**Machine Learning Based Approach for The Synthesis of Frequency Selective Surface for Bio-Medical Applications**

# Chapter 1: Introduction

## Project Overview

Technology has advanced rapidly in recent years, transforming various fields and enabling new levels of connectivity, communication, and information exchange. As the demand for wireless communication systems and smart devices continues to grow, the efficient utilization of the electromagnetic spectrum becomes increasingly critical. Traditional frequency management methods often involve bulky and expensive components or complex circuits. However, Frequency Selective Surfaces (FSS) offers a lightweight and cost-effective approach to manipulate electromagnetic waves based on their frequencies, addressing these challenges effectively.

A Frequency selective surface (FSS) is a periodic settlement of metallic patches assisted by a dielectric substrate that controls the reflection and transmission characteristics of an incident EM wave. Therefore, it shows either bandpass or bandstop characteristics, depending on the shape, geometry, unit cell size, properties of the dielectric substrate, and inter-element spacing. Due to their bandpass or bandstop characteristics, they are used for various applications, such as the design of spatial filters, antenna reflectors, radomes, absorbers, artificial magnetic conductors, electromagnetic band-gap materials, etc.

According to Munk [1], an FSS can be characterized by its unit cell, which is the repeating pattern of elements that form the overall structure. The unit cell is defined by its dimensions, such as length, width, and height, as well as the geometric properties of the elements within it. These properties determine the resonant frequencies, transmission, and reflection characteristics of the FSS.

Our project focuses on designing and synthesizing a Frequency Selective Surface (FSS) with significant benefits for the biomedical field. Healthcare facilities rely on various electronic devices operating within the 2.4 GHz and 5 GHz frequency bands, including Wi-Fi networks, wireless medical devices, and communication systems. By integrating our FSS, which has a dual bandstop behavior at 2.5 GHz and 5 GHz, into the infrastructure or shielding design, we aim to effectively reduce electromagnetic interference (EMI) between these devices. This ensures reliable operation and minimizes disruptions in critical healthcare systems, enhancing overall performance and safety.

Thus, in this project, we have designed a new single layer compact frequency-selective surface comprising of four-corner, modified, interconnected, open square headed dumbbell-shaped (OSHD) structure to provide dual-bandstop behaviour. This provides stopband behaviour over 230 MHz and 300 MHz bandwidths with resonating frequencies of 2.4 GHz and 5.0 GHz, respectively.

Using this proposed structure, we conducted a parametric analysis to gain insights into the output curves of various FSS (Frequency Selective Surface) structures. To achieve this, we employed supervised machine learning algorithms including decision tree, random forest, and multiple output gradient boosting regression. Through a comparison of the predicted output curves and subsequent error analysis for each algorithm, we found that the decision tree algorithm is particularly well-suited for accurately predicting the output curves of diverse FSS structures. This conclusion is supported by its ability to achieve a minimum relative absolute error (RAE).

## Objectives

* + - The primary objective of our project is to identify a proficient and accurate machine learning algorithm for synthesizing Frequency Selective Surfaces (FSS) effectively.
    - To achieve this, our project is divided into two subparts: (i) Designing a distinctive FSS with bandstop characteristics operating at the 2.4 GHz and 5 GHz frequency bands, and (ii) Employing supervised machine learning algorithms to predict the output curve of any given FSS structure.
    - The initial step entails utilizing Ansys HFSS Software to design the unique FSS structure that operates at the 2.4 GHz and 5 GHz frequency bands, followed by its fabrication.
    - The next step involves constructing a dataset using the proposed FSS design to train various machine learning algorithms, enabling them to predict the output curve of any unknown FSS structure.
    - Lastly, a comparison of the output curves predicted by the machine learning models and subsequent error analysis for each algorithm will be conducted to determine the most suitable algorithm.

## Applications

Our project has the following applications:

* + - Electromagnetic interference mitigation: In healthcare facilities, there are numerous electronic devices and equipment that operate within the 2.4 GHz and 5 GHz frequency bands, such as Wi-Fi networks, wireless medical devices, and communication systems. By incorporating our proposed FSS into the infrastructure or shielding design, it can help mitigate electromagnetic interference (EMI) between different devices, ensuring reliable operation and minimizing disruptions in critical healthcare systems.
* Wireless communication and connectivity: Our proposed FSS can be applied to wireless applications where it is preferable to avoid signal transmission over two broad stopbands by using a straightforward, single-layer structure. Thus, it allows seamless and reliable wireless communication between healthcare professionals, medical devices, and patient monitoring systems.
* Radio frequency identification (RFID) applications: RFID technology finds various applications in healthcare, such as patient tracking, inventory management, and medication administration. An FSS operating at 2.4 GHz and 5 GHz can assist in optimizing RFID systems by improving signal transmission, reducing interference, and enhancing overall system performance.
* Non-invasive medical imaging: FSS can be used in non-invasive medical imaging techniques, such as microwave imaging or terahertz imaging. By incorporating our proposed FSS structures into the imaging system design, it is possible to manipulate and control the electromagnetic fields, leading to improved image resolution, enhanced tissue contrast, and more accurate diagnostic information.

# Chapter 2: Background of the Project

## Literature Survey

The article [2] presents a novel design of a Frequency Selective Surface (FSS) that aims to mitigate interference and enhance network security within buildings, specifically in the unlicensed 2.4-GHz ISM band. The primary objective of this FSS is to achieve maximum transparency at broadcast frequencies while providing a narrow band response. To achieve this, a new element geometry known as the "Four Legged Loaded" is introduced which ensures a stable frequency response for TE polarization, even when the incident wave's angle varies from 0 to 70 degrees.

In [3], the authors present a novel approach for synthesizing multiband Frequency Selective Surfaces (FSSs) using supervised machine learning with the decision tree algorithm. The FSS structure consists of an array of metallic patches printed on a dielectric substrate, designed for spatial filtering microwave applications. Through the application of supervised machine learning with the decision tree algorithm, two bioinspired FSS geometries are synthesized. The accuracy of the results obtained from the decision tree algorithm is further validated using the random forest algorithm. Numerical analysis is conducted using Ansoft Designer software, and prototypes are fabricated and measured.

The work presented in [4] focuses on the design of a single-layer bandpass Frequency Selective Surface (FSS) operating at 2.6 GHz. To address the challenge of grating lobe suppression and achieve miniaturization, a metasurface unit cell is employed.

In [5], a fractal Frequency Selective Surface (FSS) is employed to effectively suppress two lower band frequencies that are utilized for 5G communication bands.

In [6], a dual-passband Frequency Selective Surface (FSS) operating in Wi-Fi bands is introduced. It serves the purpose of isolating intensive care rooms in hospitals from irrelevant signals while supporting Wi-Fi functionalities, allowing doctors to monitor patients' health conditions effectively. The design process involves the combination of the equivalent circuit method (ECM) and the genetic algorithm (GA) curve-fitting. This approach aims to determine the initial dimension parameters of the FSS. The calculated results obtained from the ECM are then compared with the numerical results obtained through HFSS optimization.

In a recent study [7], an innovative approach was proposed to enhance the performance of the dual- band multilayer FSS. The new technique involves incorporating an intermediate coupling layer between two resonant layers, further improving selectivity and miniaturization capabilities. These advancements contribute to the development of highly efficient and compact FSS designs with superior filtering properties.

In [8], a compact dual-band bandstop Frequency Selective Surface (FSS) design is presented that effectively suppresses signals within both the 2.4 and 5 GHz bands. The FSS structure features a closed loop topology, with unit elements measuring 0.084 λ0 × 0.084 λ0 in size. Notably, this design offers compactness while being immune to polarization defects and angular variations of up to 45°.

In [9], a fractal design approach for Frequency Selective Surfaces (FSSs) is introduced. The FSSs are composed of teragon metallic patches on a single-layer fiberglass dielectric, allowing for compact and efficient dual-band band-stop spatial filters.

In [10], a frequency selective surface is introduced specifically designed for Wi-Fi applications. This FSS design serves as a single-layer filter and can be utilized either as a reject or pass band filter. It combines ring loops/slots as canonical elements and is optimized for the 2.4 and 5.2 GHz Wi-Fi frequency bands. The design exhibits a stable frequency response within these bands for incidence angles ranging from 0° to 45°.Additionally, the paper proposes active variants of the FSS, allowing for on-off switching capabilities at the 2.4 and 5.2 GHz Wi-Fi bands.

## Review:

The mentioned research papers have contributed to the field of Frequency Selective Surfaces (FSS) by presenting novel designs with applications in WiFi and communication. These designs incorporate different techniques to optimize performance and ensure polarization independence. Building upon this body of work, our project focuses on developing a unique FSS structure operating at the 2.4 GHz and 5 GHz bands, specifically tailored for biomedical applications. Given the prevalence of biomedical devices operating at these frequencies, our design holds significant relevance.

Moreover, we have conducted a comprehensive parametric analysis and integrated machine learning algorithms to predict the output curves of FSS structures with various dimensions. This approach allows for efficient synthesis of FSS designs for biomedical applications. By successfully implementing a machine learning approach, we bridge the gap between theoretical analysis and practical design, enabling more effective and streamlined FSS development for the biomedical field.

# Chapter 3: Details of the Project

## Frequency Selective Surfaces

Frequency Selective Surfaces (FSS) are specialized structures that exhibit frequency-dependent electromagnetic properties. They are designed to selectively control and manipulate the transmission, reflection, or absorption of electromagnetic waves based on their frequency or wavelength. FSS structures are typically composed of periodic arrangements of conductive elements or apertures on a dielectric substrate.

The behavior of FSS is determined by the size, shape, and spacing of the elements, as well as the electrical properties of the materials used. By carefully designing these parameters, FSS can exhibit specific electromagnetic characteristics, such as transmitting certain frequencies while blocking or attenuating others.

Our project focuses on the synthesis of a novel, compact frequency-selective surface (FSS) consisting of a single layer. This innovative FSS exhibits dual stopband characteristics, covering bandwidths of 230 MHz and 300 MHz, with resonating frequencies of 2.4 GHz and 5.0 GHz, respectively. Notably, the FSS is designed to maintain its performance stability for both TE and TM polarizations, making it polarization independent. The unit-cell size of the proposed FSS is 0.12 times the free-space wavelength at the lower resonating frequency (λ0). To accomplish this, we utilized the ANSYS HFSS software for design and analysis purposes.

## Frequency Selective Surface Design and Analysis

Fig. 1(a) illustrates the unit cell geometry of our proposed FSS, featuring a periodic metallic pattern on one side of the substrate. This pattern consists of four interconnected open square dumbbell (OSHD) structures. To design the FSS, we opted for a readily available and cost-effective FR4 substrate with dimensions of 15 mm × 15 mm × 1.6 mm, possessing a dielectric constant (εr) of 4.4 and a loss tangent of 0.02. Through simulation with appropriate boundary conditions, we observed that when incident waves interacted with the FSS structure, it demonstrated dual stopband behavior. Specifically, the FSS exhibited stopbands at 2.4 GHz and 5 GHz, with transmission coefficients of - 36 dB and -31 dB, respectively. The corresponding transmission (S21) and reflection (S11) coefficients for the operating bandwidth are depicted in Fig. 1(b).

1

* + 1. (b)

Fig. 1.(a) Configuration of the proposed FSS unit cell (L1= 5 mm, L2= 4, L3= 3.5 mm, L4= 1.75 mm, L5= 3.5 mm, W1= 0.5 mm, W2= 0.5 mm, T1= 0.25 mm, T2= 0.25 mm, D= 15 mm.) (b) Simulated S21, and S11 for the proposed FSS.

In terms of bandwidth, the dual-band stop FSS showcased a -10 dB bandwidth of 690 MHz (ranging from 2.00 GHz to 2.69 GHz) and 820 MHz (ranging from 4.55 GHz to 5.37 GHz). These bandwidths correspond to fractional bandwidths of 28.75% and 16.4%, respectively. Fig. 2 demonstrates how the modification of the square head dumbbells in the FSS structure results in an expanded stop band bandwidth. As a result, the overall operating bandwidth increases by 80 MHz and 320 MHz for the respective stopbands. These findings highlight the effectiveness of our proposed FSS design in achieving dual stopband behavior and enhancing the operating bandwidth through modifications in the FSS structure.

To explain the resonance mechanism, the surface current distribution at both the resonances are shown in Fig. 3. The concentration of maximum current distribution at the dumbbells significantly varies with the resonating frequency, providing validation for the design of our proposed unit cell structure at both resonating frequencies.

Fig. 2. Effect on transmission coefficient for the tapering of each corner of OSHD

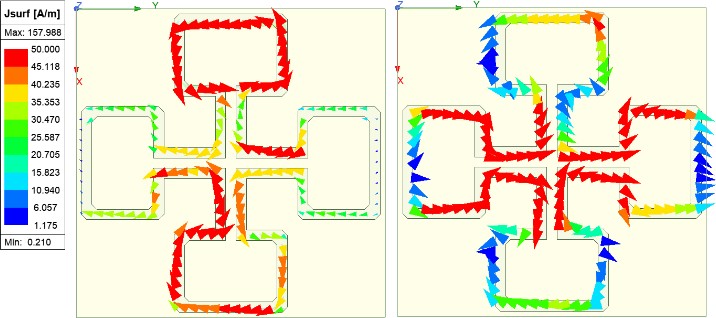
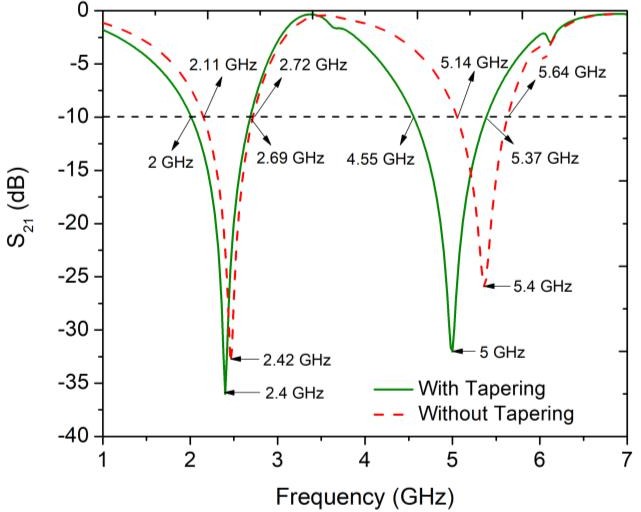
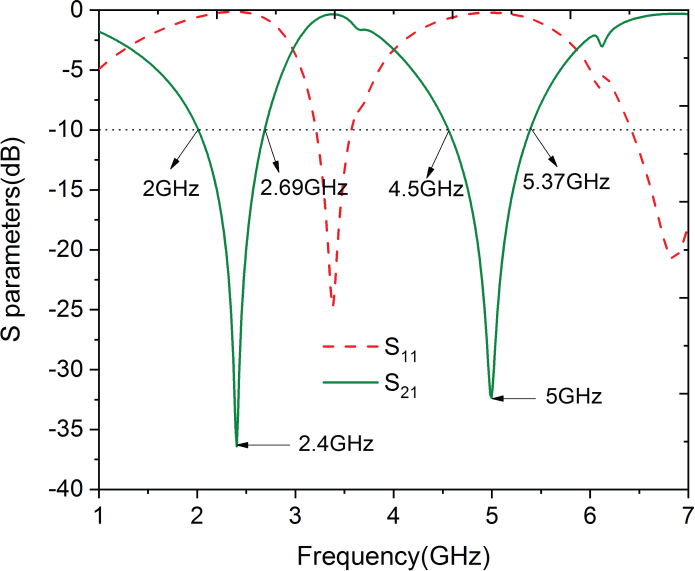
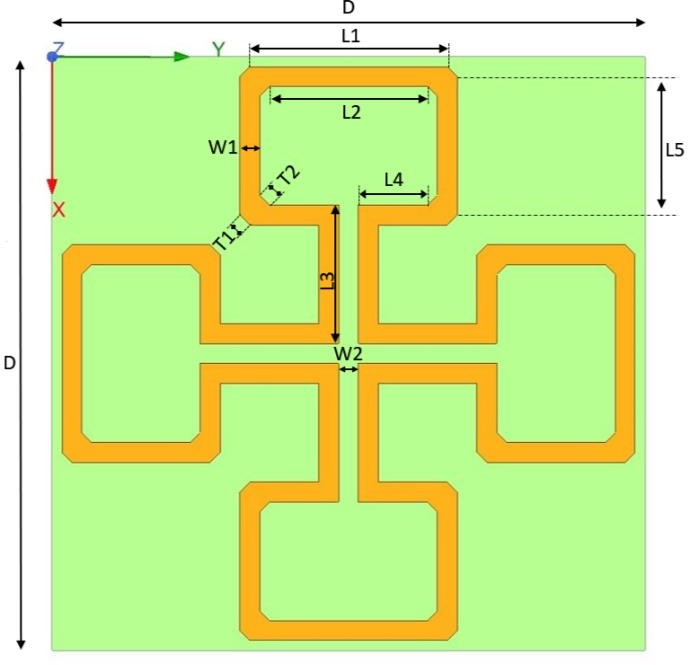
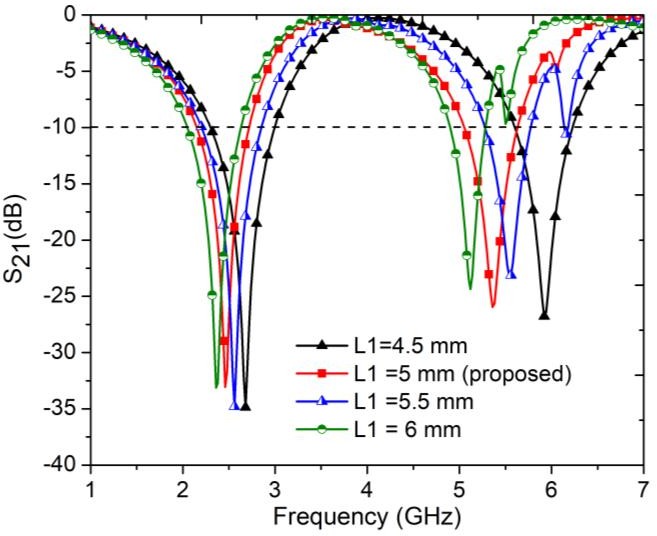
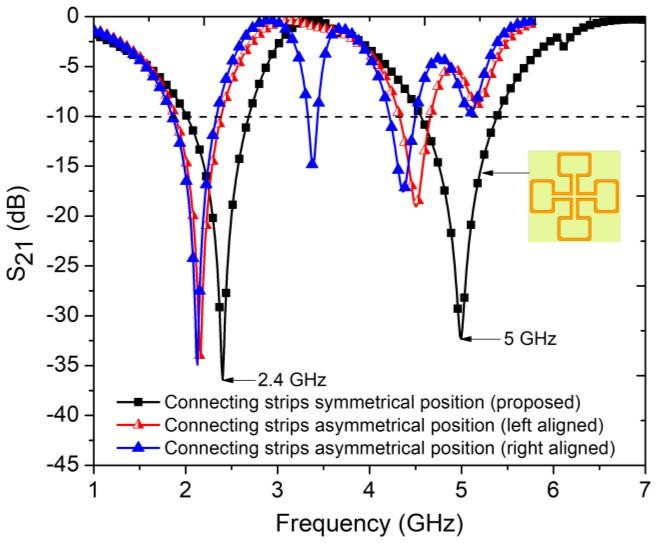
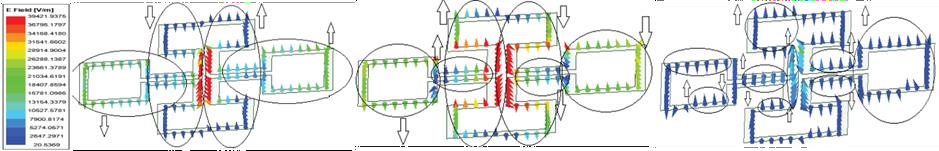


Fig. 3. Surface current distributions of the proposed FSS at (a) 2.4 GHz, and (b) 5 GHz



In order to demonstrate the bandstop characteristics of our proposed FSS, we examined the electric field distributions at two resonating frequencies and one additional frequency, as depicted in Fig.

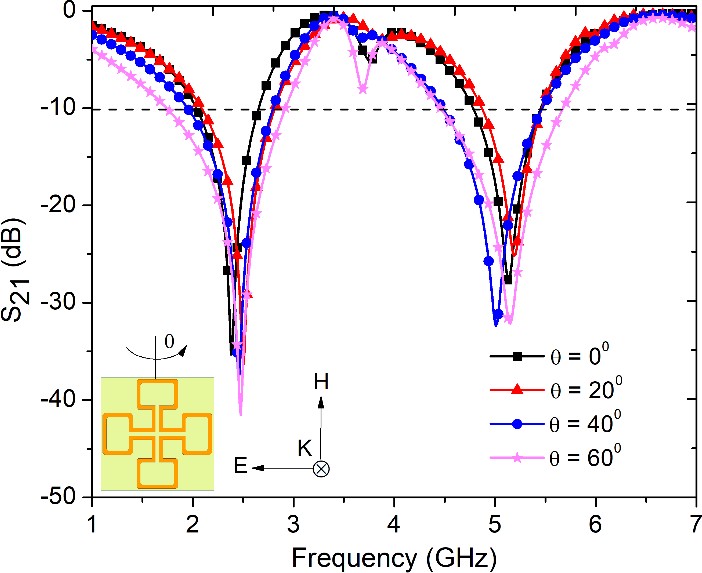
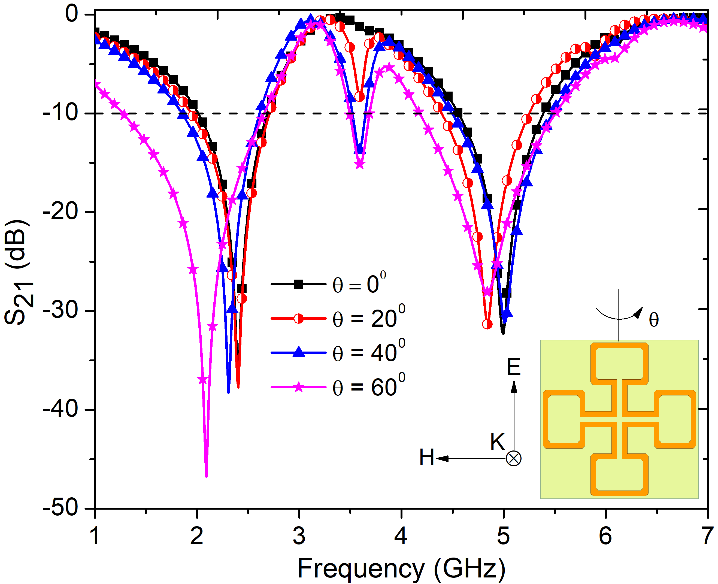
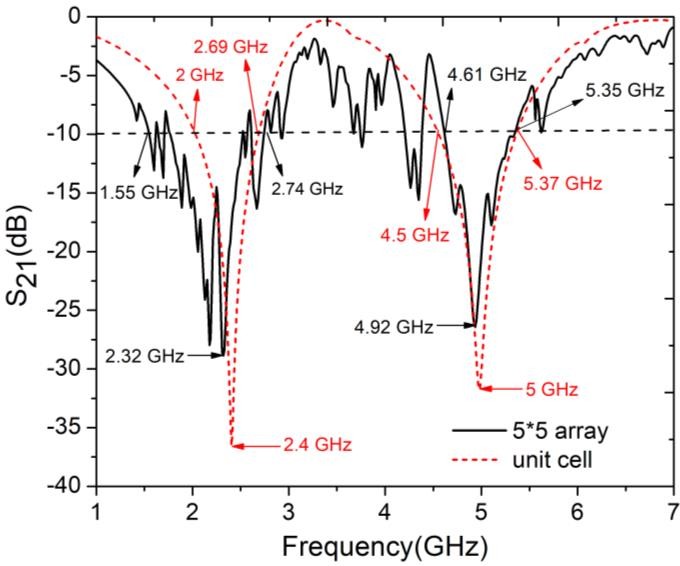
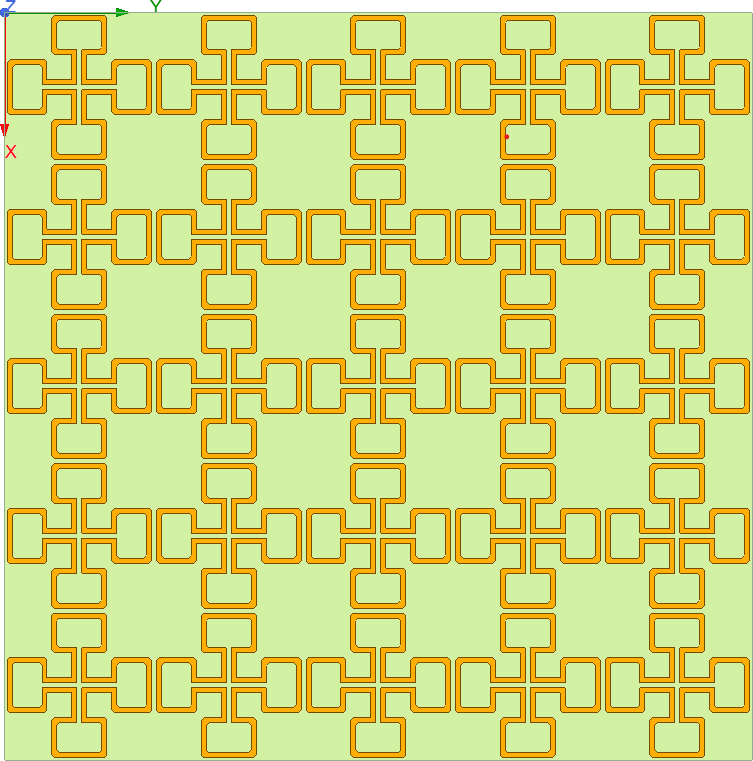
* + 1. The observations revealed that the FSS design is strategically devised to ensure that the electric fields in the opposite arms of the OSHD structure cancel each other at the resonating frequencies. In contrast, proper cancellation of the electric fields does not occur at non-resonating frequencies. Consequently, we achieved sharp and deep stopbands with high attenuation solely at the designated operating frequencies.

Fig. 4 Electric fields variation of the proposed FSS at different frequencies (a) 2.4 GHz, (b) 5 GHz and (c) 3.5GHz

Based on the results of the parametric analysis, the symmetric placement of the connection between the OSHDs in the proposed FSS structure is crucial. Fig. 5(a) illustrates that deviating from the central position of the connecting strip leads to a reduction in the overall bandwidth. This finding emphasizes the importance of maintaining the symmetric location for optimal performance. Furthermore, the parametric analysis conducted on the variation of the OSHD length, as shown in Fig. 5(b), revealed that the proposed FSS operates at the desired resonating frequencies only when the length of L1 is set to 5 mm. These results highlight the specific design requirements and the critical role played by the OSHD length in achieving the desired resonating frequencies.

* + - 1. (b)

Fig. 5. Effects on S21 curves for (a) different positions of interconnecting strip, and (b) different length of open square headed dumbbell



## Stability of the FSS

Fig. 6(a) and Fig. 6(b) depict the angular stability performance of the proposed FSS for TE and TM polarizations, respectively. Notably, both resonances exhibit stability up to an incidence angle of 600.

* + 1. (b)

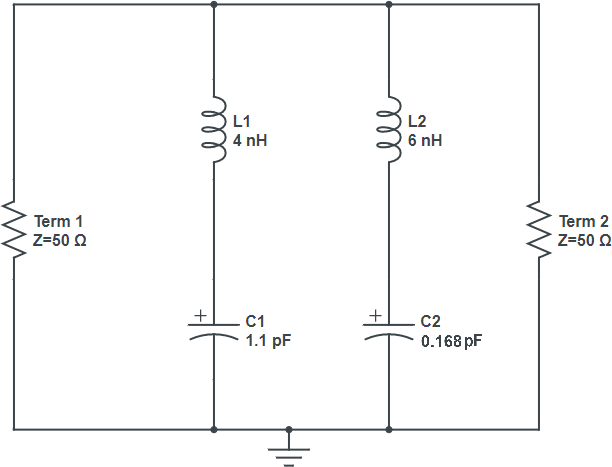
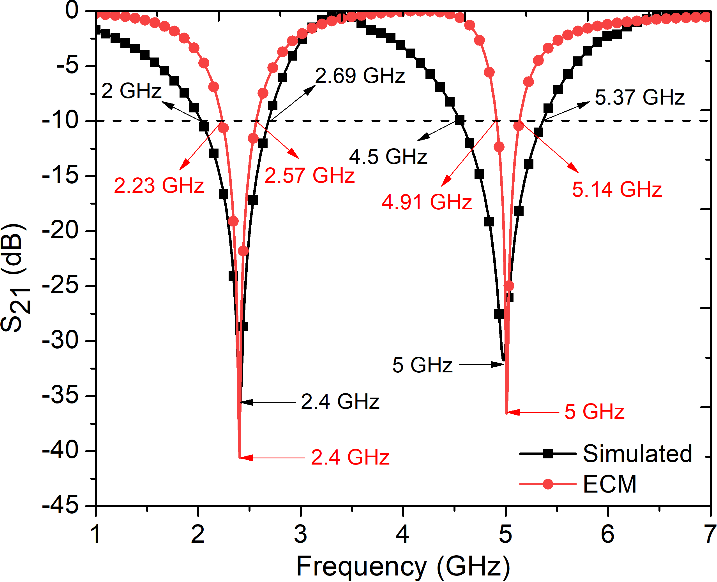
Fig. 6. Simulated S21 curves for different incidence angles(θ) (a) under TE polarization, and (b) under TM polarization

## FSS Array

To evaluate the performance of the proposed FSS, we created an FSS array consisting of a 5 x 5 grid of unit cells, that is, a total of 25 unit cells, as depicted in Fig. 7(a). Subsequently, we conducted simulations to analyze the behavior of the array. Remarkably, the results obtained from the array were found to be consistent with the output of a single unit cell, as illustrated in Fig. 7(b).

* + 1. (b)

Fig. 7. (a) 5x5 array of unit cells (b) Comparison between the simulated S21 curve of a single unit cell and that of a 5x5 array



This comparison between the FSS array and the individual unit cell output demonstrates the effectiveness and coherence of the array's behavior. It affirms that the collective response of the array aligns with the expected behavior of a single unit cell, validating the design and functionality of the FSS array.

## Equivalent Circuit Model (ECM) of the FSS

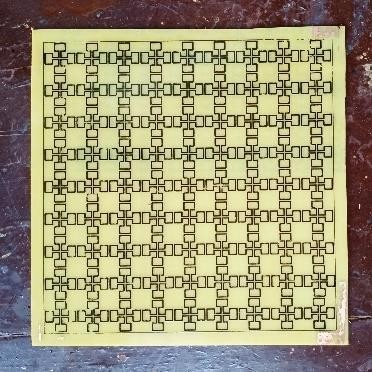
The equivalent circuit modeling of Frequency Selective Surfaces (FSSs) plays a crucial role in the analysis and design of these periodic structures for various applications. By accurately representing the behavior of FSSs in terms of electrical components, like resistors, capacitors, and inductors, the equivalent circuit models enable efficient analysis and prediction of the FSS response.

The first step in the equivalent circuit design procedure involves obtaining the equivalent circuit model of the FSS. The circuit parameters, such as inductance and capacitance, are derived based on the geometric parameters of the structure and the electromagnetic properties of the materials. Fig. 8(a) shows the equivalent circuit of our FSS design. Here, L1 = 4 nH and L2 = 6 nH are the effective inductances due to lengths of the OSHD structure. The effective capacitances resulting from series and parallel connections are denoted by C1 = 1.1 pF and C2 = 0.168 pF, respectively. It's important to note that the dielectric substrate used in constructing the FSS primarily affects the capacitances. The equivalent circuit model of the FSS is simulated using Advanced Design software (ADS), and the results are compared with the simulated response. Fig. 8(b) shows the comparison curve between the S21 response obtained from the ECM and that of the simulated response.

* + 1. (b)

Fig. 8. (a) Equivalent Circuit of the FSS model, (b) Comparison between the S21 response obtained from the ECM and that of the simulated response.

The similarity observed in the frequency response graphs of ECM and simulation validates the accuracy of the proposed model. The key features, such as the dual stopbands are faithfully reproduced by both the ECM and the simulation. This consistency suggests that the derived circuit parameters, including inductance and capacitance, adequately describe the electromagnetic behavior of the FSS.



## Fabrication of the FSS Prototype

Following the completion of the analysis using the HFSS software, we proceeded with the fabrication of the designed Frequency Selective Surface (FSS). To realize the FSS array, we meticulously executed the following steps:

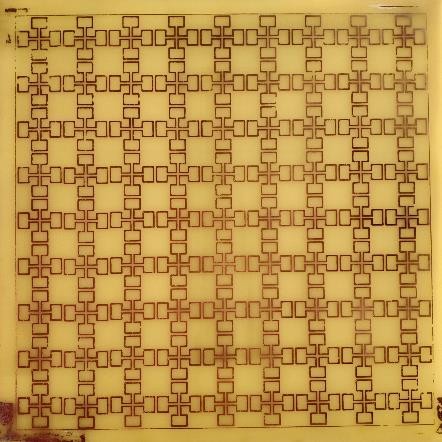
1. Substrate Preparation: FR4 substrate was chosen and prepared for the fabrication process. Due to the small size of the individual units, and its ability to block two specific frequencies, precise substrate dimensions are crucial to ensure the overall functionality and performance of the FSS array. The substrate dimensions are determined in such a way that they accommodate the required number of units while maintaining the desired spacing and arrangement.
2. Design Transfer: To replicate the FSS design accurately, we employed a method wherein the design was printed on photo-paper (Fig. 9a) and subsequently transferred onto the metal side of the prepared substrate (Fig. 9b) through a controlled heating process (Fig. 9c). This precise transfer technique ensured the faithful reproduction of the unit cell geometry patterns onto the substrate surface (Fig. 9d).

(a) (b) (c) (d)

Fig. 9. (a) The array pattern is printed on photo-paper (b) The paper is attached to the metal side of the substrate for ink transfer (c) the ink is transferred through a controlled heating process (d) reproduction of the unit cell geometry patterns onto the substrate surface

1. Etching and Removal of Excess Material: To remove the excess metal that was not part of the inked portions, the prepared substrate block was immersed in a bath of ferric chloride and hot water (Fig. 10a). This etching process selectively dissolved the un-inked metal, leaving behind the desired patterns and unit cell geometry on the substrate surface (Fig. 10b).
   1. (b)

Fig. 10. (a) Preparing a bath of ferric chloride and hot water (b) the desired patterns on the substrate surface after etching



1. Ink Removal: Following the etching process, toluene was applied to dissolve and remove the inked portions from the substrate surface, leaving behind only the clean metal structures that constitute the fabricated FSS (Fig. 11). This step ensured the elimination of any residual ink and further refined the FSS design, resulting in a pristine and precisely defined structure.

Fig. 11. Final Fabricated Prototype

By meticulously following these steps, we successfully fabricated a 9 x 9 FSS array, bringing the designed structure to physical realization.

## Introduction to Machine Learning

Machine learning is a field of study and practice that empowers computers to learn from data and improve their performance on specific tasks without explicit programming. It is a subset of artificial intelligence (AI) that focuses on developing algorithms and models capable of automatically extracting meaningful insights and making accurate predictions or decisions.

At the core of machine learning is the concept of training. During the training phase, an algorithm is exposed to a large dataset, often referred to as the training dataset, which consists of input data along with corresponding desired outputs or labels. The algorithm learns by iteratively adjusting its internal parameters to minimize the discrepancy between the predicted outputs and the actual labels. This process, known as model optimization or parameter estimation, allows the algorithm to generalize from the training data and make accurate predictions on new, unseen data.

In recent times, the application of machine learning techniques has gained significant attention from researchers globally. The appeal lies in its ability to learn from data, recognize patterns, classify outputs, and make decisions with minimal human intervention. This has led to widespread interest and exploration of machine learning's potential across various domains.

For our project, we have utilized various supervised machine learning algorithms, including Decision Tree, Random Forest, Linear Regression, and Multiple Gradient Boosting Regression. By utilizing these supervised machine learning algorithms and conducting a comprehensive analysis, we aimed to determine the most effective and reliable model for predicting the output curves of various FSS structures.

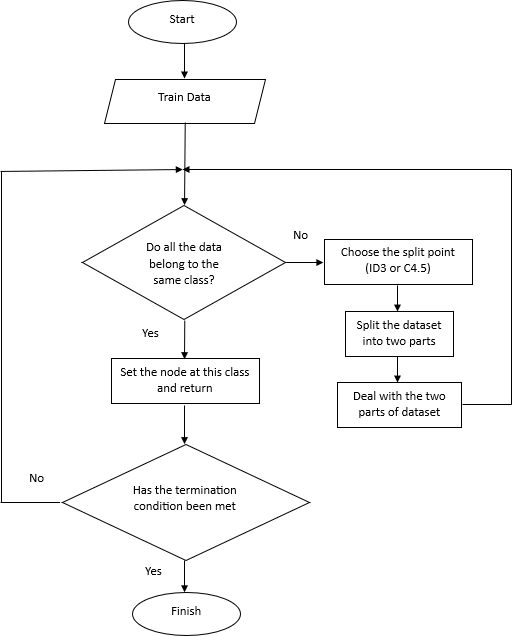
#### DECISION TREE ALGORITHM

The decision tree algorithm is a popular machine learning algorithm used for both classification and regression tasks. It is a supervised learning method that builds a tree-like model to make predictions based on a set of input features.

The simplicity and interpretability of decision trees make them a popular choice for many machine learning applications. They are especially useful when the relationships between features and outcomes are nonlinear or complex, as they can capture interactions between variables that may be missed by other models. Additionally, decision trees can be easily visualized and interpreted, making them a valuable tool for explaining the logic behind a particular prediction or decision.

The algorithm has the following steps:

* + - 1. Select the Best Feature: The algorithm starts with the entire dataset and evaluates different features based on a splitting criterion (e.g., Gini impurity, information gain). The feature that provides the most significant split or reduction in impurity is chosen as the root node of the tree.
      2. Repeat for Each Child Node: The algorithm recursively applies steps 1 and 2 for each child node, using the subset of data associated with that node. This process continues until a stopping criterion is met, such as reaching a maximum depth, achieving a minimum number of samples in a node, or other predefined conditions.
      3. Stopping Conditions: The algorithm checks if the stopping criterion is met at each node. If the criterion is satisfied, a leaf node is created, representing the predicted class or value for the corresponding subset of data.
      4. Calculate Node Impurity: At each node, the algorithm calculates the impurity or uncertainty of the data using a suitable measure (e.g., Gini impurity, entropy). This measure helps determine the quality of the split and the homogeneity of the subsets.
      5. Repeat for Each Child Node: The recursive process is repeated for each child node, selecting the best feature and splitting the data based on the splitting criterion. This step continues until the stopping criterion is met for each branch.
      6. Prediction: To make predictions on new or unseen data, the algorithm traverses the decision tree by evaluating the feature values at each node and following the appropriate branch. The prediction is determined based on the class or value associated with the reached leaf node.



#### Flowchart

Fig. 12 depicts the flowchart of the decision tree algorithm.

Fig. 12. Flowchart of Decision Tree algorithm

In our implementation, we utilized the Python programming language to develop the decision tree model. As the first step, we prepared a comprehensive dataset that included variable parameters such as Length (L1), Width (W1), and Breadth (L5). This dataset was utilized to train our decision tree model, enabling it to learn and make predictions based on the given input parameters.

To obtain predictions from the trained model, user inputs are required, which specify the values for the Length (L1), Width (W1), and Breadth (L5) parameters. These inputs serve as the basis for predicting the corresponding output according to the specified parameters.

By incorporating these user inputs into the trained decision tree model, we can obtain accurate predictions that align with the provided parameter values. This approach allows us to leverage the power of machine learning to make informed decisions and gain valuable insights from the data.

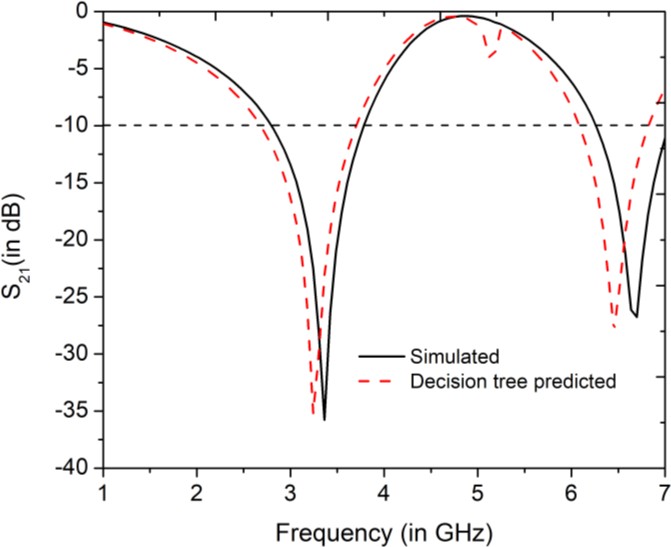
#### Code Snippet for Decision Tree

import pandas as pd

from sklearn.tree import DecisionTreeRegressor

# Load dataset

data = pd.read\_csv("D:\VS code\FSS\_project\FSS\Dataset\_ML.csv") # Split into features and labels



X = data[['Length', 'Width', 'Breadth']]

y = data.iloc[:, 3:] # select all columns starting from the 4th column

# Train decision tree model clf = DecisionTreeRegressor() clf.fit(X, y)

length = float(input("Enter the length: ")) width = float(input("Enter the width: ")) breadth = float(input("Enter the Breadth: "))

# Predict S21 values for a new input new\_input = [[length, width, breadth]] prediction = clf.predict(new\_input) print(prediction)

# Create a new DataFrame with the predicted S21 values s21\_pred = pd.DataFrame(prediction, columns=y.columns)

# Write the DataFrame to an Excel file s21\_pred.to\_excel("DT\_pred.xlsx", index=False)

This code outlines the steps involved in loading the dataset, training the decision tree regression model, taking user input, predicting the S21 values, printing the predictions, creating a Data Frame for the predicted values, and writing the Data Frame to an Excel file.

The data obtained from the decision tree model is utilized to generate a comparative graph, showcasing the simulated curve and the predicted curve. This graph, depicted in Fig. 13, visually represents the relationship between the input parameters and the corresponding output values.

By plotting both the simulated curve, obtained through rigorous simulations, and the predicted curve, derived from the decision tree model, we can visually assess the level of agreement or discrepancy between the two. This comparison aids in evaluating the accuracy and effectiveness of the decision tree model in predicting the desired output based on the given input parameters.

Fig. 13. Comparison between simulated S21 curve and Decision Tree predicted S21 curve

* + 1. **RANDOM FOREST ALGORITHM**

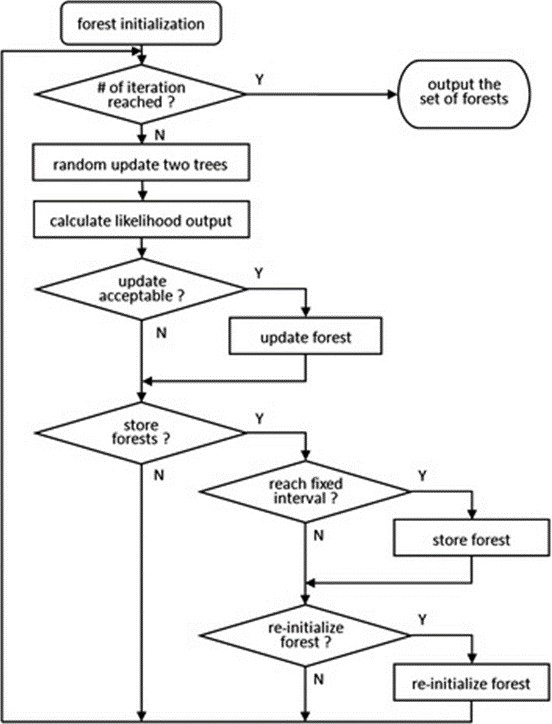
The Random Forest algorithm is a powerful machine learning technique that combines the principles of decision trees with the concept of ensemble learning. It is a versatile algorithm that can be applied to both classification and regression problems.

In Random Forest, multiple decision trees are created and trained on different subsets of the dataset, using a technique called bootstrap aggregating or bagging. Each decision tree is trained independently, considering a random subset of features at each node split. This randomness helps to introduce diversity among the trees and prevents overfitting.

During the prediction phase, each decision tree in the forest independently generates a prediction, and the final output is determined by combining the predictions of all the trees. In classification tasks, the final prediction is usually determined by majority voting, where the class with the most votes is selected. In regression tasks, the final prediction is the average or median of the individual tree predictions.

Here are the key components and steps involved in the Random Forest algorithm:

* + - 1. Ensemble of Decision Trees: Random Forest builds an ensemble of decision trees, where each tree is trained on a different subset of the training data. This process is known as bagging (bootstrap aggregating). By training multiple trees on different subsets of data, Random Forest introduces diversity and reduces overfitting.
      2. Random Feature Subsets: In addition to using different subsets of the training data, Random Forest also employs random feature subsets when constructing each decision tree. Instead of considering all features at each split, a random subset of features is selected. This randomness further enhances the diversity among the trees and reduces the correlation between them.
      3. Training Process: Random Forest trains each decision tree independently. The process begins by randomly selecting a subset of the training data (with replacement) known as a bootstrap sample. This sample is used to build a decision tree by recursively splitting the data based on feature thresholds that minimize impurity or maximize information gain (for classification) or reduce mean squared error (for regression).
      4. Voting or Averaging: Once all the decision trees are constructed, the Random Forest algorithm combines their predictions. For classification tasks, it employs majority voting, where the class predicted by the majority of the trees is chosen as the final prediction. In regression tasks, the algorithm takes the average of the predicted values from all the trees as the final prediction.
      5. Prediction and Evaluation: After training, the Random Forest model can be used to make predictions on unseen data. For classification tasks, it assigns class labels based on the majority vote of the ensemble. For regression tasks, it calculates the average of the predicted values. The performance of the Random Forest model can be evaluated using various metrics such as accuracy, precision, recall, F1 score, mean squared error, or R-squared, depending on the task.



#### Flowchart

Fig. 14 depicts the flowchart of Random Forest Algorithm.

Fig. 14. Flowchart of the Random Forest model

#### Code Snippet for Random Forest

import pandas as pd

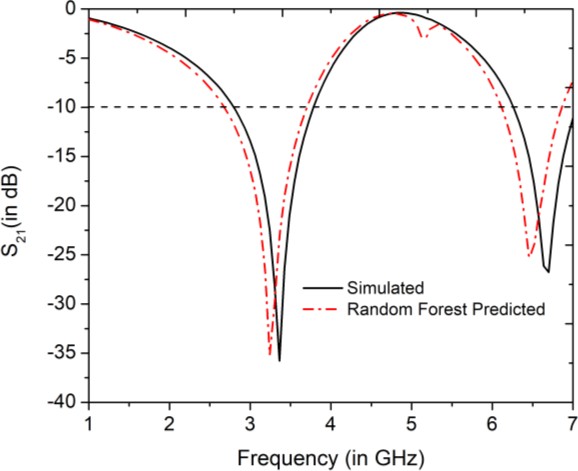
from sklearn.ensemble import RandomForestRegressor # Load dataset

data = pd.read\_csv("D:\VS code\FSS\_project\FSS\Dataset\_ML.csv") # Split into features and labels

X = data[['Length', 'Width', 'Breadth']]

y = data.drop(['Length', 'Width', 'Breadth'], axis=1) # Train random forest model

rf = RandomForestRegressor(n\_estimators=100, random\_state=42) rf.fit(X, y)



# Predict S21 values for a new input length = float(input("Enter the length: ")) width = float(input("Enter the width: "))

breadth = float(input("Enter the breadth: ")) new\_input = [[length, width, breadth]] prediction = rf.predict(new\_input) print(prediction)

# Set feature names feature\_names = list(y.columns)

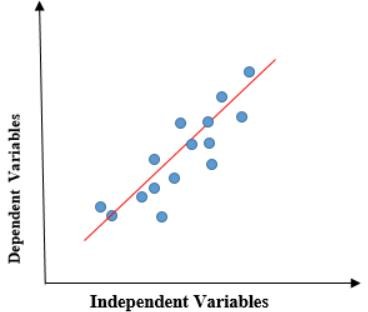
# Create a new DataFrame with the predicted S21 values s21\_pred = pd.DataFrame(prediction, columns=feature\_names)

# Write the DataFrame to an Excel file s21\_pred.to\_excel("RF\_pred.xlsx", index=False

The code snippet provided demonstrates the implementation of the Random Forest algorithm for regression using Python programming language. After obtaining the predicted S21 values using the Random Forest algorithm, a comparison analysis is conducted by plotting the predicted curve against the simulated curve. This graphical representation in Fig. 15 allows for a visual assessment of the error and accuracy of the model.

Fig. 15. Comparison between simulated S21 curve and random forest predicted S21 curve

Analyzing the plotted curve facilitates the identification of areas where the predicted values closely align with the simulated values, indicating higher accuracy. Conversely, regions where the curves diverge highlight potential areas of error or discrepancy in the predictions made by the Random Forest model. Thus, it provides valuable insights into the model's ability to capture and replicate the complex characteristics and behavior of the FSS, thereby aiding in further refinement and optimization of the algorithm for improved accuracy and performance.



### LINEAR REGRESSION ALGORITHM

Linear regression is a supervised machine learning algorithm used for predicting continuous numeric values based on a set of input features. It establishes a linear relationship between the input features and the target variable, allowing for the estimation of values within a continuous range. If there is a single input variable (x), such linear regression is called simple linear regression. And if there is more than one input variable, such linear regression is called multiple linear regression.

Fig.16. Representation of Linear Regression model

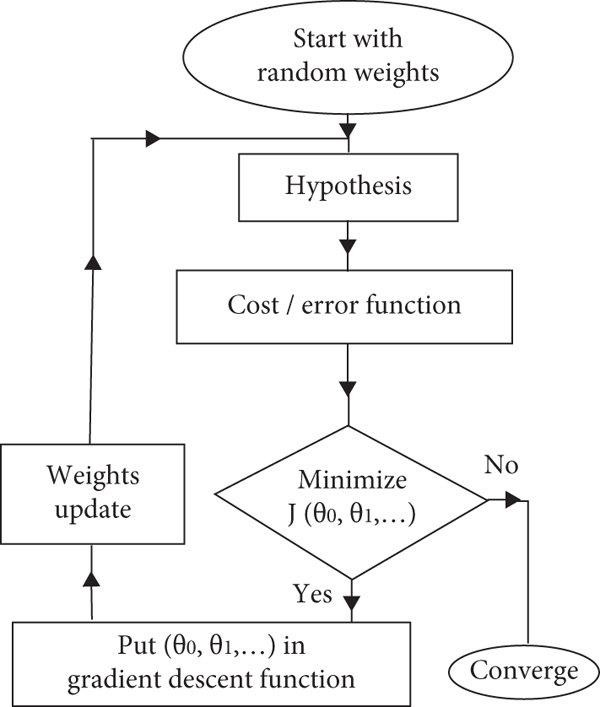
The algorithm has the following steps:

1. Data Preparation: Prepare the dataset consisting of input features (independent variables) and corresponding target values (dependent variable). Split the dataset into a training set and a test set for model evaluation.
2. Model Initialization: Initialize the model parameters, including the coefficients (slopes) and the intercept (constant term).
3. Training the Model: Use an optimization algorithm, such as gradient descent, to iteratively update the model parameters to minimize the difference between the predicted values and the actual target values in the training set. The steps involved are as follows:
   1. Compute the predicted values by multiplying the input feature values with their

corresponding coefficients and adding the intercept term.

* 1. Calculate the error or residual as the difference between the predicted values and the actual target values.
  2. Update the model parameters using the optimization algorithm to minimize the objective function, which is typically the sum of squared errors (ordinary least squares).

1. Making Predictions: Once the model parameters are learned, predictions can be made on new or unseen data. Multiply the input feature values with the learned coefficients and add the intercept to calculate the predicted target values.
2. Evaluation: Evaluate the performance of the linear regression model using appropriate evaluation metrics, such as mean squared error (MSE), root mean squared error (RMSE), or R- squared (coefficient of determination). Compare the predicted values with the actual target values in the test set to assess the model's accuracy and generalization capability.



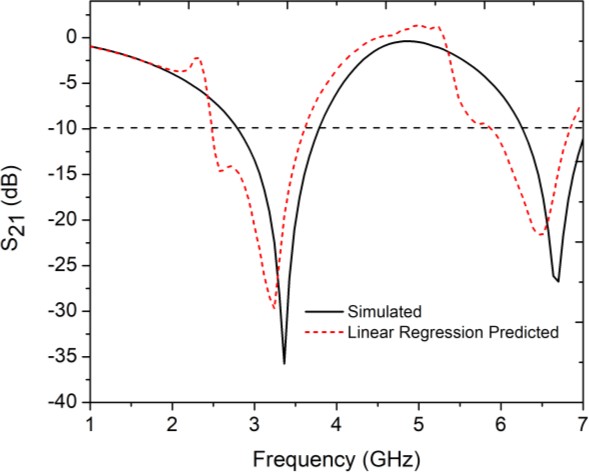
1. Model Refinement: If the model performance is not satisfactory, refinement techniques like feature selection, feature engineering, or regularization methods (e.g., Ridge regression, Lasso regression) can be applied to improve the model's predictive power and handle issues like overfitting or multicollinearity.

#### Flowchart

Fig. 17 shows the flowchart for Linear Regression algorithm.

Fig. 17. Flowchart of Linear Regression model

We implemented linear regression in our project using the Python programming language and compared its output with the simulated output. The goal was to assess the accuracy and effectiveness of the linear regression model in predicting the desired output based on the input parameters.



#### Code snippet for Linear Regression

import pandas as pd

from sklearn.linear\_model import LinearRegression

# Load dataset

data = pd.read\_csv("D:\VS code\FSS\_project\FSS\Dataset\_ML.csv")

# Split into features and labels

X = data[['Length', 'Width','Breadth']]

y = data.drop(['Length', 'Width','Breadth'], axis=1)

# Train linear regression model lr = LinearRegression() lr.fit(X, y)

# Predict S21 values for a new input length = float(input("Enter the length: ")) width = float(input("Enter the width: "))

breadth = float(input("Enter the breadth: ")) new\_input = [[length, width, breadth]] prediction = lr.predict(new\_input) print(prediction)

# Set feature names feature\_names = list(y.columns)

# Create a new DataFrame with the predicted S21 values s21\_pred = pd.DataFrame(prediction, columns=feature\_names)

# Write the DataFrame to an Excel file s21\_pred.to\_excel("LR\_pred", index=False)

The data generated from the linear regression model is used to construct a comparative graph that illustrates the simulated curve and the predicted curve. This graphical comparison as shown in Fig. 18 allows us to evaluate the accuracy and efficacy, which is comparatively lower, of the linear regression model in forecasting the desired output based on the provided input parameters.

Fig. 18. Comparison between simulated S21 curve and Linear Regression predicted S21 curve

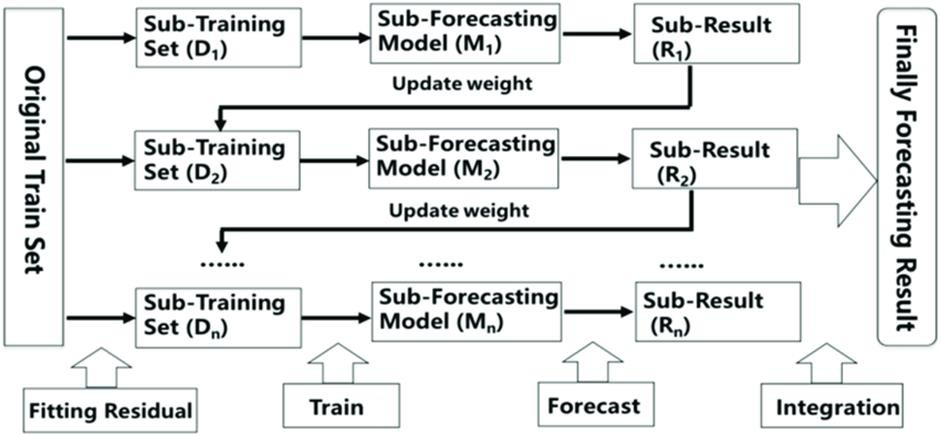
### MULTIPLE OUTPUT GRADIENT BOOSTING REGRESSION (MOGBR) ALGORITHM

The Multiple Output Gradient Boosting algorithm is a powerful machine learning technique used for regression tasks that involve predicting multiple output variables simultaneously. It is an extension of the popular Gradient Boosting algorithm.

The Multiple Output Gradient Boosting algorithm is particularly useful when dealing with regression tasks involving multiple dependent variables, providing accurate predictions across multiple dimensions.

Here is a high-level overview of how you can incorporate a multiple output gradient boosting algorithm into a feature selection framework:

* + - 1. Data preparation: Organize your dataset, ensuring it contains the input features and corresponding output variables for each instance. If you have multiple output variables, they should be arranged in a matrix or a set of vectors.
      2. Feature selection methods: Choose or develop feature selection methods that are compatible with multiple output problems. Common techniques include correlation-based methods, wrapper methods (e.g., recursive feature elimination), and regularization-based methods (e.g., L1 or L2 regularization). These methods help identify the most relevant features for your multiple output problem.
      3. Feature selection loop: Iterate through the feature selection process for each output variable. At each iteration, follow these steps:
         1. Train a multiple output gradient boosting model using the selected features from the previous iteration (or the full feature set in the first iteration).
         2. Evaluate the performance of the model using appropriate metrics for multi-output regression (e.g., mean squared error, mean absolute error, etc.) on a validation set or using cross-validation.
         3. Apply the feature selection method to determine the importance or relevance of each feature for the current output variable. Select the top-k features based on the relevance scores.
         4. Store the selected features and their relevance scores for the current output variable.
      4. Final feature selection: After completing the feature selection loop for all output variables, you can combine the relevance scores obtained from each iteration to create an overall ranking of the features. You can use simple averaging, weighted averaging, or other aggregation methods to obtain the final feature ranking.
      5. Model training and evaluation: Train a multiple output gradient boosting model using the selected features obtained from the final feature selection step. Evaluate its performance on a separate test set using appropriate metrics for multi-output regression.



#### Flowchart

Fig. 19 shows the flowchart of Multiple Output Gradient Boosting Algorithm.

Fig. 19. Flowchart of Multiple Output Gradient Boosting Regression model.

#### Code snippet for Multiple Output Gradient Boosting Regression

import pandas as pd

from sklearn.multioutput import MultiOutputRegressor from sklearn.ensemble import GradientBoostingRegressor

# Load dataset

data = pd.read\_csv("D:\VS code\FSS\_project\FSS\Dataset\_ML.csv")

# Split into features and labels

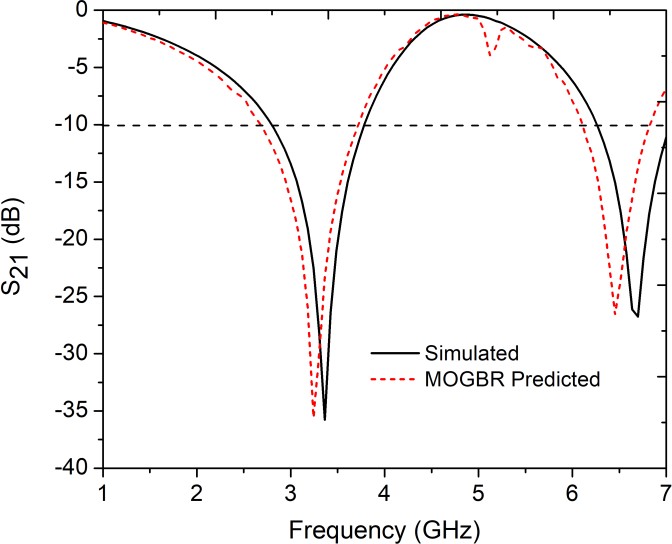
X = data[['Length', 'Width', 'Breadth']]

y = data.iloc[:, 3:] # select all columns starting from the 4th column

# Train multi-output gradient boosting model gbr = GradientBoostingRegressor() multi\_gbr = MultiOutputRegressor(gbr) multi\_gbr.fit(X, y)

# Predict S21 values for a new input length = float(input("Enter the length: ")) width = float(input("Enter the width: "))

breadth = float(input("Enter the breadth: ")) new\_input = [[length, width, breadth]] prediction = multi\_gbr.predict(new\_input) print(prediction)



# Set feature names feature\_names = list(y.columns)

# Create a new DataFrame with the predicted S21 values s21\_pred = pd.DataFrame(prediction, columns=feature\_names)

# Write the DataFrame to an Excel file s21\_pred.to\_excel("MOGBR\_pred.xlsx", index=False)

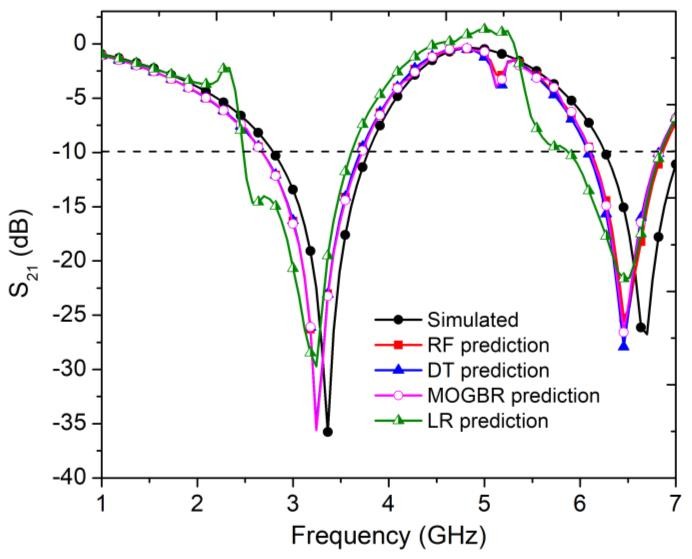
This code demonstrates the usage of Multi-Output Gradient Boosting Regression for predicting multiple output variables based on the input features.

The output obtained from the Multi-Output Gradient Boosting Regression (MOGBR) model is utilized to generate a comparative graph, illustrating both the simulated curve and the predicted curve. This graphical representation, depicted in Fig. 20, serves as a visual assessment of the performance and predictive capabilities of the MOGBR model in accurately forecasting the desired outputs based on the given input parameters.

Fig. 20. Comparison between simulated S21 curve and MOGBR predicted S21 curve

* 1. **Results**

# Chapter 4: Results and Analysis



A comparative analysis was conducted using the outputs obtained from all four algorithms, and a graph was generated, as depicted in Fig. 21. Using the input parameters of length (L1) as 4.5 mm, width (W1) as 1 mm, and breadth (L5) as 3.25 mm, the implemented algorithms generated predictions based on these values. This graph allows for a visual comparison between the predicted curves from each algorithm and the simulated output.

Fig.21. Comparative analysis of the ML algorithms

The analysis revealed that the predicted curve generated by the Decision Tree algorithm exhibited the closest alignment with the simulated output. The Decision Tree algorithm demonstrated a higher degree of accuracy in capturing the underlying patterns and trends within the data, resulting in a better match between the predicted and simulated curves.

## Error Analysis

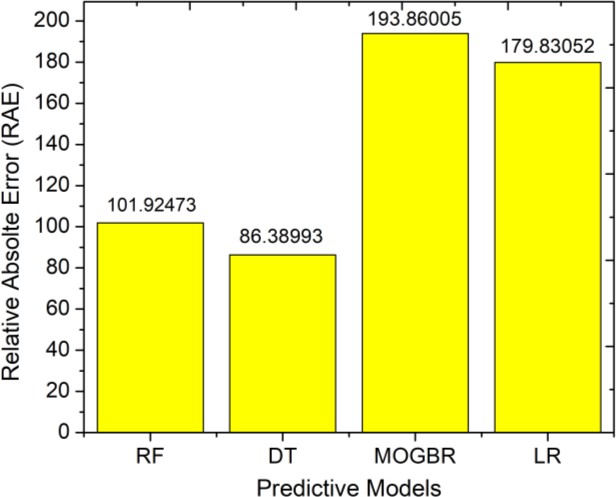
To gain a deeper understanding of the performance of each algorithm, further error analysis was conducted by calculating the relative absolute error (RAE). RAE is a commonly used metric that quantifies the difference between predicted values and actual values.

The RAE is computed by taking the absolute difference between the predicted and actual values, dividing it by the absolute value of the actual value, and then multiplying by 100 to express it as a percentage. This metric provides insights into the accuracy of the predictions made by each algorithm, allowing for a comprehensive evaluation of their performance.

RAE = sum(|Y\_true - Y\_pred|) / sum(|Y\_true - mean(Y\_true)|) Where:

Y\_true represents the actual values Y\_pred represents the predicted values

mean(Y\_true) is the mean of the actual values



By examining the RAE values obtained for each algorithm, we can assess their respective abilities to accurately predict the output curves. A lower RAE indicates a smaller discrepancy between the predicted and actual values, implying a higher level of accuracy.

Fig.22. Comparison of RAE of different ML algorithms

Based on the error analysis conducted, as depicted in Fig. 22, it was observed that the Decision Tree algorithm exhibited the lowest RAE among the tested algorithms. This suggests that the Decision Tree algorithm provides the most accurate predictions of the output curves for the given dataset. The lower RAE obtained for the Decision Tree algorithm further supports its suitability and effectiveness in this particular application.

A comprehensive table was constructed to compare the percentage errors for various parameters, including bandwidth and frequency. This table, as illustrated in Table II, provides a clear overview of the discrepancies between the predicted values and the actual values for each parameter.

#### TABLE II: Comparison of the percentage errors for various parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name of Different Algorithms Used | % error in 1st Resonating Frequency | % error in 2nd Resonating Frequency | % error in 1st Operating Bandwidth | % error in 2nd Operating Bandwidth |
| Decision Tree | 3.57 | 1.77 | 6.18 | 2.59 |
| Random Forest | 3.57 | 2.66 | 1.03 | 10.38 |
| Linear Regression | 3.57 | 2.81 | 18.55 | 55.84 |
| Multiple Output Gradient Boosting Regression | 3.57 | 4.44 | 6.1 | 5.45 |

By analyzing the percentage errors, we can assess the accuracy of the predictions made by the different algorithms for each parameter. A lower percentage error indicates a closer approximation to the actual values, reflecting a higher level of precision in the predictions.

## Performance Comparison of the Proposed FSS with other Published Works

To compare the key specifications of a few publications of a similar nature, a performance comparison table (Table II) is provided. Unit cell size, bandwidth, resonant frequency and degree of angular stability are some of the comparison parameters.

**TABLE II: PERFORMANCE COMPARISON OF PROPOSED DUAL BAND FSS WITH THE PUBLISHED RELATED WORKS.**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Ref. No.** | **Geometry** | **Polarization** | **Unit cell size** | **Bandwidth** | **Resonating frequency** | **No. of Layers** | **Angular Stability** |
| [2] | Three- legged loaded | TE | NM | 0.179 GHz | 2.4 GHz | Single layered | From 0° to  70° for TE polarizations. |
| [3] | Bioinspired (combined element) | TE and TM | 20 mm ×  20 mm | 1.09, 1.33,  0.46 (all in GHz)  1.16, 1.33,  0.36 (all in GHz) | 4.2 GHz, 9.5 GHz, and 10.5  GHz;  4.2 GHz, 9GHz, 10.5  GHz | Single layered | NM |
| [4] | Composed of a square ring, eight rectangular strips, four square patches with slots, and a deformed cross in the center. | TE and TM | 8.8mm x 8.8mm | 0.28 GHz | 2.6 GHz | Single Layered | From 0° to 60° for TE and TM polarizations. |
| [5] | A pair of Sierpiński triangles organized as a bow tie, connected by a central metal strip | TE and TM | NM | NM | 750 MHz and  3.5 GHz | Single layered | From 0° to 30° for TE and TM polarizations. |
| [6] | Double square slots | TE and TM | 25mm x 25mm | 0.08 GHz,  0.70 GHz | 2.4 GHz and 5 GHz | Single layered | From 0° to 45° for TE and TM polarizations. |
| **This work** | **Interconnected open square dumbbell (OSHD)**  **structures** | **TE and TM** | **15mm x 15mm** | **0.69 GHz,**  **0.82 GHz** | **2.4 GHz and 5 GHz** | **Single Layered** | **From 0° to 60° for TE and TM polarizations.** |

NM: Not Mentioned

* 1. **Discussion**

# Chapter 5: Conclusion

The implementation of the decision tree algorithm in our project has successfully contributed to the synthesis of a stopband FSS structure, operating at frequencies of 2.4 and 5GHz. This FSS design demonstrates remarkable angular stability and polarization insensitivity. By incorporating four interconnected dumbbells, we have effectively increased the percentage bandwidth of the FSS structure. This modification elongates the metallic strip and results in a shift of the resonant frequency towards lower frequencies. To achieve optimal performance, we have carefully determined the length and placement of the connecting strips, ensuring a four-fold symmetric structure with uniform surface current distributions and enhanced angular stability.

The fabrication process involved the use of a cost-effective FR4 substrate to create physical prototypes. The comparison between simulation and predicted results revealed a strong agreement, confirming the reliability and effectiveness of the proposed method for FSS synthesis.

In summary, our research offers a compelling and innovative approach to FSS structure synthesis, utilizing machine learning techniques, particularly the decision tree algorithm. The results we have achieved demonstrate the considerable potential of this methodology, highlighting its attractiveness for driving advancements in FSS design and development.

## Future Works

The successful implementation of our project opens up several avenues for future exploration and improvement. Some potential areas for future research and development include:

* + - Enhancement of Machine Learning Models: Further investigation can be conducted to explore the performance of additional machine learning algorithms or advanced techniques such as deep learning. Comparing and optimizing various models can lead to improved accuracy and efficiency in predicting the output curves of FSS structures.
    - Parameter Optimization: Fine-tuning the design parameters of FSS structures can be explored to achieve better performance. This includes optimizing the unit-cell size, shape, and geometry to further enhance the stopband characteristics, bandwidth, and angular stability of the FSS.
    - Multiband FSS: Expanding the scope of the project to develop multiband FSS structures can be a valuable future endeavor. This involves designing FSS structures that exhibit stopband behavior at multiple frequencies simultaneously, catering to the needs of modern wireless communication systems that operate in multiple frequency bands.
    - Experimental Validation: Conducting extensive experimental validation of the FSS structures is essential for verifying the performance and reliability of the proposed designs. Experimental measurements and comparisons with simulation results can provide valuable insights and feedback for further refinement and improvement.

By addressing these future research directions, we can continuously enhance the effectiveness, versatility, and practicality of FSS structures, paving the way for their widespread adoption and application in various industries and fields.

# Chapter 6: REFERENCES

1. B. A. Munk, Frequency Selective Surfaces: Theory and Design, 1st ed. New York, NY, USA: Wiley, 2000.
2. Mesut Kartal, Sedef Kent Pinar, Bora Doken, and Ismail Gungor, “A new narrow band frequency selective surface geometry design at the unlicensed 2.4-GHz ISM band”*,* in *MICROWAVE AND OPTICAL TECHNOLOGY LETTERS* / Vol. 55, No. 12, December 2013.
3. L. C. M. M. Fontoura, H. W. De Castro Lins, A. S. Bertuleza, A. G. D’assunção and A. G. Neto, "Synthesis of Multiband Frequency Selective Surfaces Using Machine Learning With the Decision Tree Algorithm," in IEEE Access, Vol. 9, pp. 85785-85794, June 7, 2021.
4. M. Qu, Y. Feng, J. Su and S. M. A. Shah, "Design of a Single-Layer Frequency Selective Surface for 5G Shielding," in *IEEE Microwave and Wireless Components Letters*, Vol. 31, no. 3, pp. 249- 252, March 2021.
5. B.Decoster,S.Maes, et. Al. “Dual-Band Single-Layer Fractal Frequency Selective Surface for 5G

Applications,” *Electronics 10*, 2880, 2021.

1. Yuan Xu and Mang He, “A Design Method for Synthesizing a Dual-Passband FSS for Wi-Fi Using Equivalent Circuit Model”, in *ACES JOURNAL*, Vol. 34, No. 10, October 2019.
2. Yan Zhang, Xiue Guo, Juan Xu, “A novel dual-band miniaturized frequency selective surface with

high selectivity,” *Microw Opt Technol Lett*., Vol. 65, Issue2, pp. 548-558, February 2023.

1. Ali H, Shafique MF, Riaz L, Khan MA*,* “A compact, stable, and angular independent dual-band frequency-selective surface for WiFi applications”, *Microw Opt Technol Lett*., 60:2451–2456, 2018.
2. Clarissa de Lucena Nóbregaa , Marcelo Ribeiro da Silvaa , Paulo Henrique da Fonseca Silvab and Adaildo Gomes D’Assunçãoa, “Analysis and design of frequency selective surfaces using teragon patch elements for WLAN applications”*,* in *Journal of Electromagnetic Waves and Applications*, Vol. 28, No. 11, 1282–1292, April 2014.
3. D. Ferreira, T. R. Fernandes, I. Cuiñas and R. F. S. Caldeirinha, "A dual-band single-layer frequency selective surface for Wi-Fi applications," 2015 9th European Conference on Antennas and Propagation (EuCAP), IEEE Xplore 31 August 2015, pp.1-4.
4. Snehasish SAHA, Nurnihar BEGAM, Sushanta BISWAS, Partha Pratim SARKAR, “A Cascaded Tunable Wide Stop Band Frequency Selective Surface with High Roll-off Band Edge”*,* in *RADIOENGINEERING*, Vol. 30, No. 1, April 2021.
5. N. Liu, X. Sheng, C. Zhang and D. Guo, "Design and Synthesis of Band-Pass Frequency Selective Surface With Wideband Rejection and Fast Roll-Off Characteristics for Radome Applications," in IEEE Transactions on Antennas and Propagation, Vol. 68, no. 4, pp. 2975-2983, April 2020.
6. M. Bouslama, M. Traii, T. A. Denidni and A. Gharsallah, "Active Antenna Based on a New Cylindrical Frequency Selective Surface," 2017 Mediterranean Microwave Symposium (MMS), pp. 1-2, 2017.
7. S. B. Paiva, V. P. S. Neto and A. G. D'Assunção, "A New Compact, Stable, and Dual-Band Active Frequency Selective Surface with Closely Spaced Resonances for Wireless Applications at 2.4 and

2.9 GHz," in IEEE Transactions on Electromagnetic Compatibility, Vol. 62, no. 3, pp. 691-697, June 2020.

1. Guohong Du, Chengyang Yu, and Changjun Liu, “Frequency Selective Surface with Switchable Polarization”, in *MICROWAVE AND OPTICAL TECHNOLOGY LETTERS* / Vol. 56, No. 2, February 2014.