 17 19.4...
    21.5 22 23 25 27.3];
Lvals_1111SQ = [27 30 33 36 39 43 47];
Possible_Lvals = [Lvals_0906_1606...
    Lvals_0806SQ_0807SQ_0908SQ Lvals_1111SQ];
Possible_Lvals = unique(Possible_Lvals);

% Possible capacitance values
Possible_Cvals = 0.1:0.1:47;

% Find closest inductor values in list,
% convert to index of list
L_opt2 = interp1(Possible_Lvals,Possible_Lvals,...
    LC_opt1(1:N), 'nearest');
for i = 1:length(L_Optimized_not_nH)
    L_opt2(i) = find(Possible_Lvals == L_opt2(i));
end

% Find closest capacitor values in list,
% convert to index of list
C_opt2 = interp1(Possible_Cvals,Possible_Cvals,...
    LC_opt1(N+1:2*N),'nearest');
for i = 1:length(C_Optimized_not_pF)
    C_opt2(i) = find(Possible_Cvals == C_opt2(i));
end

% Initial LC values for second optimizer
LC_opt2_initial = [L_opt2 C_opt2];

% Create initial population for genetic algorithm
Initial_Pop_1 = repmat(LC_opt2_initial,...
    length(LC_opt2_initial),1) + diag(ones(N*2,1));
Initial_Pop_2 = repmat(LC_opt2_initial,...
    length(LC_opt2_initial),1) - diag(ones(N*2,1));
```

```
for i = 1:size(Initial_Pop_1,1)
    for k = 1:N
        if Initial_Pop_1(i,k) > length(Possible_Lvals)
            Initial_Pop_1(i,k) = length(Possible_Lvals)
        end
    end
end
Initial_Pop_2(Initial_Pop_2 == 0) = 1;
Initial_Pop = [LC_opt2_initial; Initial_Pop_1;...
    Initial_Pop_2];

% Genetic algorithm
niter = 1000;
options = optimoptions(@ga,'EliteCount', 20 ,...
    'FunctionTolerance', 1e-4, 'Display','iter',...
    'PlotFcn','gaplotbestf',...
    'InitialPopulationMatrix',Initial_Pop,...
    'MaxStallGenerations', 50);

% Bounds equal to limits of the list
L_lb_opt2 = ones(1,N);
L_ub_opt2 = ones(1,N) * length(Possible_Lvals);
C_lb_opt2 = ones(1,N);
C_ub_opt2 = ones(1,N) * length(Possible_Cvals);
lb_opt2 = [L_lb_opt2 C_lb_opt2];
ub_opt2 = [L_ub_opt2 C_ub_opt2];

% Run optimizer of component value indices
[LC_opt2_final, LMS_opt2_avg] = ...
    ga(@(LC_opt2_initial)...
    Inductor_and_Capacitor_Least_Sq_EF_variable_map...
    (matchingNW_opt2, LC_opt2_initial,N,freq_interest,...
    Zl_interest,Z0,Goal,Possible_Lvals,Possible_Cvals),...
    length(LC_opt2_initial),A,b,Aeq,beq,lb_opt2,ub_opt2,...
    nonlcon,1:length(LC_opt2_initial),options);

% Convert indices to component values and update network
LC_opt2_final(1:N) = InductorMapVariables...
    (LC_opt2_final(1:N), Possible_Lvals);
LC_opt2_final(N+1:2*N) = CapacitorMapVariables...
    (LC_opt2_final(N+1:2*N), Possible_Cvals);
matchingNW_opt2.Inductances = LC_opt2_final(1:N) * 1e-9;
matchingNW_opt2.Capacitances = LC_opt2_final(N+1:2*N) * 1e-12;
%% Results of fmincon
S_mn_opt1 = sparameters(matchingNW_opt1,freq_data,Z0);
for i= 1:81
    S_opt1{i} = gammain(S_mn_opt1,Zl_data{i});
    db_S_opt1{i} = db(abs(S_opt1{i}));
    for k = 1:length(db_S_opt1{i})
        if db_S_opt1{i}(k) ≤ Goal
            LMS_opt1{i}(k) = 0;
        else
            LMS_opt1{i}(k) = (abs(db_S_opt1{i}(k)...
                - (Goal)))^2;
        end
    end
end
%% Results of ga
S_mn_opt2 = sparameters(matchingNW_opt2,freq_data,Z0);
for i= 1:81
    S_opt2{i} = gammain(S_mn_opt2,Zl_data{i});
    db_S_opt2{i} = db(abs(S_opt2{i}));
    for k = 1:length(db_S_opt2{i})
        if db_S_opt2{i}(k) ≤ Goal
            LMS_opt2{i}(k) = 0;
        else
            LMS_opt2{i}(k) = (abs(db_S_opt2{i}(k)...
                - (Goal)))^2;
        end
    end
end

Initial_Obj_Func = mean(cell2mat(LMS_initial_interest),'all')
Fmincon_Obj_Func = LMS_opt1_avg
GA_Obj_Func = LMS_opt2_avg
%% Functions
function [freq, s] = sweep_excel_to_s1p_func(file)
    Excel = readmatrix(file);
    i = 1;
    for Scan_Angle = -40:40
        [row,col] = find(Excel==Scan_Angle);
        Data = Excel(3:end,[1,col:col+1]);
        Header = {'#' 'MHZ'   'S'   'RI'   'R'   '50.0'};
        writecell(Header, 'Data.txt','delimiter',' ')
        writematrix(Data, 'Data.txt','delimiter',' ...
            ','WriteMode','append')

        file1 = 'Data.txt';
        file2=strrep(file1,'txt','s1p');
        copyfile(file1,file2);

        data = rfdata.data;
        data = read(data,'Data.s1p');
        [s_params_data,freq_data] = ...
            extract(data,'S_PARAMETERS');
        s11{i} = s_params_data(:);
        i = i+1;
    end
    freq = freq_data;
    s = s11;
end

```

```
function coefficients = butter_g(n)
    g = zeros(1,n);
    for r = 1: n
        g(r) = 2*sin((2*r-1)*pi/(2*n));
    end
    coefficients = [g];
end

function LMS = Inductor_and_Capacitor_Least_Sq_EF...
     (matchingNW,LCvalues,N,freq,ZL,Z0,Goal)

    % Ensure positive element values
    if any(LCvalues ≤ 0)
        output = Inf;
        return
    end

    % Update the element values in the matching network
    matchingNW.Inductances = LCvalues(1:N) * 1e-9;
    matchingNW.Capacitances = LCvalues(N+1:2*N) * 1e-12;

    % Perform analysis on tuned matching network
    S = sparameters(matchingNW,freq,Z0);

    % Calculate input reflection coefficient 'gammaIn'
    for i= 1:81
        gIn{i} = gammain(S,ZL{i});
        db_gIn{i} = db(abs(gIn{i}));
        for k = 1:length(db_gIn{i})
            if db_gIn{i}(k) ≤ Goal
                metric{i}(k) = 0;
            else
                metric{i}(k) = (db_gIn{i}(k) - Goal)^2;
            end
        end
        avg{i} = mean(metric{i});
    end
avg = cell2mat(avg);
LMS = mean(avg);
end

function LMS = ...
    Inductor_and_Capacitor_Least_Sq_EF_variable_map...
     (matchingNW,LCvalues,N,freq,ZL,Z0,...
    Goal,Possible_Lvals,Possible_Cvals)

LCvalues(1:N) = InductorMapVariables(LCvalues(1:N),...
    Possible_Lvals);
LCvalues(N+1:2*N) = CapacitorMapVariables(LCvalues(N+1:2*N),...
    Possible_Cvals);

```

```
% Ensure positive element values
if any(LCvalues ≤ 0)
    output = Inf;
    return
end

% Update the element values in the matching network
matchingNW.Inductances = LCvalues(1:N) * 1e-9;
matchingNW.Capacitances = LCvalues(N+1:2*N) * 1e-12;

% Perform analysis on tuned matching network
S = sparameters(matchingNW,freq,Z0);

% Calculate input reflection coefficient 'gammaIn'
for i= 1:81
    gIn{i} = gammain(S,ZL{i});
    db_gIn{i} = db(abs(gIn{i}));
    for k = 1:length(db_gIn{i})
        if db_gIn{i}(k) ≤ Goal
            metric{i}(k) = 0;
        else
            metric{i}(k) = (abs(db_gIn{i}(k) - Goal))^2;
        end
    end
    avg{i} = mean(metric{i});
end
avg = cell2mat(avg);

% Cost function
LMS = mean(avg);
end

function C = CapacitorMapVariables(C, Cvals)
    for i = 1:length(C)
        C(i) = Cvals(C(i));
    end
end

function L = InductorMapVariables(L, Lvals)
    for i = 1:length(L)
        L(i) = Lvals(L(i));
    end
end
```

A.2 Tunable Matching Network Program

```
%% Description
% Finds optimized component values for tunable matching network
% Required information:
%   User variables: outlined in code
%   S-parameters: this code takes an excel sheet to
%                 extract s-parameter data, format is
%                 shown in Appendix A.3
%% Initialization & Variables
clc;
clear all;
close all;

% User variables
BW_interest = 10e6;      % Tunable bandwidth (Hz)
BW_spec = 125e6;         % Total spec bandwidth (Hz)
fc_design = 200e6;       % Filter cutoff freq (Hz)
f_bandstop = 200e6;      % Bandstop center freq (Hz)
BW_bandstop = 100e6;     % Bandstop bandwidth (Hz)
Zs = 50;                 % Source impedance (ohm)
Z0 = 50;                 % System impedance (ohm)
Goal = -15;              % Goal S11 (dB)
C_tune_lb = 0.71;        % Lower bound of varactors (pF)
C_tune_ub = 22.47;       % Lower bound of varactors (pF)
n_angles = 81;           % Num of angles, must be odd
% Take excel file and extract s-parameters
[file,path] = uigetfile('*.xlsx');
[freq_data, s11_data] = sweep_excel_to_s1p_func(file);

% Generate freq bins and associated data
for i = 1:floor(2*BW_spec/BW_interest+2)
    % Create bins
    freq_interest{i} = 435e6 - 0.5*BW_spec + ...
        (i-2)*0.5*BW_interest:...
        312500:435e6 - 0.5*BW_spec + (i+0)*0.5*BW_interest;
    %Index of lower and upper bounds of each freq bin
    [rowLower(i),colLower(i)] = ...
        find(freq_data==freq_interest{i}(1));
    [rowUpper(i),colUpper(i)] = ...
        find(freq_data==freq_interest{i}(end));
    % Generate associated data for each bin
    for j = 1:n_angles
        s11_interest{i,j} = ...
            s11_data{(40-floor(n_angles/2)+j)}...
            (rowLower(i):rowUpper(i));
        Zl_interest{i,j} = Z0.*(1+s11_interest{i,j})./...
            (1-s11_interest{i,j});
    end
```

```
end
%% Filter Prototype
% Frequency calculations
fLower = fc_design - (BW_design/2);
fUpper = fc_design + (BW_design/2);
nfreq = (fUpper-fLower)/0.3125*10^-6+1;
freq = linspace(fLower,fUpper,nfreq);
w = 2*pi*freq;
wU = 2*pi*fUpper;
wL = 2*pi*fLower;
w0 = sqrt(wL*wU);

% Filter design
% Butterworth coefficients (N=3)
LCproto = butter_g(3);

% Initialize component value matrix
LCvals = zeros(1,4);

% First high pass capacitor
LCvals(1) = 1/(LCproto(1)*Zs*(2*pi*fc_design));

% Last high pass capacitor
LCvals(3) = 1/(LCproto(3)*Zs*(2*pi*fc_design));

% Middle bandstop L
LCvals(4) = (Zs)./((wU-wL).*LCproto(2));

% Middle bandstop C
LCvals(2) = (LCproto(2)*(wU-wL))./((w0^2)*Zs);


% Create circuit object
matchingNW = circuit;
mn_C1= add(matchingNW,[1 2],capacitor(LCvals(1)));
mn_C2= add(matchingNW,[2 4],capacitor(LCvals(2)));
mn_C3= add(matchingNW,[2 3],capacitor(LCvals(3)));
mn_L = add(matchingNW,[4 5],inductor(LCvals(4)));
setports(matchingNW,[1 5],[3 5]);

% Save initial values for comparison
LC_initial = LCvals;
%% Calculate LMS of raw data
for i = 1:floor(2*BW_spec/BW_interest+2)
    for j= 1:n_angles
         db_s11_interest{i,j} = db(abs(s11_interest{i,j}));
        for k = 1:length(db_s11_interest{i,j})
            if db_s11_interest{i,j}(k) ≤ Goal
               LMS_initial{i,j}(k) = 0;
            else
```

```
                LMS_initial{i,j}(k) = ...
                    (abs(db_s11_interest{i,j}...
                     (k) - Goal))^2;
            end
        end
        LMS_initial_avg{i,j} = mean(LMS_initial{i,j});
    end
end
%% Genetic Optimizer (Outer optimizer)
% Increase sensitivity of optimizer
% by changing order of magnitude
Cvals_scaled = LCvals(1:3).*1e12;
L = LCvals(4);
optimize_L = L*1e9;

% Set capacitance and inductance bounds
C_tuning_range = [C_tune_lb C_tune_ub];
L_lb = 1;
L_ub = length(Possible_Lvals);

% Coilcraft inductors
Lvals_0906_1606 = [1.65 2.55 3.85 5.4 5.6 7.15 8.8 9.85 12.55];
Lvals_0806SQ_0807SQ_0908SQ = [5.5 6 6.9 8.1 8.9 10.2 11.2...
    12.1 12.3 13.7 14.7 15.7 16.6 17 19.4 21.5 22 23 25 27.3];
Lvals_1111SQ = [27 30 33 36 39 43 47];
Possible_Lvals = [Lvals_0906_1606 ...
   Lvals_0806SQ_0807SQ_0908SQ Lvals_1111SQ];
Possible_Lvals = unique(Possible_Lvals);

% Set constraints to null
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];

% Find closest inductor values in list,
% convert to index of list
optimize_L = ...
   interp1(Possible_Lvals,Possible_Lvals,optimize_L,'nearest');
optimize_L = find(Possible_Lvals == optimize_L);

% Create initial population for genetic algorithm
if optimize_L > 2
    Initial_Pop = [optimize_L-1; optimize_L; optimize_L+1];
else
    Initial_Pop = [optimize_L-1; optimize_L; optimize_L+1];
end

% Set options for genetic algorithm
niter = 1000;
```

```
options = optimoptions(@ga,'FunctionTolerance', 1e-3,...
    'Display','iter','PlotFcn','gaplotbestf',...
    'MaxStallGenerations', 2,'PopulationSize',10,...
    'InitialPopulationMatrix',Initial_Pop);

% Run optimizer
[final_L_index, LMS] = ga(@(optimize_L)...
    Inductor_and_Capacitor_Least_Sq_EF_variable_map...
    (matchingNW, optimize_L,Cvals_scaled,freq_interest,...
    Zl_interest,Z0,Goal,Possible_Lvals, ...
        C_tuning_range,n_angles,...
    BW_spec,BW_interest), ...
        1,A,b,Aeq,beq,L_lb,L_ub,nonlcon,1,options);

% Update inductance value
final_L = InductorMapVariables(final_L_index, Possible_Lvals);
matchingNW.Elements(1,4).Inductance = final_L * 1e-9;

% Generate networks
[optimized_capacitors,LMS_tuned, dB_S_tuned] = ...
    Generate_Networks...
    (matchingNW, ...
        Cvals_scaled,freq_interest,Zl_interest,Z0,Goal,...
    final_L,C_tune_lb,C_tune_ub,n_angles);
%% Find best values and capacitors from overlapping bins
% Create array of frequency interesctions between bins
for i = 1:(floor(2*BW_spec/BW_interest+2)-1)
[C{i},ia{i},ib{i}] = ...
    intersect(freq_interest{i},freq_interest{i+1});
end
C{end} = C{end}(1:find(C{end} == 435e6+BW_spec/2));
freq_interest_matrix = unique(cell2mat(freq_interest));
first_intersect = find(freq_interest_matrix == ...
    435e6-BW_spec/2);
last_intersect = find(freq_interest_matrix == 435e6+BW_spec/2);
freq_interest_matrix = freq_interest_matrix(first_intersect:...
    last_intersect);

% Create matrix of best values
for i = 1:length(C)
    for j = 1:length(C{i})
        for k = 1:n_angles
            if dB_S_tuned{i,k}(ia{i}(j)) < ...
                dB_S_tuned{i+1,k}(ib{i}(j))
                best_s_db(j+length(C{1})*(i-1),k) = ...
                    dB_S_tuned{i,k}(ia{i}(j));
                best_lms(j+length(C{1})*(i-1),k) = ...
                    LMS_tuned{i,k}(ia{i}(j));
                best_cap_network{j+length(C{1})*(i-1),k} = ...
                    optimized_capacitors{i,k};
            else
```

```
                    best_s_db(j+length(C{1})*(i-1),k) = ...
                        dB_S_tuned{i+1,k}(ib{i}(j));
                    best_lms(j+length(C{1})*(i-1),k) = ...
                        LMS_tuned{i+1,k}(ib{i}(j));
                    best_cap_network{j+length(C{1})*(i-1),k} = ...
                        optimized_capacitors{i+1,k};
                end
            end
        end
    end

% Remove double values at edge of bins
for i = 1:length(C)-1
    for k = 1:n_angles
        if best_s_db(i*length(C{i+1})+first_intersect-1,k)...
                > best_s_db(i*length(C{i+1})+1+...
                first_intersect-1,k)
            best_s_db(i*length(C{i+1})+first_intersect-1,k)...
                = best_s_db(i*length(C{i+1})+1+...
                first_intersect-1,k);
            best_lms(i*length(C{i+1})+first_intersect-1,k)...
                = best_lms(i*length(C{i+1})+1+...
                first_intersect-1,k);
            best_cap_network{i*length(C{i+1})+...
                first_intersect-1,k} = ...
                best_cap_network{i*length(C{i+1})+1+...
                first_intersect-1,k};
        end
        best_cap_network{i*length(C{i+1})+1+...
            first_intersect-1,k} = [0 0 0];
    end
    idx_remove(i) = i*length(C{i+1})+1+first_intersect-1;
end

for i = 1:length(C)-1
    best_s_db(idx_remove(i)-(i-1),:) = [];
    best_lms(idx_remove(i)-(i-1),:) = [];
end

% Calculate overall obj func value from best values
avg_best_lms = mean(best_lms, 'all');

% Create matrix of best tuned capacitor values
for i = 1:size(best_cap_network,1)
    for k = 1:size(best_cap_network,2)
        best_cap_network{i,k}( ...
            all(¬best_cap_network{i,k},2), : ) = [];
    end
end
best_cap_network(all(cellfun(@isempty, ...
    best_cap_network),2),:) = [];
```

```
%% Functions
function [freq, sparameters] = sweep_excel_to_s1p_func(file)
    Excel = readmatrix(file);
    i = 1;
    for Scan_Angle = -40:40
        [row,col] = find(Excel==Scan_Angle);
        Data = Excel(3:end,[1,col:col+1]);
        Header = {'#' 'MHZ'   'S'   'RI'   'R'   '50.0'};
        writecell(Header, 'Data.txt','delimiter',' ')
        writematrix(Data, 'Data.txt','delimiter',' ...
            ','WriteMode','append')

        file1 = 'Data.txt';
        file2=strrep(file1,'txt','s1p');
        copyfile(file1,file2);

        data = rfdata.data;
        data = read(data,'Data.s1p');
        [s_params_data,freq_data] = ...
            extract(data,'S_PARAMETERS');
        s11{i} = s_params_data(:);
        i = i+1;
    end

    freq = freq_data;
    sparameters = s11;
end

function coefficients = butter_g(n)
g = zeros(1,n);
for r = 1: n
    g(r) = 2*sin((2*r-1)*pi/(2*n));
end

coefficients = [g];
end

function LMS = ...
    Optimize_Tunable_Matching_EF(matchingNW,Cvals_scaled,...
    freq_interest, Zl_interest,Z0,Goal,optimize_L, ...
        C_tuning_range,...
    n_angles)

% INNER OPTIMIZER
% Initiate optimizer
niter = 1000;
options = ...
    optimoptions(@fmincon,'MaxIter',niter,'PlotFcn',[],...
    'Display','off','StepTolerance', 1e-3);

% Set varactor bounds
```

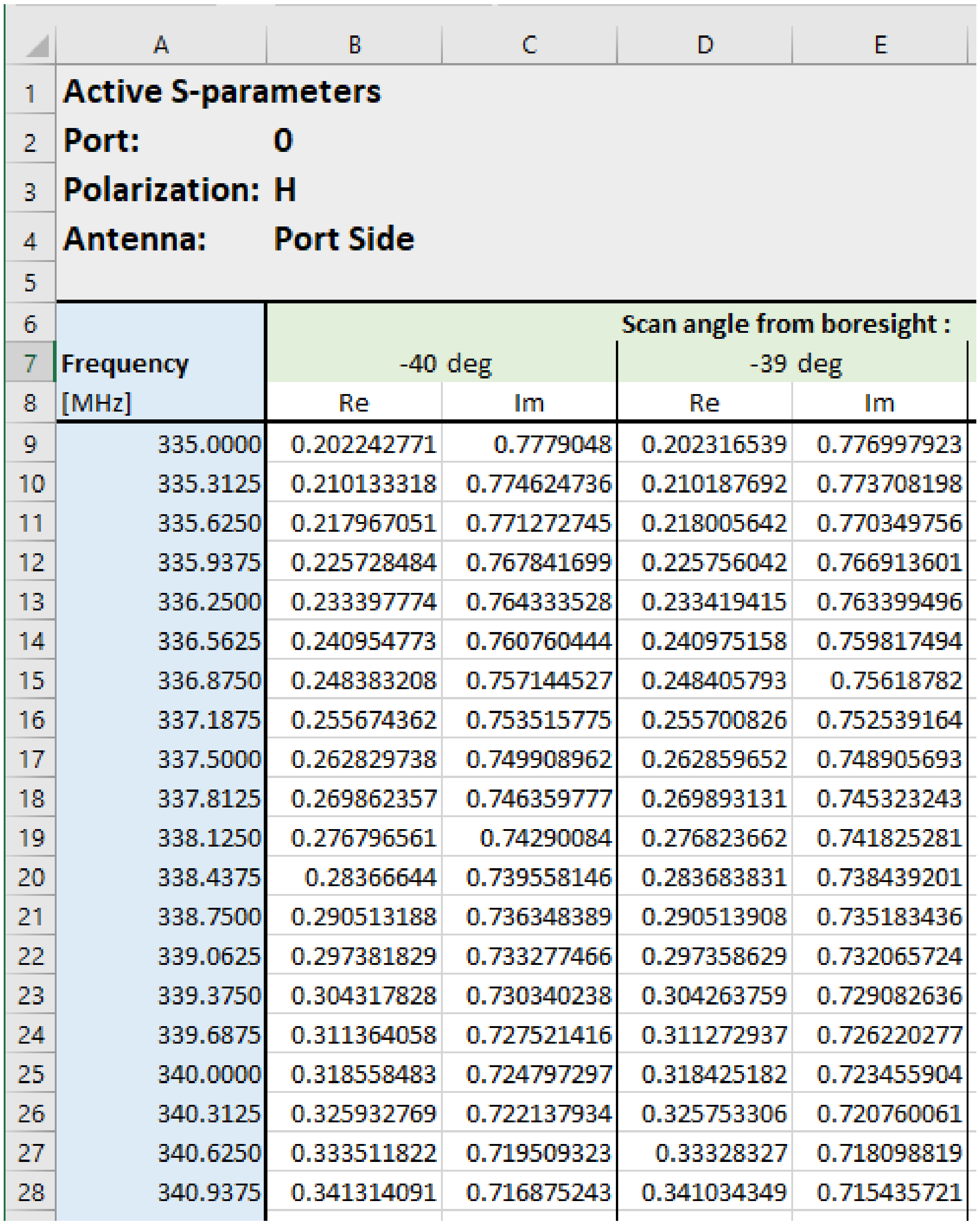
```
C_tune_lb = C_tuning_range(1);
C_tune_ub = C_tuning_range(2);
C_lb(1) = C_tune_lb;
C_lb(2) = C_tune_lb;
C_lb(3) = C_tune_lb;
C_ub(1) = C_tune_ub;
C_ub(2) = C_tune_ub;
C_ub(3) = C_tune_ub;
lb = [C_lb];
ub = [C_ub];

% Set constraints to null
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];

% Run optimizer for each freq bin and angle
f = waitbar(0,'Optimizing inductor');
for i = 1:length(freq_interest)
    for j = 1:n_angles
        [Best_Cvals_scaled{i,j}] = ...
        fmincon(@(Cvals_scaled)down_the_rabbit_hole...
        (matchingNW, Cvals_scaled,optimize_L,...
        freq_interest{i}, Zl_interest{i,j},Z0,Goal),...
        Cvals_scaled,A,b,Aeq,beq, lb,ub,nonlcon,options);
    end
    waitbar(i/length(freq_interest),f, 'Optimizing inductor');
end
close(f);

% Generate capacitor networks
[optimized_capacitors,LMS] = Generate_Networks(matchingNW, ...
    Cvals_scaled,freq_interest,Zl_interest,Z0,Goal,...
    optimize_L, C_tune_lb,C_tune_ub,n_angles);

% Evaluate best values from overlapping bins
for i = 1:(length(freq_interest)-1)
[C{i},ia{i},ib{i}] = ...
   intersect(freq_interest{i},freq_interest{i+1});
end
freq_intersect = cell2mat(C);
freq_intersect = unique(freq_intersect);
best_LMS = zeros(length(freq_intersect),n_angles);
for i = 1:length(C)
    for j = 1:length(C{i})
        for k = 1:n_angles
            if LMS{i,k}(ia{i}(j)) < LMS{i+1,k}(ib{i}(j))
                best_LMS(j+length(C{i})*(i-1),k) = ...
                    LMS{i,k}(ia{i}(j));
            else
                best_LMS(j+length(C{i})*(i-1),k) = ...
                    LMS{i+1,k}(ib{i}(j));
            end
        end
    end
end

for i = 1:length(C)-1
    for k = 1:n_angles
        if best_LMS(i*length(C{i}),k) > ...
            best_LMS(i*length(C{i})+1,k)
            best_LMS(i*length(C{i}),k) = ...
                best_LMS(i*length(C{i})+1,k);
        end
    end
    best_LMS(i*length(C{i})+1,:) = zeros(1,n_angles);
end
best_LMS( all(¬best_LMS,2), : ) = [];

% Cost function
LMS = mean(best_LMS, 'all');
end

function output = ...
    down_the_rabbit_hole(matchingNW,Cvals_scaled,...
    Lvals_scaled,freq,ZL,Z0,Goal)

% Find optimized values of individual freq bin
    if any(Cvals_scaled ≤ 0)
        output = Inf;
        return
    end

    if any(Lvals_scaled ≤ 0)
        output = Inf;
        return
    end

    % Update the element values in the matching network
    matchingNW.Elements(1).Capacitance = Cvals_scaled(1)*1e-12;
    matchingNW.Elements(2).Capacitance = Cvals_scaled(2)*1e-12;
    matchingNW.Elements(3).Capacitance = Cvals_scaled(3)*1e-12;
    matchingNW.Elements(4).Inductance = Lvals_scaled*1e-9;

    % Perform analysis on tuned matching network
    S = sparameters(matchingNW,freq,Z0);

    % Calculate input reflection coefficient 'gammaIn'
    gIn = gammain(S,ZL);
    db_gIn = db(abs(gIn));
    for k = 1:length(db_gIn)
        if db_gIn(k) ≤ Goal
            metric(k) = 0;
        else
            metric(k) = (abs(db_gIn(k) - Goal))^2;
        end
    end
    avg_dummy = mean(metric);

    % Cost function
    output = mean(avg_dummy);
end

function [tuned_capacitor_values, LMS_tuned, db_gIn] = ...
    Generate_Networks(matchingNW,Cvals_scaled,freq_interest,...
    Zl_interest,Z0,Goal,final_L,C_tune_lb,C_tune_ub,n_angles)

% CANNOT FEED VARIABLES THROUGH OPTIMIZER FUNCTION
% THIS FUNCTION GENERATES CAPACITOR NETWORK
% Initialize optimizer
niter = 1000;
options = ...
   optimoptions(@fmincon,'MaxIter',niter,'PlotFcn',[],...
    'Display','off', 'StepTolerance', 1e-3);

% Set varactor bounds
C_lb(1) = C_tune_lb;
C_lb(2) = C_tune_lb;
C_lb(3) = C_tune_lb;
C_ub(1) = C_tune_ub;
C_ub(2) = C_tune_ub;
C_ub(3) = C_tune_ub;
lb = [C_lb];
ub = [C_ub];

% Set constraints to null
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];

% Run optimizer for each bin and angle
for i = 1:length(freq_interest)
    for j = 1:n_angles
        [tuned_capacitor_values{i,j}, LMS_tuned{i,j}] = ...
        fmincon(@(Cvals_scaled)down_the_rabbit_hole...
        (matchingNW, Cvals_scaled,final_L,freq_interest{i},...
        Zl_interest{i,j}, Z0,Goal), ...
            Cvals_scaled,A,b,Aeq,beq,...
        lb,ub,nonlcon,options);
    end
end

% Create network for each bin and angle
for i = 1:length(freq_interest)
    for j= 1:n_angles
        tuned_matchingNW{i,j} = circuit;
        add(tuned_matchingNW{i,j},[1 2],...
            capacitor((tuned_capacitor_values{i,j}(1))*1e-12));
        add(tuned_matchingNW{i,j},[2 4],...
            capacitor((tuned_capacitor_values{i,j}(2))*1e-12));
        add(tuned_matchingNW{i,j},[2 3],...
            capacitor((tuned_capacitor_values{i,j}(3))*1e-12));
        add(tuned_matchingNW{i,j},[4 5],...
            inductor(final_L*1e-9));
        setports(tuned_matchingNW{i,j},[1 5],[3 5]);

        % Perform analysis on tuned matching network
        S{i,j} = sparameters(tuned_matchingNW{i,j},...
            freq_interest{i},Z0);
        gIn{i,j} = gammain(S{i,j},Zl_interest{i,j});
        db_gIn{i,j} = db(abs(gIn{i,j}));
        for k = 1:length(db_gIn{i,j})
            if db_gIn{i,j}(k) ≤ Goal
               LMS_tuned{i,j}(k) = 0;
            else
               LMS_tuned{i,j}(k) = (abs(db_gIn{i,j}(k) - ...
                   Goal))^2;
            end
        end
    end
end
end

function LMS = ...
    Inductor_and_Capacitor_Least_Sq_EF_variable_map...
     (matchingNW,optimize_L,Cvals_scaled,freq_interest,...
    Zl_interest,Z0,Goal,Possible_Lvals,C_tuning_range,...
    n_angles,BW_spec, BW_interest)
% OUTER OPTIMIZER
% Convert inductor index into value
optimize_L = InductorMapVariables(optimize_L,Possible_Lvals);

% Initiate optimizer
niter = 1000;
options = ...
    optimoptions(@fmincon,'MaxIter',niter,'PlotFcn',[],...
    'Display','off','StepTolerance', 1e-3);

% Set bounds of varactors
C_tune_lb = C_tuning_range(1);
```

```
C_tune_ub = C_tuning_range(2);
C_lb(1) = C_tune_lb;
C_lb(2) = C_tune_lb;
C_lb(3) = C_tune_lb;
C_ub(1) = C_tune_ub;
C_ub(2) = C_tune_ub;
C_ub(3) = C_tune_ub;
lb = [C_lb];
ub = [C_ub];

% Set constraints to null
A = [];
b = [];
Aeq = [];
beq = [];
nonlcon = [];

% Run optimizer for each bin and angle
for i = 1:length(freq_interest)
    for j = 1:n_angles
        [Best_Cvals_scaled{i,j}] = ...
        fmincon(@(Cvals_scaled)down_the_rabbit_hole...
        (matchingNW, ...
           Cvals_scaled,optimize_L,freq_interest{i},...
        Zl_interest{i,j}, Z0,Goal), ...
           Cvals_scaled,A,b,Aeq,beq,...
        lb,ub,nonlcon,options);
    end
end

% Generate networks to find best values in regions of bin ...
   overlap
[optimized_capacitors,LMS] = Generate_Networks(matchingNW, ...
    Cvals_scaled,freq_interest,Zl_interest,Z0,Goal,...
    optimize_L, C_tune_lb,C_tune_ub,n_angles);

% Find best values in bin overlap
for i = 1:(floor(2*BW_spec/BW_interest+2)-1)
[C{i},ia{i},ib{i}] = ...
   intersect(freq_interest{i},freq_interest{i+1});
end
C{end} = C{end}(1:find(C{end} == 435e6+BW_spec/2));
freq_intersect = cell2mat(C);
freq_intersect = unique(freq_intersect);
freq_interest_matrix = unique(cell2mat(freq_interest));
first_intersect = find(freq_interest_matrix == ...
   435e6-BW_spec/2);
last_intersect = find(freq_interest_matrix == 435e6+BW_spec/2);
freq_interest_matrix = freq_interest_matrix(first_intersect:...
    last_intersect);
best_lms = zeros(length(freq_interest_matrix),n_angles);
```

```

for i = 1:length(C)
    for j = 1:length(C{i})
        for k = 1:n_angles
            if LMS{i,k}(ia{i}(j)) < LMS{i+1,k}(ib{i}(j))
                best_lms(j+length(C{1})*(i-1),k) = ...
                    LMS{i,k}(ia{i}(j));
            else
                best_lms(j+length(C{1})*(i-1),k) = ...
                    LMS{i+1,k}(ib{i}(j));
            end
        end
    end
end

for i = 1:length(C)-1
    for k = 1:n_angles
        if best_lms(i*length(C{i+1})+first_intersect-1,k)>...
                best_lms(i*length(C{i+1})+1+...
                first_intersect-1,k)
            best_lms(i*length(C{i+1})+first_intersect-1,k)=...
                best_lms(i*length(C{i+1})+1+...
                first_intersect-1,k);
        end
    end
    idx_remove(i) = i*length(C{i+1})+1+first_intersect-1;
end

for i = 1:length(C)-1
    best_lms(idx_remove(i)-(i-1),:) = [];
end

LMS = mean(best_lms, 'all');
end

function L = InductorMapVariables(L, Lvals)
    for i = 1:length(L)
        L(i) = Lvals(L(i));
    end
end
```

A.3 Excel Document Format

The two programs import an excel sheet to extract the S-parameters of the array. Fig. A.1 shows a snippet of the excel sheet used to display the formatting.



	A	B	C	D	E
1	**Active S-parameters**				
2	**Port:**	**0**			
3	**Polarization:**	**H**			
4	**Antenna:**	**Port Side**			
5					
6		**Scan angle from boresight :**			
7	**Frequency**	-40 deg		-39 deg	
8	[MHz]	Re	Im	Re	Im
9	335.0000	0.202242771	0.7779048	0.202316539	0.776997923
10	335.3125	0.210133318	0.774624736	0.210187692	0.773708198
11	335.6250	0.217967051	0.771272745	0.218005642	0.770349756
12	335.9375	0.225728484	0.767841699	0.225756042	0.766913601
13	336.2500	0.233397774	0.764333528	0.233419415	0.763399496
14	336.5625	0.240954773	0.760760444	0.240975158	0.759817494
15	336.8750	0.248383208	0.757144527	0.248405793	0.75618782
16	337.1875	0.255674362	0.753515775	0.255700826	0.752539164
17	337.5000	0.262829738	0.749908962	0.262859652	0.748905693
18	337.8125	0.269862357	0.746359777	0.269893131	0.745323243
19	338.1250	0.276796561	0.74290084	0.276823662	0.741825281
20	338.4375	0.28366644	0.739558146	0.283683831	0.738439201
21	338.7500	0.290513188	0.736348389	0.290513908	0.735183436
22	339.0625	0.297381829	0.733277466	0.297358629	0.732065724
23	339.3750	0.304317828	0.730340238	0.304263759	0.729082636
24	339.6875	0.311364058	0.727521416	0.311272937	0.726220277
25	340.0000	0.318558483	0.724797297	0.318425182	0.723455904
26	340.3125	0.325932769	0.722137934	0.325753306	0.720760061
27	340.6250	0.333511822	0.719509323	0.33328327	0.718098819
28	340.9375	0.341314091	0.716875243	0.341034349	0.715435721

Figure A.1: S-parameter format in excel sheet used for code

Appendix B

Matching Network Fabrication Process

The fabrication process for the optimized matching networks from Chapter 4 and Chapter 5 are detailed in this appendix. The entire milling process is performed by the LPKF ProtoMat S104. The processes of the circuits from both chapters are identical.

There are three Gerber files needed to fabricate the static matching network. The required files are as follows:

1. cond_static - top layer
2. cond2_static - board outline
3. cond_cond2_static - via drill file

There are three Gerber files needed to fabricate the tunable matching network. The required files are as follows:

1. cond_tunable - top layer
2. cond2_tunable - board outline
3. cond_cond2_tunable - via drill file

B.1 Detailed Fabrication Process

The fabrication process is broken down into the 4 steps as follows:

1. Board preparation and drilling
2. Electroplating
3. Board milling
4. Component assembly

B.1.1 Board preparation

The board is placed on the LPKF S104 and the gerber files are imported into the system. Fiducial drills must be added to continue milling after electroplating. The following steps are taken on the LPKF ProtoMat S104:

1. Import Gerber files
2. Assign layers and board outline
3. Add three fiducial drills around teh outline
4. Set material properties
5. Secure substrate to bed
6. Set placement of design using camera
7. Compute toolpaths
8. Run the toolpaths step by step until vias are drilled

B.1.2 Electroplating

The board is taken from the S104 machine and the electro-less plating process is performed. After this process, the board can be electroplated to add conductivity to the via holes. All standard procedures of electro-less and electroplating must be followed.

B.1.3 Board milling

After the vias have been plated the board must be milled. Picking up from step 8 of Section B.1.1, the toolpaths will be continued to mill and route the board. This will conclude the use of the S104.

B.1.4 Assemble board

The surface mount components and edge-launch connectors must now be soldered, with the following steps:

1. Apply solder paste to pads of surface mount components
2. Position surface mount components on pads
3. Put board in reflow oven for under one minute at 260 ° C,
   until solder paste reflows
4. Solder on edge-launch connectors