國立臺灣大學電機資訊學院電信工程學研究所

碩士論文

Graduate Institute of Communication Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master's Thesis

天線設計自動化方法：機器學習與最佳化研究

Methodologies for Antenna Design Automation: A Study of Machine Learning and Optimization

陳韋丞

Wei-Cheng Chen

指導教授: 陳士元 博士

Advisor: Shih-Yuan Chen Ph.D.

中華民國 114 年 7 月

July, 2025

doi:10.6342/NTU202501972

Acknowledgements

The completion of this thesis represents the culmination of significant effort and dedication. While much of the work was conducted independently, its successful realization would not have been possible without the invaluable support, guidance, and encouragement from various individuals and organizations.

First and foremost, I extend my deepest gratitude to my supervisor, Professor Shih-Yuan Chen, of the Department of Electrical Engineering, National Taiwan University. Despite his primary expertise lying in antenna design rather than machine learning and computational optimization, Professor Chen provided crucial supervision, insightful discussions, and, perhaps most importantly, unwavering mental support throughout this challenging journey. His guidance helped shape the direction of this research and provided a stable foundation for my exploration.

I am also profoundly grateful to my friend and colleague, Bing-Jia Chen. His generous assistance and patient explanations in the early stages of this project were instrumental in building my foundational understanding of machine learning principles and optimization concepts, without which this research would not have progressed.

This research was made possible through the generous sponsorship and support of external organizations. I would like to express my sincere appreciation to ASUSTeK Com-

doi:10.6342/NTU202501972

puter Inc. for their sponsorship, which included providing a high-performance computer for electromagnetic simulations and a powerful GPU card. The provision of such hardware contributed to the overall research environment and capabilities.

Finally, I would like to thank my family and friends for their constant encouragement and understanding. Their belief in me provided the motivation needed to overcome obstacles and persist through the demanding phases of this work.

This thesis is a testament to the collaborative spirit of academia and the importance of both direct and indirect support. To everyone who contributed, in ways big or small, thank you.

doi:10.6342/NTU202501972

摘要

天線設計是一個複雜的電磁逆問題，傳統上通常透過基於經驗的啟發式方法或耗時的參數化掃描來解決。本論文旨在實現天線設計流程的自動化和一般化，並比較和討論不同的方法。

研究首先探討並實現以離線機器學習代理人模型微調天線的幾何參數。我們的模型可以實時的根據環境的變化優化操作在 2.45 GHz 之 PIFA 天線。類似的離線模型在天線領域有許多其他應用，我們也將對其進行分析討論。其次，我們也嘗試利用像素化形式表示平面天線，從而實現對材料分佈的離散控制。機器學習的引入顯著的加速了優化的流程 (MLAO)。這類採用 MLAO 架構的研究近來蓬勃發展，我們同樣大量分析並討論其方法的優劣。再者，為了應對可擴展性的挑戰，本文介紹了一種基於梯度的最佳化方法，利用帶有懲罰的固體各向同性材料 (SIMP) 方法和伴隨方法進行高效的靈敏度分析。伴隨方法顯著加快了收斂速度，即使在高維設計空間中也是如此。

論文最後討論了新興方法，例如強化學習、生成模型和基於物理的神經網絡等，這些方法或可透過實現即時自適應、更廣泛的設計空間探索和更低的模擬成本來進一步增強天線設計，具有很大的潛力。總而言之，本論文對天線設計自動化進行了深入且廣泛的探討，並統整出一個全面的理解架構，為未來數據驅動和基於物理的設計範式的整合奠定了基礎。

doi:10.6342/NTU202501972

關鍵字：天線設計自動化、人工智慧、機器學習、拓樸優化

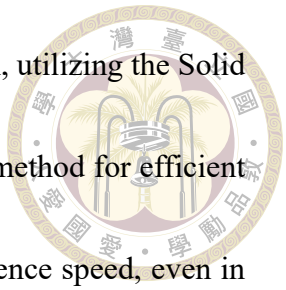
doi:10.6342/NTU202501972

Abstract

Antenna design is essentially an inverse electromagnetic problem, traditionally addressed through empirical heuristic methods or time-consuming parametric sweeps. This thesis aims to automate and generalize the antenna design process while comparing and discussing various approaches.

The study first explores and implements the fine-tuning of antennas using an offline machine learning surrogate model. Our model can optimize a PIFA antenna operating at 2.45 GHz in real-time based on environmental changes. Similar offline model frameworks have numerous other applications in the antenna domain, which will also be analyzed and discussed. Secondly, we attempt to represent planar antennas in a pixelated form, enabling discrete control over material distribution. The introduction of machine learning significantly accelerates the optimization process. Research adopting the MLAO framework has recently flourished, and we extensively analyze and discuss the strengths and weaknesses of these methods. Furthermore, to address scalability challenges, this the-

doi:10.6342/NTU202501972



sis introduces a well-developed gradient-based optimization approach, utilizing the Solid Isotropic Material with Penalization (SIMP) method and the adjoint method for efficient sensitivity analysis. The adjoint method markedly enhances convergence speed, even in high-dimensional design spaces.

Finally, the thesis discusses emerging methods, such as reinforcement learning, generative models, and physics-based neural networks, which hold significant potential to further enhance antenna design by enabling real-time adaptation, broader design space exploration, and reduced simulation costs. In summary, this thesis provides a thorough and comprehensive exploration of automated antenna design, establishing a robust framework for understanding and laying the foundation for the integration of future data-driven and physics-based antenna design paradigms.

Keywords: Antenna Design Automation, Artificial Intelligence, Machine Learning, Topology Optimization

doi:10.6342/NTU202501972

Contents

Page

Acknowledgements **i**

摘要 **iii**

Abstract **v**

Contents **vii**

List of Figures **xi**

List of Tables **xiii**

Denotation **xv**

Chapter 1 Introduction **1**

1.1 Motivation 1

1.2 Background 2

1.3 Outline 3

Chapter 2 Machine Learning in Antenna Design **5**

2.1 Introduction 5

2.2 A Brief Introduction to Machine Learning 6

2.3 Machine Learning in Antenna Design 8

2.3.1 Antenna Design as an Inverse EM Problem 8

2.3.1.1 Optimization-Based Forward Simulation 9

2.3.1.2 Learning the Inverse Function Directly 9

doi:10.6342/NTU202501972

2.3.2 Surrogate Modeling for Antenna Design 9
2.3.2.1 Surrogate Models for Optimization 10
2.3.2.2 Surrogate Models for Inverse Modeling 10
2.4 Advanced Topics in Surrogate Modeling 10
2.4.1 Surrogate vs. Generative Models 10
2.4.1.1 Generative Models 11
2.4.1.2 Surrogate Models 11
2.4.2 Training a Surrogate Model: Step-by-Step 11
2.4.3 Offline vs. Online Learning 12
2.4.3.1 Offline (Batch) Learning 12
2.4.3.2 Online Learning 12
Chapter 3 Antenna Design Using Offline Surrogate Model 13
3.1 Recent Developments 13
3.2 Design Example and Goal 14
3.3 Design Region 14
3.4 Design Flow 17
3.4.1 Data Acquisition 17
3.5 Model Selection and Performance 18
3.5.1 Feedforward Neural Network (FNN) 18
3.5.2 Weighted-FNN 20
3.5.3 Recurrent Neural Network (RNN) 21
3.5.4 TabTransformer 21
3.6 Results 23
3.7 Discussion 26

doi:10.6342/NTU202501972

Chapter 4 Antenna Design Using Online Surrogate Models **29**

4.1 Machine Learning-Assisted Optimization 29

4.2 1.5 GHz Patch Antenna Design Demonstration 31

4.2.1 Design Example and Goal 31

4.2.2 Design Region 31

4.2.2.1 Topology Coding 33

4.2.2.2 Design Space Diversity 34

4.2.3 Genetic Algorithm (GA) 35

4.2.4 Machine Learning-Assisted Optimization - Genetic Algorithm (MLAO-GA) 38

4.2.5 Results 39

4.3 Discussion 44

Chapter 5 Antenna Design Using Adjoint Method **47**

5.1 Introduction to Topology Optimization 47

5.2 Theory of Adjoint Method 49

5.2.1 Optimization Problems 50

5.2.2 The Adjoint Method 51

5.3 1.5 GHz Patch Antenna Design Demonstration 53

5.3.1 Design Example and Goal 53

5.3.2 Design Region 53

5.3.3 Implementation 53

5.3.4 Preliminary Results 56

5.3.4.1 4 × 4 Pixelated Design 56

5.3.4.2 18 × 18 Pixelated Design 58

doi:10.6342/NTU202501972

5.4 Discussion 62

Chapter 6 Conclusion **65**

6.1 Summary 65

6.2 Future Directions 66

6.2.1 Generative Model 66

6.2.2 Reinforcement Learning 67

6.2.3 Physics-Informed Neural Networks 68

6.2.4 Machine Learning with Hard Constraints 68

6.3 Contributions 69

References **71**

Appendix A — Recurrent Neural Network with Gated Recurrent Unit **79**

Appendix B — TabTransformer **81**

Appendix C — Adaptive Moment Estimation **83**

doi:10.6342/NTU202501972

List of Figures

3.1 Device environment resembling a smartphone (75 mm wide and 150 mm high). The design region is located in the upper-left corner. 15

3.2 Design region. The 11 inputs (in blue) include 10 binary block states and 1 feed position. Outputs (in red) are the four geometric parameters. 16

3.3 FNN prediction vs. ground truth. 19

3.4 Weighted-FNN prediction vs. ground truth. 20

3.5 RNN prediction vs. ground truth. 22

3.6 TabTransformer prediction vs. ground truth. 23

3.7 Visualizations of test cases. 24

3.8 $|S_{11}|$ comparison: Case 1 24

3.9 $|S_{11}|$ comparison: Case 2 25

3.10 $|S_{11}|$ comparison: Case 3 25

3.11 $|S_{11}|$ comparison: Case 4 26

4.1 Frameworks for traditional computational optimization and machine learning-assisted optimization (MLAO) 30

4.2 Design region (blue), substrate, and bottom ground plane. The feed point is indicated by a black dot. 32

4.3 Design region with the central four blocks prefilled with PEC, leaving 12 designable blocks. 32

4.4 Antenna topologies and their binary-decimal coding. 33

4.5 $|S_{11}|$ plots for topologies 1, 3, and 7. 33

4.6 $|S_{11}|$ plots of 150 randomly sampled topologies, illustrating design space diversity. 34

doi:10.6342/NTU202501972

4.7 Flow diagram of a genetic algorithm. 35
4.8 Fitness progression of GA across generations. 36
4.9 $|S_{11}|$ for initial (2763) and final (2378) topologies. 36
4.10 Workflow of MLAO-GA incorporating surrogate modeling. 39
4.11 Fitness progression using MLAO-GA. 40
4.12 $|S_{11}|$ for initial (0) and final (2392) topologies. 40
4.13 1st trial of fitness comparison. 41
4.14 2nd trial of fitness comparison. 41
4.15 3rd trial of fitness comparison. 42
4.16 4th trial of fitness comparison. 42
4.17 Average fitness comparison. 43
4.18 Final antenna topology (2392) from MLAO-GA. 43

5.1 4×4 pixelated design region. 54
5.2 18×18 pixelated design region. 54
5.3 Optimized 1.5 GHz antenna topology in the 4×4 pixelated region. 56
5.4 $|S_{11}|$ parameter of the optimized antenna in the 4×4 pixelated region. . . 57
5.5 Relative received power per iteration in the 4×4 pixelated region. 57
5.6 Design variable distribution per iteration in the 4×4 pixelated region. . . 58
5.7 Gradient update step per iteration in the 4×4 pixelated region. 59
5.8 Optimized 1.5 GHz antenna topology in the 18×18 pixelated region. . . 59
5.9 $|S_{11}|$ parameter of the optimized antenna in the 18×18 pixelated region. 60
5.10 Relative received power per iteration in the 18×18 pixelated region. . . 60
5.11 Design variable distribution per iteration in the 18×18 pixelated region. . 61
5.12 Gradient update step per iteration in the 18×18 pixelated region. 61

doi:10.6342/NTU202501972

List of Tables

3.1 Input and output of our surrogate model. 17

3.2 Example dataset entry (Sample 75). 17

3.3 Test cases. 24

3.4 FNN vs. CST optimizer under Case 3 (in mm). 24

4.1 Population Initialization 37

4.2 Fitness values of initial population 37

4.3 Parent Selection 38

doi:10.6342/NTU202501972

doi:10.6342/NTU202501972

Denotation

Adam	Adaptive Moment Estimation
CMA-ES	Covariance Matrix Adaptation Evolution Strategy
CNN	Convolutional Neural Network
DE	Differential Evolution
FDTD	Finite Difference Time Domain
FEA	Finite Element Analysis
FEM	Finite Element Method
FNN	Forward Neural Network
GA	Genetic Algorithm
GAN	Generative Adversarial Network
GNN	Graph Neural Network
GPR	Gaussian Process Regression

doi:10.6342/NTU202501972

GRU	Gated Recurrent Unit
IL	Imitation Learning
KNN	K-Nearest Neighbor
LLM	Large Language Model
LSM	Level-Set Method
ML	Machine Learning
MLAO	Machine Learning-Assisted Optimization
MoM	Method of Momentstt
MSE	Mean Square Error
NLP	Natural Language Processing
NN	Neural Network
NSGA-II	Non-dominated Sorting Genetic Algorithm II
PEC	Perfect Electric Conductor
PIFA	Planar Inverted-F Antenna
PINN	Physics-Informed Neural Network
PSO	Particle Swarm Optimization
RL	Reinforcement Learning

doi:10.6342/NTU202501972

RNN	Recurrent Neural Network
RS	Random Search
SA	Simulated Annealing
SIMP	Solid Isotropic Material with Penalization
SVM	Support Vector Machine
TRF	Trust Region Framework
VAE	Variational Autoencoder

doi:10.6342/NTU202501972

doi:10.6342/NTU202501972

Chapter 1 Introduction

1.1 Motivation

The rapid advancement of wireless communication technologies, including 5G, satellite internet, the Internet of Things (IoT), and the forthcoming 6G systems, has imposed increasingly stringent requirements on antenna performance. Modern antennas must operate efficiently across broader frequency bands, adapt to dynamic environments, integrate into compact or conformal platforms, and support complex configurations such as multiple-input multiple-output (MIMO). These demands significantly increase the complexity of antenna design while tightening constraints related to size, material selection, and manufacturability [1].

At the same time, emerging applications, ranging from wearable and implantable medical devices to autonomous vehicles and aerospace systems, call for antennas that are high-performing, reconfigurable, and rapidly deployable [2]. Traditional antenna design approaches, which typically involve iterative manual tuning and time-consuming electromagnetic (EM) simulations, are increasingly inadequate in meeting these evolving challenges.

Consequently, there is growing interest in automating the antenna design process.

doi:10.6342/NTU202501972

Design automation aims to reduce reliance on expert knowledge, shorten development cycles, and enable exploration of complex and non-intuitive design spaces. With the convergence of optimization techniques, high-performance computing, and machine learning (ML), antenna design is being transformed into a data-driven, algorithmically guided discipline [3][4].

1.2 Background

Historically, antenna design has been a manual, expertise-intensive process. Designers would rely on analytical models and empirical tuning guided by experience and trial-and-error. The advent of numerical EM solvers, such as the finite-difference time-domain (FDTD) method and the method of moments (MoM), shifted the field toward simulation-driven design. Despite these advances, the workflow remained highly iterative and computationally demanding.

To improve design efficiency, optimization algorithms were introduced to automate the tuning of key parameters. Early methods employed heuristic algorithms such as Simulated Annealing (SA), Genetic Algorithms (GAs), Particle Swarm Optimization (PSO), and the Covariance Matrix Adaptation Evolution Strategy (CMA-ES). These techniques are well-suited for non-convex, black-box problems typical of EM-based optimization, though they often require a large number of simulations to reach convergence.

From the perspective of problem formulation, antenna optimization can generally be categorized into parameter optimization and topology optimization. Parameter optimization focuses on adjusting geometric parameters or boundary conditions within a predefined structure. In contrast, topology optimization treats the design domain as a material dis-

doi:10.6342/NTU202501972

tribution problem, enabling the discovery of entirely new structures. These structures are often represented as binary or continuous material grids. While parameter optimization is more straightforward and easily integrates with commercial EM solvers, topology optimization provides greater design freedom and the potential to discover unconventional, high-performance geometries that are difficult to conceive using traditional methods.

With advances in computational resources, machine learning (ML) has recently emerged as a powerful complement to traditional optimization techniques. ML methods, such as surrogate modeling, neural networks, and reinforcement learning, are being explored to accelerate optimization, reduce simulation costs, and even enable inverse design capabilities.

This thesis builds upon these developments by reviewing foundational methodologies and proposing novel strategies for automated antenna design. Both emerging ML-assisted techniques and classical optimization approaches are investigated and extended.

1.3 Outline

This thesis is organized into two major parts: the first part focuses on machine learning, and the second part concentrates on topology optimization. Chapter 2 provides an introduction to machine learning techniques and outlines how they can be integrated into antenna design workflows. Chapters 3 and 4 present case studies of ML-based antenna design, including original work conducted in this research.

In Chapter 5, we revisit fundamental optimization principles and demonstrate how topology optimization, particularly when combined with the adjoint method, can dramatically accelerate the design process. Finally, Chapter 6 concludes the thesis and discusses

doi:10.6342/NTU202501972

promising future directions and potential frameworks for further advancing antenna design automation.

doi:10.6342/NTU202501972

Chapter 2 Machine Learning in Antenna Design

2.1 Introduction

To navigate the complex field of machine learning (ML) in antenna research, it is helpful to establish a high-level understanding of their integration. According to Sarker et al. [4], ML applications in antennas can be broadly categorized into three areas: antenna design, antenna optimization, and antenna selection. This thesis focuses on the first two—design and optimization—while excluding antenna selection. Notably, when the design space is sufficiently large, antenna optimization can be viewed as a subset or alternative formulation of antenna design.

This thesis explores ML integration into antenna design workflows, followed by an examination of topology optimization using adjoint methods. We begin with a brief overview of machine learning.

doi:10.6342/NTU202501972

2.2 A Brief Introduction to Machine Learning

In strategic games like poker, Go (board game), or team sports, players constantly seek optimal strategies. For simple scenarios, rule-based systems can be designed to perform well. However, in complex cases, where relationships are nonlinear or difficult to define explicitly, the solution becomes a black-box problem. Although we may not know the precise functional mapping from input to output, we can often learn its behavior using data. This concept lies at the heart of machine learning: modeling unknown functions from data using statistical techniques.

Machine learning enables computers to learn patterns and make predictions from data without being explicitly programmed. Core components of ML include:

- Data acquisition: Collecting and preparing relevant training data.
- Model training: Using algorithms (e.g., linear regression, neural networks) to learn mappings from inputs to outputs.
- Loss functions: Measuring prediction error (e.g., mean squared error for regression or cross-entropy for classification) and guiding optimization.

Conceptually, an ML model represents a function $f(x)$. For instance, in $f(x) = ax^2 + bx + c$, the parameters a, b, and c start randomly and are optimized using training data. More sophisticated models, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and variational autoencoders (VAEs), differ structurally, with each architecture imposing a form of model bias. This "bias" refers to the assumptions embedded in the model's structure and determines its ability to generalize. A poorly

doi:10.6342/NTU202501972

chosen architecture may systematically fail to capture meaningful relationships in the data.

On the other hand, overly complex models risk overfitting, where the model learns noise instead of the underlying pattern. For example, using a high-degree polynomial to fit a simple quadratic process may yield good training accuracy but poor generalization. Balancing model complexity and structural suitability is therefore essential to effective learning.

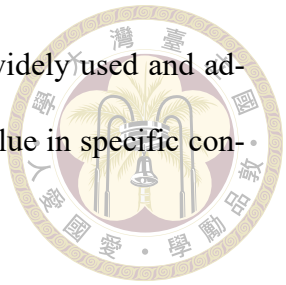
While terms like CNNs, generative adversarial networks (GANs), autoregression, Gaussian Process Regression (GPR), and reinforcement learning can seem overwhelming, ML techniques can be organized into broad categories:

- By task: Regression, classification, etc.
- By learning scenario: Supervised, unsupervised, and reinforcement learning.

Another useful lens is the historical progression of ML paradigms:

- 1970s–1980s: Expert systems based on hand-coded rules.
- 1980s–1990s: Decision trees for interpretable, rule-based predictions.
- 1990s–2000s: Support Vector Machines (SVMs), introducing kernel methods.
- 2000s–2010s: Neural networks and deep learning leveraging GPUs and big data.
- 2017–present: Transformer architectures revolutionizing sequence modeling and dominating state-of-the-art NLP (Natural Language Processing) and computer vision tasks.

doi:10.6342/NTU202501972



Although neural networks and transformers are now the most widely used and advanced architectures, older models like GPR and SVMs still offer value in specific contexts.

In this thesis, references to machine learning generally imply the use of neural networks unless otherwise stated. These networks are loosely inspired by the brain: each neuron performs a weighted sum of inputs, applies an activation function (e.g., ReLU or sigmoid), and passes the result to the next layer. Learning occurs through backpropagation, a gradient descent-based algorithm that adjusts weights to minimize prediction error, similar to fine-tuning a guitar by adjusting string tension.

For instance, a CNN trained on cat and dog images adjusts its weights iteratively to recognize features like ears or fur textures. In language tasks, an RNN learns to predict the next word in a sentence based on sequential patterns. Through such iterations, neural networks learn to model complex relationships from raw data.

2.3 Machine Learning in Antenna Design

2.3.1 Antenna Design as an Inverse EM Problem

Antenna design is fundamentally an inverse electromagnetic (EM) problem. While forward EM problems predict performance (e.g., gain, impedance, radiation pattern) from a given antenna geometry, inverse problems aim to determine the geometry or material configuration that satisfies a set of performance requirements. Solving inverse problems is inherently more difficult due to the nonlinear and often non-unique nature of the design-to-performance relationship.

doi:10.6342/NTU202501972

There are two main approaches to the inverse design:

2.3.1.1 Optimization-Based Forward Simulation

This is the traditional approach, where forward EM solvers (e.g., CST, HFSS) are used to simulate antenna performance. Designers iteratively adjust parameters using trial-and-error or optimization algorithms to converge on a suitable design. While effective, this method can be computationally expensive, especially for high-dimensional or complex geometries, due to the large number of simulations required.

2.3.1.2 Learning the Inverse Function Directly

Alternatively, one can attempt to learn a direct inverse mapping from performance targets to design parameters. However, this is more challenging due to the non-uniqueness and instability of inverse functions. ML-based models can be trained to approximate such mappings, but care must be taken to handle issues like underdetermined mappings, solution ambiguity, and overfitting. Techniques such as regularization, multi-objective optimization, and probabilistic modeling are often employed to address these challenges.

2.3.2 Surrogate Modeling for Antenna Design

Surrogate modeling offers a compelling solution to the computational bottlenecks of traditional antenna design. A surrogate model is a data-driven ML model that approximates the relationship between design parameters and performance. By serving as a fast and efficient proxy for EM simulations, surrogate models can dramatically accelerate both optimization and inverse design workflows.

doi:10.6342/NTU202501972

2.3.2.1 Surrogate Models for Optimization

In this context, a surrogate model learns to approximate the forward function: it predicts antenna performance based on input parameters. For example, a neural network can be trained on a dataset of antenna shapes and their corresponding simulated gains, bandwidths, or radiation patterns. Once trained, this model enables rapid evaluation during optimization, reducing the number of costly simulations and accelerating convergence.

2.3.2.2 Surrogate Models for Inverse Modeling

Here, the surrogate model is trained to map desired performance specifications to design parameters, approximating the inverse function. For example, given a target radiation pattern, the model suggests corresponding antenna geometries. This is particularly useful in cases where analytical inverse solutions are intractable. However, non-uniqueness and data sparsity pose challenges that require careful treatment, such as incorporating physical constraints or generating multiple candidate solutions [5].

2.4 Advanced Topics in Surrogate Modeling

2.4.1 Surrogate vs. Generative Models

ML models in antenna design can generally be divided into two categories: generative models and surrogate models [6].

doi:10.6342/NTU202501972

2.4.1.1 Generative Models

Generative models, such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), are trained to produce new samples that resemble the training data. In antenna design, they can be used to generate novel antenna shapes or layouts. While creative and useful for exploring diverse design options, generative models typically do not target specific performance goals and are less directly useful for solving the inverse EM problem.

2.4.1.2 Surrogate Models

In contrast, surrogate models are designed to approximate a known function: either the forward mapping (design → performance) or the inverse (performance → design). Surrogate models are task-specific and optimized for accuracy and efficiency, making them well-suited for performance-driven design tasks. This thesis focuses on surrogate models due to their relevance to targeted and practical antenna design.

2.4.2 Training a Surrogate Model: Step-by-Step

Training a surrogate model involves several key steps:

1. **Data Collection**: Gather a dataset of antenna designs and their corresponding EM performance, typically using simulation tools.

2. **Model Selection**: Choose an appropriate ML model (e.g., neural network, SVM, GPR) based on problem complexity and data availability.

doi:10.6342/NTU202501972

3. **Training**: Use optimization algorithms (e.g., backpropagation) to minimize the prediction error on the training set.

4. **Validation and Testing**: Evaluate the model on unseen data to ensure generalization and prevent overfitting.

5. **Deployment**: Apply the trained model to predict performance or suggest designs during optimization or exploration.

2.4.3 Offline vs. Online Learning

Two training paradigms are commonly used in ML:

2.4.3.1 Offline (Batch) Learning

In offline learning, the model is trained once on a fixed dataset. This approach is efficient and stable, making it well-suited for problems with clearly defined data and objectives.

2.4.3.2 Online Learning

In online learning, the model is updated incrementally as new data becomes available. This is useful for adaptive systems, where the design problem evolves or is data-driven.

The choice between offline and online learning depends on the use case. For most antenna design applications where data can be precomputed, offline learning provides faster development cycles and simpler integration, which is the focus in the next chapter.

doi:10.6342/NTU202501972

Chapter 3 Antenna Design Using Offline Surrogate Model

3.1 Recent Developments

Over the past decade, machine learning (ML) has seen increasing integration into the field of antenna design. One common approach is Machine Learning-Assisted Optimization (MLAO), where ML models are combined with optimization algorithms to accelerate the design process. Recent studies have applied neural networks (NNs) and simulated annealing to design wideband antennas [7], and have employed advanced graph neural networks (GNNs) with particle swarm optimization (PSO) for designing printed inverted-F antennas (PIFAs) [8]. (A more detailed discussion on MLAO is provided in Chapter 4.)

Outside of the MLAO paradigm, inverse neural networks have been successfully applied for multi-objective antenna synthesis without the need for optimization loops [5], and have leveraged time-domain representations to reduce data volume [9].

While offline surrogate models are widely used, they can become redundant when the data collection process is significantly more time-consuming than traditional optimization. However, for recurrent problems, such as adjusting antennas in changing environments,

doi:10.6342/NTU202501972

offline models offer a practical and efficient solution. In this chapter, we present a proof-of-concept study that demonstrates the utility of offline surrogate models outside of both MLAO and inverse modeling frameworks. Using a dataset of 280 samples, we train and evaluate several ML architectures to demonstrate their effectiveness.

3.2 Design Example and Goal

A common issue in consumer electronics is the need to fine-tune the antenna in smartphones, which changes in every production cycle. This makes the task ideal for offline surrogate models. In this design example, our goal is to build an ML model capable of predicting optimal geometric parameters for a template planar inverted-F antenna (PIFA) operating within the 2.4-2.5 GHz range (with $|S_{11}| < -6$ dB), given arbitrary feed locations and surrounding environments.

Instead of redesigning the antenna from scratch each time, our model can rapidly suggest optimal configurations, saving substantial time during product development cycles. The core of our approach is training a neural network to emulate and replace an EM solver's parametric optimizer.

3.3 Design Region

The chassis environment, resembling a phone-like background structure, is shown in Fig. 3.1. The PIFA design region situated in the upper-left corner of it.

The design region is illustrated in Fig. 3.2. To mimic environmental variation, we surround a well-designed PIFA with 10 blue blocks (4.5 mm $\times$ 4.5 mm) that can be either

doi:10.6342/NTU202501972

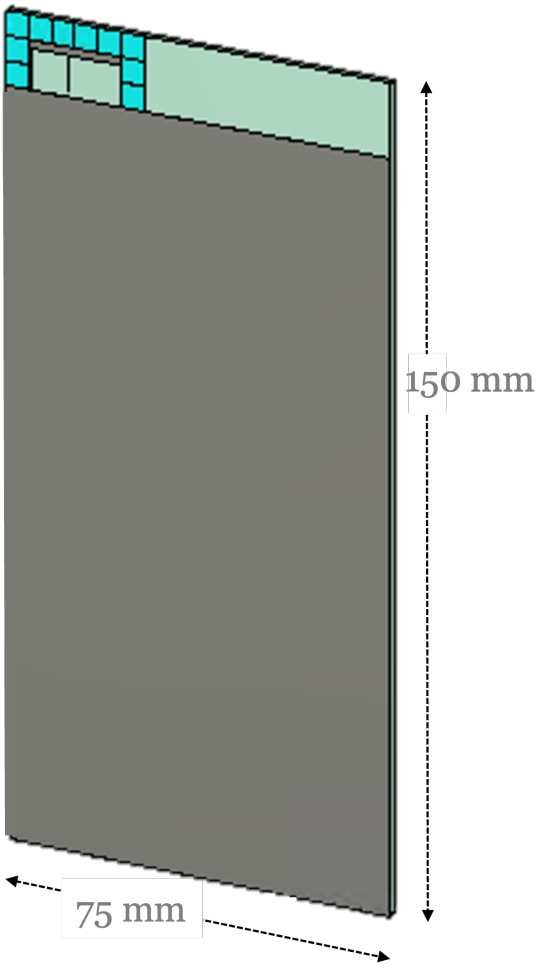


Figure 3.1: Device environment resembling a smartphone (75 mm wide and 150 mm high). The design region is located in the upper-left corner.

doi:10.6342/NTU202501972

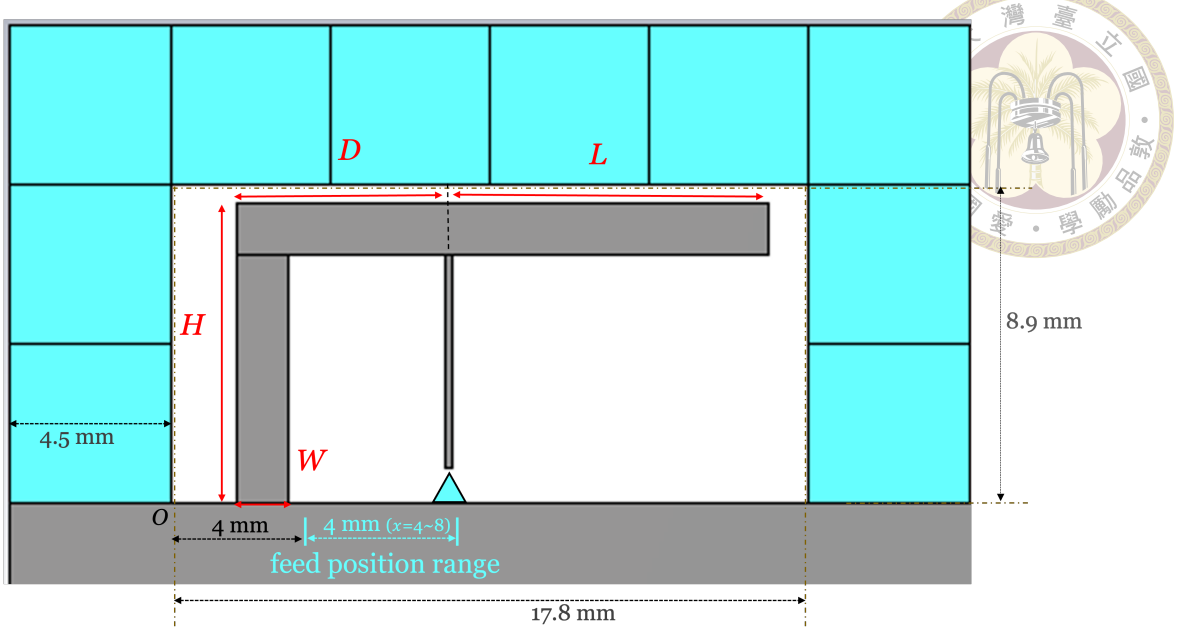


Figure 3.2: Design region. The 11 inputs (in blue) include 10 binary block states and 1 feed position. Outputs (in red) are the four geometric parameters.

air or PEC. These surroundings are encoded as binary inputs (0 for air, 1 for PEC). The feed position, indicated by a blue triangle, serves as a continuous input, varying between 4 mm and 8 mm from the origin (marked as *O* and located in the lower-left of Fig. 3.2). Although other ranges were considered, the 8 mm upper limit, approximately half the 17.8 mm design region width, was selected to provide a meaningful adjustment range without introducing excessive noise in the subsequent ML training process. The 4 mm lower bound ensures that the left antenna region retains sufficient structural integrity.

The model takes 11 input values (10 binary and 1 continuous) and predicts four output parameters: L, D, H, and W (Table 3.1). These four parameters configure the PIFA to maintain $|S_{11}| < -6$ dB in the 2.4-2.5 GHz range. The PIFA's design region is constrained to 17.8 mm $\times$ 8.9 mm, ensuring it remains within predefined boundaries (18 mm $\times$ 9 mm) and does not touch the surrounding blocks.

doi:10.6342/NTU202501972

Table 3.1: Input and output of our surrogate model.

	Description
Input	10 binary numbers and 1 continuous value
Output	4 geometric parameters L, D, H, and W

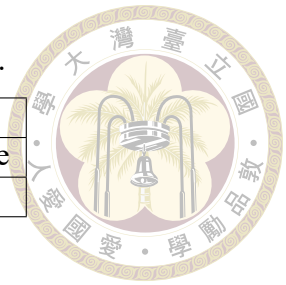


Table 3.2: Example dataset entry (Sample 75).

Sample	Feed Position (mm)	Surrounding Blocks	L (mm)	D (mm)	H (mm)	W (mm)
75	6.83	0110010001	11.17	6.59	8.94	0.24

3.4 Design Flow

Our process involves three main steps: data collection, model training, and performance verification.

Despite the design space being large (2^{10} binary combinations and a continuous feed position), a surrogate model can generalize well with a limited dataset. We use only 280 samples to train and evaluate our models.

We use supervised learning, where each data point consists of input-output pairs saved in a CSV file. Table 3.2 shows an example data point. The following subsection describes how to obtain the data.

3.4.1 Data Acquisition

We automate data collection using CST with Python and VBA scripts. For each configuration, we define the feed location and block materials, and then run CST's optimizer to find optimal antenna parameters that meet the $|S_{11}| < -6$ dB requirement in the target frequency band.

The optimization algorithm we selected in CST is the Trust Region Framework (TRF),

doi:10.6342/NTU202501972

a well-established gradient-based technique. At each iteration, an approximate model of the objective function (typically quadratic) is optimized within a small adaptive trust region. Covariance matrix adaptation evolution strategy (CMA-ES), as another very powerful algorithm, is also available in CST but excessive for this low-dimensional problem. The total 280-sample data acquisition process took approximately three days.

3.5 Model Selection and Performance

We compare the performance of four models:

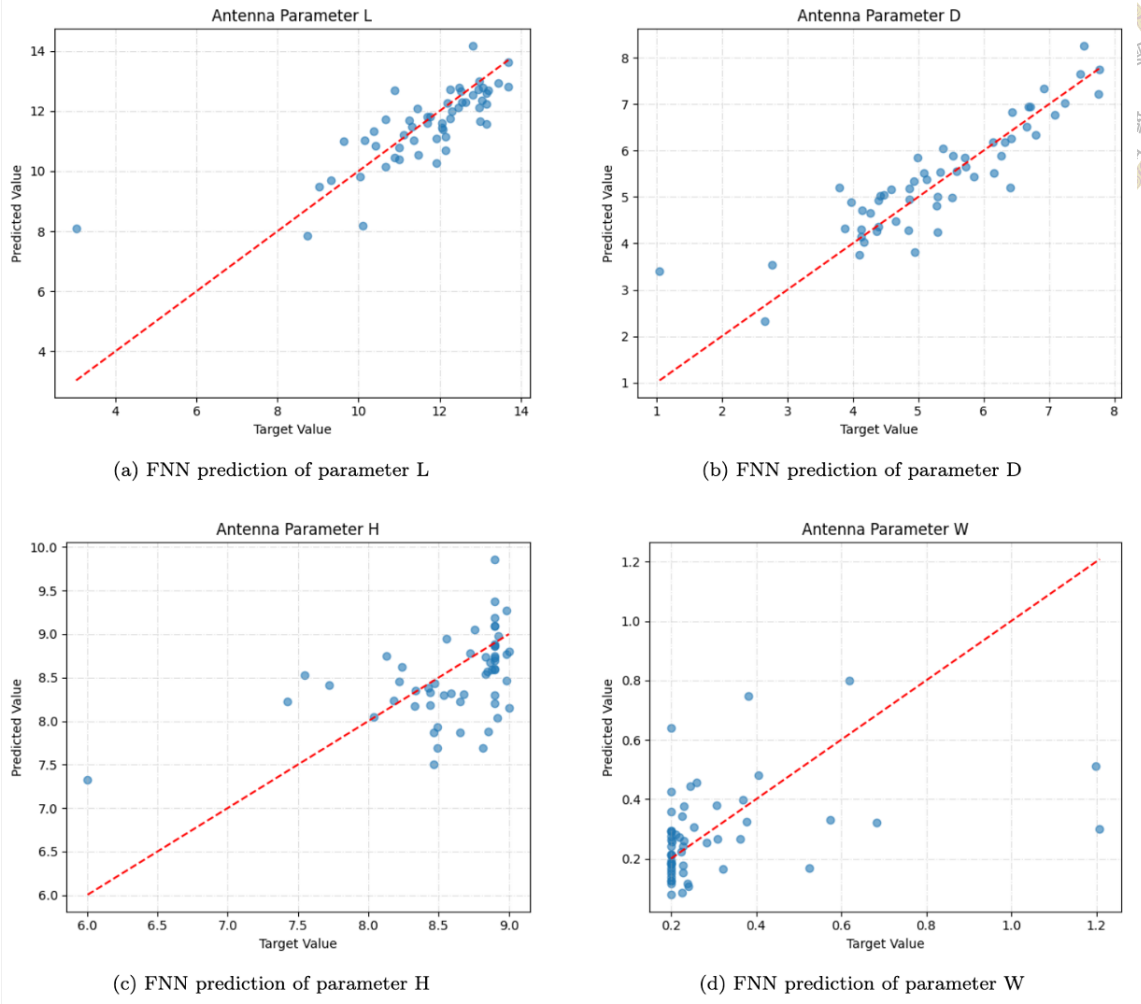
- Feedforward Neural Network (FNN)
- Weighted-FNN
- Recurrent Neural Network (RNN)
- TabTransformer

The 280-sample dataset is split into 224 training samples (80%) and 56 testing samples (20%). Mean squared error (MSE) and scatter plots are used for evaluation.

3.5.1 Feedforward Neural Network (FNN)

The Feedforward Neural Network (FNN), as the simplest kind of neural network architecture, forms the cornerstone of our offline surrogate modeling approach. An FNN is an artificial neural network where data moves from the input layer, passing through one or more hidden layers, to the output layer, without any cycles or feedback loops. In our work, a basic FNN is used with two hidden layers The architecture is:

doi:10.6342/NTU202501972



(a) FNN prediction of parameter L

(b) FNN prediction of parameter D

(c) FNN prediction of parameter H

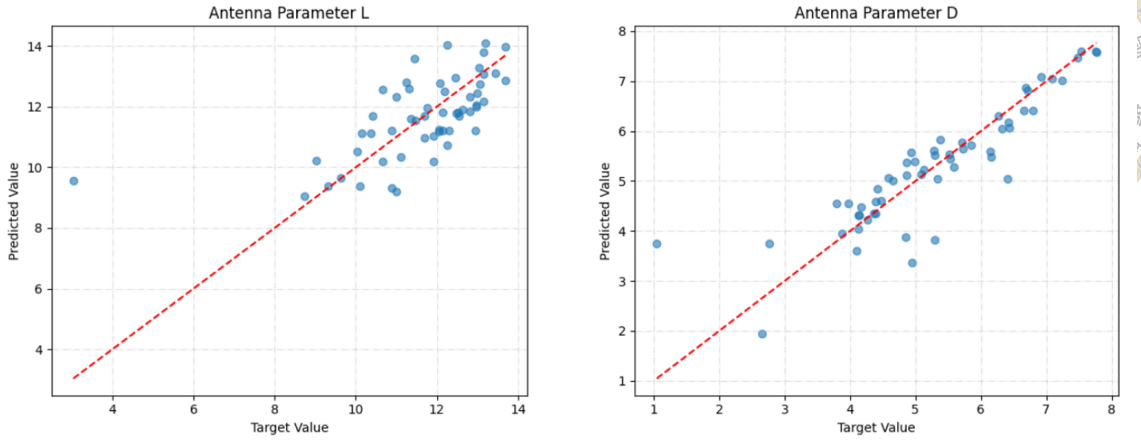
(d) FNN prediction of parameter W

Figure 3.3: FNN prediction vs. ground truth.

- Input: 11 neurons (surrounding blocks and feed position)
- Hidden Layers: 2 layers, each of 64 neurons (black box)
- Output: 4 neurons (L, D, H, W)

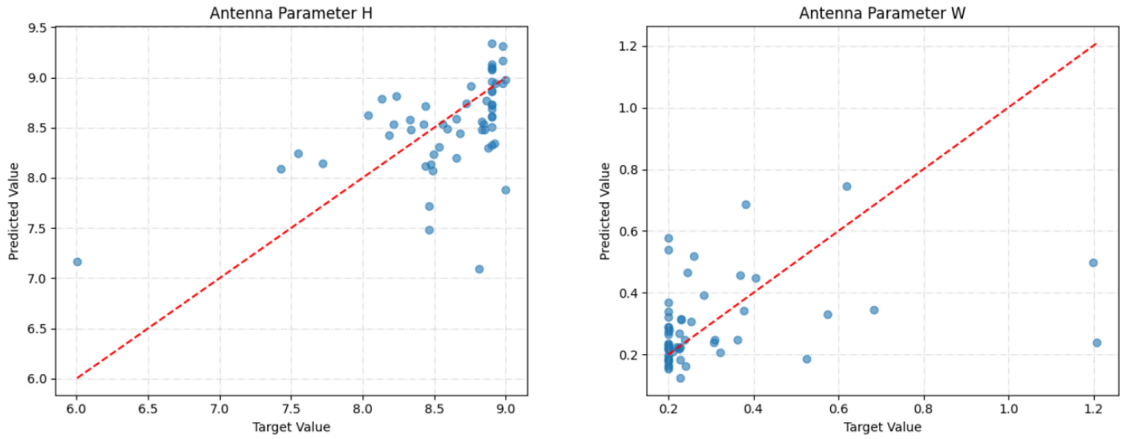
FNN achieves the best performance with MSE = 0.416. As shown in Fig. 3.3, the test data and corresponding predictions are plotted. Ideally, the scattering points would lie precisely on the red dashed line ($y = x$), indicating that our predictions perfectly match the target values (which are the optimized test output data acquired from CST).

doi:10.6342/NTU202501972



(a) Weighted-FNN prediction of parameter L

(b) Weighted-FNN prediction of parameter D



(c) Weighted-FNN prediction of parameter H

(d) Weighted-FNN prediction of parameter W

Figure 3.4: Weighted-FNN prediction vs. ground truth.

3.5.2 Weighted-FNN

The weighted-FNN extends the standard FNN by applying different weighting criteria to the loss function during training, prioritizing certain output parameters over others based on their importance to antenna performance.

In our work, this variant assigns weights to outputs in the loss function, e.g., (1,3,1,1) for L, D, H, and W. Best MSE = 0.578 (Fig. 3.4), with other weightings performing worse. Results suggest that output weighting offers no benefits and may not fully address the inherent limitations of the dataset when applied to modern complex ML models [10]. Stochastic errors, such as random initialization or noisy gradients, could also contribute

doi:10.6342/NTU202501972

to the error.

3.5.3 Recurrent Neural Network (RNN)

The RNN with Gated Recurrent Unit (GRU) was selected to investigate the hypothesis that antenna design involves temporal or spatial feedback loops, such as coupling effects from surrounding objects or variations in antenna size. GRU, a variant of RNN, are engineered to process sequential data by retaining information from prior inputs, making them ideal for tasks where context or feedback is significant (refer to Appendix A).

It was tested in our work to explore feedback effects from surrounding structures. Data enters through 11 input neurons, processed by 2 GRU layers that maintain a memory state (feedback loop).

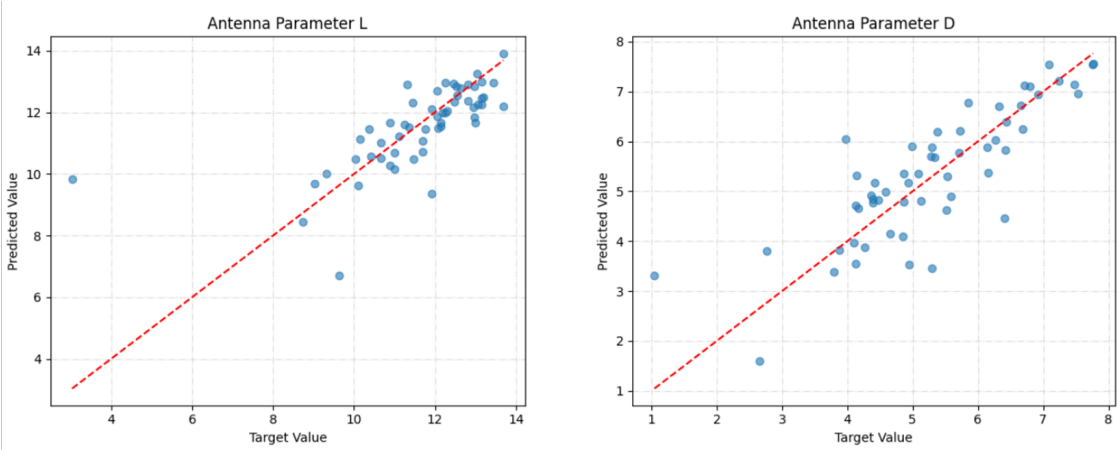
RNN achieves MSE = 0.464 (Fig. 3.5). The slightly poorer performance, compared to FNN, may reflects RNN's over-specialization.

3.5.4 TabTransformer

The TabTransformer model uses attention mechanisms to weigh the importance of different input features (e.g., PEC states and feed position), potentially capturing complex interactions. Its inclusion in this thesis aimed to explore whether advanced techniques could outperform simpler models based on the same small dataset.

MSE = 1.404 (Fig. 3.6) confirms its unsuitability for small dataset.

doi:10.6342/NTU202501972



(a) RNN prediction of parameter L

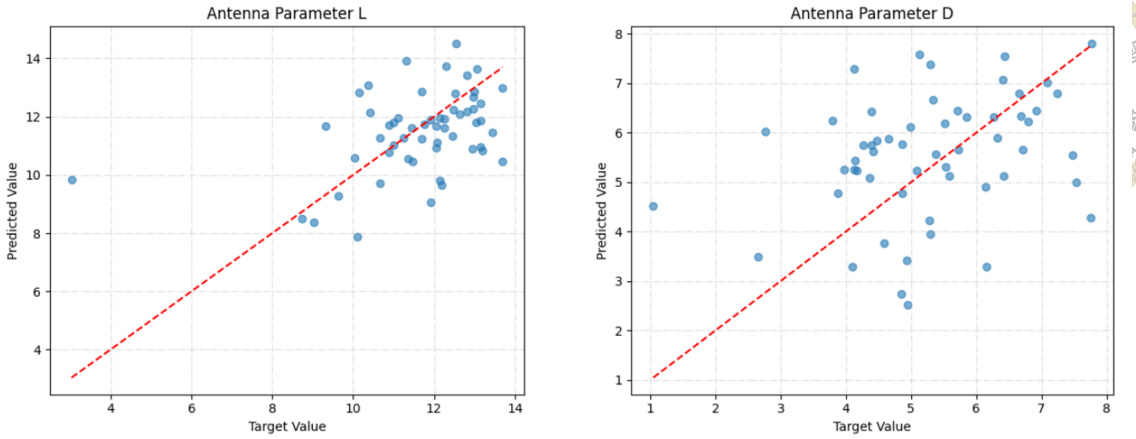
(b) RNN prediction of parameter D

(c) RNN prediction of parameter H

(d) RNN prediction of parameter W

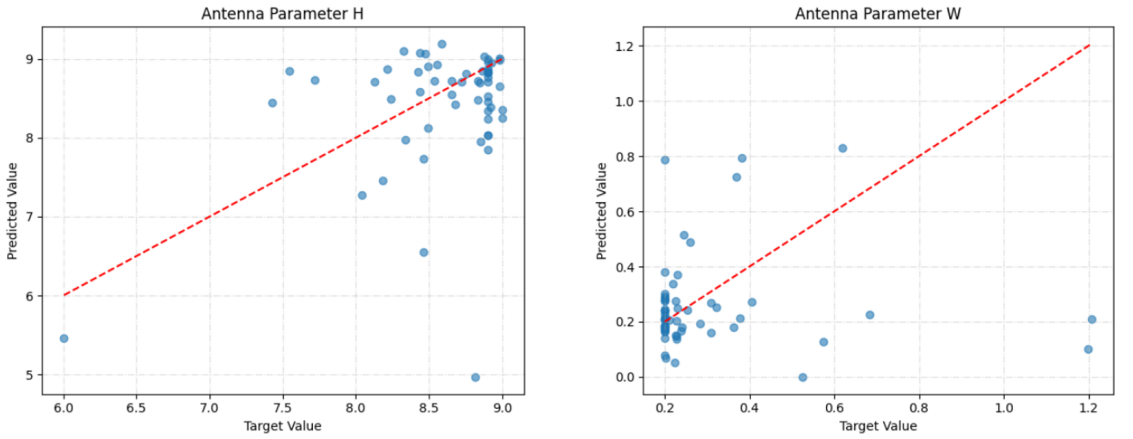
Figure 3.5: RNN prediction vs. ground truth.

doi:10.6342/NTU202501972



(a) TAB prediction of parameter L

(b) TAB prediction of parameter D



(c) TAB prediction of parameter H

(d) TAB prediction of parameter W

Figure 3.6: TabTransformer prediction vs. ground truth.

3.6 Results

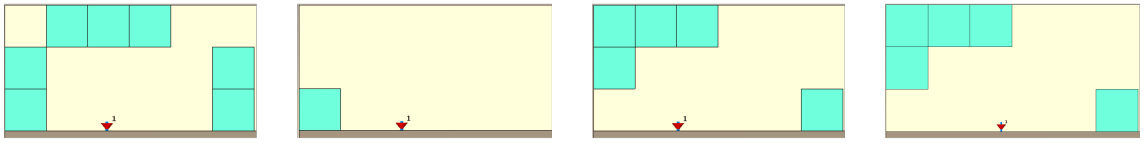
To validate real-world performance, we tested four cases of randomly generated feed locations and states of surrounding blocks (Table 3.3 and Fig. 3.7). The predictions of each model are compared with the CST optimization in Figs. 3.8-3.11, with the expectation that these predictions will align closely with CST's optimized performance. Table 3.4 compares the actual value of the parameters produced by our FNN-based model and CST optimizer, as an example, in Case 3.

TabTransformer produced noticeably worse results. In harsher environments (e.g.,

doi:10.6342/NTU202501972

Table 3.3: Test cases.

Case	Feed Position (mm)	Surrounding Blocks
1	6.58	1111001011
2	6.65	1000000000
3	4.65	0110101010
4	7.85	0110101010



(a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

Figure 3.7: Visualizations of test cases.

Table 3.4: FNN vs. CST optimizer under Case 3 (in mm).

	CST Optimizer	Our FNN-based Model
L	13.35	13.25
D	4.39	4.55
H	8.82	8.74
W	0.27	0.21

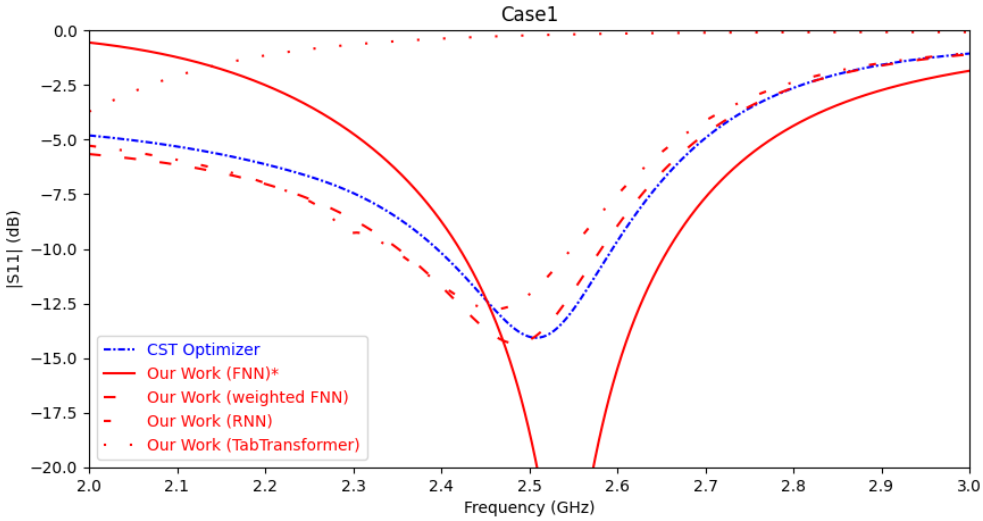


Figure 3.8: $|S_{11}|$ comparison: Case 1

doi:10.6342/NTU202501972

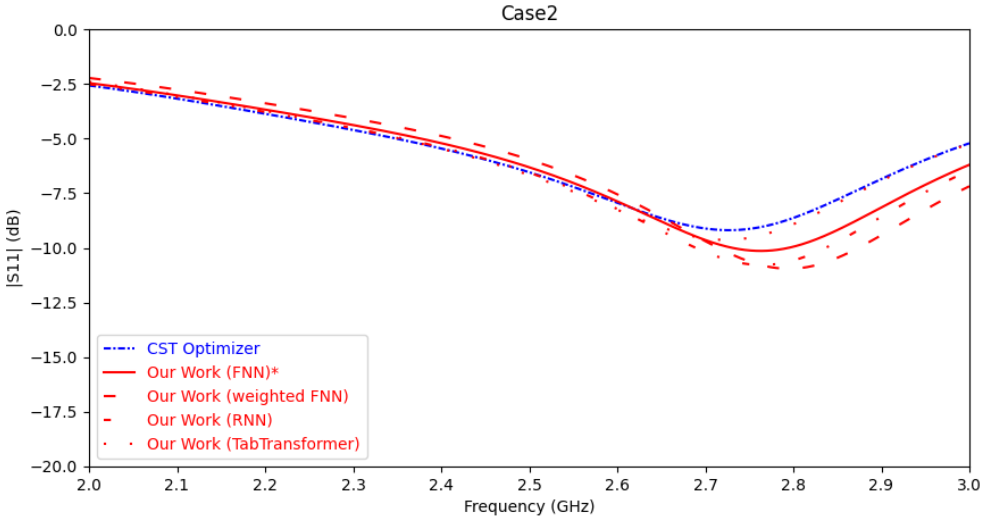


Figure 3.9: $|S_{11}|$ comparison: Case 2

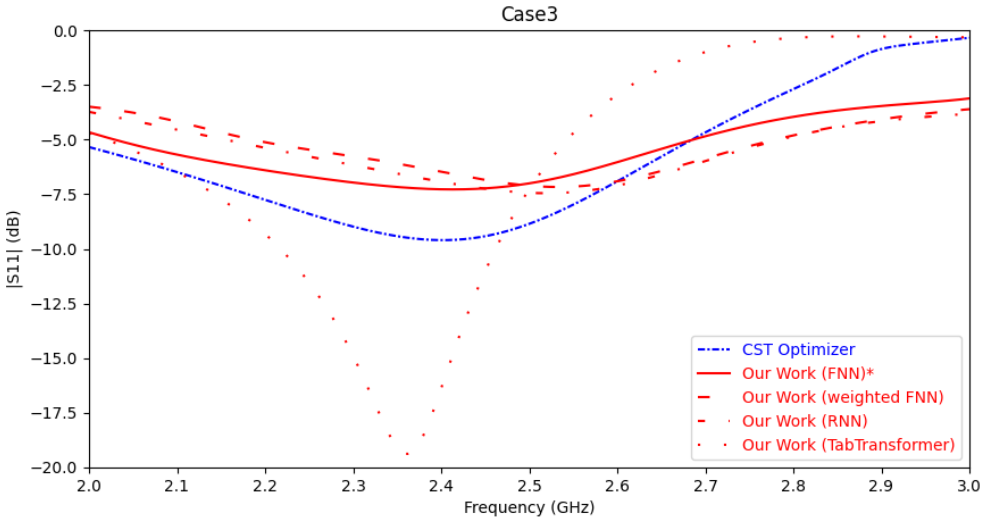


Figure 3.10: $|S_{11}|$ comparison: Case 3

doi:10.6342/NTU202501972

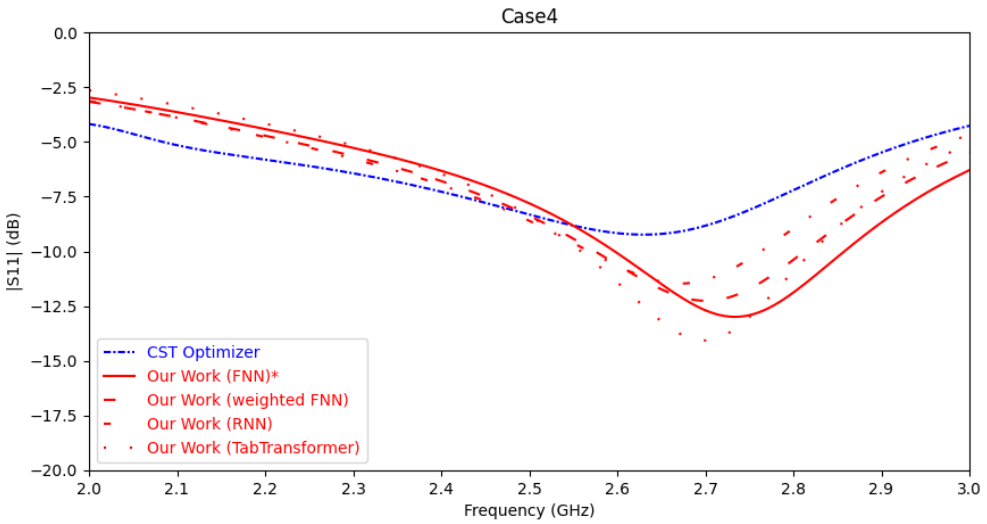


Figure 3.11: $|S_{11}|$ comparison: Case 4

Case 1), model alignment with CST is also less accurate. However, FNN and RNN perform reasonably well, with MSE serving as a useful performance indicator.

3.7 Discussion

In this chapter, we review advances in antenna design using ML. We discuss ML-assisted antenna optimization (MLAO) and inverse modeling, noting that models become obsolete unless they can fully replace electromagnetic (EM) solvers, which is a challenging goal. Therefore, using PIFA as a design example, we propose training a ML model to predict PIFA's geometric parameters under varying environmental conditions, with clearly defined design goals.

To reduce model bias, we selected ML architectures based on the characteristics of the problem, hypothesizing that different approaches could capture unique antenna behaviors. We tested an FNN as a baseline, a weighted FNN prioritizing specific parameters, an RNN to explore feedback loops from coupling effects, and a TabTransformer for mixed

doi:10.6342/NTU202501972

data. The $|S_{11}|$ comparisons (Figs. 3.8-3.11) demonstrate that the PIFAs designed using parameters from our FNN, Weighted-FNN, and RNN models all exhibit performances that align quite well with those optimized by CST. While FNN might appear slightly worse in specific harsh environments (e.g., Case 1) despite its better overall MSE, the overall fidelity across cases is strong for these three models, consistently meeting the target $|S_{11}|$ criteria. TabTransformer, as expected from its higher MSE, produced noticeably inferior results.

This work lays the foundation for ML-driven auto-tuning. Our offline ML models tune PIFAs in less than a second, compared to about 350 seconds for commercial optimizers. Our key contribution is a new approach to using offline surrogate models in antenna design, potentially inspiring researchers to integrate ML more effectively until surrogate models, unlikely soon, fully replace EM solvers.

Incidentally, hyperparameter tuning is worth noting for antenna researchers using ML. We applied random search (RS) for fair model comparisons. Bayesian optimization, though smarter, performs similarly for small models like ours and is harder to implement. For larger models, Bayesian optimization is advised [11].

This part serves as a precursor to the online surrogate modeling approach explored in Chapter 4, highlighting the strengths and limitations of offline models. As mentioned, offline models, trained on static datasets, excel only in narrowly defined scenarios and serve only for specific purposes depending on different use cases. Unlike online learning approaches, which adapt in real-time to new data, offline models lack the flexibility to handle diverse or evolving design requirements, rendering them less powerful than full-wave EM solvers. Therefore, we explore the usage and advantages of online surrogate

doi:10.6342/NTU202501972

modeling in the following chapter.

doi:10.6342/NTU202501972

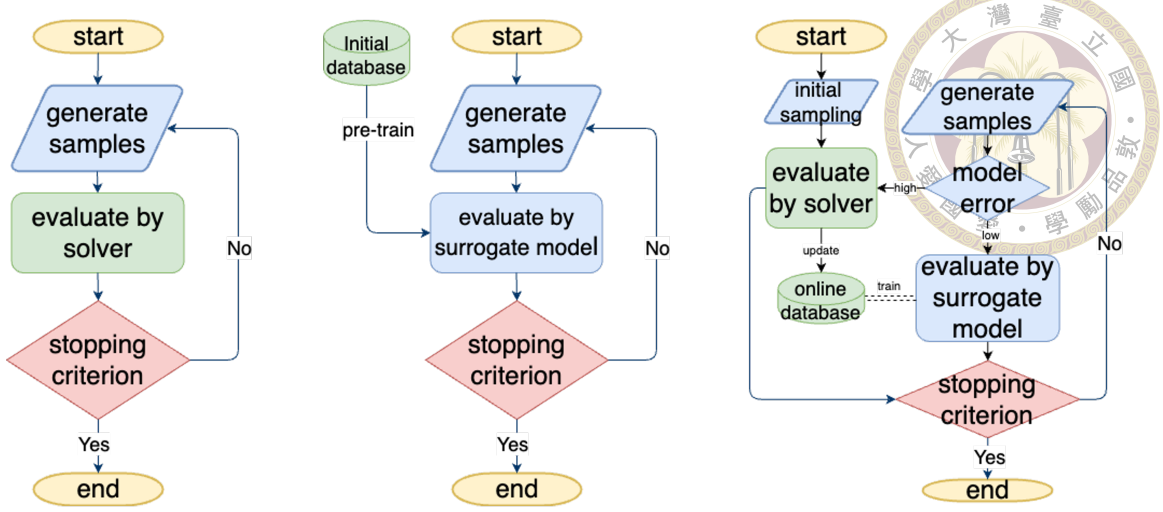
Chapter 4 Antenna Design Using Online Surrogate Models

4.1 Machine Learning-Assisted Optimization

The term “machine learning-assisted optimization” (MLAO) is widely used, though loosely defined, particularly over the past two decades [12]. MLAO has become a standard framework for design automation. Given the prior chapters’ systematic introduction of machine learning and its integration with optimization, we can now interpret MLAO not as a novel concept, but rather as a natural combination of established methods. In essence, MLAO couples computational optimization algorithms with surrogate models to replace time-consuming electromagnetic (EM) solvers. As a result, both offline and online workflows for MLAO have emerged, as illustrated in Fig. 4.1.

MLAO significantly reduces the number of simulations required to achieve optimal results, making it highly beneficial in domains such as engineering and materials science. Applications include polymer blending for desired properties, photonic metadevice design [13], synthesis of Ni-rich cathode materials for improved electrochemical performance [14], and of course, antenna geometry optimization.

doi:10.6342/NTU202501972



(a) Traditional approach (b) Offline MLAO (c) Online MLAO

Figure 4.1: Frameworks for traditional computational optimization and machine learning-assisted optimization (MLAO)

Three main types of surrogate models are commonly used in MLAO frameworks [15]: Gaussian Process Regression (GPR), support vector machines (SVMs), and neural networks. Among them, neural networks are the most widely adopted due to their ability to model complex nonlinear relationships. While SVMs are now largely considered outdated, GPR (also known as the Kriging method) remains popular thanks to its ability to generate both point predictions and uncertainty estimates.

Once a surrogate model is constructed, it can be combined with optimization algorithms. A variety of such methods are used, including gradient-based algorithms like Newton's method or trust-region frameworks (TRF). However, heuristic approaches such as genetic algorithms (GA), particle swarm optimization (PSO), and differential evolution (DE) are more commonly employed with surrogate models, particularly for design spaces that are highly nonlinear and non-differentiable.

Early work in 2014 used DE with GPR surrogate models to approximate antenna responses, thereby reducing simulation costs for structures like inter-chip antennas [16]. A similar framework using parallel computing was later proposed for the design of hybrid

doi:10.6342/NTU202501972

dielectric resonator antennas [17]. Other efforts, such as using k-nearest neighbors (KNN) for patch antenna optimization, claim improved efficiency and performance compared to traditional methods like conjugate gradient descent [18].

As antenna design problems grow more complex, the focus is shifting from parameter optimization to topology optimization. For instance, Wu et al. introduced the MLAO-AGD (Antenna Geometry Design) framework in 2024, which combines GPR and convolutional neural networks (CNNs) to optimize both antenna parameters and topologies [19]. Han et al. proposed a multiphysics MLAO method to design a low-wind-load, dual-polarized antenna using sophisticated spatial transformations to reduce the complexity of the design space [20].

In the following section, we demonstrate how online MLAO can be applied to topology optimization using a simple example that combines GA and CNN.

4.2 1.5 GHz Patch Antenna Design Demonstration

4.2.1 Design Example and Goal

Design a coaxial probe-fed patch antenna operating at 1.5 GHz with $|S_{11}| < -6$ dB.

4.2.2 Design Region

We restrict the design region (blue area in Fig. 4.2) to a 36 mm $\times$ 36 mm square located on a larger 105 mm $\times$ 105 mm FR4 substrate ($\epsilon_r = 4.3$) with a height of 1.6 mm and a copper ground plane of thickness of 0.035 mm at the bottom. The coaxial probe feed is located at $(x, y) = (5, 0)$ mm, relative to the center of the design region at $(0, 0)$.

doi:10.6342/NTU202501972

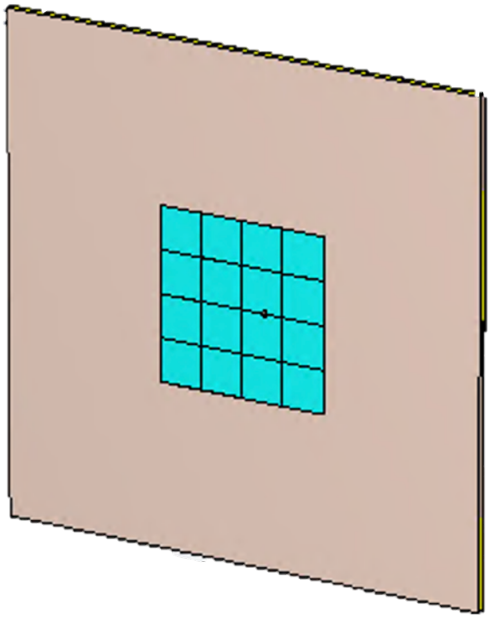


Figure 4.2: Design region (blue), substrate, and bottom ground plane. The feed point is indicated by a black dot.

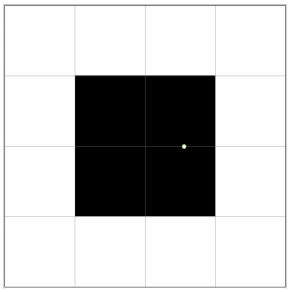
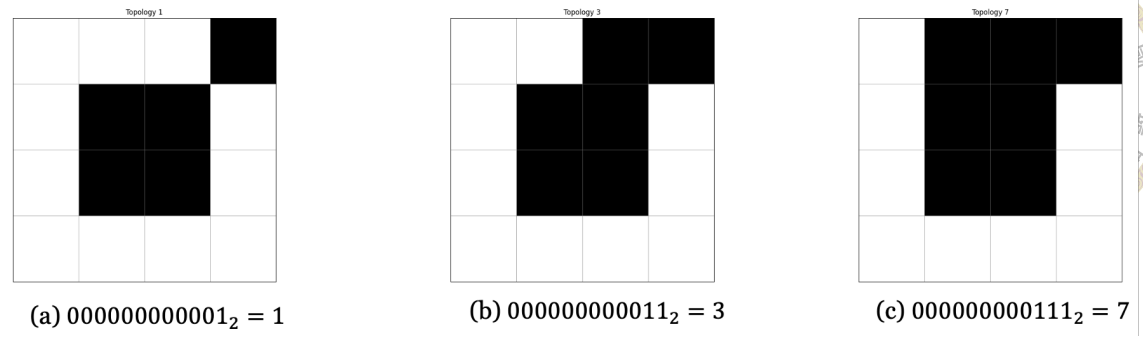


Figure 4.3: Design region with the central four blocks prefilled with PEC, leaving 12 designable blocks.

The design region is divided into a grid of 4×4 pixels. Each pixel can be filled with air or PEC, resulting in a total design space of 2^{16} configurations (or 2^{15} if antenna symmetry is considered). To reduce complexity for demonstration purposes, we prefill the central 4 pixels with PEC, reducing the design space to 2^{12}. This prefilled configuration serves as our baseline and is indexed as topology 0, where all 12 free pixels are air (binary value 0). We introduce a simple coding scheme to represent each topology for better readability.

doi:10.6342/NTU202501972



(a) $000000000001_2 = 1$ (b) $000000000011_2 = 3$ (c) $000000000111_2 = 7$

Figure 4.4: Antenna topologies and their binary-decimal coding.

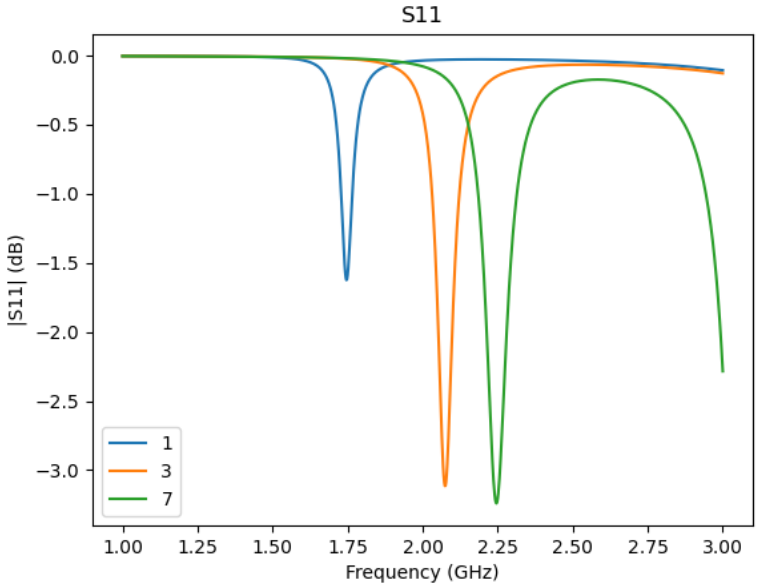


Figure 4.5: $|S_{11}|$ plots for topologies 1, 3, and 7.

4.2.2.1 Topology Coding

Each of the 12 designable blocks is assigned a value of 0 (air) or 1 (PEC). From lower left to upper right, the resulting 12-bit binary string of an antenna topology is encoded into a decimal representation. Figure 4.4 shows examples of topologies 1, 3, and 7 and their actual design topologies.

doi:10.6342/NTU202501972

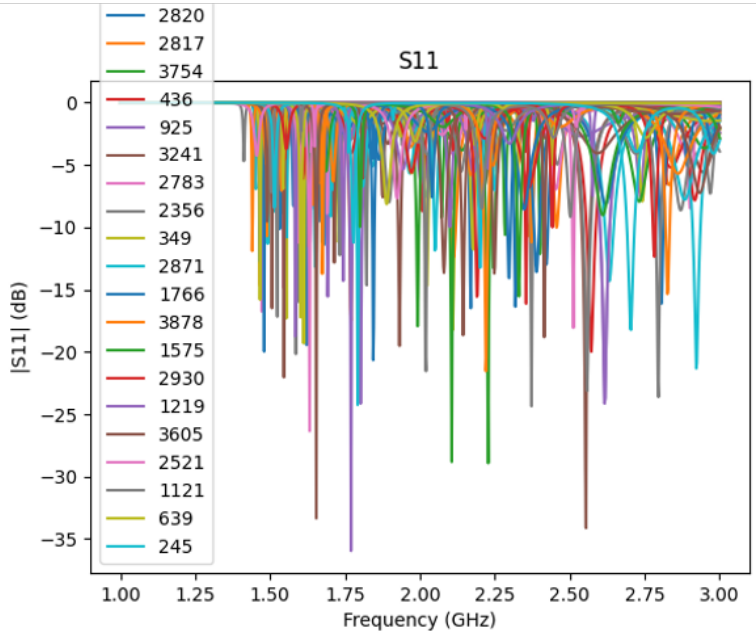


Figure 4.6: $|S_{11}|$ plots of 150 randomly sampled topologies, illustrating design space diversity.

4.2.2.2 Design Space Diversity

The $|S_{11}|$ results of topology 1, 3, and 7 are shown in Fig. 4.5. These poor performances make us wonder whether the overall design space is large enough to accommodate our solution. To assess the diversity of the design space, we randomly generated 150 topologies and evaluated their $|S_{11}|$ performances as shown in Fig. 4.6. The results indicate coverage across 1.4 to beyond 3 GHz, suggesting that a 1.5 GHz design is feasible. For reference, topology 0 resonates at 3.84 GHz (correspond to patch size 18 mm $\times$ 18 mm), while topology 4095 resonates at 1.92 GHz (correspond to patch size 36 mm $\times$ 36 mm). Our target of 1.5-GHz resonant frequency is thus challenging and encourages optimization beyond size-based heuristics.

We now proceed with genetic algorithm (GA) optimization as a baseline before integrating machine learning.

doi:10.6342/NTU202501972

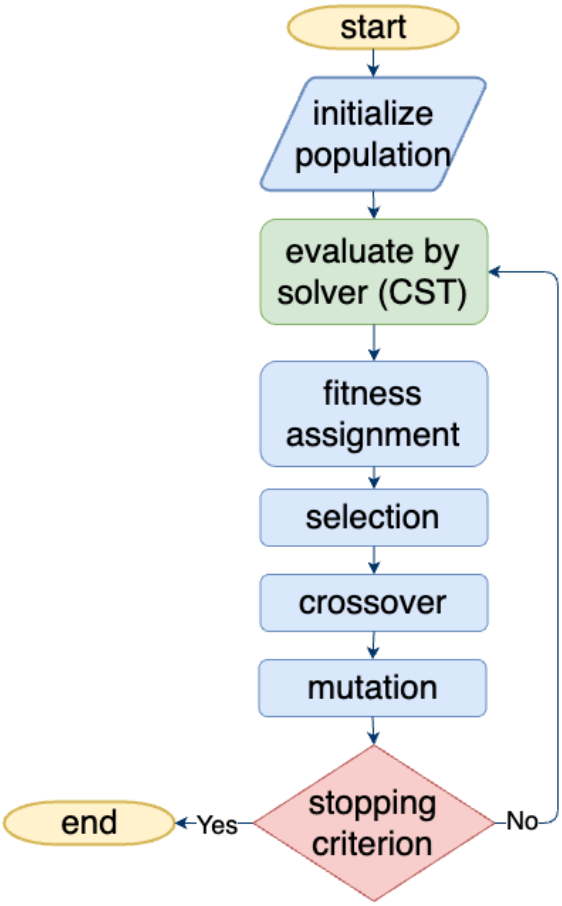


Figure 4.7: Flow diagram of a genetic algorithm.

4.2.3 Genetic Algorithm (GA)

Evolutionary algorithms such as binary Particle Swarm Optimization (PSO) and Genetic Algorithms (GA) are commonly employed for topology optimization due to their natural compatibility with discrete design variables. In this demonstration, we use a GA. Its workflow is illustrated in Fig. 4.7.

GA is an optimization method inspired by biological evolution. It simulates natural processes, such as selection, crossover, and mutation to iteratively evolve a population of potential solutions toward an optimized design. The core steps are:

1. **Population Initialization:** A set of individuals (topologies) is generated randomly. Each individual is a binary string representing one design configuration. For exam-

doi:10.6342/NTU202501972

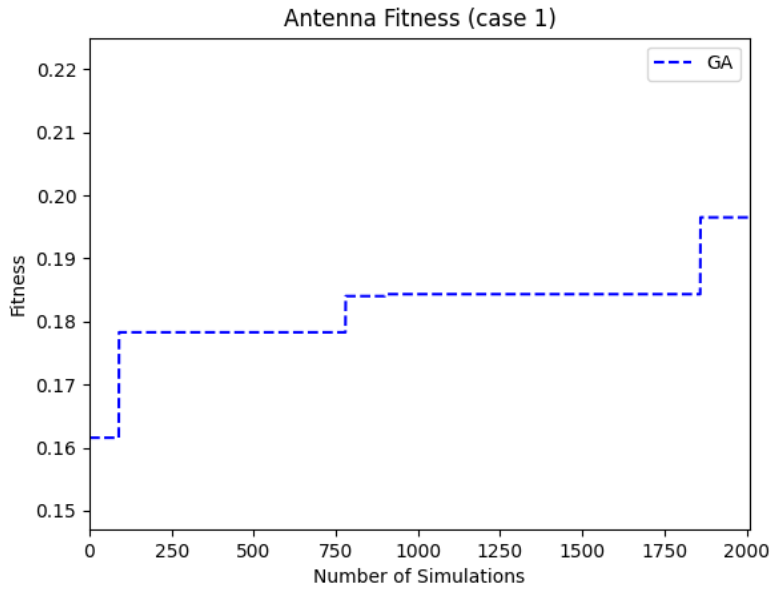


Figure 4.8: Fitness progression of GA across generations.

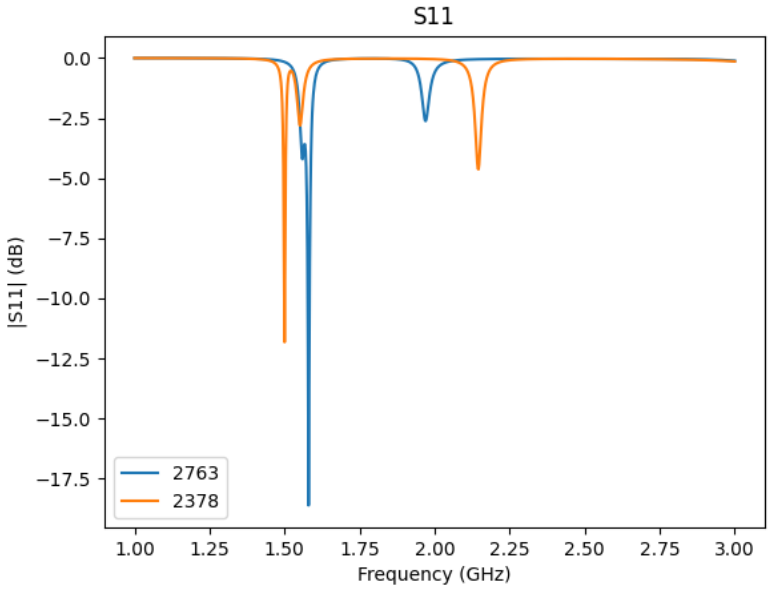


Figure 4.9: $|S_{11}|$ for initial (2763) and final (2378) topologies.

doi:10.6342/NTU202501972

Table 4.1: Population Initialization

Index	Binary Topology	Decimal Code
1	001000100000	544
2	110100010100	3348
3	101011001011	2763
4	001100000110	774
5	000011111101	253
6	000000100110	38
7	011101000111	1863
8	011011000110	1734
9	010101000010	1346
10	001101000111	839

Table 4.2: Fitness values of initial population

Index	Binary Topology	Decimal Code	Fitness
1	001000100000	544	0.00231
2	110100010100	3348	0.00326
3	101011001011	2763	0.16157
4	001100000110	774	0.02684
5	000011111101	253	0.00589
6	000000100110	38	0.00162
7	011101000111	1863	0.00082
8	011011000110	1734	0.00122
9	010101000010	1346	0.00083
10	001101000111	839	0.00122

ple, Table 4.1 lists 10 such topologies.

2. **Fitness Assignment:** Each topology is evaluated using a fitness function. For our design, we define fitness based on how closely $|S_{11}|$ matches the target value over the band 1.425–1.575 GHz:

$$\text{Fitness} = \frac{\displaystyle\int_{1.425}^{1.575} \max(-6, |S_{11}(f)|)\, df}{-6 \cdot (1.575 - 1.425)}$$

Table 4.2 shows fitness values assigned to the initial population.

3. **Selection:** Parents are selected using tournament selection. For example, the top four individuals with the highest fitness values are chosen (Table 4.3).

doi:10.6342/NTU202501972

Table 4.3: Parent Selection

Parent	Binary Topology	Decimal Code	Fitness
1	101011001011	2763	0.16157
2	001100000110	774	0.02684
3	000011111101	253	0.00589
4	110100010100	3348	0.00326

4. **Crossover:** New offspring are generated by recombining parts of two parent topologies. For example, crossing topology 2763 and 774 may yield topology 2566 and 971.

5. **Mutation:** To preserve diversity, random mutations are introduced by flipping bits in the binary strings. Mutation prevents premature convergence.

This process continues for several generations until a termination criterion, such as a certain fitness value or an iteration upper bound, is met. Here, we end the optimization process after 2000 iterations to keep the format consistent. In one experiment, the GA began with topology 2763 and converged to topology 2378, achieving the desired performance, as shown in Fig. 4.9. The optimization progression is shown in Fig. 4.8.

4.2.4 Machine Learning-Assisted Optimization - Genetic Algorithm (MLAO-GA)

The MLAO-GA follows the same workflow as GA but replaces the electromagnetic (EM) solver with a surrogate model when the model's loss is sufficiently low (Fig. 4.10). A convolutional neural network (CNN) is retrained after each batch of EM simulations. When the validation loss is small enough, the surrogate model is used in the next iteration to simulate the topologies in 20 additional iterations, generating a population that is expected to perform well. This population is passed to the EM solver for validation and is

doi:10.6342/NTU202501972

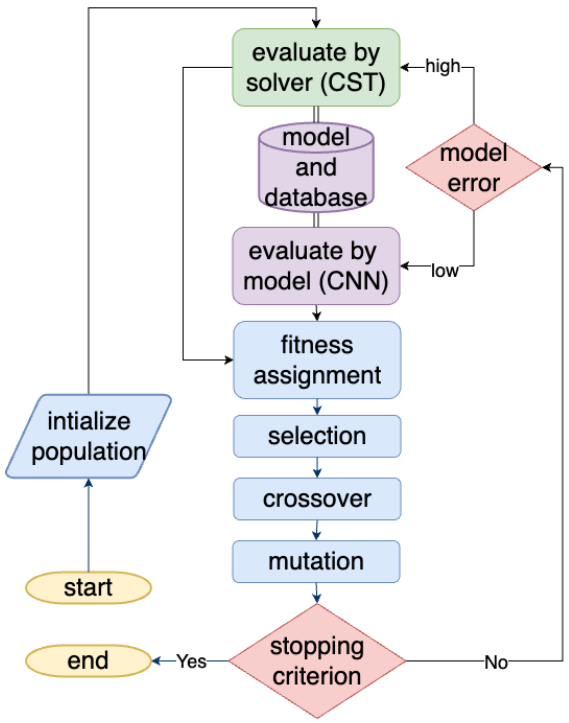


Figure 4.10: Workflow of MLAO-GA incorporating surrogate modeling.

used to update the database for further training.

Although surrogate-based iterations may occasionally yield slightly worse fitness [19], especially early on, they introduce valuable diversity and reduce computational cost. Fig. 4.14 shows one such instance. The problem is handled by eliminating the current population if the fitness is way worse than the last entry.

In one experiment, MLAO-GA started from topology 0 and ended with topology 2392, which met the design specification (Figs. 4.11 and 4.12).

4.2.5 Results

We performed four independent experiments starting from random populations and compared the performance between GA and MLAO-GA. Results are shown in Figs. 4.13-4.16.

doi:10.6342/NTU202501972

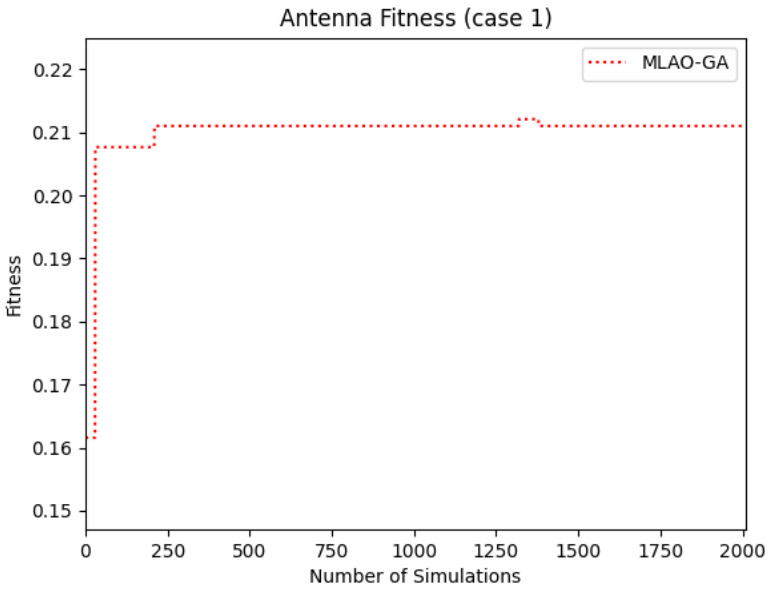


Figure 4.11: Fitness progression using MLAO-GA.

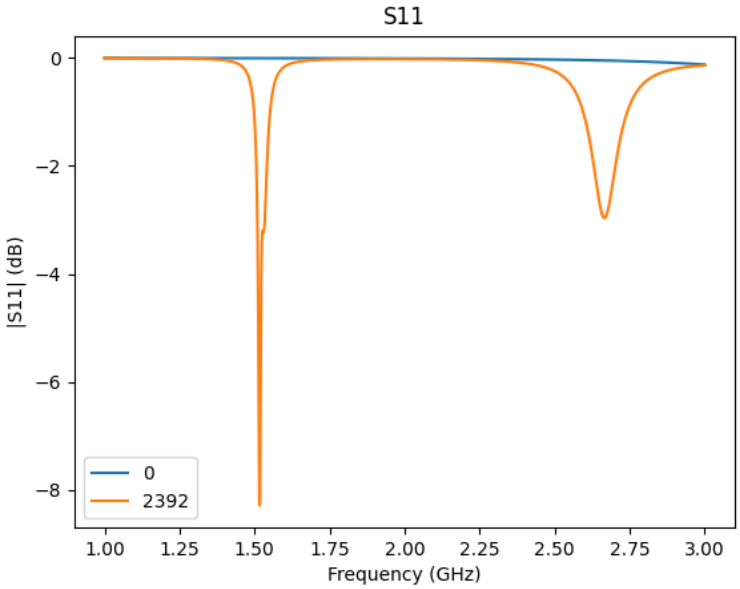


Figure 4.12: $|S_{11}|$ for initial (0) and final (2392) topologies.

doi:10.6342/NTU202501972

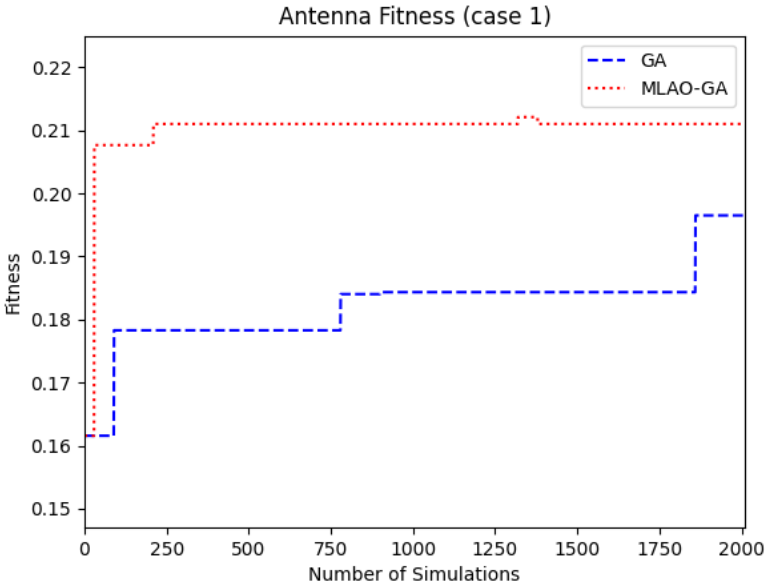


Figure 4.13: 1st trial of fitness comparison.

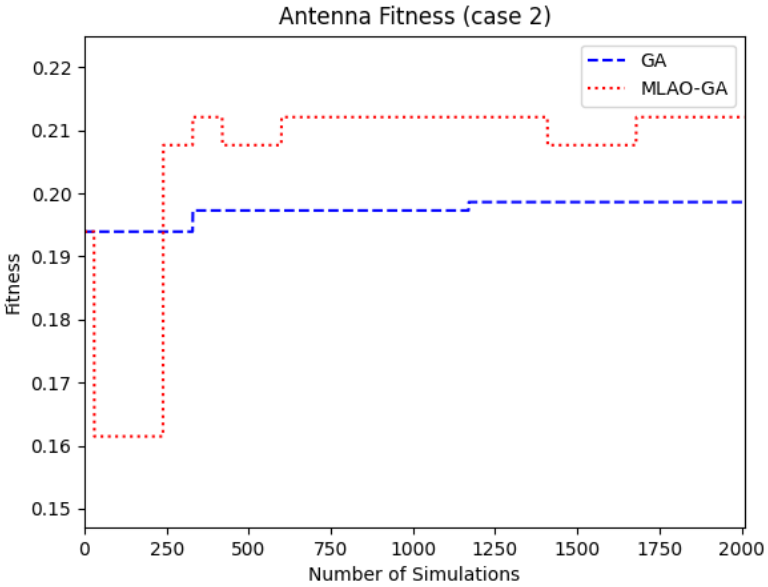


Figure 4.14: 2nd trial of fitness comparison.

doi:10.6342/NTU202501972

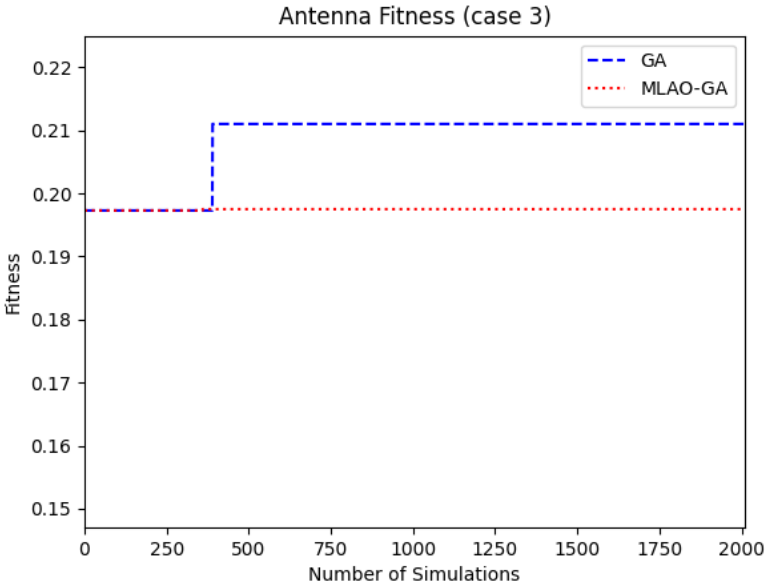


Figure 4.15: 3rd trial of fitness comparison.

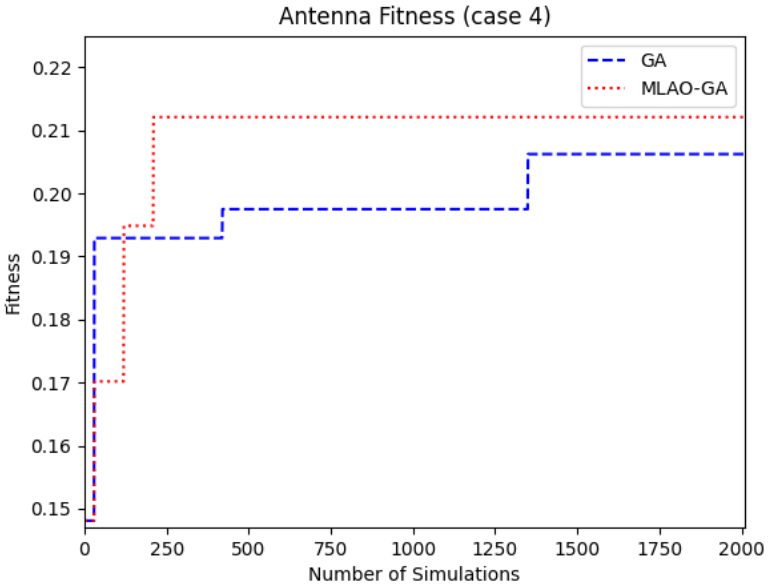


Figure 4.16: 4th trial of fitness comparison.

doi:10.6342/NTU202501972

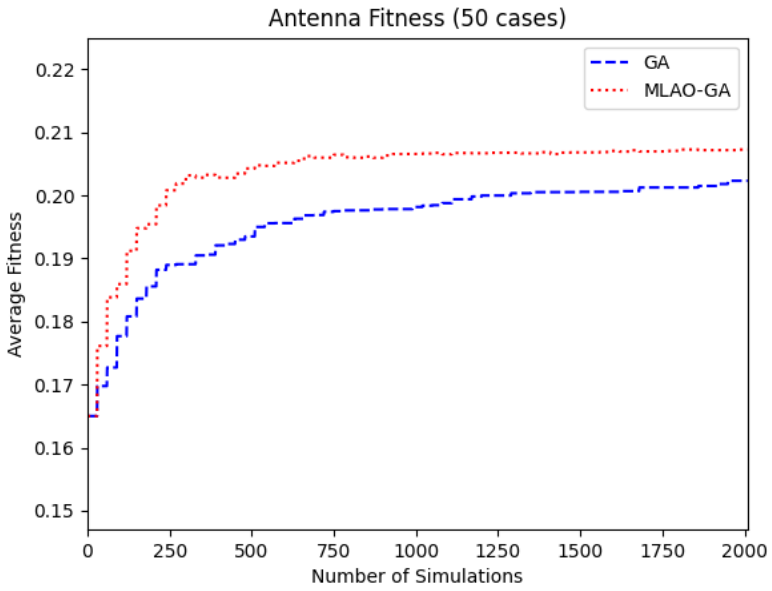


Figure 4.17: Average fitness comparison.

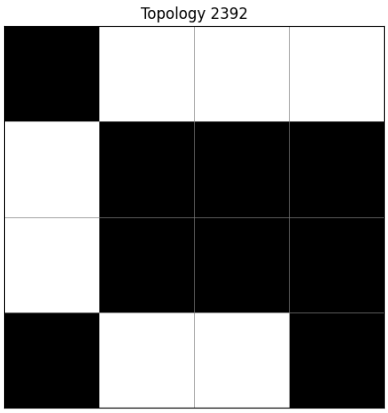


Figure 4.18: Final antenna topology (2392) from MLAO-GA.

One can see that which approach is better is indecisive and depends case by case. To fairly evaluate GA and MLAO-GA, researchers tend to average over many samples. We averaged the results over 50 trials to eliminate randomness and better visualize performance trends. As shown in Fig. 4.17, MLAO-GA consistently outperforms GA, highlighting the feasibility and effectiveness of online surrogate models.

The final design in case 1 (topology 2392) is shown in Fig. 4.18, successfully meeting the specification (Fig. 4.12).

doi:10.6342/NTU202501972

4.3 Discussion

Surrogate models enable a trade-off between computational efficiency, prediction accuracy, and ease of optimization. The choice of surrogate and optimization method depends on the nature of the problem, available resources, and dimensionality of the design space. This chapter demonstrated that MLAO is a practical and effective method for accelerating antenna design workflows.

Wu et al.'s MLAO-AGD framework [19] extends this idea by supporting both parameter and topology optimization, validated with monopole and patch antennas.

Multi-objective optimization is another important direction. Many designs must balance conflicting objectives such as gain and bandwidth, struggling to find the Pareto front. NSGA-II (Non-dominated Sorting Genetic Algorithm II), a well-established multi-objective genetic algorithm, has been used to design dual-band helix antennas [21]. More importantly, Chen et al. applied a MATLAB-based NSGA-II variant along with co-Kriging to design slot antenna arrays satisfying seven performance criteria [12], demonstrating that MLAO can be extended to high-complexity, multi-objective scenarios.

Despite these advances, most studies still focus on problems with fewer than 30 design variables. For high-dimensional cases, model performance degrades unless highly specialized models are used. The curse of dimensionality due to exponentially-growing design space makes it difficult to scale MLAO frameworks efficiently. Unlike large language models (LLMs), surrogate models in physics lack the training diversity and generality to"learn" Maxwell's equations from data alone. Thus, surrogate models are unlikely to fully replace physics-based solvers.

doi:10.6342/NTU202501972

To address such limitations, gradient-based topology optimization using adjoint method offers an alternative approach. In Chapter 5, we will discuss the development and implement one such framework, providing another pathway for optimizing complex antenna geometries.

doi:10.6342/NTU202501972

doi:10.6342/NTU202501972

Chapter 5 Antenna Design Using Adjoint Method

5.1 Introduction to Topology Optimization

Antenna design is fundamentally an inverse electromagnetic problem. Researchers have long sought general and interdisciplinary approaches to address such challenges. Imagine designing a bridge or an aircraft component not by sketching shapes but by allowing a computer to determine material placement, guided solely by performance goals and constraints. This is the essence of topology optimization, a computational method that reimagines structures by optimizing material distribution within a predefined space. Unlike traditional approaches that refine predefined forms, topology optimization begins with a blank slate, iteratively adding or removing material to create efficient, often intricate, designs. Today, it is a cornerstone in fields such as aerospace, automotive, and civil engineering, where the demand for lightweight, high-performance structures aligns with the realities of resource scarcity and environmental concerns.

The development of topology optimization began in the late 20th century, when engineers and mathematicians aimed to integrate structural mechanics with computational power. In the 1980s, pioneers like M.P. Bendsøe and N. Kikuchi laid the foundation with

doi:10.6342/NTU202501972

the homogenization method, modeling materials as porous microstructures to optimize layouts. By the 1990s, the Solid Isotropic Material with Penalization (SIMP) method emerged, simplifying the process by assigning material densities, ranging from void (0) to solid (1), within a finite element analysis (FEA) framework. Other methods like evolutionary algorithms, discussed in the previous chapter, also play a role in topology optimization but struggle to scale effectively. This chapter focuses on gradient-based topology optimization, which relies on sensitivity analysis to guide material placement, often using the adjoint method, a computationally efficient technique for large-scale problems, to compute sensitivity of design changes.

Early efforts established sensitivity analysis for electromagnetic quantities [22] and material optimization in magnetic devices [23], evolving into adjoint techniques for dielectric resonator antenna designs using the finite difference time domain (FDTD) method in 2007 [24]. The level-set method (LSM) was introduced to design dipole antennas with broadband capabilities [25], followed by a SIMP-based approach for sub-wavelength designs [26]. The SIMP method has since become the state-of-the-art, frequently leveraging the adjoint method to compute sensitivities. It was applied to design metallic antennas, including a UWB planar monopole and a dual-band microstrip antenna in 2014 [27], and for near-field sensing in 2015 [28]. In 2016, a Method of Moments (MoM)-based approach was proposed to address bottlenecks caused by the finite element method (FEM) [29]. A hybrid method combining LSM and SIMP was developed to design microstrip antennas with good impedance matching, high polarization purity, maximum gain, and desired beam directions [30]. The same team later proposed a design involving a conical beam in 2018 [31]. Recent advances include hybrid gradient-evolutionary strategies, combining genetic algorithms with SIMP for efficient optimization [32], and hybrid parameter-

doi:10.6342/NTU202501972

topology optimization for antenna designs [33].

The Solid Isotropic Material with Penalization (SIMP) method is a widely used approach in topology optimization, particularly for structural and antenna designs. SIMP assigns each finite element in the design domain a relative density value, ranging from 0 (void) to 1 (solid material). A penalized power-law interpolation governs the material properties, discouraging intermediate densities values to favor clear solid-or-void configurations. In topology optimization using SIMP, the large number of design variables often renders direct sensitivity computation impractical. The adjoint method, a powerful mathematical tool rooted in functional analysis, addresses this challenge by efficiently calculating the gradient of an objective function with respect to numerous design variables. The following sections provide a detailed explanation of the adjoint method, following a practical demonstration of its application.

5.2 Theory of Adjoint Method

The adjoint method is a computationally efficient technique for calculating gradients in optimization problems, particularly those involving large numbers of design variables, such as antenna topology optimization. It is widely used in fields like electromagnetics, structural mechanics, and fluid dynamics to solve inverse problems, where the goal is to determine unknown parameters (e.g., conductivity or material density) that optimize a performance metric (e.g., antenna gain or efficiency). By solving an auxiliary problem (the adjoint problem), the method computes sensitivities with significantly less computational effort than direct methods, making it ideal for complex systems governed by partial differential equations (PDEs).

doi:10.6342/NTU202501972

5.2.1 Optimization Problems

1. **Objective**: Minimize (or maximize) a functional $\mathcal{F}$ in an appropriate function space, such as $H^1(\Omega)$.

$$\mathcal{F}(\mathbf{u}, \mathbf{p}) = \int_{\Omega} F(\mathbf{u}, \mathbf{p}, \mathbf{x})\, d\Omega, \tag{5.1}$$

   where Ω is the domain, $\mathbf{u}(\mathbf{x})$ is the state variable (e.g., a field like temperature or electric field), $\mathbf{p}(\mathbf{x})$ is the design variable (e.g., material properties), and F is some function (e.g., squared field intensity at a point). (Supplementarily, in functional analysis, a functional is a mapping from a function space to a real or complex number and $H^1(\Omega)$ is a Sobolev space.)

2. **Constraint**: The state $\mathbf{u}$ satisfies a PDE:

$$G(\mathbf{u}, \mathbf{p}) = 0 \text{ in } \Omega, \tag{5.2}$$

   with appropriate boundary conditions (e.g., Dirichlet, Neumann, or absorbing).

3. **Goal**: Compute the gradient $\frac{\delta \mathcal{F}}{\delta \mathbf{p}}$ (in the continuous case, this is a functional derivative) to use in gradient-based optimization (e.g., gradient descent).

Directly computing $\frac{\delta \mathcal{F}}{\delta \mathbf{p}}$ involves perturbing each component of $\mathbf{p}$, solving the PDE for the new $\mathbf{u}$, and evaluating the change in $\mathcal{F}$. This is impractical for large dimensional $\mathbf{p}$. The adjoint method avoids this by introducing an auxiliary variable (the adjoint variable) to reformulate the gradient calculation. The core is to elegantly deal with constrained problems by Lagrange multipliers. It is appreciable that Lagrange multipliers are also universally applicable in functional analysis.

doi:10.6342/NTU202501972

5.2.2 The Adjoint Method

1. **Formulate the Lagrangian**: To account for the PDE constraint $G(\mathbf{u}, \mathbf{p}) = 0$, introduce a Lagrange multiplier field $\mathbf{u}^{adj}(\mathbf{x})$ (the adjoint variable). Define the Lagrangian:

$$\mathcal{L}(\mathbf{u}, \mathbf{p}, \mathbf{u}^{adj}) = \mathcal{F}(\mathbf{u}, \mathbf{p}) + \int_{\Omega} \mathbf{u}^{adj} \cdot G(\mathbf{u}, \mathbf{p})\, d\Omega. \tag{5.3}$$

The term $\int_{\Omega} \mathbf{u}^{adj} \cdot G\, d\Omega$ enforces the PDE constraint (when $G = 0$, this term is zero, so $\mathcal{L} = \mathcal{F}$).

2. **Compute the Total Derivative**: Perturb $\mathbf{p} \to \mathbf{p} + \delta\mathbf{p}$ and $\mathbf{u} \to \mathbf{u} + \delta\mathbf{u}$. We have:

$$\delta\mathcal{L} = \frac{\delta\mathcal{F}}{\delta\mathbf{u}} \cdot \delta\mathbf{u} + \frac{\delta\mathcal{F}}{\delta\mathbf{p}} \cdot \delta\mathbf{p} + \int_{\Omega} \mathbf{u}^{adj} \cdot \left(\frac{\partial G}{\partial \mathbf{u}} \cdot \delta\mathbf{u} + \frac{\partial G}{\partial \mathbf{p}} \cdot \delta\mathbf{p} \right) d\Omega. \tag{5.4}$$

Group terms:

$$\delta\mathcal{L} = \left(\frac{\delta\mathcal{F}}{\delta\mathbf{u}} + \int_{\Omega} \mathbf{u}^{adj} \cdot \frac{\partial G}{\partial \mathbf{u}}\, d\Omega \right) \cdot \delta\mathbf{u} + \left(\frac{\delta\mathcal{F}}{\delta\mathbf{p}} + \int_{\Omega} \mathbf{u}^{adj} \cdot \frac{\partial G}{\partial \mathbf{p}}\, d\Omega \right) \cdot \delta\mathbf{p}. \tag{5.5}$$

Since we want to minimize (or maximize) the functional, $\delta\mathcal{L} = 0$. We can then choose $\mathbf{u}^{adj}$ to eliminate the $\delta\mathbf{u}$ term (which we don't want to compute), which is called an adjoint equation.

3. **Adjoint Equation**:

$$\int_{\Omega} \mathbf{u}^{adj} \cdot \frac{\partial G}{\partial \mathbf{u}}\, d\Omega = -\frac{\delta\mathcal{F}}{\delta\mathbf{u}}. \tag{5.6}$$

The difficulty in implementing the adjoint method stems from this tricky aspect: both the boundary conditions and the field sources vary with the setup. For simplicity, we assume that the boundary term vanishes after integration by parts, the

doi:10.6342/NTU202501972

adjoint equation then becomes:

$$G^{adj}\mathbf{u}^{adj} = -\frac{\delta \mathcal{F}}{\delta \mathbf{u}}, \tag{5.7}$$

G^{adj} is the adjoint operator, often similar to, or even the same as, G. In other words, the adjoint equation $G^{adj}\mathbf{u}^{adj} = G^{adj}(\mathbf{u}^{adj}, \mathbf{p}) = -\frac{\delta \mathcal{F}}{\delta \mathbf{u}}$ may differ from $G(\mathbf{u}, \mathbf{p}) = 0$ only by the source term $-\frac{\delta \mathcal{F}}{\delta \mathbf{u}}$. For example, $G = 0$ and $G^{adj} = 0$ may both represent Maxwell's equations, except that the adjoint equation has a different source. Solving the adjoint equation yields $\mathbf{u}^{adj}$.

4. **Functional Derivatives (Gradient)**: With $\mathbf{u}^{adj}$ satisfying the adjoint equation, the functional derivative of $\mathcal{F}$ is:

$$\frac{\delta \mathcal{F}}{\delta \mathbf{p}} = -\int_{\Omega} \mathbf{u}^{adj} \cdot \frac{\partial G}{\partial \mathbf{p}} \, d\Omega. \tag{5.8}$$

This requires only the solution of the adjoint equation (5.7) and the state equation (5.2), regardless of the number of design variables in $\mathbf{p}$. (In most cases, G is linear in $\mathbf{p}$, therefore $\frac{\partial G}{\partial \mathbf{p}}$ is analytical and requires no extra computation.)

The adjoint method is like a "reverse engineering" trick. Instead of perturbing each design variable and solving the PDE, it solves a single adjoint PDE that encapsulates the sensitivity of the objective to all design variables. In the following section, we demonstrate the power of adjoint method and how to implement it.

doi:10.6342/NTU202501972

5.3 1.5 GHz Patch Antenna Design Demonstration

This section draws heavily on the formulations derived in “Topology Optimization of Metallic Antennas” by Hassan et al. [27].

5.3.1 Design Example and Goal

The goal is to develop a coaxial probe-fed planar antenna optimized for operation at 1.5 GHz, which is the same as that discussed in Chapter 4.2.

5.3.2 Design Region

The design region aligns with the configuration detailed in Chapter 4, illustrated in Fig. 5.1. The material is distributed across a 4×4 pixel grid. Unlike the binary material distribution used in Chapter 4, this study employs the SIMP method. Here, the material is characterized by a conductivity distribution $\sigma(\mathbf{x})$, allowing each pixel to have a unique conductivity value ranging from 0 to 5.8×10^7 (S/m).

To showcase the capabilities of the adjoint method, a finer 18×18 pixel design region is also evaluated, as shown in Fig. 5.2, with all other parameters unchanged.

5.3.3 Implementation

The objective function is formulated as:

$$\arg \max_{\sigma \in [\sigma_{\min}, \sigma_{\max}]} \mathcal{W}_{\text{out, coax}}(\sigma),$$

doi:10.6342/NTU202501972

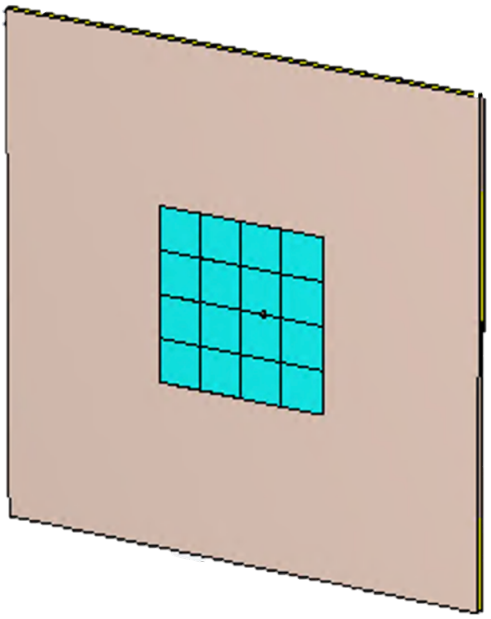


Figure 5.1: 4×4 pixelated design region.

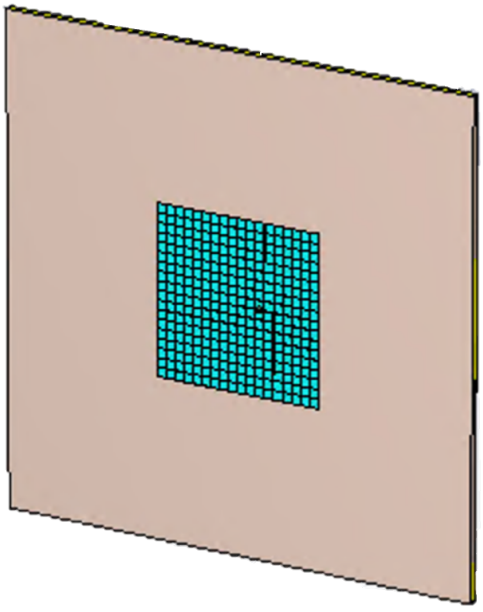


Figure 5.2: 18×18 pixelated design region.

doi:10.6342/NTU202501972

aiming to maximize the antenna's received power in receive mode.

Following Hassan et al., the gradient of the objective function is computed as:

$$\frac{\delta \mathcal{W}_{\text{out, coax}}}{\delta \sigma} = -\int_{\Omega}\int_{0}^{T} \mathbf{E}^{\text{adj}}(t) \cdot \mathbf{E}(T-t)\, dt\, d\Omega,$$

where $[0, T]$ is a sufficiently long time interval to ensure the system reaches a steady state. The time-domain approach facilitates optimization across a frequency band, making it suitable for wideband designs. The electric field $\mathbf{E}$ is obtained by solving the forward problem using CST's time-domain solver. In the forward problem, the antenna operates in receive mode, excited by a 1.5 GHz plane wave. For the adjoint problem, the antenna operates in transmit mode, excited by a pulse at the coaxial feed boundary, which, as derived by Hassan et al., is the time-reversed outgoing signal from the forward problem's coaxial cable. The adjoint electric field $\mathbf{E}^{\text{adj}}$ is then computed by solving the adjoint problem.

The optimization employs gradient descent, with the gradient of the antenna topology calculated iteratively using the adjoint method. To enhance convergence speed, the Adaptive Moment Estimation (Adam) optimizer, detailed in Appendix C, is utilized. This optimization algorithm, extensively used in ML, is well-suited for high-dimensional problems and has been validated for use with the SIMP method by our demonstrative work. The entire process is automated using CST, integrated with Python and VBA scripts.

doi:10.6342/NTU202501972

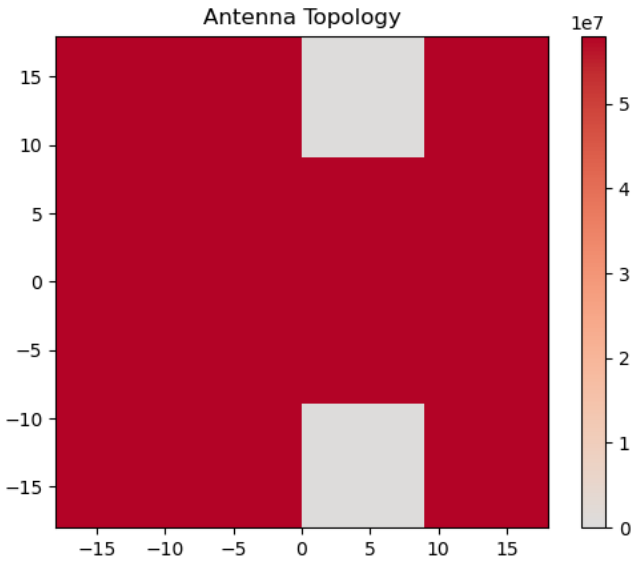


Figure 5.3: Optimized 1.5 GHz antenna topology in the 4×4 pixelated region.

5.3.4 Preliminary Results

5.3.4.1 4 × 4 Pixelated Design

The optimized antenna topology is depicted in Fig. 5.3, with its performance shown in Fig. 5.4.

The optimization converged in 12 iterations, taking approximately 4,692 seconds on the computational system, as shown in Fig. 5.5. This outperforms the MLAO-GA approach from Chapter 4, which required around 300 iterations to converge. Although comparing methods with different material resolutions is not entirely equitable, the result indicates significant potential for further improvement.

The evolution of the antenna topology is illustrated in Fig. 5.6, showing the distribution of actual design variable (for enhanced visualization) updated in the optimization process. The optimization iterations progress from the upper-left to the lower-right of the figure, reading from left to right and then top to bottom. (The design variable is exponen-

doi:10.6342/NTU202501972

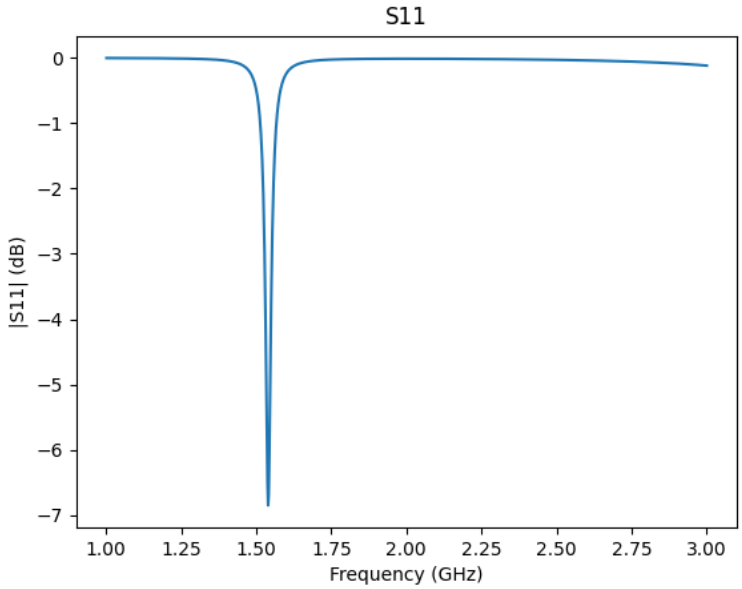


Figure 5.4: $|S_{11}|$ parameter of the optimized antenna in the 4×4 pixelated region.

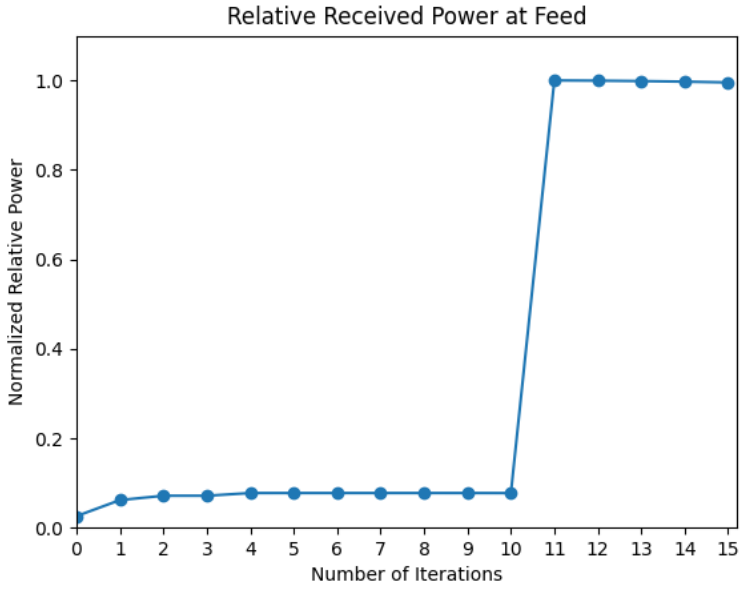


Figure 5.5: Relative received power per iteration in the 4×4 pixelated region.

doi:10.6342/NTU202501972

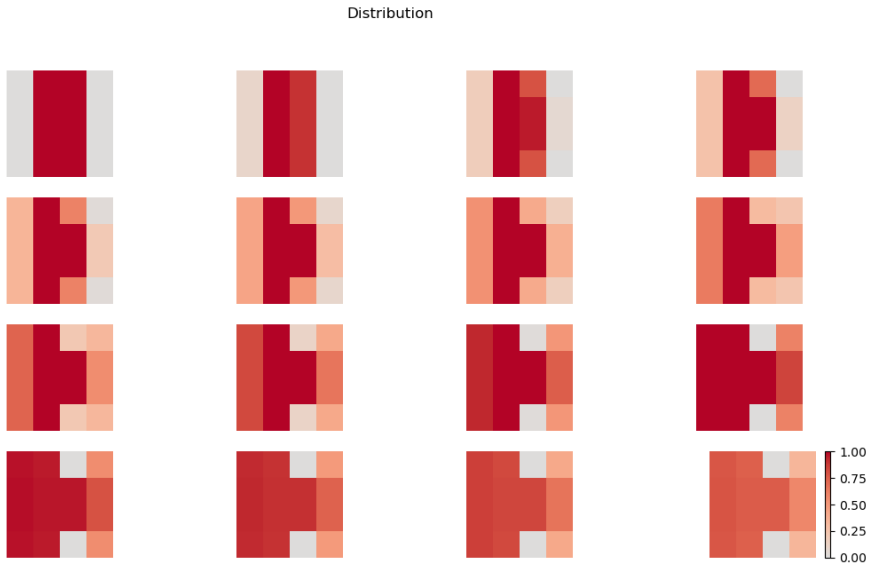


Figure 5.6: Design variable distribution per iteration in the 4×4 pixelated region.

tially mapped from $[0, 1]$ to $[0, 5.8 \times 10^7]$.) The gradient update steps (adaptively processed by Adam) are shown in Fig. 5.7.

5.3.4.2 18 × 18 Pixelated Design

To test scalability, the optimization was applied to an 18×18 pixelated design region. The resulting topology is shown in Fig. 5.8, with performance metrics in Fig. 5.9.

Convergence occurred in 7 iterations, as depicted in Fig. 5.10, demonstrating that the adjoint method's efficiency is largely independent of the design space's dimensionality. However, the increased design flexibility elevates the computational complexity of solving the forward and adjoint problems, resulting in a convergence time of approximately 28,772 seconds.

The topology evolution is shown in Fig. 5.11, and the gradient update steps are presented in Fig. 5.12.

doi:10.6342/NTU202501972

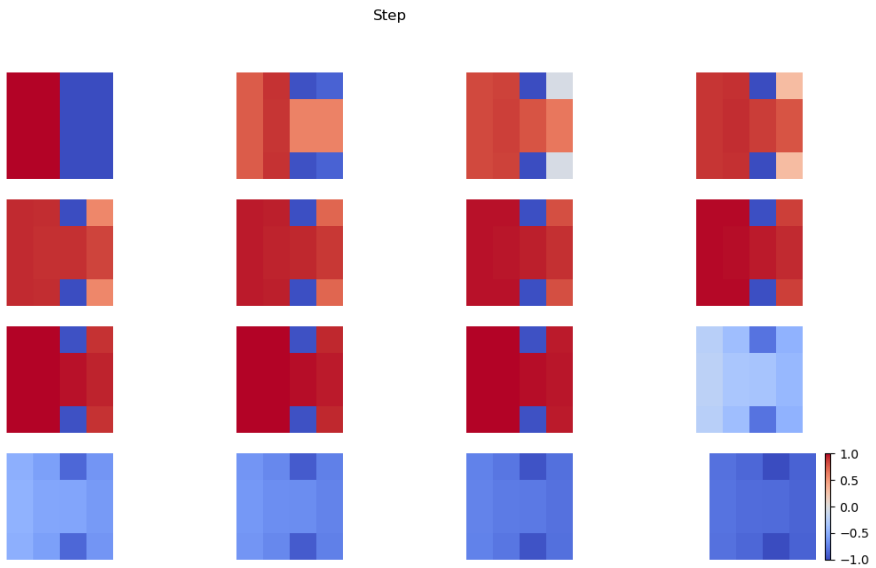


Figure 5.7: Gradient update step per iteration in the 4×4 pixelated region.

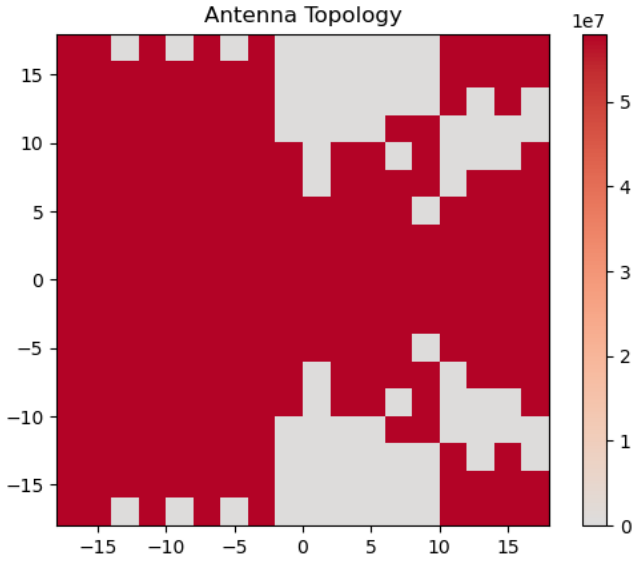


Figure 5.8: Optimized 1.5 GHz antenna topology in the 18×18 pixelated region.

doi:10.6342/NTU202501972

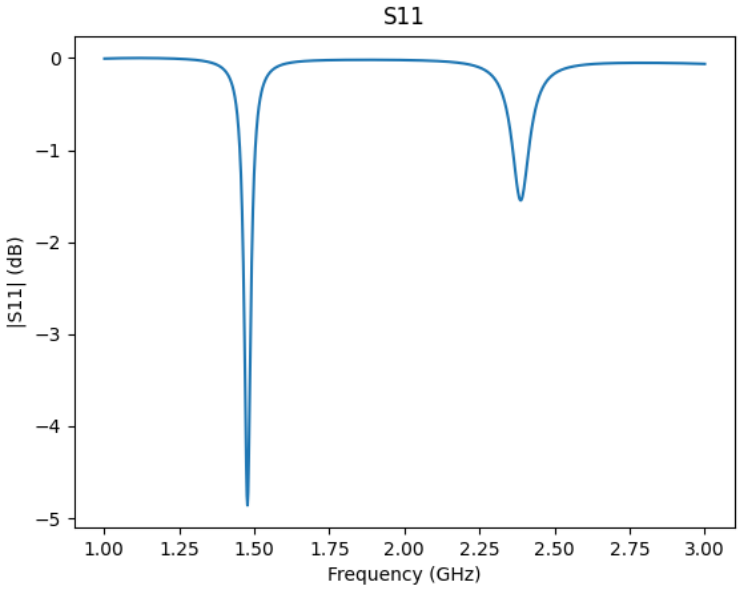


Figure 5.9: $|S_{11}|$ parameter of the optimized antenna in the 18×18 pixelated region.

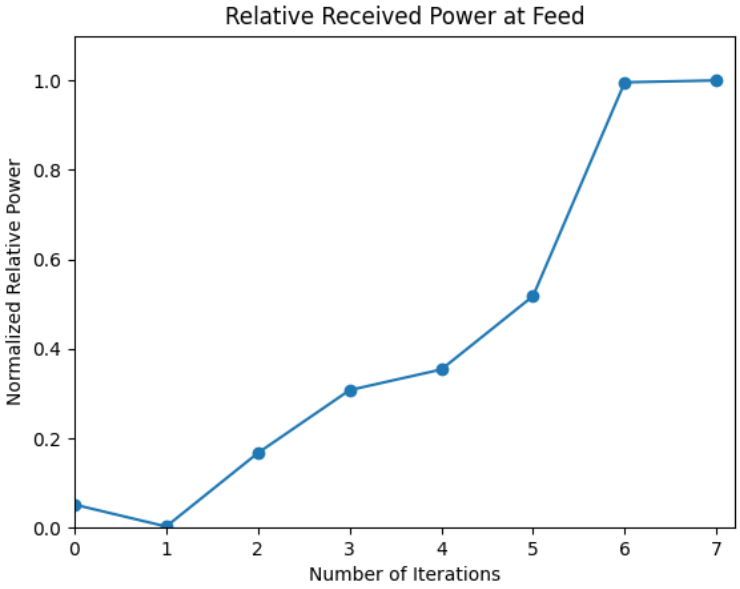


Figure 5.10: Relative received power per iteration in the 18×18 pixelated region.

doi:10.6342/NTU202501972

Figure 5.11: Design variable distribution per iteration in the 18×18 pixelated region.

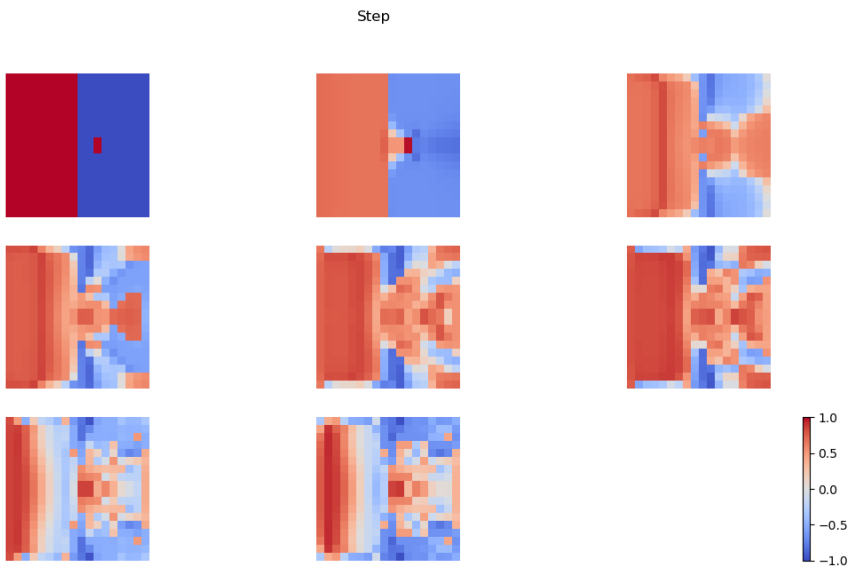


Figure 5.12: Gradient update step per iteration in the 18×18 pixelated region.

doi:10.6342/NTU202501972

The adjoint method demonstrates robust performance across varying design space dimensions, offering significant promise for antenna topology optimization. However, its formulation can be complex, though foundational work by researchers like Hassan et al. and many others mitigates this challenge.

Further investigation is required to enhance the stability, scalability, efficacy, and efficiency of this demonstrative work. Additional testing will help refine the algorithm and broaden its applicability.

5.4 Discussion

The adjoint method, combined with the SIMP approach, remains a state-of-the-art technique for material distribution in antenna design. Its ability to efficiently compute gradients for large-scale problems makes it indispensable for topology optimization. Recent advancements suggest that the adjoint method can also be extended to compute Hessians, as explored in [34]. Calculating the Hessian, which captures second-order sensitivities, could enhance convergence rates in optimization algorithms, potentially leading to faster and more robust antenna designs. This approach is worth noting for its potential to improve performance in complex design problems.

Antenna design has evolved significantly with methods like the adjoint-based topology optimization and emerging machine learning (ML) techniques. The adjoint method excels in efficiently handling large-scale problems by computing gradients. Meanwhile, ML approaches, such as neural network surrogates, offer flexibility by learning complex mappings from design parameters to performance metrics, potentially bypassing traditional sensitivity analysis. Looking ahead, promising directions include hybrid adjoint-

doi:10.6342/NTU202501972

ML frameworks [32], multi-physics optimization, and adaptive algorithms that leverage real-time data for dynamic antenna designs.

doi:10.6342/NTU202501972

doi:10.6342/NTU202501972

Chapter 6 Conclusion

6.1 Summary

The antenna design problem is formulated as an inverse problem, highlighting the need for generalizable and automated design methodologies. Well-trained offline models can provide real-time solutions, while online surrogate models offer greater flexibility and adaptability across varying tasks. Adjoint method presents a clever and elegant approach to expanding both the design space and its complexity. By and large, MLAO framework remains widely used due to its robustness, simplicity, and foundation in traditional optimization principles. As for addressing more complex antenna design challenges, it is suggested to consider hybrid frameworks that combine the physics-based strengths of adjoint method with the prediction and automation capabilities of machine learning.

There are still many more complicated scenarios, diverse goals, and promising framework that cannot be fully included, the following section illustrates some methods that I believe to possess great potential and are worth more attention.

doi:10.6342/NTU202501972

6.2 Future Directions

6.2.1 Generative Model

A generative model is a type of machine learning model that learns to generate new data similar to the training dataset by capturing its underlying distribution. Examples include Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and diffusion models.

Generative Adversarial Networks (GANs) are a distinct class of generative models. They operate through the simultaneous training of two competing neural networks: a generator and a discriminator. Within antenna design, GANs have seen application, such as in the creation of a dual-resonance antenna in 2022 [35]. Despite their capacity to yield innovative and unconventional antenna geometries, GANs are well-known for their training complexities, often encountering issues like mode collapse or vanishing gradients. Techniques like Wasserstein GANs [36] or progressive growing GANs [37] could mitigate training instability, making GANs possibly more reliable for antenna design.

Researchers also used VAE to optimize antenna decoupling structure [38]. VAE is known to reduce the design space into a lower-dimensional latent space by training an encoder. The idea of latent space might be the key to more complex antenna structure designs. Incidentally, while the latent space training isn't as robust, we can still reduce the full design space to a parametric design space defined by human-engineered variables. For instance, Koziel et al. reduced the design space by using specific geometry generation strategy involving re-sizable elliptical patches [39].

Returning to generative models, the challenge to latent space is its quality, a poorly

doi:10.6342/NTU202501972

learned latent space may produce infeasible designs, requiring careful model training and validation. As a consequence, Pang et al. suggested to use diffusion models and verified the proposed approach by designing three microstrip antennas with different band variations [40].

Although the prospects of generative model are promising, there are still a long way to go because of its instability.

6.2.2 Reinforcement Learning

AlphaZero, developed by DeepMind, stunned the world by mastering complex games like Go, achieving superhuman performance through self-play and trial-and-error optimization. The magic behind is reinforcement learning (RL). These breakthroughs highlight RL's power to tackle intricate problems.

RL is a machine learning paradigm in which an agent develops decision-making abilities through interaction with an environment. The agent executes actions, gets feedback as rewards or penalties, and then modifies its approach to maximize its total accumulated rewards over time. Distinct from supervised learning, RL operates without the need for labeled datasets, relying instead on a reward function to guide its trial-and-error discovery.

In antenna design, the environment, in the sense of RL could mimic electromagnetic behavior, and the reward function could reflect how well the antenna meets design goals. RL's ability to explore vast design spaces and handle nonlinear, high-dimensional problems makes it ideal for automated design and complex optimization.

RL was used in antenna design recently, with domain-knowledge-informed RL optimizing antenna layouts [41] and localized RL enhancing pixelated patch bandwidths [42],

doi:10.6342/NTU202501972

alongside deep RL improving patch antenna performance for specific bands [43].

6.2.3 Physics-Informed Neural Networks

Physics-Informed Neural Networks (PINNs) are a class of neural networks that integrate physical laws. Most ML schemes, serving as a stochastic model, struggle with the high-dimensional, nonlinear nature of PDEs or require extensive computational resources. Unlike traditional data-driven neural networks, PINNs embed governing equations directly into their loss function, enabling them to respect physical constraints.

Purely data-driven ML models can act as "black boxes", offering little insight into why predictions are made. PINNs, by contrast, produce physically consistent results, increasing trust and interpretability for critical applications.

PINNs have been applied to heat transfer problems [44] and fluid mechanics [45], demonstrating their versatility since 2021. Krishnapriyan et al. provided a comprehensive analysis of PINNs' limitations [46]. In 2024, novel variations of PINNs were proposed, using eigenfunctions to represent manifolds [47]. Although the application of PINNs to electromagnetic theory is still emerging, studies like Liu et al. have explored their use in the inverse design of horn antennas [48], indicating significant potential for antenna design advancements.

6.2.4 Machine Learning with Hard Constraints

PINNs are designed to incorporate physical constraints, such as those derived from governing differential equations. However, as highlighted by Krishnapriyan et al. [46], even PINNs struggle to achieve a perfect balance between satisfying constraints and en-

doi:10.6342/NTU202501972

suring convergence. The inherent flexibility of neural networks, while advantageous for approximating complex functions, can lead to solutions that deviate from strict physical requirements, particularly when constraints are non-negotiable or "hard" in nature.

Hard constraints, which demand exact satisfaction of conditions like boundary conditions, design limitations, or conservation laws, cannot always be guaranteed. Post-processing methods become essential to refine outputs and ensure compliance with these constraints. For instance, techniques such as projection onto feasible solution spaces can be employed to adjust predictions, ensuring they align with the physical requirements of the problem at hand. However, how to project the output in high-dimensional space becomes unexpectedly difficult.

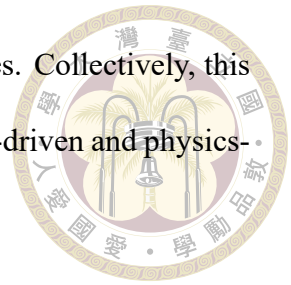
One promising approach for handling hard constraints is the use of homeomorphism [49]. This method leverages topological transformations to preserve the structure of the solution while enforcing exact compliance with physical constraints, offering a robust way to correct deviations in ML outputs. In the context of antenna design, homeomorphism could be used to ensure that the predicted antenna geometry, for example, strictly stays in the design region.

6.3 Contributions

In this thesis, we propose a novel framework, distinct from traditional forward and inverse surrogate models, to automate the real-time fine-tuning of a PIFA antenna. The efficacy of online machine learning is validated through our MLAO-GA demonstration, supported by related studies. By integrating the Adam optimization algorithm with the adjoint method, we demonstrate that contemporary machine learning optimization tech-

doi:10.6342/NTU202501972

niques effectively complement well-established mathematical theories. Collectively, this thesis establishes a robust foundation for future advancements in data-driven and physics-based antenna design paradigms.



doi:10.6342/NTU202501972

References

[1] T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez, "Millimeter wave mobile communications for 5g cellular: It will work!," *IEEE Access*, vol. 1, pp. 335–349, 2013.

[2] S. I. H. Shah, S. Bashir, M. Ashfaq, A. Altaf, and H. Rmili, "Lightweight and low-cost deployable origami antennas—a review," *IEEE Access*, vol. 9, pp. 86429–86448, 2021.

[3] S. O. Slawomir Koziel, *Antenna Design by Simulation-Driven Optimization*. Springer Cham, 2014.

[4] N. Sarker, P. Podder, M. R. H. Mondal, S. S. Shafin, and J. Kamruzzaman, "Applications of machine learning and deep learning in antenna design, optimization, and selection: A review," *IEEE Access*, vol. 11, pp. 103890–103915, 2023.

[5] L.-Y. Xiao, W. Shao, F.-L. Jin, B.-Z. Wang, and Q. H. Liu, "Inverse artificial neural network for multiobjective antenna design," *IEEE Transactions on Antennas and Propagation*, vol. 69, no. 10, pp. 6651–6659, 2021.

[6] M. Kim, J. Gu, Y. Yuan, T. Yun, Z. Liu, Y. Bengio, and C. Chen, "Offline model-based optimization: Comprehensive review," 2025.

doi:10.6342/NTU202501972

[7] Y. He, J. Huang, W. Li, L. Zhang, S.-W. Wong, and Z. N. Chen, "Hybrid method of artificial neural network and simulated annealing algorithm for optimizing wideband patch antennas," *IEEE Transactions on Antennas and Propagation*, vol. 72, no. 1, pp. 944–949, 2024.

[8] Y. Zhou, X. Zhang, and H. Li, "Efficient reflection coefficient prediction and optimization for printed inverted f antenna based on graph neural networks patch antenna design and optimisation by gnn," in *2023 7th International Conference on Electrical, Mechanical and Computer Engineering (ICEMCE)*, pp. 64–67, 2023.

[9] Y. Su, Z. Su, H. Chen, H. Zhao, and X. Yin, "Inverse design of rectangular microstrip patch antenna using neural network combining with time-domain representation of s-parameters," in *2022 16th European Conference on Antennas and Propagation (EuCAP)*, pp. 1–4, 2022.

[10] K. A. Wang, N. S. Chatterji, S. Haque, and T. Hashimoto, "Is importance weighting incompatible with interpolating classifiers?," in *International Conference on Learning Representations*, 2022.

[11] J. P. Jacobs, "Accurate modeling by convolutional neural-network regression of resonant frequencies of dual-band pixelated microstrip antenna," *IEEE Antennas and Wireless Propagation Letters*, vol. 20, no. 12, pp. 2417–2421, 2021.

[12] W. Chen, Q. Wu, B. Han, C. Yu, H. Wang, and W. Hong, "Efficient incremental variable-fidelity machine-learning-assisted hybrid optimization and its application to multiobjective antenna design," *IEEE Transactions on Antennas and Propagation*, vol. 72, no. 12, pp. 9347–9354, 2024.

doi:10.6342/NTU202501972

[13] B. Duan, B. Wu, J.-h. Chen, H. Chen, and D.-Q. Yang, "Deep Learning for Photonic Design and Analysis: Principles and Applications," *Frontiers in Materials*, vol. 8, Jan. 2022. Publisher: Frontiers.

[14] K. Min, B. Choi, K. Park, and E. Cho, "Machine learning assisted optimization of electrochemical properties for Ni-rich cathode materials," *Scientific Reports*, vol. 8, p. 15778, Oct. 2018. Publisher: Nature Publishing Group.

[15] Q. Wu, Y. Cao, H. Wang, and W. Hong, "Machine-learning-assisted optimization and its application to antenna designs: Opportunities and challenges," *China Communications*, vol. 17, no. 4, pp. 152–164, 2020.

[16] B. Liu, H. Aliakbarian, Z. Ma, G. A. E. Vandenbosch, G. Gielen, and P. Excell, "An efficient method for antenna design optimization based on evolutionary computation and machine learning techniques," *IEEE Transactions on Antennas and Propagation*, vol. 62, no. 1, pp. 7–18, 2014.

[17] M. O. Akinsolu, B. Liu, V. Grout, P. I. Lazaridis, M. E. Mognaschi, and P. D. Barba, "A parallel surrogate model assisted evolutionary algorithm for electromagnetic design optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 3, no. 2, pp. 93–105, 2019.

[18] L. Cui, Y. Zhang, R. Zhang, and Q. H. Liu, "A modified efficient knn method for antenna optimization and design," *IEEE Transactions on Antennas and Propagation*, vol. 68, no. 10, pp. 6858–6866, 2020.

[19] Q. Wu, W. Chen, C. Yu, H. Wang, and W. Hong, "Machine-learning-assisted optimization for antenna geometry design," *IEEE Transactions on Antennas and Propagation*, vol. 72, no. 3, pp. 2083–2095, 2024.

doi:10.6342/NTU202501972

[20] B. Han, Q. Wu, C. Yu, H. Wang, and W. Hong, "Low-wind-load broadband dual-polarized antenna and array designs using sequential multiphysics machine-learning-assisted optimization," *IEEE Transactions on Antennas and Propagation*, vol. 73, no. 1, pp. 135–148, 2025.

[21] J. Moreno, I. Gonzalez, and D. Rodriguez, "Using simulation and the nsga-ii evolutionary multi-objective algorithm in the design of a compact dual-band equatorial helix antenna," in *2017 6th International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, pp. 56–60, 2017.

[22] D. Dyck, D. Lowther, and E. Freeman, "A method of computing the sensitivity of electromagnetic quantities to changes in materials and sources," *IEEE Transactions on Magnetics*, vol. 30, no. 5, pp. 3415–3418, 1994.

[23] D. Dyck and D. Lowther, "Automated design of magnetic devices by optimizing material distribution," *IEEE Transactions on Magnetics*, vol. 32, no. 3, pp. 1188–1193, 1996.

[24] T. Nomura, K. Sato, K. Taguchi, T. Kashiwa, and S. Nishiwaki, "Structural topology optimization for the design of broadband dielectric resonator antennas using the finite difference time domain technique," *International Journal for Numerical Methods in Engineering*, vol. 71, no. 11, pp. 1261–1296, 2007.

[25] S. Zhou, W. Li, and Q. Li, "Level-set based topology optimization for electromagnetic dipole antenna design," *Journal of Computational Physics*, vol. 229, no. 19, pp. 6915–6930, 2010.

[26] A. Erentok and O. Sigmund, "Topology optimization of sub-wavelength antennas," *IEEE Transactions on Antennas and Propagation*, vol. 59, no. 1, pp. 58–69, 2011.

doi:10.6342/NTU202501972

[27] E. Hassan, E. Wadbro, and M. Berggren, "Topology optimization of metallic antennas," *IEEE Transactions on Antennas and Propagation*, vol. 62, no. 5, pp. 2488–2500, 2014.

[28] E. Hassan, D. Noreland, R. Augustine, E. Wadbro, and M. Berggren, "Topology optimization of planar antennas for wideband near-field coupling," *IEEE Transactions on Antennas and Propagation*, vol. 63, no. 9, pp. 4208–4213, 2015.

[29] S. Liu, Q. Wang, and R. Gao, "Mom-based topology optimization method for planar metallic antenna design," *Acta Mechanica Sinica*, vol. 32, pp. 1058–1064, Dec. 2016.

[30] J. Wang, X.-S. Yang, X. Ding, and B.-Z. Wang, "Antenna radiation characteristics optimization by a hybrid topological method," *IEEE Transactions on Antennas and Propagation*, vol. 65, no. 6, pp. 2843–2854, 2017.

[31] J. Wang, X.-S. Yang, X. Ding, and B.-Z. Wang, "Topology optimization of conical-beam antennas exploiting rotational symmetry," *IEEE Transactions on Antennas and Propagation*, vol. 66, no. 5, pp. 2254–2261, 2018.

[32] L.-L. Wang, X.-S. Yang, and J. Wang, "An efficient topology optimization strategy based on the combination of ga and simp," in *2022 IEEE 10th Asia-Pacific Conference on Antennas and Propagation (APCAP)*, pp. 1–2, 2022.

[33] L.-L. Wang, X.-S. Yang, and C.-J. Ma, "An efficient gradient-based hybrid parameter-topology optimization for antenna design," *IEEE Transactions on Antennas and Propagation*, vol. 71, no. 12, pp. 9477–9486, 2023.

[34] H. S. Bhat, "Second-order adjoint method for quantum optimal control," 2025.

doi:10.6342/NTU202501972

[35] Y. Zhong, P. Renner, W. Dou, G. Ye, J. Zhu, and Q. H. Liu, "A machine learning generative method for automating antenna design and optimization," *IEEE Journal on Multiscale and Multiphysics Computational Techniques*, vol. 7, pp. 285–295, 2022.

[36] L. Weng, "From gan to wgan," 2019.

[37] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," 2018.

[38] H. Huang and X.-S. Yang, "An efficient antenna decoupling optimization method using variational auto-encoders," in *2024 IEEE International Symposium on Antennas and Propagation and INC/USNC-URSI Radio Science Meeting (AP-S/INC-USNC-URSI)*, pp. 2007–2008, 2024.

[39] S. Koziel, A. Pietrenko-Dabrowska, and S. Szczepanski, "Versatile unsupervised design of antennas using flexible parameterization and computational intelligence methods," *Scientific Reports*, vol. 14, p. 29753, Nov. 2024. Publisher: Nature Publishing Group.

[40] Q. Pang, J. Ouyang, F. Yang, S. Yang, and J. Hu, "Inverse design method of antenna based on generative artificial intelligence," *IEEE Antennas and Wireless Propagation Letters*, pp. 1–5, 2025.

[41] Z. Wei, Z. Zhou, P. Wang, J. Ren, Y. Yin, G. F. Pedersen, and M. Shen, "Automated antenna design via domain knowledge-informed reinforcement learning and imitation learning," *IEEE Transactions on Antennas and Propagation*, vol. 71, no. 7, pp. 5549–5557, 2023.

[42] Q. Wang, Z. Pang, D. Gao, P. Liu, X. Zhang, X. Pang, and X. Yin, "Bandwidth enhancement of pixelated patch antennas based on localized reinforcement learning,"

doi:10.6342/NTU202501972

in *2024 14th International Symposium on Antennas, Propagation and EM Theory (ISAPE)*, pp. 1–4, 2024.

[43] Y. Su, Y. Yin, S. Li, H. Zhao, and X. Yin, "Bandwidth improvement for patch antenna via knowledge-based deep reinforcement learning," *IEEE Antennas and Wireless Propagation Letters*, vol. 23, no. 12, pp. 4094–4098, 2024.

[44] S. Cai, Z. Wang, S. Wang, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks for heat transfer problems," *Journal of Heat Transfer*, vol. 143, Apr. 2021.

[45] S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis, "Physics-informed neural networks (PINNs) for fluid mechanics: a review," *Acta Mechanica Sinica*, vol. 37, pp. 1727–1738, Dec. 2021.

[46] A. S. Krishnapriyan, A. Gholami, S. Zhe, R. M. Kirby, and M. W. Mahoney, "Characterizing possible failure modes in physics-informed neural networks," 2021.

[47] F. Sahli Costabal, S. Pezzuto, and P. Perdikaris, "Δ -pinns: Physics-informed neural networks on complex geometries," *Engineering Applications of Artificial Intelligence*, vol. 127, p. 107324, Jan. 2024.

[48] J.-P. Liu, B.-Z. Wang, C.-S. Chen, and R. Wang, "Inverse design method for horn antennas based on knowledge-embedded physics-informed neural networks," *IEEE Antennas and Wireless Propagation Letters*, vol. 23, no. 6, pp. 1665–1669, 2024.

[49] K. Deng, H. Zhang, J. Lu, and H. Sun, "Hop: Homeomorphic polar learning for hard constrained optimization," 2025.

doi:10.6342/NTU202501972

doi:10.6342/NTU202501972

Appendix A — Recurrent Neural Network with Gated Recurrent Unit

The Recurrent Neural Network (RNN) with Gated Recurrent Unit (GRU) is a special type of machine learning model designed to remember and learn from sequences of information, much like how we might recall a story by connecting one event to the next. Imagine you are adjusting an antenna's design based on how its surroundings, like nearby objects or its own size, affect its performance. These effects might build on each other over time or space, creating a kind of "memory" or feedback loop. (We hypothesized that this memory could help predict antenna parameters more accurately, especially when the antenna's behavior depends on the coupling with its environment.)

An RNN works by processing data step by step, carrying forward what it learned from previous steps to influence its next prediction. However, standard RNNs can struggle to remember long sequences because they tend to forget earlier information. This is where the Gated Recurrent Unit (GRU) comes in: it's an improved version of the RNN that acts like a smart memory manager. The GRU has special "gates" (imagine doors that open or close) that decide what information to keep or forget. It has two main gates:

- Update Gate: This gate decides how much of the past information to carry forward,

doi:10.6342/NTU202501972

like keeping the most relevant details of the antenna's surroundings.

- Reset Gate: This gate decides what old information to discard, allowing the model to focus on new data, such as changes in the antenna's configuration.

doi:10.6342/NTU202501972

Appendix B — TabTransformer

The TabTransformer is a more advanced machine learning model that combines ideas from a technology called transformers, originally developed for understanding languages like English. Think of it as a super-smart assistant that not only looks at each piece of information but also figures out how important each piece is compared to the others, much like prioritizing key words in a sentence. The TabTransformer is specially built to handle two types of information: categorical data (like yes/no choices) and continuous data (like measurements on a sliding scale).

In traditional models, all input features are treated equally, but some tasks involve a mix of different types of data For example, one continuous number (the feed position) and ten binary choices (the PEC states, either 0 or 1). The TabTransformer is designed for this kind of mixed data. Here's how it works:

- Embedding Layer: First, it transforms the input features into a format it can understand better, turning the input features into a set of numerical patterns, like translating words into a language the model can process.
- Transformer Encoder: Next, it uses a mechanism called "attention" to focus on the most important relationships between these features.
- Output Prediction: Finally, it passes this learned information through a simple neu-

doi:10.6342/NTU202501972

ral network layer to predict the outputs.

doi:10.6342/NTU202501972

Appendix C — Adaptive Moment Estimation

Adam (Adaptive Moment Estimation) is an optimization algorithm, commonly used to minimize the loss function during the training process of ML. It combines ideas from earlier methods like momentum and RMSProp, offering efficient and adaptive learning rates for gradient-based optimization.

The momentum method, inspired by physical momentum, accelerates gradient descent by accumulating past gradients to smooth out updates, especially in noisy or steep loss landscapes. Instead of using only the current gradient, it maintains a “velocity” term that aggregates previous gradients, helping to escape shallow local minima and speed up convergence in relevant directions.

RMSProp (Root Mean Square Propagation) addresses the issue of diminishing or exploding learning rates by scaling the gradient using a moving average of squared gradients. This normalization ensures that updates are more stable, especially for non-stationary objectives, by adapting the step size for each parameter based on the magnitude of recent gradients.

Proposed by Kingma and Ba in 2014, Adam combines the strengths of momentum

doi:10.6342/NTU202501972

and RMSProp. It maintains two moving averages: one for the gradients (first moment, like momentum) and one for the squared gradients (second moment, like RMSProp). By bias-correcting these moments, Adam provides an adaptive learning rate that is both fast and stable, making it widely used for deep learning tasks due to its robustness across various problems.

The update rule of Adam is:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1)\nabla_\theta J(\theta_t)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2)(\nabla_\theta J(\theta_t))^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

- m_t: First moment (mean of gradients, momentum-like).
- v_t: Second moment (uncentered variance of gradients, RMSProp-like).
- β_1: Decay rate for first moment (typically 0.9).
- β_2: Decay rate for second moment (typically 0.999).
- $\hat{m}_t, \hat{v}_t$: Bias-corrected moments to account for initialization at $t = 0$.
- α: Learning rate (e.g., 0.001).
- ϵ: Small constant (e.g., 10^{-8}) for numerical stability.

The bias correction ($\hat{m}_t, \hat{v}_t$) ensures accurate moment estimates early in training when m_t and v_t are initialized to zero.

doi:10.6342/NTU202501972

Adam's ability to adaptively scale updates and handle non-stationary objectives has made it a default choice in machine learning, though its hyperparameters ($\alpha, \beta_1, \beta_2, \epsilon$) may require tuning for optimal performance.

doi:10.6342/NTU202501972