**T.C.**
**KONYA TEKNİK ÜNİVERSİTESİ**
**LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ**

# SELÇUKLU YILDIZI ŞEKİLLİ MİKROŞERİT ANTENİN YAPAY ZEKA YÖNTEMLERİ KULLANILARAK OPTİMUM TASARIMI VE GERÇEKLEŞTİRİLMESİ

**Erdem YELKEN**

**YÜKSEK LİSANS TEZİ**

**Elektrik-Elektronik Mühendisliği Anabilim Dalı**

**Temmuz-2021**
**KONYA**

## TEZ KABUL VE ONAYI

Erdem YELKEN tarafından hazırlanan "**Selçuklu Yıldızı Şekilli Mikroşerit Antenin Yapay Zeka Yöntemleri Kullanılarak Optimum Tasarımı Ve Gerçekleştirilmesi**" adlı tez çalışması 13/07/2021 tarihinde aşağıdaki jüri tarafından oy birliği / oy çokluğu ile Konya Teknik Üniversitesi Lisansüstü Eğitim Enstitüsü Elektrik-Anabilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

| **Jüri Üyeleri** | **İmza** |
|---|---|
| **Başkan**<br>Dr. Öğr. Üyesi Özgür DÜNDAR | …….. |
| **Danışman**<br>Dr. Öğr. Üyesi Dilek UZER | …….. |
| **Üye**<br>Doç. Dr. Seyfettin Sinan GÜLTEKİN | …….. |

Yukarıdaki sonucu onaylarım.

Prof. Dr. Saadettin Erhan KESEN
Enstitü Müdürü

Bu tez çalışması Konya Teknik Üniversitesi Bilimsel Araştırma Projeleri Birimi tarafından 201002070 nolu proje ile desteklenmiştir.

## TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

## DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Erdem YELKEN

Tarih: 13.07.2021

# ÖZET

## YÜKSEK LİSANS TEZİ

## SELÇUKLU YILDIZI ŞEKİLLİ MİKROŞERİT ANTENİN YAPAY ZEKA YÖNTEMLERİ KULLANILARAK OPTİMUM TASARIMI VE GERÇEKLEŞTİRİLMESİ

**Erdem YELKEN**

**Konya Teknik Üniversitesi**
**Lisansüstü Eğitim Enstitüsü**
**Elektrik-Elektronik Mühendisliği Anabilim Dalı**

**Danışman: Dr. Öğr. Üyesi Dilek UZER**
**2021, 100 Sayfa**

**Jüri**
**Danışmanın Dr. Öğr. Üyesi Dilek UZER**
**Doç. Dr. Seyfettin Sinan GÜLTEKİN**
**Dr. Öğr. Üyesi Özgür DÜNDAR**

Son yıllarda gelişen teknoloji ve kablosuz haberleşme sistemlerinin sıklıkla tercih edilmesiyle birlikte daha küçük, hafif ve yüksek performans gerektiren mikroşerit antenler sıklıkla tercih edilmektedir. Bu antenlerin kullanım amacına yönelik olarak istenilen performansın elde edilmesi için farklı tasarımları için yapılan çalışmalar artmaktadır. Antenlerin tasarımında, ışıma frekansı, giriş empedansı ve kazanç gibi arzu edilen anten karakteristiklerini sağlayacak fiziksel parametrelerin hesaplanması önemli bir problem haline gelmiştir. Son yıllarda anten tasarımı problemine dayalı olarak yapay zeka yöntemleri ile tasarımı yapılan antenler başarılı sonuçlar elde edebilmektedir. Büyük boyutlu optimizasyon problemlerinde sıklıkla tercih edilen Meta-Sezgisel Algoritmalar mikroşerit anten optimizasyonu yapılan bilimsel çalışmalarda popüler hale gelmiştir.

Bu çalışmada, geri yayılma algoritması ve Meta-Sezgisel algoritma birleştirilerek Yapay Sinir Ağı eğitimi için yeni bir model geliştirilmiştir. Geri yayılma algoritmasının çözüm bulmadaki en büyük dezavantajı, küresel minimumdan ziyade yerel minimuma sıkışmasıdır. Bu yeni hibrit eğitim algoritmasında ise, yerel ve küresel arama eş zamanlı olarak yapılmıştır. Başlangıçta, uzun atlama sayesinde yerel minimuma yakalanma olasılığının düşük olması nedeniyle Yapay Sinir Ağı ağırlıklarını elde etmek için optimizasyon algoritmaları kullanılmıştır. Daha sonra, yerel arama yeteneğinin nöral ağ eğitiminde kullanılması için Geri Yayılım algoritması ile birleştirilmiştir. Yapay Sinir Ağının eğitim aşamasında Levenberg-Marquardt geri yayılma algoritması kullanılmıştır.

Bu çalışmada yaygın olarak kullanılan antenlere göre bant genişliğinde iyileştirme sağlayabilen Selçuklu Yıldızı şekilli mikroşerit anten tasarımı gerçekleştirilmiş ve simüle edilmiştir. Simülasyon sonuçlarından elde edilen 1342 mikroşerit antenden oluşan veri seti, 7 farklı optimizasyon algoritması kullanılan YSA modeli ile eğitilmiş ve test edilmiştir. Ayrıca farklı dielektrik katsayılara sahip malzemelerle tasarlanan antenler üretilerek ölçüm sonuçları alınmış ve simülasyon sonuçları ile karşılaştırılmıştır.

**Anahtar Kelimeler:** mikroşerit anten, Meta-Sezgisel Algoritmalar, Selçuklu Yıldızı, Yapay Sinir Ağı

# ABSTRACT

## MS THESIS

## OPTIMUM DESIGN AND PRODUCTION OF SELJUK STAR SHAPED MICROSTRIP ANTENNA USING ARTIFICIAL INTELLIGENCE METHODS

**Erdem YELKEN**

**Konya Technical University**
**Institute of Graduate Studies**
**Department of Electrical Electronics Engineering**

**Advisor: Asst. Prof. Dr. Dilek UZER**

**2021, 100 Pages**

**Jury**
**Advisor Asst. Prof. Dr. Dilek UZER**
**Assoc. Prof. Seyfettin Sinan GÜLTEKİN**
**Asst. Prof. Özgür DÜNDAR**

With the developing technology and wireless communication systems being preferred frequently in recent years, microstrip antennas that are smaller, lighter and require high performance are frequently preferred. Studies for different designs of these antennas are increasing in order to achieve the desired performance for the purpose of use. In the design of antennas, the calculation of the physical parameters that will provide the desired antenna characteristics such as radiant frequency input impedance and gain has become an important problem. In recent years, antennas designed with artificial intelligence methods based on the antenna design problem can achieve successful results. Meta-Heuristic Algorithms, which are frequently preferred in large-scale optimization problems, have become popular in scientific studies on microstrip antenna optimization.

In this study, a new model for Neural Network training has been developed by combining back propagation algorithm and Meta-Heuristic algorithm. The biggest disadvantage of the back propagation algorithm in finding solutions is that it gets stuck at the local minimum rather than the global minimum. In this new hybrid training algorithm, local and global search are performed simultaneously. Initially, optimization algorithms were used to obtain the Neural Network weights, since the probability of catching the local minimum is low due to the long jump. Then, it is combined with Backpropagation algorithm to use local search capability in neural network training. Levenberg-Marquardt back propagation algorithm was used in the training phase of the Artificial Neural Network.

In this study, Seljuk Star shaped microstrip antenna design, which can improve bandwidth compared to commonly used antennas, has been realized and simulated. The data set consisting of 1342 microstrip antennas obtained from the simulation results was trained and tested with the ANN model using 7 different optimization algorithms. In addition, antennas designed with materials with different dielectric coefficients were produced and the measurement results were obtained and compared with the simulation results.

**Keywords:** microstrip antenna, Meta-Heuristic Algorithms, Seljuk Star, Artificial Neural Network

# ÖNSÖZ

Çalışmalarım boyunca bilimsel katkılarıyla beni aydınlatan, bilgi, tecrübe ve yardımlarını esirgemeyen danışman hocam Sayın Dr. Öğr. Dilek Uzer ve değerli hocam Sayın Doç. Dr. S. Sinan Gültekin' e teşekkürlerimi borç bilirim.

Deneysel çalışmalarım sırasında karşılaştığım zorlukları aşmamda bana her zaman yardımcı olan ve yön gösterici tavsiyelerinden dolayı değerli dostlarım Hasan Basri Öksüz ve Muhammed Isayev' e değerli katkılarından dolayı teşekkür ederim.

Ayrıca hayatım boyunca her türlü maddi ve manevi desteğini esirgemeyen aileme en içten teşekkürlerimi sunarım.

Erdem YELKEN
KONYA-2021

# İÇİNDEKİLER

## SİMGELER VE KISALTMALAR

### Simgeler

$S_{11}$ : Return loss; Geri dönüş kaybı

$\Gamma$ : Yansıma katsayısı

$D(\theta, \phi)$ : Yönelticilik

$U(\theta, \phi)$ : Işıma şiddeti yoğunluğu

$R_L$ : Kayıp direnci

$R_r$ : Işıma direncinde

$Z_R$ : Anten Empedansı

G : Anten Kazancı

$D$ : Anten Yönelticiliği

C : Boş uzaydaki ışık hızı

$f_r$ : Rezonans Frekansı

$e$ : Anten Verimi

$\varepsilon_r$ : Bağıl dielektrik sabiti

W : Mikroşerit Anten Yama Genişliği

L : Mikroşerit Anten Yama Uzunluğu

h : Dielektrik alt tabakanın kalınlığı

$c_1$ ve $c_2$ : PSO için hız katsayıları

pa : GKAA için yumurtlama oranı

$\beta$ : AA için çekicilik katsayısı

$\sigma$ : YOA için rastgelelik katsayısı

$\vec{D}$ : BOA için konum vektörü

$\vec{D}_\alpha, \vec{D}_\beta, \vec{D}_\delta$ : GKA için av ve kurt (alfa, beta, delta) arasındaki vektör mesafesi

$\vec{X}_\alpha, \vec{X}_\beta, \vec{X}_\delta$ : GKA için avın pozisyon vektörü

$\vec{C}_\alpha, \vec{C}_\beta, \vec{C}_\delta, \vec{A}_\alpha, \vec{A}_\beta, \vec{A}_\delta$: GKA için alfa, beta ve delta kurtları için katsayı vektörleri

$\vec{U}_\alpha, \vec{U}_\beta, \vec{U}_\delta$ : GKA için alfa, beta ve delta kurtları için deneme vektörleri

**Kısaltmalar**

KİS : Kişisel İletişim Sistemleri

YSA : Yapay Sinir Ağı

MŞA : Mikroşerit Anten

LM : Levenberg-Marquardt algoritması

MSE : Ortalama Kare Hatası

RMSE : Kök Ortalama Kare Hatası

BÇS : Bulanık Çıkarım Sistemi

ÇK-İBSA : Çok Katmanlı İleri Beslemeli Sinir Ağı

GA : Genetik Algoritma

PSO : Parçacık Sürü Optimizasyonu

GKAA : Guguk Kuşu Arama Algoritması

AA : Ateş Böceği Algoritması

YOA : Yabani Ot Algoritması

BOA : Balina Optimizasyon Algoritması

GKA : Gri Kurt Algoritması

HFSS : High Frequency Structural Simulator

DİMA : Dikdörtgen Şekilli Mikroşerit Antenler

DAMA : Dairesel Şekilli Mikroşerit Antenler

SYMŞA : Selçuklu Yıldızı Mikroşerit Anten

FDTD : Finite Difference Time Domain; Zaman domeninde sonlu arklar yöntemi

FEM : (Finite Element Method) Sonlu Eleman Metodu

RL : Return Loss; Geri Dönüş Kaybı

BG : Bant Genişliği

WLAN : Wireless Local Area Network; Kablosuz yerel alan ağı

VSWR : Voltage standing wave ratio; Voltaj duran dalga oranı

# 1. GİRİŞ

Günümüzde kablo karmaşası ve maliyetlerinden kurtulma ve kullanıcıların hareket özgürlüklerini sağlama avantajlarından dolayı kablosuz sistemler sıklıkla tercih edilmektedir. Son yıllarda gelişen teknoloji ile yüksek performans gerektiren uçaklar, radar sistemleri, uzay araçları uydu ve füze uygulamaları gibi pek çok alanda düşük maliyetli, küçük boyutlu, kolay kuruluma sahip ve yüksek bant genişliği gereksinimlerini karşılayabilen antenlere ihtiyaç duyulmaktadır.

Son yıllarda kablosuz iletişim sistemlerinin sıklıkla tercih edilmesi ile kullanıcılar tarafından beklenilen yüksek hız ve daha geniş kullanım alanları beraberinde tasarlanacak kablosuz iletişim cihazlarının düşük maliyetli ve geniş bir haberleşme bandı ihtiyacını karşılayabilecek düzeyde olması beklenmektedir. Bu nedenle, mobil haberleşme sistemlerinde oldukça fazla kullanılan antenlerden en önemlisi olan mikroşerit antenler yoğun ilgi görmektedir.

İlk mikroşerit yama anten fikri Deschamps tarafından 1953 yılında ortaya atılmıştır(Deschamps, 1953). Ancak, baskı devre teknolojisinin ilerlemesinden sonra mikroşerit antenlerin ilk pratik uygulaması Howell ve Mudson tarafından 1970' li yılların başlarında yapılmıştır(Munson, 1974; Howell, 1975). Küçük boyutlu, hafif oluşları ve üretim maliyetlerinin düşüklüğü gibi avantajlarından dolayı mikroşerit antenler günümüzde bu ihtiyaçları karşılayabilecek duruma gelmişlerdir. Gelişen teknoloji ve artan ihtiyaçlar mikroşerit antenler üzerindeki ilgi aralığını artırmış ve istenilen performanslarda çalışması adına yeni üretim metodolojileri geliştirilmiştir.

Mikroşerit antenlerin gittikçe artan kullanımı ve kullanıldığı yere uyum sağlayabilmeleri için farklı tasarım gereksinimleri gerekmekte ve bu antenlerin en önemli dezavantajı olan dar bant, düşük kazanç gibi özelliklerinin iyileştirilmesi gibi nedenlerden dolayı yeni tasarımlar denenmektedir. Mikroşerit antenlerin tüm fiziksel ve elektriksel parametrelerinde ve besleme türlerinde farklı varyasyonlar yapılarak istenilen uygulamalara daha elverişli tasarımlar elde edilmeye çalışılmaktadır.

Mikroşerit antenleri yüksek verimle çalıştırmak için, ilgili birçok teknolojiye ihtiyaç vardır. Günümüzde mikroşerit antenlerin elektronik haberleşme pazarında kullanımının artması, performans analizinin yapılmasında daha basit ve etkili yöntemlerin kullanılmasını gerektirmektedir. Bu nedenle mikroşerit anten tasarımcıları, çok fazla hesaplama zamanı gerektirmeyen basit yaklaşımları tercih etmektedirler. Son yıllarda bilgisayar bilimindeki hızlı gelişmelerle yapay sinir ağları ve optimizasyon algoritmaları

gibi yapay zekaya dayanan teknikler, geleneksel optimizasyon ve klasik analitik yaklaşımlardan daha esnek ve elverişli sonuçlar üreten güçlü alternatif araçlar olmuştur.

Mikroşerit antenler birçok önemli özelliğine rağmen, pek çok sistemde dar bant genişliği ile sınırlıdırlar. Kablosuz iletişim sistemlerinde daha hızlı ve kaliteli haberleşme için gereken geniş bant genişliği, mikroşerit antenlerin dezavantajı olan bant genişliğini artırmayı önemli hale getirmiştir.

Son yıllarda Yapay Sinir Ağları ve doğadan esinlenen optimizasyon algoritmalarının geliştirilmesi ve bilimsel çalışmalarda karşılaşılan çok boyutlu problemlere uygulanabilir olmasıyla haberleşme alanında önemli çözümler elde edilerek önemli çalışmalara konu olmaya başlamıştır. Mikroşerit antenlerin dar bant genişliği, düşük kazanç, düşük güç kapasitesi gibi dezavantajları uygulama alanlarındaki en önemli sorun haline gelmiş ve kullanım alanlarında kısıtlamaya yol açmıştır. Bu yapay zekâ yöntemleri ile mikroşerit antenlerinde iyileştirmeler yapmak mümkündür.

Mikroşerit antenin geometrik şekli, anten performansını büyük ölçüde etkilemektedir. Mikroşerit antenlerin birçok şekilde tasarımları yapılmaktadır. Literatürde sunulan çalışmaların büyük bir kısmı analizlerinin ve tasarımlarının kolay olmasından dolayı dikdörtgen, üçgen ve daire gibi bilinen geometriler üzerinde yapılan çalışmalar bulunmaktadır. Ancak bu bilinen geometrilere sahip mikroşerit antenlerin boyutları, yüksek bant genişliği uygulamaları için göreceli olarak büyüktürler. Bu yama şekillerine göre yama alanında küçültmeye imkân tanıyan yeni bir yama şekli olarak Selçuklu Yıldızı şekilli mikroşerit anten (SYMŞA) bant genişliğinde iyileştirmeye olanak sağlamıştır(Uzer, 2016; Uzer ve ark., 2016).

### 1.1. Tezin Amacı ve Önemi

Kablosuz iletişim sistemlerinin sıklıkla tercih edilmesi ile kullanıcılar tarafından beklenilen yüksek hız ve daha geniş kullanım alanları beraberinde tasarlanacak kablosuz iletişim cihazlarının düşük maliyetli ve geniş bir haberleşme bandı ihtiyacını karşılayabilecek düzeyde olması beklenmektedir. Bu nedenle, söz konusu cihazlarda kullanılan antenlerden en önemlisi olan mikroşerit antenler yoğun ilgi görmektedir.

Bu çalışmada Selçuklu Yıldızı mikroşerit anten için yapay sinir ağları ve sezgisel optimizasyon algoritmaları kullanılarak uygun anten parametrelerle optimum tasarımı gerçekleştirilmesi hedeflenmiştir. Basit şekilli mikroşerit antenlerin tasarım parametrelerinin Meta-Sezgisel Algoritmalar yardımıyla bulunması amaçlanmaktadır. Selçuklu Yıldızı mikroşerit anten basit yapılı ve kolay üretimi ile geniş bant

uygulamalarında yer alacak olan mikroşerit antendir. Optimizasyon işlemi için mikroşerit antenlerde çokça tercih edilen Genetik algoritma, Parçacık Sürü Optimizasyonu, Guguk Kuşu Algoritması, Ateş Böceği Algoritması, Yabani Ot Algoritması, Balina Optimizasyonu ve Gri Kurt Algoritması kullanılması amaçlanmıştır. Ayrıca bu optimizasyon algoritmaları ile Yapay Sinir Ağları birleştirilerek hibrit bir problem çözüm mekanizması geliştirilmek istenmiştir.

Bu tez çalışmasında mikroşerit anten simülasyonu için ANSYS HFSS (High Frequency Structural Simulator) elektromanyetik yapıları sonlu elemanlar yöntemi ile çözen yazılımsal simülasyon aracı kullanılarak elde edilmesi amaçlanmıştır. Sonuçların YSA ve Meta-Sezgisel algoritmalar kullanılarak değerlendirilmesi MATLAB yazılımı aracılığıyla gerçekleştirilmiştir. Selçuklu Yıldızı mikroşerit antenlerin üretimi yapılmış ve ölçüm sonuçları değerlendirilmiştir (Uzer, 2016, Uzer, 2012).

## 2. KAYNAK ARAŞTIRMASI

Literatüre Uzer (Uzer ve Gultekin, 2012)(Uzer, 2016) tarafından kazandırılan Selçuklu yıldızı şeklinde yeni bir yıldız şekli mikro yama anten 5mm kalınlığında dairesel bir Köpük (Foam) tabakası şeklinde tasarlanmıştır. Tasarımlar HFSS simülatörü kullanılarak simüle edilmiştir. Antenin bant genişliği, radyasyon şekli ve geri dönüş kaybı gibi elektriksel parametreleri araştırılmıştır. Yamanın geometrik boyutları ile antenin elektriksel parametre özellikleri arasındaki ilişki araştırılmıştır. Bu yeni anten tasarımının farklı uygulamalar için kullanılabileceği ve yaygın bant genişliği geliştirme tekniklerinin bu tip yama antenine uyarlanabileceği sonucuna varılmıştır (Uzer ve Gultekin, 2012).

Diğer bir çalışmada ise, Selçuklu Yıldızı yama anten boyutlarının hedeflenen çalışma frekansı için belirlenmesi amacıyla temel bir formülasyon elde edilmiştir. Yama anten çalışma frekansı hesaplamalarında, openEMS kütüphanesi kullanılmıştır. OpenEMS, FDTD yöntemini kullanan serbest ve açık bir elektromanyetik alan çözücüdür. Oktav komut dosyası arayüzü olarak kullanılmıştır. Performans göstergesi olarak Göreli Kök Ortalama Kare Hatası (RMSE) kullanılmıştır. Sonuç olarak, önerilen formülasyon, belirli bir çalışma frekansı için Selçuklu Yıldız yama anteninin boyutlarını %2.08 doğrulukla hesapladığı görülmüştür. Önerilen ifade teorik altyapıdan yoksun olsa da, yama anten tasarımının ilk çalışmasında yararlı olduğu düşünülmektedir (Imeci ve ark., 2018).

Selçuklu yıldızı adı verilen yeni bir mikroşerit anten yama geometrisi önerilmiştir. Tasarlanan antenin performansı, literatürdeki diğer iki popüler yama geometrisi olan kare ve dairesel mikroşerit antenlerin performanslarıyla karşılaştırılmıştır. Dairesel ve karenin yama boyutları, Selçuklu yıldızıyla aynı olarak alınmıştır. Daha sonra antenlerin yama yüzey alanları, Selçuklu yıldız anteni ile eşleştirilmiş ve tasarımlar için yeni yama boyutları hesaplanmıştır. Anten performansları bant genişliği açısından karşılaştırılmıştır. Selçuklu Yıldızı şekilli mikroşerit antenin her rezonans frekansında kare ve daire şekilli antenlere oranla üstün performansa sahip olduğu görülmüştür. Ancak, WiFi uygulamaları için gerekli %2.5 bant genişliği kriterleri %1.48 bant genişliği sonucuyla eşleşememiştir. Bununla birlikte, ek bant genişliği geliştirme yöntemleri uygulayarak bant genişliğini daha da geliştirmenin mümkün olacağı belirtilmiştir. (Uzer ve ark., 2016)

Yapay Sinir Ağı (YSA) ve Bulanık Çıkarım Sistemi (BÇS) kombinasyonuna dayanan bir hibrit yöntem, çeşitli mikroşerit antenlerin (MSA'lar) düzenli geometrilerin rezonans frekanslarını eşzamanlı olarak hesaplamak için sunulmuştur. YSA' nın eğitimi, Bayesian düzenleme algoritması ile yapılmıştır. BÇS parametrelerini tanımlamak için en

küçük kareler yöntemini ve geri yayılım algoritmasını birleştiren bir algoritma kullanılmıştır. Dikdörtgen, dairesel ve üçgen MSA' lar için önerilen hibrit yöntemin rezonans frekans sonuçları, literatürde mevcut deneysel sonuçlarla çok iyi bir uyum içinde olduğu belirtilmiştir. Yüksek hızlı gerçek zamanlı hesaplama özelliğine sahip Hibrit yönteminin, anten bilgisayar destekli tasarım programlarında kullanılmasını önerilmiştir.(Guney ve ark., 2007)

Dikdörtgen yama anten tasarımı için gereken yama boyutlarını tahmin etmek için bir Yapay Sinir Ağı modeli geliştirilmiş ve test edilmiştir. Yapay sinir ağının performansı, IE3D EM Simulator' den elde edilen benzetilmiş değerlerle karşılaştırılmıştır. Yapay Sinir Ağı (YSA), dielektrik sabiti ($\varepsilon_r$), alt tabakanın (h) kalınlığını ve antenin baskın mod rezonans frekansını ($f_r$) içeren verileri, yama boyutlarına, yani, yamanın uzunluğuna (L) ve genişliğine (W) dönüştürür. Ağ modelini uygulamak için, Çok Katmanlı İleri Beslemeli Geri Besleme Yayılımlı Yapay Sinir Ağı ve Radyal Tabanlı Yapay Sinir Ağı geri yayılım eğitim algoritmasının farklı varyantları kullanılmıştır. Yapay sinir ağından elde edilen sonuçlar simülasyon sonuçlarıyla karşılaştırıldığında, tatmin edici bulunmuştur ve aynı zamanda Radyal Tabanlı Yapay Sinir Ağı' nın, Çok Katmanlı Yapay Sinir Ağı' nın farklı geri yayılma eğitim algoritmaları varyantlarına kıyasla daha doğru ve hızlı olduğu sonucuna varılmıştır. YSA' nın sonuçları simülasyon bulgularıyla daha uyumludur. Yapay sinir ağları temelli tahmin, tüm çıktıların çok hızlı ve aynı anda yanıt vermesinin olağan avantajına sahiptir.(Thakare ve ark., 2011)

PSO (Parçacık Sürüsü Optimizasyonu) algoritması ile optimize edilmiş Yapay Sinir Ağı kullanılarak mikroşerit yama anten tasarımını yapılmıştır. Yapay Sinir Ağının dikdörtgen şekilli mikroşerit antenin yama boyutlarını (genişlik ve uzunluk) tasarlama yeteneği incelenmiştir. Yapay sinir ağı tasarım sonuçlarına göre bir anten tasarlanmıştır. Hesaplanan ve sinir ağı sonuçları ölçülen anten değerleri ile karşılaştırılmıştır. Eğitimli sinir ağlarının elde edilen sonuçları, bu modelin deneysel olarak doğrulanan yama antenlerinin hızlı ve doğru tasarımında bir araç olarak kullanılabileceğini göstermektedir. Yapay sinir ağı modelleri, karmaşık ve uzun ömürlü hesaplamaları önlemek için anten tasarımı için alternatif bir teknik olarak kullanılabilir.(Vilović ve ark., 2013)

Koaksiyel beslemeli 2.4GHz E şekilli mikroşerit yama anten tasarımı tam dalga elektromanyetik simülatör tabanlı IE3D programı ile tasarlanmış ve simüle edilmiştir. Daha sonra aynı anten farklı yapay sinir ağı modelleri kullanılarak modellenmiş ve analiz edilmiştir. En iyi besleme noktasını bulmadaki zorluklardan dolayı hem Geri Beslemeli hem de Radyal Tabanlı Yapay Sinir Ağı modeli en iyi besleme noktasını bulmak için

tasarlanmıştır. Radyal Tabanlı Yapay Sinir Ağının Geri Beslemeli Yapay Sinir Ağı' na göre ortalama hata değerinin daha az olması ve parametre tahmininin gerçeğe daha yakın olmasından dolayı daha verimli olduğu sonucuna varılmıştır.(Singh ve ark., 2015)

Daire şekilli mikroşerit antenin geri dönüş kaybı (RL), gerilim durgun dalga oranı (VSWR), rezonans frekansı ($f_r$), bant genişliği (BW), kazanç (G), yönelticilik (D) anten verimliliği (η) gibi tüm performans parametrelerinin tahmini için nörobilişsel (NC) yaklaşım uygulanmıştır. Geri Yayılım Levenberg Marquardt algoritması ile eğitilmiş Yapay Sinir Ağı, Dairesel Mikro Şerit Anteninin farklı performans parametrelerinin tahmini için uygulanmıştır. Önerilen ANN modeli, daire şekilli mikroşerit antenin geniş kapsamlı giriş parametreleri için iyi bir doğruluğa ve geçerliliğe sahiptir. Bu YSA modelinin çok kısa hesaplama süreleri aldığı ve çok çeşitli uygulamalar için uygun olduğu belirtilmiştir (Sivia ve ark., 2016).

Mikroşerit anteninin modellenmesinde YSA kullanılmasının yanı sıra, yüksek kazanç ve optimum empedans eşleşmesi elde etmek için verimli anten tasarlamaya yönelik bir çalışma yapılmıştır. Önerilen eliptik yama anteni, kablosuz uygulamalar için kullanılan 2.4GHz' de çalışmaktadır. Yapay Sinir Ağı modelini eğitmek ve test etmek için CST EM simülatörü tarafından türetilen veri seti ile beslenir. İleri beslemeli geri yayılım YSA, anteni modellemek için Levenberg-Marquart (LM) öğrenme algoritması ile kullanılmıştır. Ortalama kare hata (MSE), ortalama hata ve standart sapma hatası gibi istatistiksel ölçütler kullanılarak verimli bir Yapay Sinir Ağı modeli sağlamak için kapsamlı analizler yapılmıştır. Ayrıca, önerilen anten üretilmiş ve ölçülmüştür. Simüle edilen ve ölçülen anten geri dönüş kaybı arasında yüksek bir uyum gösterilmektedir (Abbassi ve ark., 2019).

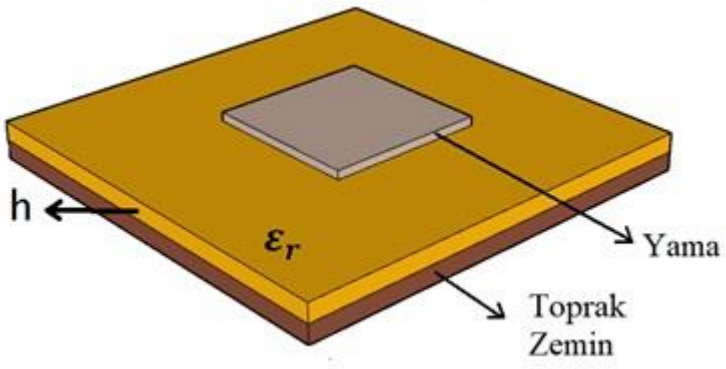
# 3. MATERYAL VE YÖNTEM

## 3.1. Mikroşerit Antenler

Mikroşerit antenler son yıllarda iyi özellikleri nedeniyle yaygın olarak haberleşme alanında kullanılmaktadır. Elektriksel olarak ince, hafif, düşük maliyetli, kullanıldığı yere uyumlu ve benzeri özellikleri mikroşerit antenlerin önemini artırmaktadır (Balanis, 2016). Hem teorik olarak hem de teknolojideki ilerlemeyle birlikte, mikroşerit antenlerin dezavantajlarının bazıları aşılmış ya da en azından bir dereceye kadar hafiflemiştir. Özellikle kişisel iletişim sistemlerinde (PCS), mobil uydu haberleşmelerinde, doğrudan yayınlarda (DBS), kablosuz yerel alan ağlarında (WLAN) ve akıllı araç karayolu sistemlerinde (IVHS) hızla gelişen pazarlar, mikroşerit antenler için talebin daha da artacağını göstermektedir. Bu nedenle, artan talep mikroşerit antenlerin iyileştirilmesi için yapılan bilimsel çalışmaları artırmıştır.

### 3.1.1. Mikroşerit Antenin Genel Özellikleri

En temel haliyle bir mikroşerit anten, tek tarafında bir iletken tabaka ve diğer tarafında topraklama yüzeyi bulunan bir yalıtkan katmandan oluşur. Çeşitli yama geometrisi ile yapılan mikroşerit anten konfigürasyonları, haberleşme sistemlerinde geniş bir alanda kullanılmaktadır. Temel mikroşerit yama anten yapısı Şekil 3.1' de gösterilmektedir.




**Şekil 3.1.** Temel Mikroşerit Anten Yapısı

Mikroşerit antenlerin analizinde ve tasarımında kullanılacak modelin antenler ilgili ışıma diyagramı, giriş empedansı, kazanç, bant genişliği, huzme genişliği, verimlilik, kayıplar ve kalite faktörü gibi parametreleri deney sonuçları ile uyumlu olacak şekilde belirlenir.

Mikroşerit anten, yama anten olarak da adlandırılır. Yama, antenin ışımasına izin verir. Mikroşerit antenlerin yama şekilleri kare, dikdörtgen, daire, halka ve daha birçok şekillerde üretilebilir. Son yıllarda kişisel ve hareketli haberleşme sistemlerinde kullanılan cihazlara olan ilginin artması, daha düşük maliyetle üretilebilen, küçük ve daha az yer kaplayan, bulunduğu yere uyum sağlayabilen antenlerin ön plana çıkmasına sebep olmuştur. Mikroşerit antenler de uygulama alanlarında bu gereksinimleri karşılayan anten tiplerinin öncüsüdür.

Mikroşerit yama antenler, alışılmış mikrodalga antenlere göre çeşitli avantajlara sahip olduğundan pek çok uygulaması 100MHz – 60GHz frekans aralığındadır. Kablosuz sistemler, uydu haberleşmesi, silahların otomatik ateşlenmesi, biyomedikal ışınlayıcı ve giyilebilir anten tasarımları vb. uygulamalarda mikroşerit antenler yaygın bir şekilde kullanılır. Mikroşerit antenler, geleneksel mikrodalga antenlerle kıyaslandığında birçok avantaj sağlamaktadır. Bunlardan bazıları şöyle sıralanabilir;

- Mikroşerit antenler küçük boyutludurlar.
- Baskı devre teknolojisi kullanılarak seri üretimlerinin yapılabilmesi düşük maliyet sağlar.
- Diğer mikrodalga devrelerle aynı katman üzerinde üretilebilirler.
- Hem dairesel hem de doğrusal polarizasyona izin verirler
- Kişisel mobil haberleşmede kullanılabilmeleri boyutları dolayısıyla mümkündür.
- Düşük ağırlığa sahip olması
- Yarım dalga boyunda ya da daha azında ayrılabilen geniş dizi oluşturmada kolaylık sağlaması

Mikroşerit antenlerin avantajlarının yanı sıra bazı önemli dezavantajları da bulunmaktadır:

- Dar bant genişliği
- Düşük kazanç
- Güç aktarabilme kapasitesinin düşüklüğü

Mikroşerit antenlerin haberleşme sistemlerinde daha çok kullanılmasının önündeki en önemli engel olan dar bant genişliğinin aşılabilmesi konusu en önemli çalışma alanlarından birisi haline gelmiştir. Son yıllardaki çalışmalar, mikroşerit antenlerle yüzde 70’ lere varan bant genişliklerine ulaşılmasını sağlamıştır (James ve ark., 1986; Kumar ve Ray, 2003). Bu gelişmeyi sağlayan en önemli değişiklik, kalın ve düşük

elektrik geçirgenliğine sahip yalıtkan katmanların kullanılması ve farklı şekillerde yamaların denenmesidir.

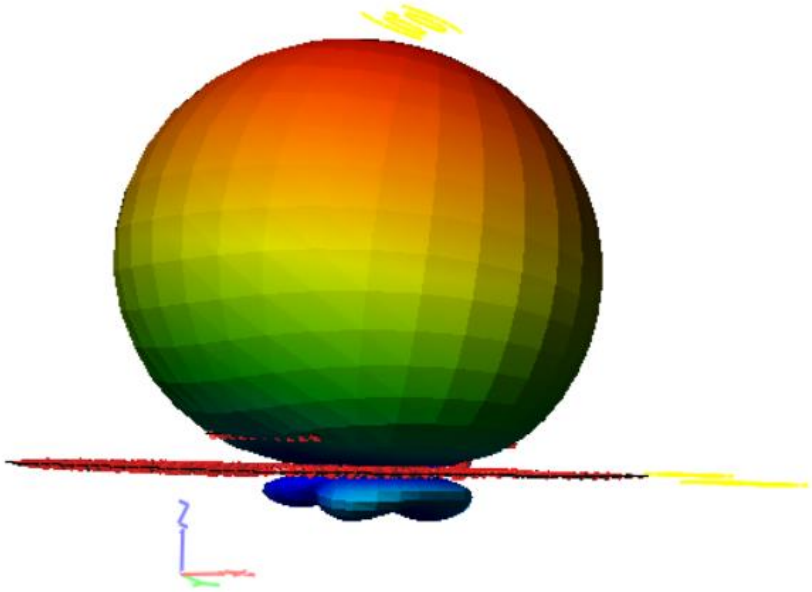
### 3.1.2. Mikroşerit Antenin Önemli Parametreleri

#### 3.1.2.1. Işıma Karakteristiği

Işıma özelliği, yaymak için bir güç anteninin veya gerekli gücün bir fonksiyonudur. Işıma karakteristiği, anten gücünün nasıl yönlendirildiğini ifade eder. Şekil 3.2' deki aşağıdaki çizim, yeniden yönlendirilebilen bir antenin genel ışıma özelliklerini göstermektedir. Bu sonuçlar şekilden elde edilebilir.

- Verici anten ana lobu, diğer lobdan daha büyük ve daha uzun olmalıdır.

Arka ve küçük loblar istenmeyen örneklerdir. Verici antenler açısından arka ve küçük loblar, harcanan enerjiyi ana loba aktarılmamış olarak ifade eder. İletim ortamından gelen gürültüyü temsil ederler. Şekil 3.2' de temel bir ışıma deseni örnek olarak verilmiştir.

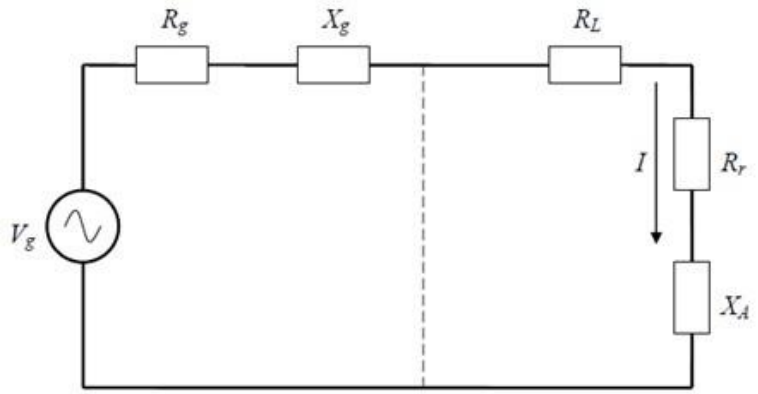


**Şekil 3.2.** Temel Anten Işıma Deseni

#### 3.1.2.2. Giriş Empedansı

Anten giriş empedansı, antenin bir terminalindeki empedans ya da bir çift terminaldeki voltaj akım oranı olarak tanımlanabilir. Alıcı anten uyarıldığında iç

empedanslı bir jeneratör gibi görünür. İletim durumunda bu iç empedans giriş empedansı ile aynıdır. Anten empedansının devre eşdeğeri Şekil 3.3' te görülmektedir.




**Şekil 3.3.** Anten Giriş Empedansının Devre Gösterimi

Anten empedansını veren voltaj akım oranını Denklem 3.1' de verilmiştir. Burada $R_A$ ifadesi empedansın reel kısmı, $X_A$ise sanal kısımdır. Reel kısım ise Denklem 3.2' deki formülde görüldüğü gibi $R_L$ kayıp direnci ve $R_r$ ışıma direncinde oluşur.

$$Z_R = R_A + jX_A \tag{3.1}$$

$$Z_R = R_r + R_L \tag{3.2}$$

Anten tasarımında, anten empedansının kaynağın empedansı ile uyumlu olması en verimli sonucu verir.

### 3.1.2.3. Geri Dönüş Kaybı

Geri dönüş kaybı, kaynaktan antene gönderilen gücün ne kadarının geri döndüğünün bir ölçüsüdür. Antenin en önemli parametrelerinden biri olan bu parametre antenle iletim hattı arasındaki empedans uyumunu ve gücün maksimum transferini göstermektedir. Eğer empedans uyumu yok ise tüm güç antene aktarılamayacak ve kayıp meydana gelecektir. Bu durumda yansımaların oluşması söz konusudur. Logaritmik skalada gösterildiği için birimi dB cinsinden gösterilir. Bir antenin belirli bir frekans bölgesinde çalışıyor olabilmesi için o frekans aralığında geri dönüş kaybı -9.95 dB' nin altında olmalıdır. Aşağıdaki denklemlerdeki gibi ifade edilir:

$$RL = -20\log_{10}|\Gamma| \tag{3.3}$$

$$RL = -20\ \log_{10}\left|\frac{z_L - z_0}{z_L + z_0}\right| \tag{3.4}$$

### 3.1.2.4. Voltaj Duran Dalga Oranı

İletim hattındaki maksimum gerilimin minimum gerilime oranı gerilim duran dalga oranını verir. Γ yansıma katsayısını $V_{max}$ ve $V_{min}$ ise maksimum ve minimum gerilim değerlerini ifade eder. Aşağıdaki eşitlikte gerilim duran dalga oranının ifadesi verilmiştir.

$$VSWR = \frac{V_{max}}{V_{min}} = \frac{1+|\Gamma|}{1-|\Gamma|} \tag{3.5}$$

Gerilim duran dalga oranı iletim hattının karakteristik empedansı ile anten giriş empedansının uyumunu gösterir. Yansıma katsayısının 1 veya -1 olduğu durumlarda VSWR sonsuza gider bu en istenmeyen durumdur. İstenen durumda VSWR, 1 değerini alır.

### 3.1.2.5. Yönelticilik ve Kazanç

Yönelticilik, antenin yönlendiriciliğinin maksimum olduğu değeri ifade eder. Yönelticilik girişe gelen sinyalin anten tarafından ne kadar yönlendirildiğini gösterir. Kayıpsız antenlerde anten kazancı yönelticiliğe eşittir. Aşağıdaki ifadede görüldüğü üzere yönelticilik D, ışıma yoğunluğu U' nun 4π katının toplam ışıma gücü $P_{rad}$' ne oranıdır. $U_0$ yönsüz kaynağın ışıma yoğunluğu.

$$D = \frac{U}{U_0} = \frac{4\pi U}{P_{rad}} \tag{3.6}$$

Yönün belirli olmadığı durumda maksimum ışıma yoğunluğuna bakılır. Aşağıdaki ifadede, $D_{max}$ maksimum yönelticilik, $Umax$ maksimum ışıma yoğunluğu olarak kullanılmıştır.

$$D_{max} = \frac{U_{max}}{U_0} = \frac{4\pi U_{max}}{P_{rad}} \tag{3.7}$$

Burada D = Yönlülük (Birimsiz),

U = Işıma şiddeti (W/Birim katı açı),

$U_0$= Yön bağımsız kaynağın ışıma şiddeti (W/Birim katı açı),

$P_{rad}$ = Toplam ışıyan güç (W)' tür.

Bir antenin ışıma şiddeti yoğunluğunun yönsüz bir antendekine olan oranı yönelticilik kazancı $D(\theta, \phi)$' dır. Aşağıdaki ifadede $U(\theta, \phi)$ anten ışıma şiddeti yoğunluğu, P antenin yaydığı toplam güçtür.

$$\mathrm{D}(\theta,\phi) = \frac{4\pi \mathrm{U}(\theta,\phi)}{P} \tag{3.7}$$

Kazanç bir antenin aldığı giriş gücünü belirli bir yönde ışımaya dönüştürme kabiliyetinin bir ölçüsüdür ve ışıma gücünü tepe yaptığı yerde ölçülür. Gerçek bir antenin kazancı ışımanın zirve yaptığı yöndeki güç yoğunluğunu arttırır. Kazanç bu açıdan antenin performansını değerlendirme önemli bir ölçüdür. Kazanç yönelticilikle ile ilgili olduğu kadar verimlilik ile de ilgilidir. Kazanç, ışıma şiddetinin $U(\theta, \phi)$ ve toplam giren gücün $P_{in}$ oranının $4\pi$ ile çarpımına eşittir.

$$G = 4\pi \frac{\mathrm{U}(\theta,\phi)}{P_{in}} \tag{3.8}$$

### 3.1.2.6. Verimlilik

Anten ile ilgili birden fazla verimlilik ölçütü belirlenebilir. Genel olarak anten verimliği, değişik verimlilik etkenlerinin birleşimi olarak tanımlanabilir. Toplam anten verimliliği $e_o$ antenin yapısından kaynaklanan kayıpları ve giriş terminalindeki kayıpları kullanır. Toplam verimlilik; yansıma verimliliği $e_r$, iletim verimliliği $e_c$ ve dielektrik verimliliği $e_d$ olmak üzere Denklem 3.9' daki ifadede verilmiştir.

$$e_0 = e_r e_c e_d \tag{3.9}$$

Burada $e_0$= Toplam verimlilik (Birimsiz),

$e_r$= Yansıma verimliliği (1-$|\Gamma|^2$)(Birimsiz)

$e_c$= İletim verimliliği (Birimsiz),

$e_d$= Dielektrik verimliliği (Birimsiz)'dir.

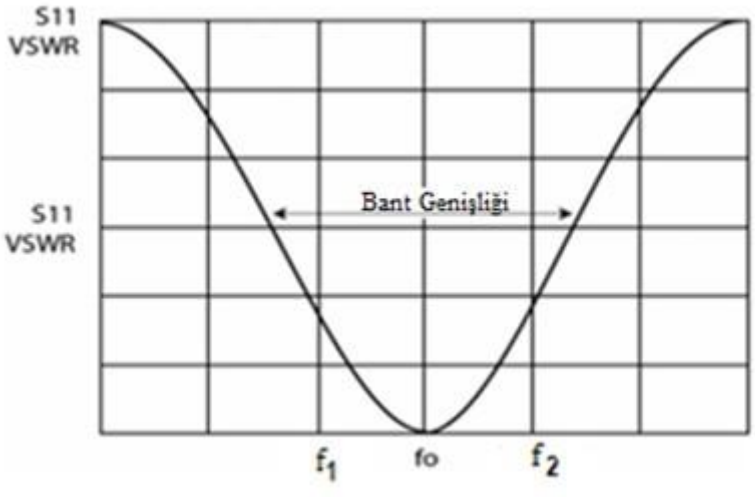
### 3.1.2.7. Bant Genişliği

Bir antenin bant genişliği, antenin çalışabileceği frekans aralığı anlamına gelir. Bir antenin bant genişliği, antenin performansının bazı özelliklere göre belirli bir standarda uyduğu frekans aralığı olarak tanımlanır (James ve ark., 1986). Diğer bir deyişle, bant genişliğinin benzersiz bir özelliği yoktur ve özellikleri her bir özel uygulamanın

ihtiyaçlarını karşılayacak şekilde ayarlanır. Anten bant genişliği standardı için farklı tanımlar vardır.

Geri dönüş kaybı, bir antenden yansıma ölçüsüdür. 0 dB, tüm gücün yansıtıldığı anlamına gelir; bu nedenle eşleşme iyi değildir. -10 dB, gelen gücün %10' unun yansıtıldığı anlamına gelir; yani gücün %90' ı anten tarafından kabul edilir. Dolayısıyla, bant genişliği referansı olarak -10 dB' ye sahip olmak, enerji kaybının %10 olduğu bir varsayımdır. Şekil 3.4' teki ifadeler dikkate alınarak, bant genişliği değeri aşağıdaki Denklem 3.10' daki gibi yüzde şeklinde hesaplanabilir;

$$Bant\ Genişliği = \frac{f_2 - f_1}{f_2 + f_1} x\ 100\% \quad (3.10)$$




**Şekil 3.4.** Bant Genişliği

### 3.1.3. Mikroşerit Antenin Tasarım Parametreleri

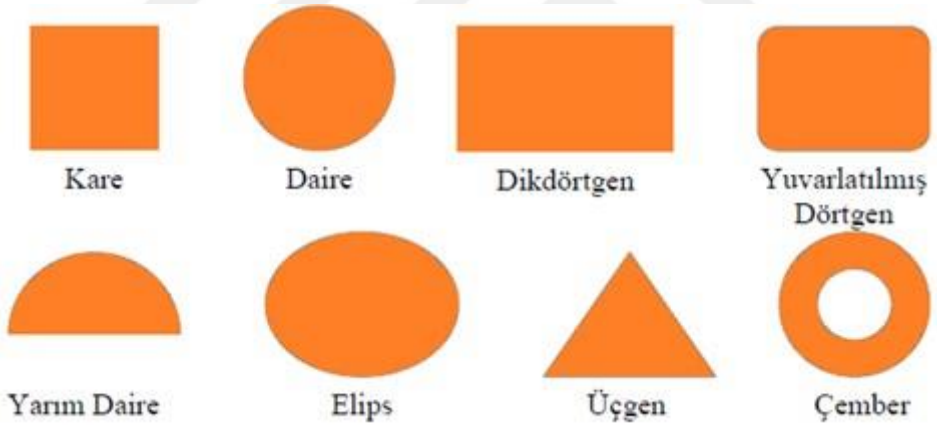
Anten tasarımlarının boyut parametrelerinin küçültülmesi ışıma performansının düşmesine ve bant genişliğinin azalmasına neden olmaktadır. Anten tasarımında yama şekli ve boyutunda yapılan değişiklikler sayesinde anten sonucunda iyileştirmeye olanak sağlamaktadır. Literatürdeki çalışmalar incelendiğinde, mikroşerit antenlerin performansını ve bant genişliğini etkileyen en önemli parametreler; empedans uyumu, anten geometrisi, dielektrik katsayısı ve taban malzemesinin kalınlığının olduğu görülmektedir (Zhang ve ark., 2016; Aoad ve Tuereli, 2018; Deshmukh ve ark., 2018; Luzon ve Gerasta, 2018; Sharma ve Sharma, 2018; Yohandri ve ark., 2018; Rathod ve ark., 2019).

**Anten Geometrisi**

Mikroşerit antenin yama geometrisi seçimi, uygun bir alt tabaka dielektiriği seçiminden sonra gerçekçi ve kullanışlı bir anten tasarlamak için en önemli faktörlerden biridir. Yamanın boyutları ve şekilleri antenin özelliklerinde en belirleyici etkenlerden biridir. Ancak şeklin değişmesi ışıma karakteristiğinde büyük değişiklikler getirmez.

Yama şekli belirlenirken kullanılacak yamanın boyutları ve bu boyutların anten performansı üzerindeki etkileri göz önünde bulundurulmalıdır.

Mikroşerit antenin geometrik şekli, anten performansını büyük ölçüde etkilemektedir. Mikroşerit antenlerin birçok şekilde tasarımları yapılmaktadır. Literatürde sunulan çalışmaların büyük bir kısmı analizlerinin ve tasarımlarının kolay olmasından dolayı dikdörtgen, üçgen ve daire gibi bilinen geometriler üzerinde yapılan çalışmalar bulunmaktadır. Ancak bu bilinen geometrilere sahip mikroşerit antenlerin boyutları, yüksek bant genişliği uygulamaları için göreceli olarak büyüktürler. Şekil 3.5' te literatürde en yaygın kullanılan anten geometrileri verilmiştir.




**Şekil 3.5** Yaygın Mikroşerit Anten Yama Şekilleri

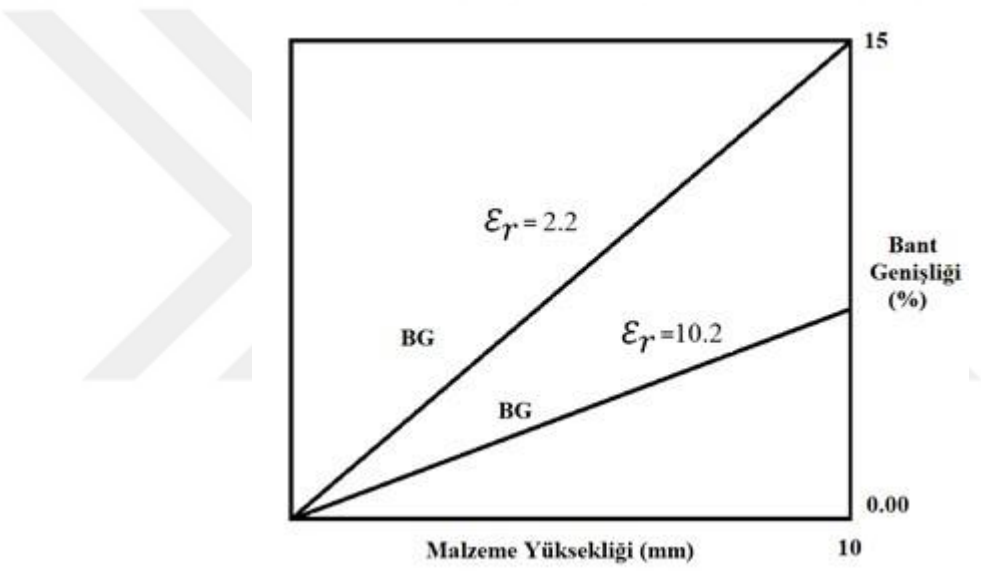
Bu tez çalışmasında kullanılan geleneksel yama şekillerine göre yama alanında küçültmeye imkân tanıyan yeni bir yama şekli olarak Selçuklu Yıldızı şekilli mikroşerit anten bant genişliğinde iyileştirmeye olanak sağlamıştır.

**Dielektrik Tabaka Kalınlığı**

Elektriksel olarak kalın bir dielektrik taban kullanımı, mikroşerit anten performansını önemli ölçüde artıracaktır. Dielektrik taban kalınlığı ile anten bant genişliği orantılıdır. Bir mikroşerit antenin taban malzemesinin görevi, temel olarak ışıma yapan yamaya mekanik destek sağlamak ve toprak düzlemi ile yama arasında gereken

boşluğu yaratmaktır. Yüksek dielektrik sabitine sahip taban malzemeleriyle, yükleme etkisine bağlı olarak yama boyutunda küçültmeye gidilebilir. Fakat küçülen anten hacmiyle birlikte, büyük dielektrik sabiti antenin bant genişliğini de azaltacaktır. Mikroşerit anten için taban malzemesi seçilirken istenilen yama boyutu, bant genişliği, ilave kayıplar, ısıl kararlılık, maliyet, vb. etmenlerin göz önünde bulundurulması gerekmektedir (Balanis, 2016).

Şekil 3.6' da görüldüğü üzere anten bant genişliğinin yüksek olması için dielektrik katsayının küçük değerler alınırken malzeme kalınlığının fazla olması sağlanmalıdır. Çalışma frekansının düşük olduğu anten tasarımlarında ise tabaka dielektrik katsayısının fazla olması antenin boyutlarının küçük olabilmesini sağlar.




**Şekil 3.6** Bant Genişliğindeki Dielektrik Katsayılar

### 3.1.4. Daire Şekilli Mikroşerit Antenler

Pratikte en çok kullanılan mikroşerit anten tiplerinden biri, anten dizilerinde diğer mikroşerit anten tiplerine göre daha fazla üstünlüklere sahip olan dairesel mikroşerit antendir.

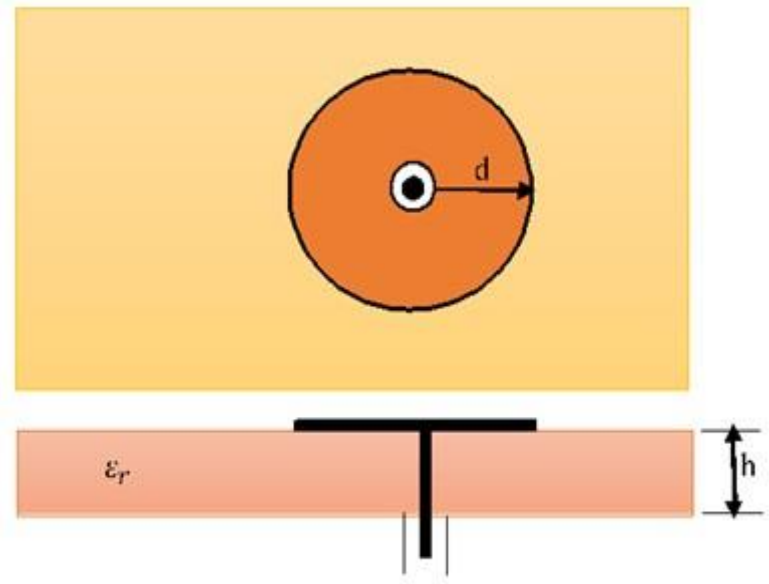
Dairesel mikroşerit antenlerin fiziksel parametrelerini değiştirerek antenin elektromanyetik karakteristiklerinin istenilen seviyelere getirilmesi mümkün olabilmektedir. Yama yarıçapı a ve dielektrik taban kalınlığı h olan bir dairesel mikroşerit anten geometrisi, Şekil 3.7' de gösterilmiştir.

Dairesel mikroşerit antenlerin fiziksel parametrelerini değiştirerek antenin elektromanyetik karakteristiklerinin istenilen seviyelere getirilmesi mümkün olabilmektedir.

Bu tip bir antende dielektrik katmanın kalınlığı h, dairesel iletken tabakanın çapı a ve antenin besleme noktası d değiştirilerek antenin istenen kazanca sahip olması, istenen frekansta ışıma yapması ve istenilen bant genişliğinde olması sağlanabilmektedir.

Dairesel mikroşerit antenin rezonans frekansı aşağıdaki Denklem 3.11 ile hesaplanabilir (Karaboga ve ark., 1999).

$$f_r = \frac{c\alpha_{nm}}{2\pi a_{eff}\sqrt{\varepsilon_r}} \tag{3.11}$$




**Şekil 3.7** Daire Şekilli Mikroşerit Anten Yapısı

Burada c elektromanyetik dalgaların boşluktaki yayılma hızı, $\mathcal{E}_r$ yalıtkan alt tabakanın dielektrik katsayısıdır. $\alpha_{nm}$, n. Dereceden Bessel fonksiyonunun m. Türevi olup mikroşerit antenlerde yaygın olarak kullanılan bir $TM_{nm}$ modu olan $TM_{11}$ için bu değer 1.84118' dir. $a_{eff}$ ise dairesel iletken yüzeyin etkin yarıçapı olup Denklem 3.12' de hesaplanır.

$$a_{eff} = \frac{\pi a^2}{4} + \left[0.13 + \left(\frac{3.64}{{\varepsilon_r}^{2.6}}\right)\right] h + \left(\frac{0.22}{a}\right) h^2 \tag{3.12}$$

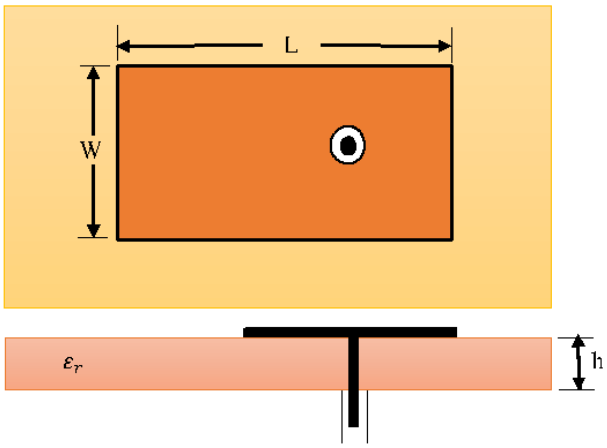
### 3.1.5. Dikdörtgen Şekilli Mikroşerit Antenler

Dikdörtgen şekilli mikroşerit antenler, basit bir geometriye sahip olması nedeniyle modellenmesi, analizi ve parametrelerinin hesaplanması nispeten daha kolay olduğundan dolayı mikroşerit antenler konfigürasyonları içerisinde çok tercih edilen bir anten çeşididir.

Dikdörtgen şekilli mikroşerit anten yapısı Şekil 3.8' de gösterilmiştir. Genişliği (W), uzunluğu (L) olan dikdörtgen yama ve h kalınlığında bir dielektrik malzemeden yapılan alt tabakadan oluşur.

Bir dikdörtgen mikroşerit antenin rezonans frekansı Denklem 3.13' teki gibi hesaplanır (Kara ve Letters, 1996).

$$f_r = \frac{c}{2(L+\Delta W)\sqrt{\varepsilon_{eff}(W)}} \tag{3.13}$$




**Şekil 3.8** Dikdörtgen Şekilli Mikroşerit Anten Yapısı

Bu denklemde c elektromanyetik dalgaların boşluktaki hızı, L iletken tabakanın uzunluğu, $\Delta W$ iletken tabakanın bir fonksiyonu, $\varepsilon_{eff}(W)$ ise dielektrik alt tabakanın etkin elektrik geçirgenliğidir. $\Delta W$ ve $\varepsilon_{eff}(W)$ aşağıdaki formüllerle hesaplanabilir.

$$\Delta W = \frac{[\varepsilon_e(W)+0.300](\frac{W}{h}+0.264)}{[\varepsilon_e(W)-0.258](\frac{W}{h}+0.813)} \tag{3.14}$$

$$\varepsilon_{eff}(W) = \frac{\varepsilon_r+1}{2} + \frac{\varepsilon_r-1}{2(1+\frac{10}{\mu})^{1/2}} \tag{3.15}$$

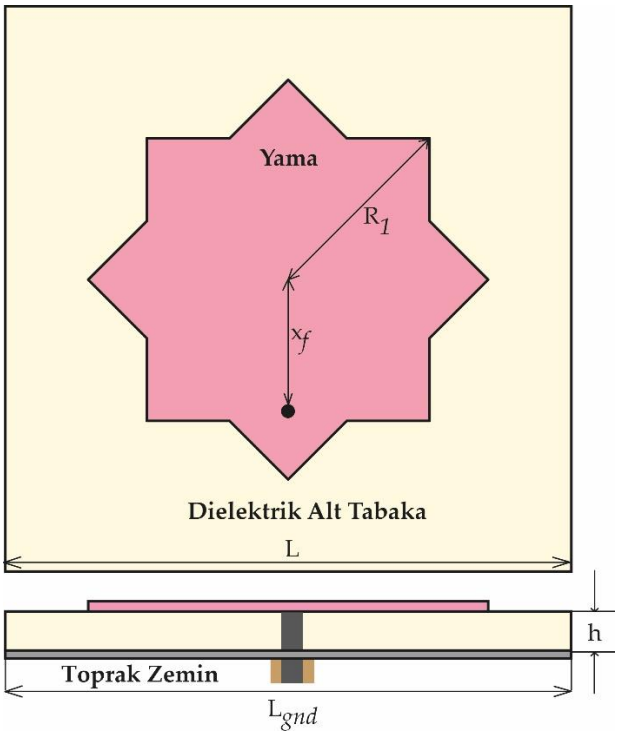
### 3.1.6. Selçuklu Yıldızı Şekilli Mikroşerit Antenler

Selçuklu Yıldızı mikroşerit anten, mikroşerit antenlerin bant genişliğini artırmak amacıyla son yıllarda literatürde performansı değerlendirilen bir anten çeşididir. Selçuklu yıldızı, eşit boyutlu iki karenin birbirine $45^{\circ}$ derece farkla yerleştirilip 8 köşeli yıldız oluşturularak kolayca elde edilebilir.

Geleneksel ve literatürde mevcut olan mikroşerit yama anten şekilleri ile Selçuklu Yıldızı olarak isimlendirilen geometrik şekle sahip bir mikroşerit yama anten tasarımı bant genişliği ve diğer anten parametreleri üzerindeki etkileri karşılaştırmalı olarak incelenmiştir (Uzer, 2012 ).

Selçuklu Yıldızı mikroşerit anten için tasarım parametrelerini belirlemek adına bir tasarım formülü elde edilememiştir. Basit yapısı ve kolay üretimi ile rahatlıkla geniş band uygulamalarında yer alacak olan bu yeni yama anten şekli, kendine has formüller yardımıyla tasarlandığı frekansta yüksek band genişliğiyle beraber çalışabilecek hale getirilecektir (Uzer, 2016).

Bu tez çalışmasında; Yapay Sinir Ağları ve optimizasyon algoritmaları kullanılarak Selçuklu Yıldızı şekilli mikroşerit antenin optimum tasarımı gerçekleştirilecektir. Elde edilen parametreler yardımıyla antenin benzetimi ve sonrasında uygulaması yapılarak ölçümleri tamamlanmıştır.




**Şekil 3.9** Selçuk Yıldızı Mikroşerit Anten Yapısı

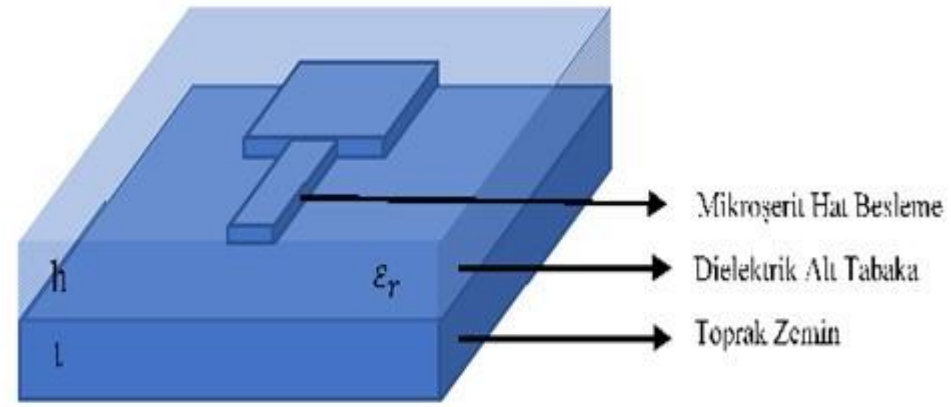
### 3.1.7. Mikroşerit Antenlerde Besleme Şekilleri

Bu bölüm literatürde en çok kullanılan besleme tekniklerini içermektedir. Genel olarak, besleme tekniğinin adı ne olursa olsun, temel amaç empedansla uyumluluğu sağlamaktır. Empedansla uyum sağlanması, iletilen gücün büyük bir kısmının kaynaktan metalik yamaya geçmesini garanti eder. Öte yandan, bir anten uyumluluğu için bir besleme tekniğine odaklanmak da çok önemlidir.

#### 3.1.7.1 Mikroşerit Hat Besleme

Mikroşerit beslemede güç, Şekil 3.10' da gösterilen ince bir iletken yardımıyla yamaya aktarılır. Bu tekniğin ana avantajı, düzlemsel bir yapı oluşumu durumunda yamayı aynı metalik tabaka ile beslemektir. Bu durum, yüzey dalgasını azaltan bir durumdur. Antene kolayca uyarlanabilir. Metalik yama, besleme şeridinin bir devamıdır. Bununla birlikte, kullanılan dielektrik alt tabakanın kalınlığı arttıkça, yüzey dalgaları ve yanıltıcı besleme ışıması da artar, bu da antenin bant genişliğini etkiler. Besleme ışıması ayrıca istenmeyen çapraz kutuplama ışımasına yol açar.

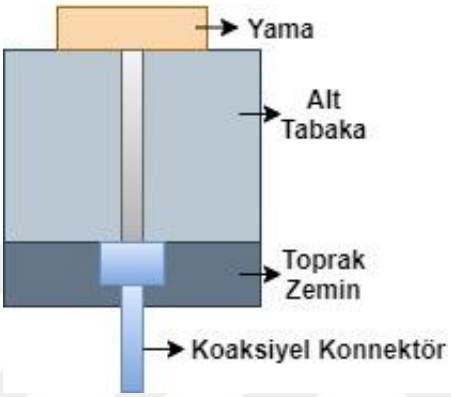
Mikroşerit hat besleme, yamayı bağlayan sadece iletken bir şerit olduğu ve bu nedenle yamanın uzantısı olarak düşünülebildiği için üretilmesi daha kolay yöntemlerden biridir. Yamayla besleme hattının eklendiği konumun kontrol edilerek modellemesi basit ve eşleştirilmesi kolaydır. Bununla birlikte, bu yöntemin dezavantajı, alt tabaka kalınlığının artmasıyla bant genişliğini sınırlayan yüzey dalgası ve sahte besleme ışımasının artmasıdır.




**Şekil 3.10** Mikroşerit Hat Besleme

### 3.1.7.2. Koaksiyel Hat Besleme

Bu besleme şeklinde, koaksiyel kablo veya konektörün iç iletkeni yamaya bağlayarak beslenir ve dış iletken toprağa bağlanır. Bu yapıyı elde etmek için zemin düzleminde ve dielektrik alt tabakada bir delik açılır. Koaksiyel kablonun iç iletkeni bu delikten geçirilerek yamaya bağlanır. Dış iletken bu delikten geçmeden toprağa bağlanır (Bisht ve ark., 2014). Temel koaksiyel hat beslemesi Şekil 3.11' de verilmiştir.




**Şekil 3.11** Koaksiyel Hat Besleme

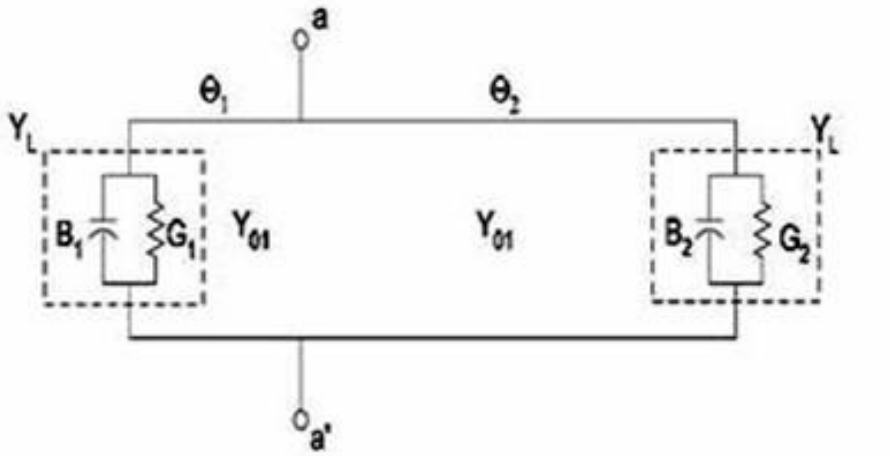
Yama içinde istenen konuma yerleştirilen prob ile empedans uyumu sağlanabilir. Bu yapıyı düşük kalınlıktaki dielektrik alt tabaka kullanarak tasarlamak ve üretmek daha kolaydır. Daha kalın yapılarda prob uzunluğu da artacağı için giriş empedansı endüktif hale gelir ve empedans uyumlandırma sorunları ortaya çıkar. Bu nedenle bu yapının tasarımı ve imalatı zorlaşır. Bu yöntem dar bant genişliğine neden olsa da ışımadaki parazit düşüktür.

### 3.1.8. Mikroşerit Antenler için Analitik ve Sayısal Modeller

Anten analizinin amacı, hem kutuplanma, kazanç ve ışıma diyagramı gibi ışıma karakteristiklerini, hem de verim, ortak kuplaj, empedans band genişliği ve giriş empedansı gibi yakın alan karakteristiklerini tahmin etmektir. Mikroşerit antenlerin analizi ise dielektrik homojensizliği, homojen olmayan sınır şartları, dar frekans bandı karakteristikleri ile çok farklı besleme, yama şekli ve dielektrik taban konfigürasyonlarının varlığı sebebiyle oldukça karmaşıktır. Bu sebeple, seçilen yöntemin karmaşıklığı ile çözümün doğruluğu arasında bir denge sağlanmalıdır.

### 3.1.8.1. İletim (Transmisyon) Hattı Modeli

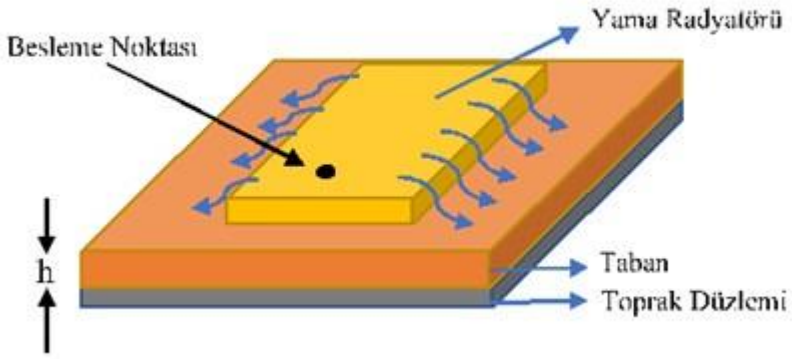
İletim hattı modeli, diğer yöntemlere kıyasla daha basittir ve iyi bir fiziksel bakış açısı sağlar. İletim hattı modelinin eksik yanı, doğruluğunun az olmasıdır. Diğer modellere kıyasla daha az karmaşık olması nedeniyle en çok tercih edilen yöntemlerden biridir. Şekil 3.12' de bu modele ait yapı görülmektedir.




**Şekil 3.12** Mikroşerit Antenin İletim Hattı Modeli

*Saçılma Etkileri:*

Yama boyutlarının uzunluk ve genişlik olarak sonlu olması dolayısıyla, yama kenarlarındaki alanlar dielektriğin dışında oluşan ve etkin dielektrik sabitinde değişikliğe yol açan alanlar gibi saçılmaya uğrar. Bu da yama boyutlarının ve taban kalınlığının bir fonksiyonudur.




**Şekil 3.13** Mikroşerit Antenin Saçılma Etkileri

Yukarıdaki Şekil 3.13 yama antenin iletim hattı modelinin yandan görünüşünü göstermektedir. Etkin uzunluğu artırarak kenarlardaki saçılmalar gözlemlenebilir.

$$\varepsilon_{eff} = \frac{\varepsilon_r+1}{2} + \frac{\varepsilon_r-1}{2}\left[1 + 10\frac{h}{w}\right]^{-1/2} \quad (3.16)$$

*Etkin Uzunluk ve Genişlik:*

Saçılma etkisinde bağlı olarak, elektriksel yama boyutları fiziksel boyutlardan daha büyük olacaktır. Etkin uzunluk $L_{eff}$' i hesaplamak için aşağıdaki formül kullanılabilir:

$$L_{eff} = L + 2\Delta L \quad (3.17)$$

Burada, normalize edilmiş uzunluk uzantısı $\Delta L$ için yaklaşık bir ilişki aşağıdaki gibi verilebilir:

$$\frac{\Delta L}{h} = 0.412\frac{(\varepsilon_{eff}+0.3)(W/h+0.264)}{(\varepsilon_{eff}-0.258)(W/h+0.8)} \quad (3.18)$$

Verimli bir radyatör için, iyi bir ışıma verimliliği sağlayacak uygulanabilir genişlik ile tanımlanabilir.

$$W = \frac{c}{2f_r}\sqrt{\frac{2}{\varepsilon_r+1}} \quad (3.19)$$

*İletkenlik:*

Her ışıyan yarık paralel eşdeğer Y admitansı (G iletkenliği ve B suseptansıyla beraber) ile temsil edilir. Yarıklar $\Theta_1$ ve $\Theta_2$ olarak adlandırılır. Yarığın eşdeğer admitansı aşağıdaki gibi verilir:

$$Y_1 = G_1 + B_1 \quad (3.20)$$

Sonlu (W) genişliğindeki yarık için

$$G_1 = \frac{W}{120\lambda_0}\left[1 - \frac{1}{24}(k_0 h)^2\right] \quad (3.21)$$

$$B_1 = \frac{W}{120\lambda_0}[1 - 0.636\ln(k_0 h)] \quad (3.22)$$

$\frac{h}{\lambda_0} < \frac{1}{10}$ için bulunabilir.

Her iki yarık da ideal olduğu için, eşdeğer admitansı,

$$Y_1 = Y_2 \quad (3.23)$$

$$G_1 = G_2 \quad (3.24)$$

$$B_1 = B_2 \quad (3.25)$$

Her yarığın iletkenliği rezonatör modelindeki alan ifadesi kullanılarak elde edilebilir. Genel olarak, iletkenlik şeklinde tanımlanır. İletim hattı modelinden elde edilen tüm bu denklemler antenin sentezi ve analizinde anten parametrelerinin hesabında kullanılabilir. Bu amaçla, modeller birbiriyle bağlantılı olarak kullanılmaktadır.

$$G_1 = \frac{2P_{rad}}{|V_0|^2} \tag{3.26}$$

### 3.1.8.2. Rezonatör Modeli

Mikroşerit yama antenler dar bantlı rezonans antenlerdir ve kayıplı rezonatörler olarak nitelenirler. Bu sebeple, rezonatör model yama antenleri analiz etmek için doğal bir tercih haline gelir. Rezonatör modeli Lo ve arkadaşları tarafından geliştirilmiştir. Bu modelde yamanın iç bölgesi, alttan ve üstten elektrik duvarlarla çevresi boyunca manyetik duvarla sınırlandırılmış bir rezonatör olarak modellenir (Bayraktar ve ark., 2019). Bu yaklaşımın temelleri ince tabanlar (h $<<\lambda_0$) için aşağıda verilen gözlemlere dayanır (Lo ve ark., 1979; Lo ve Richards, 1981).

- İç bölgedeki alanlar z ile değişmez (yani $\partial/\partial z \equiv 0$) çünkü taban çok incedir (h $<<\lambda_0$).
- Elektrik alan sadece z yönünde olup, manyetik alan yama iletkeni ve toprak düzlemi tarafından sınırlanan bölgede sadece enine bileşenlere sahiptir. Bu gözlem altta ve üstte elektrik duvarlar için sağlanır.
- Yamadaki elektrik akımı, yama iletkeninin kenarına dik bileşene sahip değildir. Bu durum H manyetik alanın kenar boyunca olan teğet bileşeninin ihmal edilebileceğini gösterir. Böylece çevre boyunca bir manyetik duvar yerleştirilebilir. Matematiksel olarak $\partial E_Z/\partial n = 0$ dır.

Rezonatör modeli, (dikdörtgen, dairesel ve üçgen dahil) hemen hemen bütün düzenli yama şekillerine uygundur. Bu model aynı zamanda, halka şeklindeki dairesel yamalara başarılı bir şekilde uygulanmıştır (Kumar ve ark., 1984; Bhattacharyya ve ark., 1985).

## 3.2. Yapay Sinir Ağları

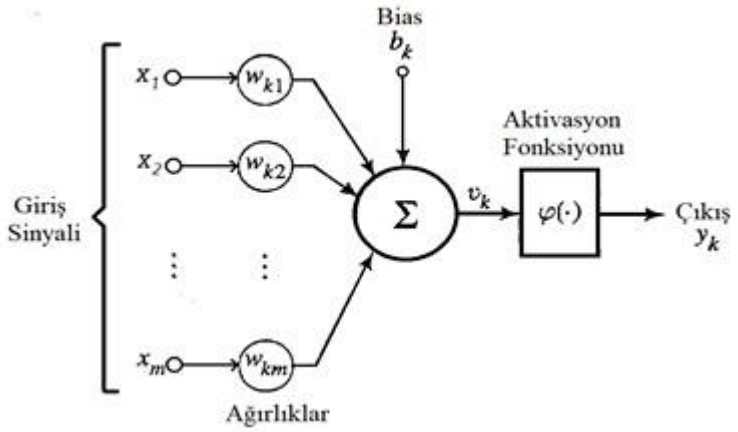
Bilgisayar teknolojisinin gelişmesi ile bilim adamları, insanlarda bulunan zekayı bilgisayarlara yerleştirmeye çalıştılar. Görsel algılama, konuşma tanıma, karar verme ve diller arasında çeviri gibi normalde insan zekâsı gerektiren görevleri yerine getirebilen bilgisayar sistemlerinin teorisi ve gelişimi ile yapay zekâ uygulamaları yaygın bir şekilde

kullanılmaktadır. Son yıllarda bu yöntem karmaşık hesaplamalardan kurtulmak için özellikle mikroşerit anten uygulamalarında verileri tahmin etmek için etkili bir yöntemdir.

Yapay Sinir Ağı, canlılara özgü sinir ağları sisteminden ilham alan bilinen hesaplama yöntemlerinden farklı bir hesaplama yöntemi öneren yapay zekâ yöntemidir. Yapay sinir ağları bir programcının geleneksel yeteneklerini gerektirmeyen, kendi kendine öğrenme düzenekleridir. Dolayısıyla, Yapay Sinir Ağları veya (YSA), makine öğrenimi uygulamalarında en popüler terimlerdir. YSA, verileri sınıflandırmak, tahmin etmek ve kümelemek için aynı biyolojik beyin hücreleri kavramını kullanır. Bu tür bir makine öğrenimi, çoğunlukla olay/problem hakkında öğrenerek gelecekteki olayları veya sorunları tahmin etmek için kullanılır.

Sinir ağları ileri teknolojidir ve birçok farklı bilim ve mühendislik alanında çok faydalıdır. Sinir ağları, yapay zekanın daha da geliştirilmesini temsil eder ve hızları, doğrulukları ve verimlilikleriyle bilinir. Adından da anlaşılacağı gibi sinirlerin işlevi, insan beynindeki sinir hücrelerinin işlevi ile aynıdır. Biyolojik olarak konuşursak, sinir hücreleri düzenli sinirleri aktive eder ve daha sonra bunları nöronlar yoluyla daha yüksek sıradaki hücrelere geçirir. Bir nöron bir komut aldığında, komut son aşamaya veya son hücreye ulaşana kadar durum değiştikçe güncellenir.

1. Her biri kendi ağırlığı veya gücü ile karakterize edilen bir dizi sinaps veya bağlantı halkası. Özellikle k nöronuna ile bağlı j sinapsının girişindeki $x_j$ sinyali, sinaps ağırlığı olan $w_{kj}$ ile çarpılır. $w_{kj}$' deki ilk alt indis k söz konusu nöronu ifade eder. İkinci alt indis ağırlığın ilgili olduğu sinapsın giriş ucunu belirtir. Beyindeki bir sinapsın ağırlığından farklı olarak, bir yapay nöronun sinaptik ağırlığı hem negatif hem de pozitif değerler içeren bir aralıkta olabilir.
2. Giriş sinyallerini toplamak için nöronun ilgili sinaptik güçleri ile ağırlık değerleri atanan bir toplayıcı; burada açıklanan işlemler doğrusal bir birleştirici oluşturur.
3. Bir nöronun çıktısının genliğini sınırlamak için bir aktivasyon işlevi. Aktivasyon fonksiyonu, çıkış sinyalinin izin verilen genlik aralığını bazı sonlu değerlere sıkıştırması (sınırlandırması) açısından bir sıkıştırma fonksiyonu olarak da adlandırılır (Haykin, 2009).




**Şekil 3.14** Temel Yapay Sinir Ağı Yapısı

Beyindeki sinir hücrelerinin işlevinden esinlenen YSA yapısı Şekil 3.14' de gösterilmiştir. Bu doğrusal girdi kombinasyonunun çıktısına ağırlıklı toplam denir. Burada $y_k$ aşağıdaki Denklem 3.27' deki gibi hesaplanır.

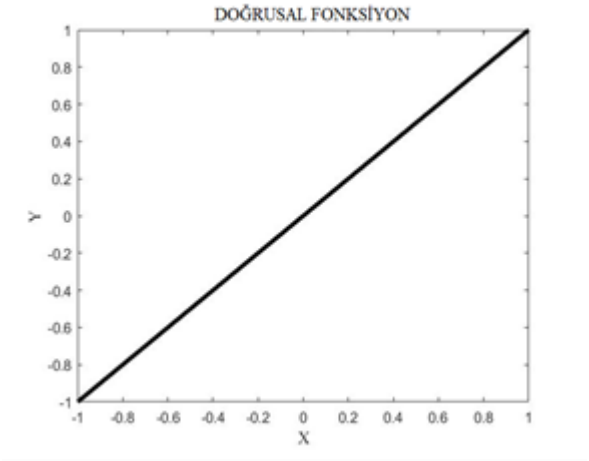
$$y_k = \sum_{i=t}^{n} w_{km} x_i + b \tag{3.27}$$

Nöronun ikinci kısmında, ağırlıklı toplamlara aktivasyon fonksiyonu adı verilen doğrusal olmayan bir fonksiyon uygulanır. Burada probleme uygun olarak türevinin eğitimde kullanılması için sorunsuz bir aktivasyon fonksiyonu seçilmelidir.

Aktivasyon fonksiyonu olmayan bir sinir ağı, sınırlı öğrenme gücüne sahip doğrusal bir regresyon gibi davranacaktır. Bu nedenle literatürde YSA ile yapılan çalışmalarda çeşitli aktivasyon fonksiyonları kullanılmaktadır. Çalışmada kullanılan aktivasyon fonksiyonu olarak, gizli ve çıktı katmanlarının transfer fonksiyonu aşağıdaki gibi doğrusal (lineer) fonksiyon, sigmoid fonksiyon ve hiperbolik tanjant sigmoid transfer fonksiyonu kullanılmıştır:

Doğrusal fonksiyonda, girdi aynı şekilde çıktı olarak üretilir. Bu fonksiyon doğrusaldır, türevlenebilir, üst ve alt sınırları yoktur, tekdüze artan ve azalan, başlangıç noktasında yakınsaktır. Aşağıdaki Şekil 3.15 doğrusal fonksiyonunu gösterir ve bu fonksiyonun matematiksel denklemi Denklem 3.28' de gösterilebilir.

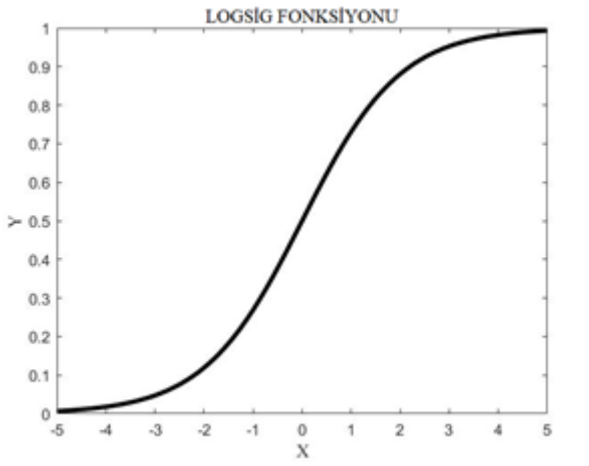
$$x=y \tag{3.28}$$




**Şekil 3.15** Doğrusal Fonksiyon

En çok uygulanan aktivasyon fonksiyonlarından biri sigmoid fonksiyonudur. Sigmoid işlevi doğrusal değildir, ancak hem doğrusal hem de doğrusal olmayan işlevler için dengeli çıktılar üreterek modelleme yapılabilir, türevlenebilir, alt sınırı, üst sınırı, monoton artan ve azalan işlevi vardır. Aşağıdaki Şekil 3.16' da sigmoid fonksiyonu görülebilir. Denklem 3.29, sigmoid fonksiyonunun matematiksel bir temsilidir.

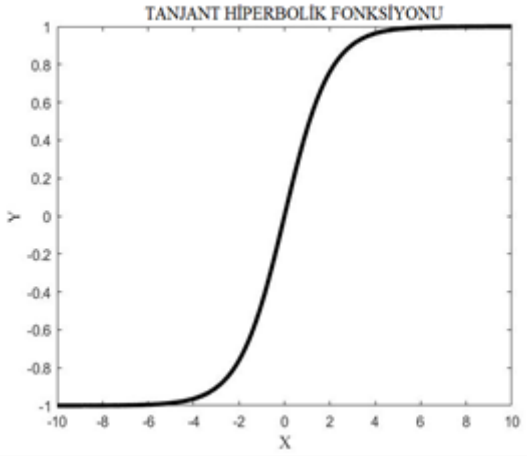
$$f(x) = logsig(x) = \frac{1}{1+e^{-x}} \tag{3.29}$$




**Şekil 3.16** Sigmoid Fonksiyonu

Tanjant hiperbolik fonksiyonu, YSA' da en sık kullanılan başka bir aktivasyon fonksiyonudur. Bu işlevi kullanabilmek için giriş değerleri 1 ile -1 aralığında normalize edilir. Aşağıdaki Şekil 3.17' de hiperbolik tanjant (Tansig) fonksiyonu görülebilir. Denklem 3.28, tansig fonksiyonunun matematiksel bir temsilidir.

$$f(x) = tansig(x) = \frac{2}{(1+exp\,(-2*x))} - 1 \tag{3.28}$$




**Şekil 3.17** Tanjant Hiperbolik Fonksiyonu

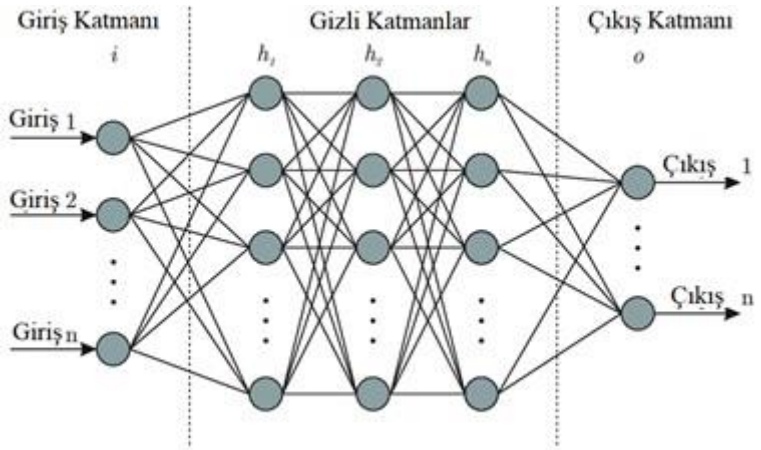
Son aşamada, YSA yapısına göre bir nöronun çıktısı ya başka bir nörona ya da YSA modelinin çıktı nöronuna iletilir.

### 3.2.1. İleri Beslemeli Yapay Sinir Ağı

En popüler Yapay Sinir Ağları türü, Çok Katmanlı İleri Beslemeli Sinir Ağlarıdır (ÇK-İBSA). Doğrusal olmayan aktivasyon işlevine sahip birçok nöronun mimari hiyerarşik bir şekilde birbirine bağlandığı bir yapıya sahip olan çok katmanlı sinir ağları, Yapay Sinir Ağlarının özel bir şeklidir. Bilgi, ÇK-İBSA' larda yalnızca ileri yönde akar. İleri beslemeli sinir ağında, hücreler katmanlar halinde düzenlenir ve bir katmandaki hücrelerin çıktıları bir sonraki katmana ağırlıklarla girilir. Giriş katmanı, dış ortamlardan aldığı bilgileri herhangi bir değişiklik yapmadan ara (gizli) katmandaki hücrelere iletir. Ağ çıktısı, ara ve çıktı katmanlarında bilgi işlenerek belirlenir. Bu yapı ile ileri beslemeli ağlar, doğrusal olmayan statik bir işlev gerçekleştirir. İleri beslemeli 3 katmanlı bir yapay sinir ağının, orta katmanda yeterli hücre olması koşuluyla herhangi bir sürekli işlevi istenen doğrulukla yaklaştırabileceği gösterilmiştir. Bu tür yapay sinir ağlarının eğitiminde en çok bilinen algoritma olan geri yayılım öğrenme algoritması etkin bir şekilde kullanılmaktadır.

Bu nedenle, eğitim temelde bir optimizasyon problemidir ve doğrusal olmayan bir en küçük kare minimizasyon yöntemi kullanılarak basitçe çözülebilir. Geri yayılım, gradyantların nasıl hesaplanacağını anlatır, böylece daha sonra doğrusal olmayan küçültme yöntemleri, bazı başlangıç değerleriyle ağırlıkları düzeltmek için uygulanabilir.

Şekil 3.18, İleri Beslemeli Sinir Ağının topolojisini göstermektedir; girdi, gizli ve çıktı katmanı olmak üzere üç katmandan oluşur. Katmanlar birkaç düğümden oluşur; nöronlar bu katmanları birbirine bağlamak için kullanılır.




**Şekil 3.18** İleri Beslemeli Yapay Sinir Ağı

### 3.2.2. Geri Yayılım Algoritması

Geri yayılım algoritması, hata gradyantlarını hesaplamak için sistematik olarak bir dizi zincir kuralı uygular. Geri yayılım ağları iki adımda çalışır. İlk aşamada bilgi ileri yönde yayılır, giriş katmanı tarafından alınır, gizli nöronlarda işlenir ve çıkış katmanına aktarılır. Daha sonra üretilen çıktı ve hedef değerlerden hata hesaplanır. Hata yeterince küçük değilse, her bağlantının ağırlıklarını bir optimizasyon algoritması ile değiştirerek çıktı katmanından giriş katmanına geri yayılır. Bu yineleme, eğitim hatası izin verilen aralığa düşene kadar devam edecektir(He ve Xu, 2010).

Geri yayılım, "hataların geriye doğru yayılması" teriminin kısaltılmış bir versiyonudur. Bu yöntemde YSA bir eğitim veri kümesi kullanılarak öğretilmelidir. Eğitim veri kümesi, eşleşen bir hedefle (istenen çıktı) ayrılan girdi değerlerinden oluşur. Ağın eğitimi, yinelemeli bir prosedürle gerçekleştirilir. Ağırlıklar, eğitim veri setlerinin yeni verilerinin kullanılmasıyla her yinelemede benimsenir. Her öğretme adımı, eğitim setinden giriş sinyalleri uygulanarak başlatılır. Bu adımdan sonra her katmandaki her bir nöronun çıktı değerleri belirlenebilir. Ardından, sonraki adımda, ağın çıktı değeri, eğitim veri setinde bulunan istenen çıktı değeri ile karşılaştırılır. Ağın çıktısı ile tercih edilen değer arasındaki farka hata değeri denir.

Geri yayılım fikri, hata değerini, incelenen nöron için çıktı değerlerinin girildiği tüm nöronlara geri yaymaktır. Hataları geri yaymak için kullanılan ağırlıklar, çıktı değerini hesaplarken kullanılanlarla aynıdır. Değişen sadece veri akışının yönüdür.

### 3.3. Meta-Sezgisel Optimizasyon Algoritmaları

Optimizasyon, mühendislik tasarımlarından iş planlamasına kadar birçok disiplinde önemlidir. Bu tür optimizasyon problemlerinin çoğu, çözülmesi için sofistike optimizasyon araçları gerektirir ve literatürde geleneksel gradyant tabanlı algoritmalardan ve basit yöntemlerden evrimsel algoritmalara ve doğadan ilham alan Meta-Sezgisel Algoritmalara kadar çeşitli algoritmalar kullanılmaktadır. Son yıllarda, doğadan ilham alan algoritmalar, oldukça doğrusal olmayan problemler ve zorlu optimizasyon problemleriyle başa çıkmak için yaygın olarak kullanılmaktadır. Doğadan esinlenen bu tür algoritmaların çoğu, doğal sistemlerdeki çeşitli özellikleri taklit etmeyi amaçlayan sürü zekasına dayanmaktadır (Yang, 2017).

Geçmişte, hesaplamalı mühendislik tasarım süreci çoğunlukla deneysel temelliydi. Tasarımcı kendi deneyimini veya makine kataloğunu kullanarak bir üretim süreci için uygun işlem parametreleri kombinasyonunu seçebilir. Ancak seçilen parametreler genellikle kısıtlı ve optimum olmaktan uzaktır. Deney yoluyla uygun bir süreç parametreleri kombinasyonu seçmek maliyetli, zaman alıcıdır ve bir dizi deneysel denemeyi gerektirir. Bu nedenle araştırmacılar, üretim süreçlerinin süreç parametresi optimizasyonu için optimizasyon tekniklerini kullanmayı tercih etmişlerdir.

Bilgisayarların geliştirilmesinden sonra, mühendisler gerçek sistemlerin farklı yönlerini hesaplamak ve araştırmak için bilgisayarlarda modelleri simüle etmeye başlamışlardır. Araştırmacılar, üretim süreçlerinin süreç parametresi optimizasyonu için geometrik programlama, doğrusal olmayan programlama, sıralı programlama, hedef programlama, dinamik programlama vb. gibi bir dizi geleneksel optimizasyon tekniğini kullanmışlardır. Geleneksel optimizasyon yöntemlerinin birçok pratik durumda iyi performans göstermesine rağmen, esas olarak içsel arama mekanizmalarıyla ilgili bazı sınırlamaları vardır. Bu geleneksel optimizasyon yöntemlerinin arama stratejileri genellikle amaç ve kısıtlama fonksiyonlarının türüne (doğrusal, doğrusal olmayan vb.) ve problem modellemede kullanılan değişkenlerin türüne (tamsayı, ikili, sürekli vb.), verimlilikleri de büyük ölçüde çözüm alanının boyutuna bağlıdır, problem modellemede kullanılan değişken ve kısıtların sayısı ve çözüm uzayının yapısı (dışbükey, dışbükey

olmayan vb.), elde edilen çözümler büyük ölçüde seçilen ilk çözüme bağlıdır. Ayrıca, bu tür yöntemler farklı değişken türlerinin, amaç ve kısıtlama fonksiyonlarının kullanıldığı problemleri çözmek için kullanılabilecek genel çözüm yaklaşımları sağlayamazlar.

Gerçek dünyadaki optimizasyon problemlerinde değişkenlerin sayısı çok büyük olabilir ve optimize edilecek hedef fonksiyon üzerindeki etki çok karmaşık olabilir. Tasarımcı veya süreç planlayıcı (veya karar verici) küresel optimumla ilgilenirken, amaç fonksiyonunun birçok yerel optiması olabilir. Bu tür sorunlar, yalnızca yerel optimayı hesaplayan klasik yöntemlerle veya geleneksel yöntemlerle etkili bir şekilde çözülemez. Bu nedenle, tasarım problemleri için verimli ve etkili optimizasyon yöntemlerine ihtiyaç vardır. Bu alanda sürekli araştırmalar yürütülmektedir ve doğadan esinlenen sezgisel optimizasyon yöntemlerinin deterministik yöntemlerden daha iyi olduğu ve yaygın olarak kullanıldığı kanıtlanmıştır.

Bu nedenle, geleneksel optimizasyon tekniklerinin sınırlamalarını göz önünde bulundurarak, araştırmacılar, popüler olarak meta-sezgisel olarak bilinen gelişmiş optimizasyon algoritmaları geliştirmişlerdir (Rao, 2019).

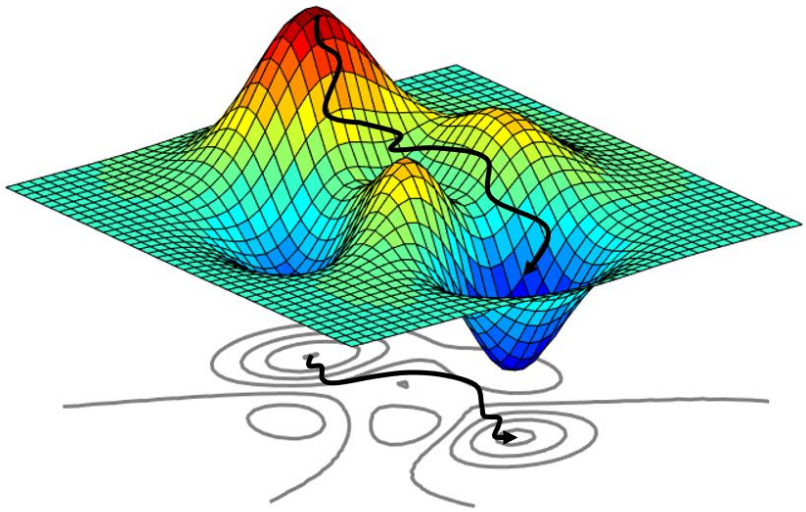
Meta-Sezgisel Algoritmalarda meta, "ötesinde" veya "daha yüksek seviye" anlamına gelir. Genellikle basit buluşsal yöntemlerden daha iyi performans gösterirler. Tüm meta-sezgisel algoritmalar, yerel arama ve küresel aramanın bazı dönüşümlerini kullanır. Çözümlerin çeşitliliği genellikle rastgele seçim yoluyla gerçekleştirilir. Meta-sezgisel tanımının popülaritesine rağmen, literatürde sezgisel ve meta sezgisel tanımı üzerinde mutabık kalınan bir tanım yoktur. Bu nedenle bazı araştırmacılar sezgisel ve meta sezgisel tanımını birbirinin yerine kullanır. Bununla birlikte, son eğilim, rastgele arama ve küresel keşif içeren tüm stokastik algoritmaları meta-sezgisel olarak adlandırmaya eğilimlidir. Rastgele arama, yerel aramadan küresel ölçekte aramaya geçmek için iyi bir yol sağlar. Bu nedenle, neredeyse tüm meta-sezgisel algoritmalar genellikle doğrusal olmayan modelleme ve küresel optimizasyon için uygundur (Gandomi ve ark., 2013).

Meta-Sezgisel Algoritmaların büyük çoğunluğu aşağıdaki prosedürü gerçekleştirmektedir.

1. Bir başlangıç toplumunun rastgele yaratılması
2. Her bir eleman için bir uygunluk değerinin belirlenmesi. Bu değer problemin minimizasyon ya da maksimizasyon problemi olmasına bağlı olarak maliyet ya da kar olarak isimlendirilebilir.
3. Uygunluk değerlerine göre toplumun yeniden yapılandırılması

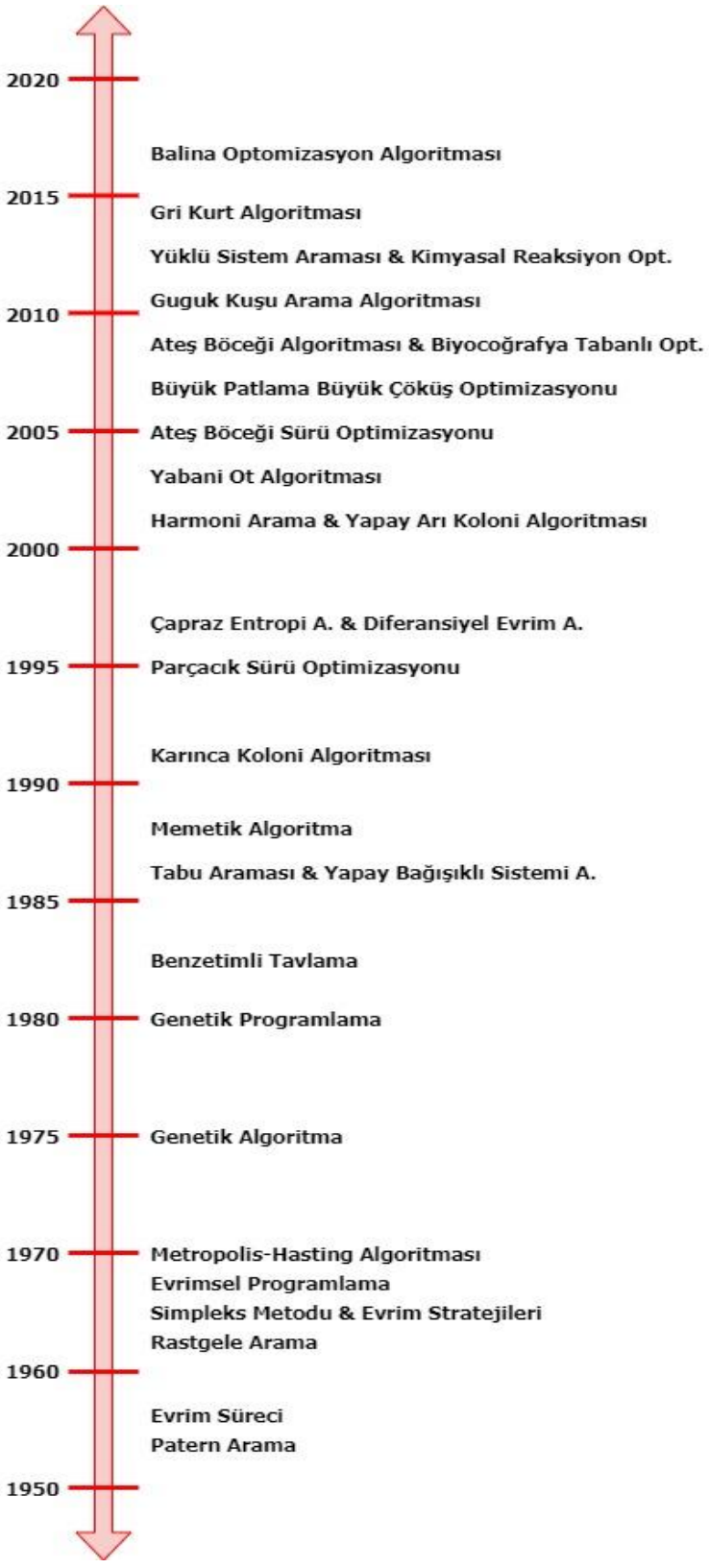
4. Gereksinimler karşılanana kadar iterasyonun devam etmesi. Burada karar mekanizması olarak belli bir iterasyon sayısı verilebilir ya da bir hata değeri belirlenebilir(Akbulut, 2008).

Şekil 3.9’ da optimizasyon algoritmasının arama uzayı ve minimum-maksimum hata değerindeki çözüm kümesine ulaşma şekli gösterilmiştir.



**Şekil 3.19** Optimizasyon Algoritmasının Arama Uzayı

Meta-Sezgisel Algoritmalar, optimizasyon problemlerini çözmek için doğadaki hayvanların, bireylerin, canlıların hareketlerinden esinlenerek oluşturulan algoritmalardır. Temel sezgisel arama algoritmaları kronolojiye göre Şekil 3.20’ de gösterilmektedir.




**Şekil 3.20** Meta Sezgisel Algoritmaların Kronolojisi

### 3.3.1. Genetik Algoritma

İlk olarak John Holland tarafından geliştirilen genetik algoritma (GA), biyolojik sistemlerin Darwinist evrimine dayanan evrimsel bir algoritmadır. Başlıca özellikleri üç genetik operatördür: çaprazlama, mutasyon ve seçim. İkili veya gerçek karakter dizisi olarak kodlanan bir popülasyondan gelen bir çözüm kümesi kromozom olarak adlandırılır. Bu tür genetik operatörler kullanılarak yeni bir çözüm popülasyonu üretilir (Yang, 2010b).

Genetik algoritmalar, problemlerin çözümü için evrimsel süreci bilgisayar ortamında taklit ederler. Diğer en iyileme yöntemlerinde olduğu gibi çözüm için tek bir yapının geliştirilmesi yerine, böyle yapılardan meydana gelen bir küme oluştururlar. Problem için olası pek çok çözümü temsil eden bu küme genetik algoritma terminolojisinde nüfus adını alır. Nüfuslar vektör, kromozom veya birey adı verilen sayı dizilerinden oluşur. Birey içindeki her bir elemana gen adı verilir. Nüfustaki bireyler evrimsel süreç içinde genetik algoritma işlemcileri tarafından belirlenirler (Goldberg ve Holland, 1988).

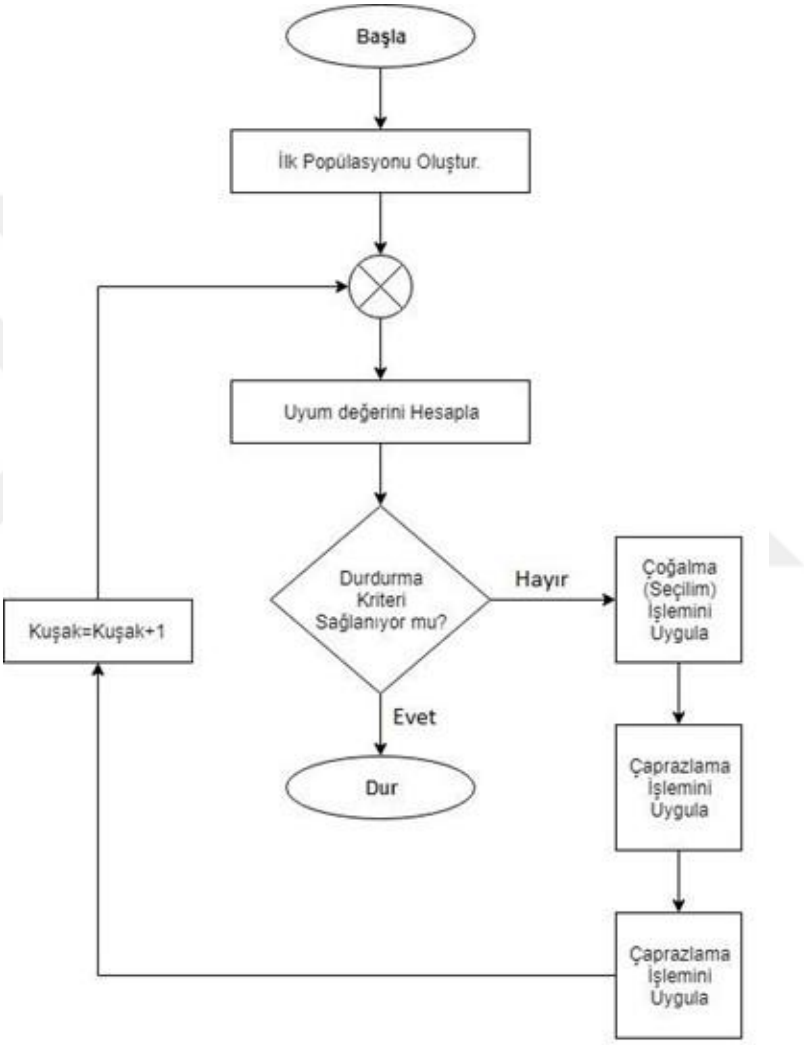
Genetik algoritma, yönlendirilmiş rastgele araştırma algoritmalarının bir türüdür. Doğal seçilim ile canlılarda bulunan genetik gelişimin benzetimini gerçekleştirmektedir. Algoritma diğer evrimsel algoritmalar gibi araştırma uzayında bulunan çözümlerin bazılarının oluşturduğu bir başlangıç popülasyonunu kullanmaktadır. Başlangıç popülasyonu her jenerasyonda, doğal seçilim ve tekrar üreme işlemleri aracılığıyla art arda geliştirilir. En son kuşağın en uygun yani kaliteli bireyi, problem için optimal çözüm olmaktadır. Bu çözüm her zaman optimum olmayabilir ama kesinlikle optimuma yakın bir optimal çözümdür (Karaboğa, 2014).

Genetik algoritmanın gerçekleştirilmesi için önce başlangıç popülasyonu oluşturulmakta ve sonra doğal seçilim işlemi ile birlikte genetik operatörler çaprazlama ve mutasyon gelecek jenerasyondaki çözümleri üretmek amacıyla kullanılmaktadır. Kalite veya uygunluk değerlendirme işlemi tekrar üreme olayında uygulanan seçme işlemini gerçekleştirebilmek için her bir bireye uygulanmaktadır. Birbirini takip eden jenerasyonların geliştirilmesi ve değerlendirilmesi çevrimi, optimal bir çözüm bulununcaya kadar devam etmektedir.

Bir problemin Genetik Algoritma ile çözümünde izlenecek işlem adımları aşağıda verilmektedir.

**Çizelge 3.1.** Genetik Algoritma Adımları

1- Çözümlerin bir başlangıç popülasyonunu oluştur.
2- Popülasyondaki her çözümün uygunluk değerini hesapla.
3- Durdurma kriteri sağlanıyorsa araştırmayı durdur
Yoksa, aşağıdaki adımları gerçekleştir.
a. Doğal seçilim işlemini uygula (uygunluk değerleri daha yüksek olan çözümler yeni popülasyonda daha fazla temsil edilirler.)
b. Çaprazlama işlemini uygula (Mevcut iki çözümden yeni iki yapı üretilir.)
c. Mutasyon işlemini uygula (Çözümlerde rastgele değişim meydana getirilir.)
4- 2. adıma git.




**Şekil 3.21** Genetik Algoritması Akış Diyagramı

### 3.3.2. Parçacık Sürü Optimizasyonu

Parçacık Sürü Optimizasyonu, 1990' lı yılların ortalarında bir sosyal psikolog olan James Kennedy ve bir elektrik mühendisi olan Russell Eberhart tarafından geliştirilmiş olan bir optimizasyon algoritmasıdır. Tabanında sosyal etkileşim yatan ve sürü zekâsını

temel alan bu algoritma, biyolojik toplumlardaki toplu zekâyı araştıran sosyobilişsel bir çalışma sırasında kuş sürülerinin hareketlerinin temsilinin gerçekleştirilmesi sonucunda ortaya koyulmuştur. Parçacık Sürü Optimizasyonu' nun kökleri iki ana yönteme uzanmaktadır: yapay hayat (kuşların, balıkların yani genel olarak sürülerin hareketlerini inceleyen teori) ve evrimsel programlama.

Doğası gereği toplum temelli olan ve toplumun değişim geçirmesiyle işleyen bu algoritma, yalnızca basit matematiksel operatörler kullanması sayesinde diğer evrimsel hesaplama tekniklerinden ayrılmaktadır. Parçacık Sürü Optimizasyonu' nun birçok global optimizasyon problemi için etkili bir çözüm yöntemi olduğu ve diğer evrimsel hesaplama yönteminin karşılaştığı bazı zorlukların üstesinden kolaylıkla gelebildiği Eberhart ve Kennedy tarafından gösterilmiştir (Kennedy ve Eberhart, 1995).

Sürüdeki her bir birey, bir parçacık olarak adlandırılır. Sürüdeki her bir parçacık, bireysel olarak aynı hükmedici prensip altında hareket eder; bir yandan her an için kendi konum değerini kontrol ederken kendi kişisel en iyi bölgesine ve bütün sürünün bildiği en iyi bölgeye doğru ivmelenir.

PSO yalnızca parçacıkların hızlarına dayandığından teorik olarak sonsuz boyutlu bir problemin çözümüne ulaşmak mümkündür. PSO' da parçacıkların konumu çözülmek istenen problemin 'n' boyutlu olduğu varsayılırsa, n-boyutlu bir uzay problemimizin çözüm uzayını ifade eder ve bu uzaydaki herhangi bir koordinat, incelenen problem için olası çözüm kümesidir.

Her evrimsel hesaplama yönteminde olduğu gibi, PSO' da da herhangi bir konumun uygunluğunu değerlendirmek için kullanılan bir fonksiyon vardır. Uygunluk fonksiyonu, konumun performansını değerlendirebilmek amacıyla o konumdan tek bir sayıdan oluşan bir değer oluşturur. Kısaca, ulaşılmak istenen uygunluk fonksiyonu değerine göre konumun çözüme olan uygunluğu belirlenir. Optimizasyonun amacı, problem için en uygun sonucu sağlayan konumu bulmaktır.

Geleneksel sezgisel arama algoritmalarında olduğu gibi PSO' da da parçacıklar arama uzayına, arama uzayı sınırları dikkate alınarak rastgele dağıtılır. En iyi amaç fonksiyonu değerine sahip olan parçacık diğer parçacıklara haber verir. Diğer parçacıklar denklemde formülü gösterilen hızla konumu en iyi olan parçacığa doğru yönelir.

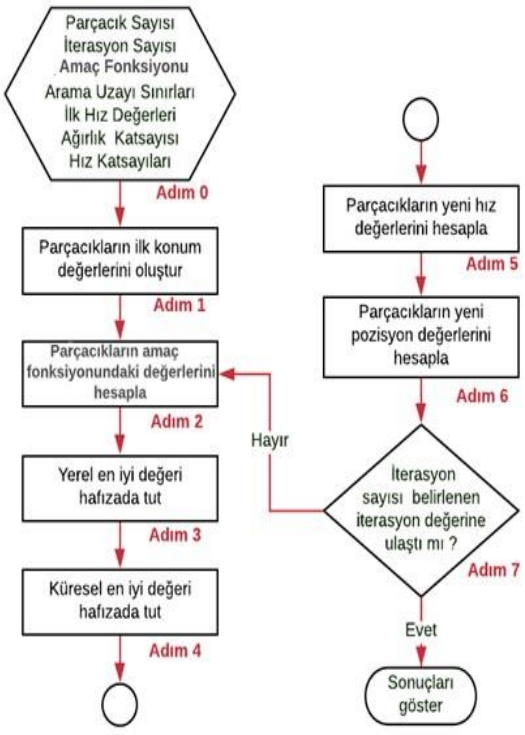
$$V_{iy} = wV_{\text{i}} + c_1 rand\left(P_y - X_i\right) + c_2 rand(P_g - X_i) \tag{3.29}$$

$$X_{iy} = X_i + V_i \tag{3.30}$$

$V_{iy}$ i. parçacığın yeni hızı, w atalet ağırlık faktörü, $V_i$ i. parçacığın hızı, $c_1$,$c_1$ hızlanma katsayıları, rand 0-1 arasında rastgele sayı, $P_y$ parçacıkların i. iterasyondaki en iyi(yerel en iyi) konumu, $X_i$ i. parçacığın bulunduğu mevcut konumu, $P_g$ parçacıkların o ana kadarki tüm iterasyonlardaki en iyi (küresel en iyi) konumudur. Denklem 3.30 parçacıkların yeni konumunu belirlemektedir (Öksüz, 2019).

**Çizelge 3.2.** PSO Algoritma Adımları

Başla
Başlangıç popülasyonunu yarat.
Her parçacığı bazı (küçük) rastgele hızlarla problem uzayındaki bir noktaya rastgele atayın ve her parçacığın mevcut konumundaki tutarlılığını değerlendirin.
Parçacıkların ilk hız değeri, hızlanma katsayıları ve atalet ağırlık değeri belirlenir.
Amaç Fonksiyonu $f(x), x = [x_1, x_2, x_3, \dots, x_d]^T$
**while (t<Jenerasyon Sayısı || durdurma kriteri)**
Her bir parçacığın amaç fonksiyonundaki değeri hesaplanır.
    **if($I_j < I_i$ )**
    En düşük amaç fonksiyonu değerini değiştir.
    **End**
Parçacıkların yeni hızları hesaplanır.
Parçacıkların yeni konumu bulunu
**end**




**Şekil 3.22** Parçacık Sürü Optimizasyonu Akış Diyagramı

### 3.3.3. Guguk Kuşu Arama Algoritması

Guguk Kuşu Arama (CS) algoritması, bazı guguklu türlerin zorunlu kuluçka parazitik davranışından esinlenen stokastik bir küresel arama algoritmasıdır ve 2009 yılında Xin She Yang ve Deb tarafından geliştirilmiştir (Yang ve Deb, 2009). Ek olarak, bu algoritma basit izotropik rastgele yürüyüşler yerine L'evy uçuşları yardımı ile geliştirilmiştir. Guguk Kuşu Arama algoritması anten dizilerinde optimizasyon uygulamalarında kullanılmaktadır (Singh ve Rattan, 2014).

Guguk kuşları herhangi bir kuş yuvasının hakiki sahibi kuş uzaklaşır uzaklaşmaz gözetlediği yerden gelir ve bir yumurtasını ev sahibi kuşun yumurtaları arasına bırakır. Yuva sahibi kuşun yumurtalarından birisini de gagasıyla alarak yuvadan uzaklaşır. Daha sonra yavru guguk kuşu, yuva sahibi kuştan daha büyük hale gelmesine rağmen onun tarafından beslenmeye devam edilir. Guguk kuşu, ziyaret ettiği her yuvaya sadece bir yumurta bırakır. Yumurtadan yeni çıkan bir guguk yavrusu, henüz gözleri bile açılmadan sanki öğretilmiş gibi, ev sahibi kuşun yumurtalarını sırtını ve kanatlarını kullanarak yuvadan atar.

Guguk Kuşu Arama algoritması üç idealleştirilmiş kuralı kullanır:

- Her guguk kuşu bir seferde bir yumurta bırakır ve onu rastgele seçtiği bir yuvaya bırakır.
- Yüksek kaliteli yani iyi çözüme sahip yumurtaya sahip en iyi yuvalar gelecek nesillere taşınır.
- Uygun konakçı yuvaların sayısı sabittir ve bir konakçı, pa $\in$ [0,1] olasılığı ile yabancı bir yumurtayı keşfedebilir. Bu durumda, ev sahibi kuş yeni bir yerde tamamen yeni bir yuva yapmak için yumurtayı atabilir veya yuvayı terk edebilir.

Basitlik açısından, bu son varsayım, yeni yuvaların (yeni konumlarda yeni rastgele çözümlerle) değiştirilmesiyle yuvaların pa fraksiyonu ile yaklaşık olarak tahmin edilebilir. Bir maksimizasyon problemi için, bir çözümün kalitesi veya uygunluğu, basitçe amaç fonksiyonuyla orantılı olabilir. Diğer uyum biçimleri, genetik algoritmalardaki uyum işlevine benzer şekilde tanımlanabilir.

GKA, bir anahtarlama parametresi geçişi ile kontrol edilen yerel bir rastgele yürüyüş ve küresel keşif rastgele yürüyüşünün dengeli bir kombinasyonunu kullanır. Denklem 3.31' de Guguk Kuşu Arama algoritmasının en iyi çözüm kümesini ulaşma adımı ifade edilmektedir.

$$x_i^{t+1} = x_i^t + \alpha s \otimes H(p_a - \epsilon) \otimes (x_j^t - x_k^t) \quad (3.31)$$

Burada $x_j^t$ ve $x_k^t$ rastgele seçilen iki farklı çözümdür, H(u) Heaviside fonksiyonu, $\epsilon$ tek tip bir dağılımdan alınan rastgele bir sayıdır, s adım boyutudur. $\otimes$ iki vektörün giriş açısından çarpımı anlamına gelir.

Guguk Kuşu Arama algoritmasının en güçlü özelliklerinden biri, yeni aday çözümler (yumurta) üretmek için L´evy uçuşlarının kullanılmasıdır.

$$x_i^{t+1} = x_i^t + \alpha L(s, \lambda) \quad (3.32)$$

$$L(s, \lambda) \sim \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}} \qquad (s > 0) \quad (3.33)$$

Burada, $\alpha > 0$, ilgilenilen problemin ölçekleriyle ilişkili olması gereken adım boyutu ölçeklendirme faktörüdür. Burada L (s, $\lambda$)' nin Lévy dağılımından alınan rastgele bir adım boyutu olduğuna dikkat çekmek önemlidir; doğrudan cebirsel hesaplamalar değil, rastgele bir sayı üretecidir, bu nedenle bu ince farklılığı vurgulamak için ∼ kullanılır (Yang, 2018). Denklem 3.32, Lévy uçuşu kullanılarak rastgele dağılımı göstermektedir. Denklem 3.33, Lévy uçuşu dağılımını formülize eder.

Algoritmanın temel üç kuralına dayanarak, Guguk Kuşu Arama algoritmasının temel adımları aşağıdaki sözde kodda özetlenebilir (Yang ve ark., 2010).

**Çizelge 3.3.** Guguk Kuşu Arama Algoritması Adımları

Başla
Amaç Fonksiyonu $f(x), x = [x_1, x_2, x_3, \dots, x_d]^T$
N adet konak yuvası başlangıç pozisyonu belirle.
Bu yuvaların $F_i$ amaç fonksiyonu değerlerini hesapla
**while (t<Jenerasyon Sayısı || durdurma kriteri)**
L'evy uçuşları aracılığıyla guguk kuşlarının konum değerlerini güncelle
$F_j$ amaç fonksiyonu değerlerini hesapla.
N adet konak yuvası arasından rastgele yuva seç.
**if($F_j < F_i$ )**
En düşük amaç fonksiyonu değerini değiştir.
**End**
En kötü yuvaları pa oranınca terk et. Yerlerine yenilerini bul.
İyi çözümleri hafızada tut.
Elde edilen çözümleri amaç fonksiyonu değerlerine göre küçükten büyüğe sırala ve en iyi çözümü bul
**end while**

(Yang ve ark., 2010)

### 3.3.4. Ateşböceği Algoritması

Ateşböceği Algoritması biyolojik olarak esinlenilmiş, nüfusa dayalı bir Meta-Sezgisel Algoritmadır. Ateşböceği Algoritması, Xin She Yang tarafından 2009 yılında doğadan ilham alan en başarılı algoritmalardan biri olarak önerildi ve geliştirildi (Yang, 2010a). Bu algoritma ateş böceklerinin parlama davranışı ve hareketleri incelenerek doğal davranışlarına dayanan doğadan esinlenen bir algoritma olarak çalışır. Ateşböceklerinin sosyal davranışlarında, her ateşböceğinin çekiciliği parlaklığına göre belirlenir ve parlaklığa doğru ilerlemeye çalışırlar. Her ateş böceğinin parlaklığı, o ateş böceğinin zindelik değerine bağlıdır. Popülasyondaki her bir ateş böceği bir probleme potansiyel bir çözüm temsil eder ve ateş böceği popülasyonunun başlangıçta çözüm alanı boyunca düzgün ve rastgele dağıtılması gerekmektedir.

Temelde, FA aşağıdaki üç idealleştirilmiş kuralı kullanır:

- Ateşböcekleri cinsiyeti ne olursa olsun diğer ateşböceklerine karşı çekiciliği olacaktır.
- Çekicilik parlaklık ile orantılıdır ve mesafe arttıkça ikisi de azalır. Bu yüzden herhangi iki yanıp sönen ateşböceği için daha az parlak olan daha fazla parlak olana doğru hareket edecektir. Eğer kendisinden daha parlak olmayan ateş böceği yoksa rastgele hareket edecektir.
- Bir ateşböceğinin parlaklığı, amaç fonksiyonunun uygunluk değerine göre belirlenir.

Bir ateşböceğinin çekiciliği, bitişik ateşböceğinin gördüğü ışık yoğunluğuyla orantılı olduğundan, çekicilik $\beta$ değişimi r mesafesi ile Denklem 3.34 gösterildiği gibi tanımlanabilir;

$$\beta = \beta_0 e^{-\gamma r^2} \tag{3.34}$$

Burada, $\beta_0$, r=0'daki çekiciliği ifade eder.

Bir ateş böceğinin hareketi i daha çekici başka bir ateşböceği j tarafından kendisine çekilir. Denklem 3.35, ışık yoğunluğuyla orantılı olarak ateş böceklerinin konumlarını güncelleyerek en iyi çözüme ulaşan iterasyon denklemini ifade eder.

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}\left(x_j^t - x_i^t\right) + \alpha_t \epsilon_i^t \tag{3.35}$$

Buradaki ikinci terim çekicilikten kaynaklanmaktadır. Üçüncü terimi $\alpha_t$ rastgelelik parametresidir. $\epsilon_i^t$, t zamanında bir Gauss dağılımdan veya sabit dağılımdan elde edilen rastgele sayıların bir vektörüdür. Eğer $\beta_0$ = 0 olursa, basit rastgele yürüyüşe dönüşür. Diğer yandan $\gamma$ = 0 ise Ateşböceği algoritması, Parçacık Sürü Optimizasyonunun bir varyantına indirgenir. Ayrıca, rastgelelik $\epsilon_i^t$ kolaylıkla L'evy uçuşları gibi diğer dağılımlara genişletilebilir (Yang, 2013).

Ateşböceği algoritmasının sözde kodu aşağıda gösterilmiştir.

**Çizelge 3.4.** Ateş Böceği Algoritması Adımları

Başla
Amaç Fonksiyonu $f(x), x = [x_1, x_2, x_3, \dots, x_d]^T$
Ateşböceklerinin başlangıç popülasyonunu üret.
Işık yoğunluğunu I formüle et.
Işık soğurma katsayısını $\gamma$ tanımla.
**while (t<Jenerasyon Sayısı || durdurma kriteri)**
**For i=1:n (Tüm ateşböcekleri için n)**
**For j=1:n (Tüm ateşböcekleri için n)**
**if($I_j < I_i$ )**
En düşük amaç fonksiyonu değerini değiştir.
**End**
Yeni çözümleri değerlendir ve ışık yoğunluğunu güncelle.
**End for *j***
**End for *i***
Ateşböceklerini uygunluk değerlerine göre derecelendir ve en iyi değerini bul.
**End while**
**End**

### 3.3.5. Yabani Ot Algoritması

Yabani ot algoritması ilk olarak Mehrabian tarafından 2006 yılında ortaya konulmuş bir sezgisel optimizasyon algoritmasıdır. Bu algoritma, doğadaki yabani otların, tüm zirai mücadelelere rağmen ayakta kalıp, soyunu sürekli güçlendirerek devam etmesinden esinlenilerek ortaya konulmuştur. Yabani ot ekolojisinde, iyilerin daima ayakta kalabildiği bir mekanizma vardır. Yabani otların bu özellikleri YOA' nın temelini oluşturmaktadır (Mehrabian ve Lucas, 2006).

Günümüzde YOA yeni bir optimizasyon metodu olarak anten konfigürasyonları için optimizasyon (Mallahzadeh ve ark., 2008), ileri beslemeli sinir ağları (Giri ve ark., 2010), elektromanyetik alanında anten düzenlemeleri (Karimkashi ve ark., 2010), anten dizilerinin tasarımı (Zaharis ve ark., 2013) ve bir çok farklı alandaki konularda kullanılmıştır.

Yabani otlar, rastgele dağılma yoluyla bir tarlayı istila eder ve mahsuller arasında bulunan kullanılmayan alanları kaplarlar. Her işgalci ot, bu alanlardaki kullanılmayan kaynakları alır, çiçeklenerek tohumlar yetişir ve bağımsız olarak yeni yabani otlar üretir. Her çiçeklenen ot tarafından üretilen yeni tohumların sayısı, koloni içindeki çiçeklenen otun uygunluğuna bağlıdır. Çevreye daha iyi adapte olan ve daha fazla kullanılmayan kaynak alanına sahip olan yabani otlar daha hızlı büyür ve daha fazla tohum elde eder. Yeni üretilen tohumlar rastgele tarlaya yayılır ve çiçeklenen yabani otlara dönüşür. Bu işlem, tarladaki sınırlı kaynaklardan dolayı arazide azami yabani ot sayısına ulaşılana kadar devam eder. Azami yabani ot sayısına ulaşıldıktan sonra yalnızca daha iyi uygunluk gösteren yabani otlar hayatta kalabilir ve yeni yabani otlar üretebilir. Yabani otlar arasındaki bu rekabetçi çekişme, zamanla kendilerini iyi adapte etmelerine ve gelişmelerine neden olur (Okkan ve ark., 2017).

Algoritmanın başlangıcında belirli sayıda yabani ot, n boyutlu arama alanının her tarafına rastgele dağıtılır. Rastgele üretilen yabancı otların tohum üretmesine izin verilir. Tohumların üretilmesi, kendi yeteneği ve kolonilerinin yeteneğiyle ilgilidir. Daha iyi uygunluk değerine sahip olan ot, daha fazla tohum üretirken, daha kötü uygunluk değerine sahip olan ot, daha az tohum üretir. Yabani otlar tarafından üretilen tohumlar, en kötü durumla başlayıp en iyi durumla biten doğrusal bir büyüme gerçekleştirirler. Yeniden çoğalma formülü Denklem 3.36' da verilmiştir.

$$weed_i = \frac{f_{anlık\ değer} - f_{minimum}}{f_{maksimum} - f_{minimum}} * (S_{maksimum} - S_{minimum}) + S_{minimum} \tag{3.36}$$

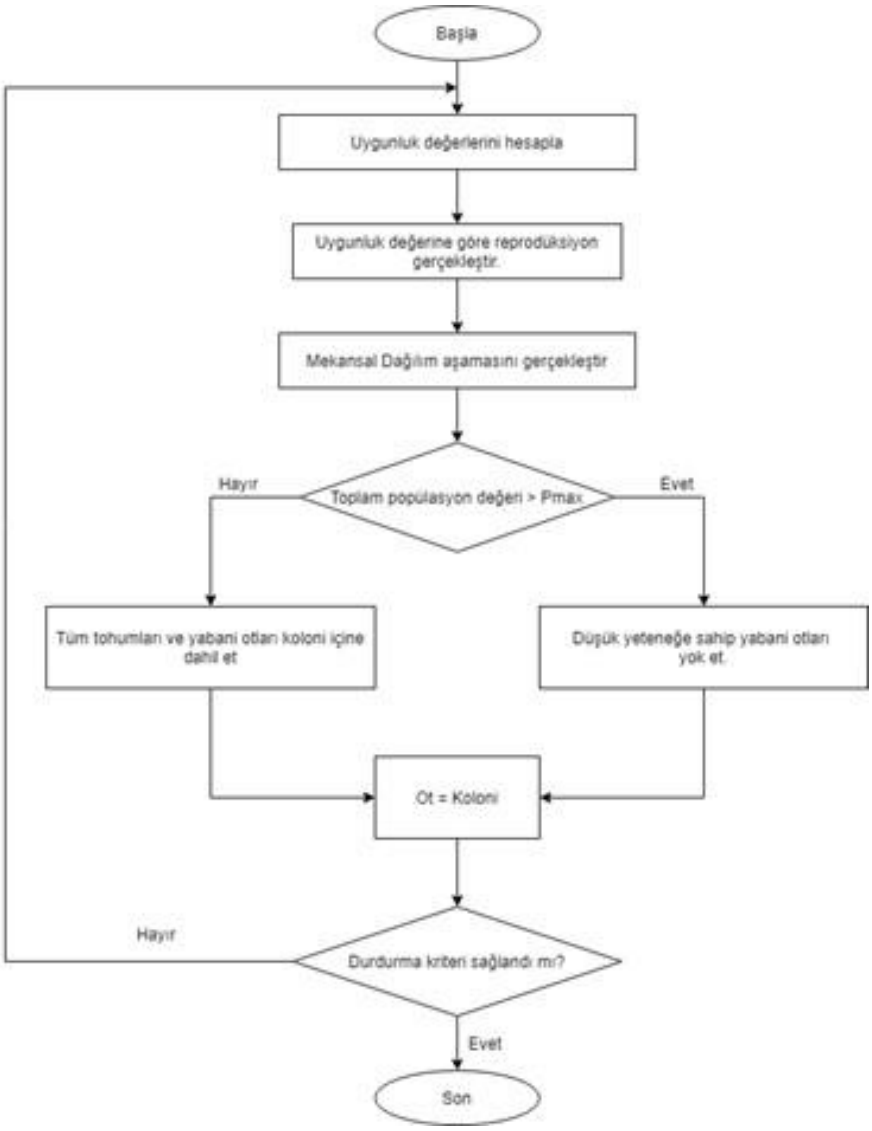
Burada $f_{anlık\ değer}$, mevcut yabani otun uygunluk değeridir. $S_{maksimum}$ ve $S_{minimum}$ sırasıyla bir otun maksimum ve minimum değerlerini ifade eder. $F_{maksimum}$ ve $F_{minimum}$, popülasyonun maksimum ve minimum uygunluk değerlerini temsil etmektedir.

Üretilen tohumlar rasgele olarak arama alanına yayılır ve ortalamaya göre sıfıra eşit ve değişken varyantına dayalı olarak, üretilen bu tohumlar ana ot yakınında bulunurlar. Burada, rastgele fonksiyonun standart sapması ($\sigma$) önceden belirlenmiş bir başlangıç değerinden ($\sigma_{başlangıç}$) başlayıp, son bir değere ($\sigma_{sonuç}$) kadar iterasyon boyunca azaltılacaktır. $\sigma_{iter}$ sonucu Denklem 3.37' de verilmiştir.

$$\sigma_{iter} = \frac{(iter_{maksimum} - iter)^n}{(iter_{maksimum})^n} \left(\sigma_{başlangıç} - \sigma_{sonuç}\right) + \sigma_{sonuç} \tag{3.37}$$

Burada, $iter_{maksimum}$ iterasyon sayısıdır, geçerli aman adımında standart sapma $\sigma_{iter}$ ve n doğrusal olmayan modülasyon indeksidir.

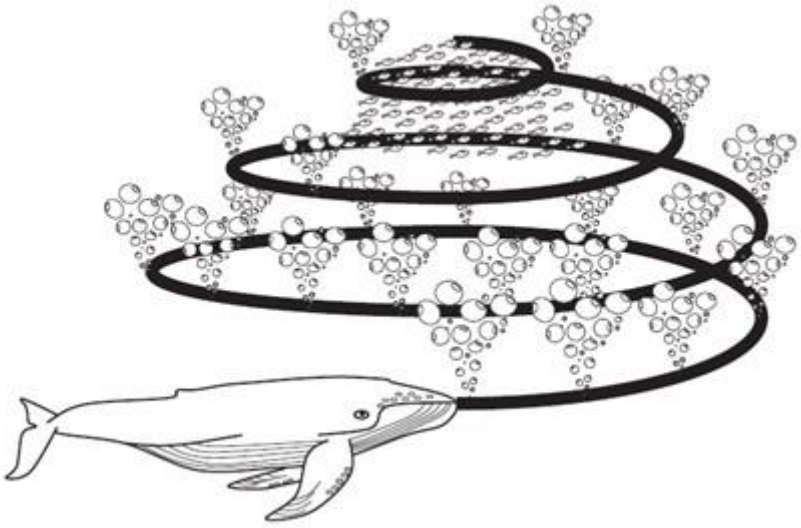
Hayatta kalmak için bitkiler arasında bir rekabet vardır. Kolonideki bitkiler çok hızlı çoğalır ve tüm bitkiler koloni olarak kabul edilir. Fakat kolonideki toplam bitki sayısı popülasyonun maksimum değerini aşmamalıdır. Bu nedenle, daha fazla uygunluk gösteren bitkiler koloni içine dâhil edilirken, daha az uygunluk gösteren bitkiler koloniden çıkarılır. Son olarak, kolonideki uygunluk değeri yüksek bitkiler, üst bitkiler olarak kabul edilir, maksimum iterasyon sayısına ulaşılana kadar tekrarlanır (Koç ve ark., 2018).




**Şekil 3.23** Yabani Ot Algoritması Akış Diyagramı

### 3.3.6. Balina Optimizasyon Algoritması

Balina sürüsü algoritması, Mirjalili ve Lewis tarafından 2016 yılında balinaların avlanma stratejilerinden faydalanılarak geliştirilmiş bir algoritmadır. Özellikle kambur balinaların davranışları balık sürüsünü çıkarmış oldukları hava kabarcıkları ile yönlendirmelerinden esinlenilmiştir. Aşağıda gösterilen Şekil 3.21' de balinaların avlanma davranışları verilmiştir (Mirjalili ve Lewis, 2016).



**Şekil 3.24** Balinaların Su Kabarcık Ağları ile Avlanması

Kambur balinalar avın yerini tanıyabilme ve onları çevreleyebilme yeteneğine sahiptir. Arama alanındaki optimal tasarımın konumu önceden bilinmediği için Balina Optimizasyon algoritması mevcut en iyi aday çözümün hedef av olduğunu veya optimum seviyeye yakın olduğunu varsayar. En iyi arama arayıcısı tanımlandıktan sonra, diğer arama avcıları konumlarını en iyi arama arayıcısına doğru güncellemeye çalışır. Bu davranış aşağıdaki denklemler ile ifade edilir:

$$\vec{D} = \left|\overrightarrow{DX^*}(k) - \vec{X}(k)\right| \tag{3.38}$$

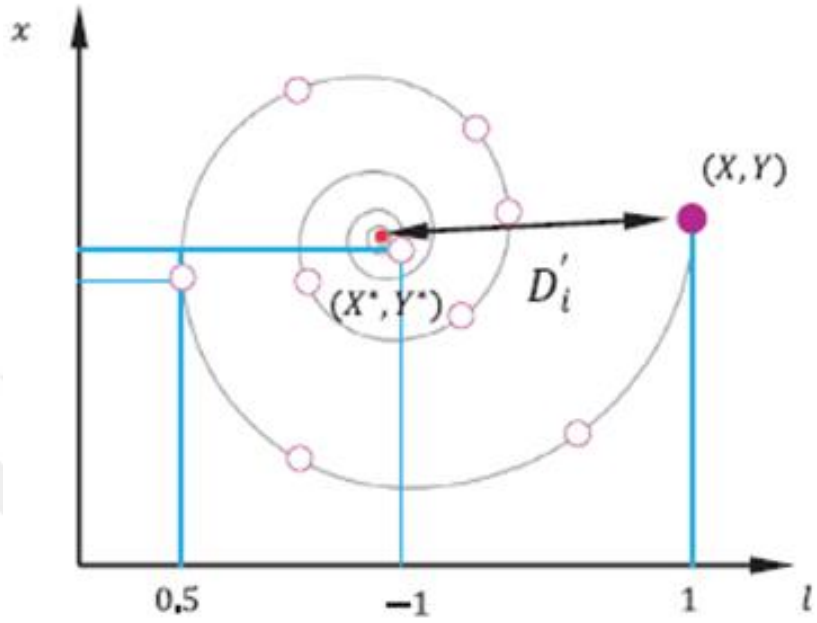
$$\vec{X}(k+1) = \left|\overrightarrow{X^*}(k) - \vec{D}\vec{A}\right| \tag{3.39}$$

$$\vec{A} = 2\vec{a}\vec{r} - \vec{a} \tag{3.40}$$

$$\vec{C} = 2\vec{a} \tag{3.41}$$

Burada, $\vec{D}$ vektörü sürü içerisindeki en iyi konumu ile sürü içerisindeki bir balina konumu arasındaki mesafeyi ifade eder, $\overrightarrow{X^*}(k)$ çözüm uzayında balinanın en iyi konumunu ifade etmektedir. $\vec{A}$ ve $\vec{C}$ ise yakınsamayı sağlayan vektörlerdir. $\vec{r}$ ifadesi

rastgele yönelim vektörü iken $\vec{a}$ ise iterasyon sayısına bağlı olarak azalan bir lineer vektördür. Bu parametrik değişkenler sürü içerisindeki her bir balina için her iterasyonda hesaplanmaktadır.




**Şekil 3.25** Ava Doğru Hareket

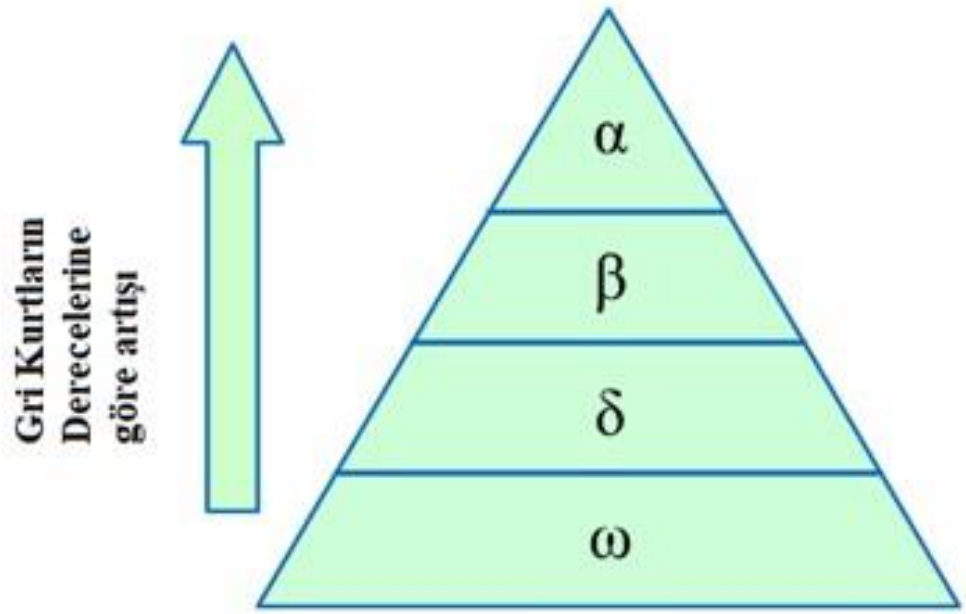
Balina sürüsünde yerel arama balinaların ava spiral şeklince veya lineer şekilde hareket etmektedir. Şekil 3.25' de balinaların ava doğru hareketi gösterilmiştir. Balina Optimizasyon algoritmasının sözde kodu aşağıda gösterilmiştir.

**Çizelge 3.5.** Balina Optimizasyonu Algoritma Adımları

```
Başla
Sürüdeki balinalara başlangıç yerlerini rastgele ata
Her bir balinanın uygunluk fonksiyonundaki değerini hesapla
X* en iyiyi belirle
while (t<Jenerasyon Sayısı || durdurma kriteri)
   for i=1:Sürü boyutu
      a,A,C,I ve p parametrelerini güncelle
      if(p<0.5)
           if(|A|<1)
           Denklem 3.38 ile balinaların konumunu güncelle
           else
           Denklem 3.39 ile balinaların konumunu güncelle
           end if
       else if (p>=0.5)
           Denklem 3.40 ile balinaların konumunu güncelle
       end if
     end for
Balinaların yeni konumlarını karşılaştır ve en iyiyi güncelle
        end while
```

### 3.3.7. Gri Kurt Optimizasyonu

Gri kurt optimizasyon algoritması (GKO), 2014 yılında Mirjalili tarafından önerilen gri kurtların avlanma stratejisini ve sosyal liderliğini taklit eden bir optimizasyon algoritmasıdır. Gri kurtlar çoğunlukla grup olarak yaşamayı tercih ederler. Grup büyüklüğü ortalama 5–12 birey arasındadır. Lider kurtlar alfa kurdu olarak adlandırılırlar. Gri kurtların hiyerarşisi, alfa, beta, delta ve omega kurtları olmak üzere dört grup şeklindedir. Lider ya da baskın kurda alfa kurdu denir ve alfa kurt gruptaki diğer kurtları yönetmek açısından en iyi kurttur ve genellikle avlanma, uyuma yeri, uyanma zamanı ve benzeri konularda karar vermekle sorumludur. Kurt grubunun sosyal hiyerarşisindeki ikincisi beta kurttur. Beta birçok etkinlikte lider kurt (alpha) yardımcısı konumundadır. Delta kurdu alfa ve beta kurtlarına uymakla zorunlu üçüncü kurttur ve sadece omega kurtlarına hükmedebilir. Sonuçta omega kurdu en düşük seviyedeki gri kurttur (Mirjalili ve ark., 2014). Şekil 3.23' te gri kurt hiyerarşisi gösterilmiştir.




**Şekil 3.23** Gri Kurt Hiyerarşisi

Gri kurt avcılığına ilişkin matematiksel model aşağıda verilmiştir:

$$\vec{D}_\alpha = \left|\vec{C}_\alpha . \vec{X}_\alpha - \vec{X}_i\right| \quad (3.42)$$

$$\vec{D}_\beta = \left|\vec{C}_\beta . \vec{X}_\beta - \vec{X}_i\right| \quad (3.43)$$

$$\vec{D}_\delta = \left|\vec{C}_\delta . \vec{X}_\delta - \vec{X}_i\right| \quad (3.44)$$

$$\vec{U}_\alpha = \vec{X}_\alpha - \vec{A}_\alpha \vec{D}_\alpha \quad (3.45)$$

$$\vec{U}_\beta = \vec{X}_\beta - \vec{A}_\beta \vec{D}_\beta \quad (3.46)$$

$$\vec{U}_\delta = \vec{X}_\delta - \vec{A}_\delta \vec{D}_\delta \quad (3.47)$$

$$\vec{X}_i = (\vec{U}_\alpha + \vec{U}_\beta + \vec{U}_\delta)/3 \quad (3.48)$$

Burada $\vec{D}_\alpha$, $\vec{D}_\beta$, $\vec{D}_\delta$, av ve kurt (alfa, beta, delta) arasındaki vektör mesafesini, $\vec{X}_\alpha$, $\vec{X}_\beta$, $\vec{X}_\delta$ avın pozisyon vektörünü, $\vec{X}_i$ gri kurtun i'inci iterasyondaki pozisyon vektörünü, $\vec{C}_\alpha$, $\vec{C}_\beta$, $\vec{C}_\delta$, $\vec{A}_\alpha$, $\vec{A}_\beta$, $\vec{A}_\delta$ alfa, beta ve delta kurtları için katsayı vektörlerini, $\vec{U}_\alpha$, $\vec{U}_\beta$, $\vec{U}_\delta$ alfa, beta ve delta kurtları için deneme vektörlerini göstermektedir.

Alfa, beta ve delta kurtları için katsayı vektörleri aşağıdaki gibi hesaplanır:

$$\vec{A}_\alpha = 2\vec{a}\vec{r}_{\alpha 1} - \vec{a} \quad (3.49)$$

$$\vec{C}_\alpha = 2\vec{r}_{\alpha 2} \quad (3.50)$$

$$\vec{A}_\beta = 2\vec{a}\vec{r}_{\beta 1} - \vec{a} \quad (3.51)$$

$$\vec{C}_\beta = 2\vec{r}_{\beta 2} \quad (3.52)$$

$$\vec{A}_\delta = 2\vec{a}\vec{r}_{\delta 1} - \vec{a} \quad (3.53)$$

$$\vec{C}_\delta = 2\vec{r}_{\delta 2} \quad (3.54)$$

Burada, $\vec{a}$ vektörün optimizasyon sırasında doğrusal olarak 2' den 0' a düştüğünü gösterir. $\vec{r}_{\alpha 1}$, $\vec{r}_{\beta 1}$ ve $\vec{r}_{\delta 1}$ ifadeleri [0,1] de ilk rastgele vektörleri, $\vec{r}_{\alpha 2}$, $\vec{r}_{\beta 2}$, $\vec{r}_{\delta 2}$ ifadeleri de [0,1]' de ikinci rastgele vektörleri gösterir.

Gri Kurt Optimizasyon algoritmasının sözde kodu aşağıdaki gibidir.

**Çizelge 3.6.** Gri Kurt Optimizasyonu Algoritma Adımları

Başla
Gri kurtların pozisyonlarını başlat ve maliyet değerlerini hesapla
En iyi gri kurdu alfa, ikinci en iyiyi beta, üçüncü en iyiyi delta kurt olarak kaydet
$X^*$ en iyiyi belirle
**while (t<Jenerasyon Sayısı || durdurma kriteri)**
$\vec{a}$ yı azalt
  **for i=1:Sürü boyutu(Her gri kurt için)**
    Alfa, beta, delta katsayı vektörlerini oluştur.
    Mesafe ve deneme vektörlerini hesapla.
    Gri kurdun konumunu güncelle
  **end for**
Güncellenen gri kurtların maliyet değerini hesapla.
  **for (her gri kurt için)**
    **if (gri kurt < alfa kurt)**
      Alfa kurdu güncelle
    **else if (gri kurt < beta kurt)**
      Beta kurdu güncelle
    **else if (gri kurt < delta kurt)**
      Delta kurdu güncelle
    **end if**
    **end for**
    Seçilenlerin durumunu güncelle
Gri kurtların yeni konumlarını karşılaştır ve en iyiyi güncelle
    **end while**

# 4. MİKROŞERİT ANTENLERİN TASARIMI VE UYGULAMALARI

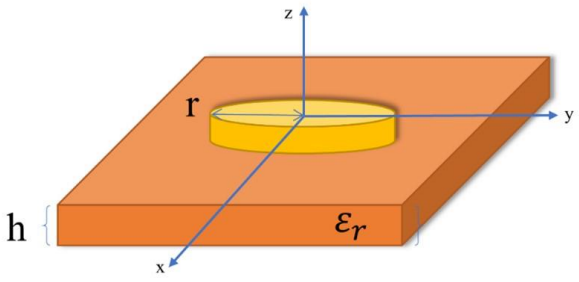
## 4.1. Mikroşerit Antenlerin Parametrelerinin Optimizasyonu

### 4.1.1. Daire Şekilli Mikroşerit Anten Parametrelerinin Optimizasyonu

Daire şekilli mikroşerit anten, diğer yama antenlere göre biraz daha küçük olduğu için kullanıldığı alanlara daha iyi uyum sağlayabilmektedir. Anten dizi uygulamaları gibi bazı uygulamalarda, diğer yapılanmalara göre dairesel şekiller belirgin avantaj sağlamaktadır. Ayrıca dairesel mikroşerit anten, belli bir aralıkta empedans değeri ışıma deseni ve çalışma frekansı üretmesi için kolayca modifiye edilebilirler. Basit yapısı ile üretimi daha kolay olan daire şekilli mikroşerit antenler, değişken parametrelerinin diğer yama şekillerine göre daha az olması sebebi ile daha kolay tasarlanabilir.

Daire şekilli mikroşerit antenler üzerine çalışan araştırmacılar, bu antenlerin üretimlerinin yanı sıra anten parametrelerinin iyileştirilmesi üzerine de çeşitli çalışmalar gerçekleştirmişlerdir. Son yıllarda bant genişliğini artırmada farklı yöntemler denenmiş ve anten parametreleri optimizasyonu sonucu istenen rezonans frekansında antenler üretilmiştir.

Şekil 4.1' de görülen anten bir daire şekilli mikroşerit antendir. Daire şekilli mikroşerit antenin rezonans frekansını belirlemede etkili olan dairesel iletken tabakanın yarıçapı 'r', dielektrik alt tabakanın kalınlığı 'h' ve yalıtkan tabakanın dielektrik katsayısı '$\varepsilon_r$' Meta-Sezgisel optimizasyon algoritmaları ile iyileştirilmiştir.




**Şekil 4.1** Daire Şekilli Mikroşerit Anten Parametreleri

İstenen ve optimize edilmiş parametreler sonucunda hesaplanan sonuçların hatasını en aza indirecek bir uygunluk fonksiyonu oluşturulmuştur. Denklem 4.1' de uygunluk fonksiyonu gösterilmektedir.

$$Uygunluk\ Değeri = w_1 \left| \frac{f_r^i - f_r^h}{f_r^i} \right| \tag{4.1}$$

Bu fonksiyonda $w_1$ ağırlık katsayısı olup uygulama için 5 olarak seçilmiştir. $f_r^h$ algoritma ile hesaplanan rezonans frekansı, $f_r^i$ istenen rezonans frekansıdır.

Daire şekilli mikroşerit anten için belirlenen uygunluk fonksiyonuna Genetik algoritma, Parçacık Sürü Optimizasyonu ve Guguk Kuşu algoritmasının uygulanması sonucu elde edilen fiziksel anten karakteristikleri ve bu karakteristiklere göre elde edilen rezonans frekansları belirlenmiştir. Optimizasyon algoritmalarının rastgeleliğinden dolayı, her bir algoritmanın her bir frekans değerinde onar kez çalıştırılması sonrasında elde edilen değerlerin en iyilerine ulaşılmıştır. Uygulamalarda her üç algoritma için toplum sayıları 100 olarak seçilmiştir. Genetik algoritmada mutasyon katsayısı 0.2 seçilmiştir. Parçacık Sürü Optimizasyonunda sosyal bileşen 0.5, bilişsel bileşen 1 ve atalet momenti 0.7 seçilmiştir. Guguk Kuşu algoritmasında yumurtlama çapı parametresi 0.25 olarak seçilmiştir.

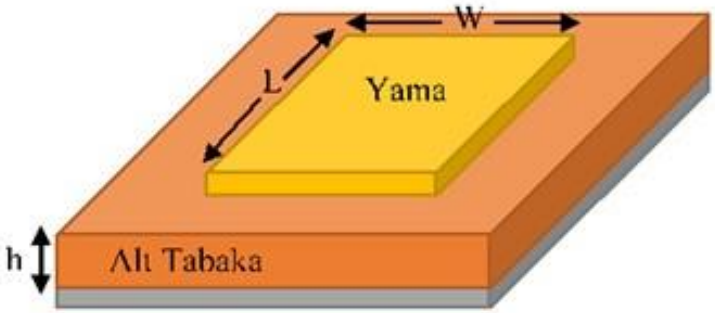
**Çizelge 4.1:** Daire Şekilli Mikroşerit Anten Optimizasyon Sonuçları

| | ***a (mm)*** | ***h (mm)*** | $\boldsymbol{\varepsilon_r}$ | ***Rezonans Frekansı (Hz)*** | ***Uygunluk Değeri*** | ***Hesaplama Süresi (sn)*** |
|---|---|---|---|---|---|---|
| *GA* | 9.8314 | 8.2124 | 2.0589 | 3 e+09 | 3.2512 e-05 | 2.14 |
| *PSO* | 14.6730 | 6.0537 | 2.7925 | 3 e+09 | 1.6134 e-05 | 1.24 |
| *GKA* | 12.4548 | 9.5965 | 2.3834 | 3 e+09 | 1.2855 e-05 | 1.57 |

Yapılan uygulamalarda daire şekilli mikroşerit anten için Genetik Algoritma, Parçacık Sürü Optimizasyonu ve Guguk Kuşu Arama Optimizasyonu yapılan sonuçlar Çizelge 4.1' de karşılaştırılmıştır. Sonuçlarda Guguk Kuşu Arama Algoritması daha iyi sonuç verdiği gözlemlenmiştir. Ayrıca PSO' nun daha çabuk sonuç vermesine rağmen, GKAA' nın daha çabuk hedeflenen sonuca yakınsadığı görülmüştür.

### 4.1.2. Dikdörtgen Şekilli Mikroşerit Anten Parametrelerinin Optimizasyonu

Dikdörtgen şekilli mikroşerit antenler literatürde en yaygın kullanılan anten çeşididir. Özellikle optimizasyon algoritmalarının denenmesinde ve kullanıma uygun mikroşerit anten üretilmesinde çokça tercih edilir (Yilmaz ve ark., 2007; Gollapudi ve ark., 2008).




**Şekil 4.2** Dikdörtgen Şekilli Mikroşerit Anten

Şekil 4.2' de görülen anten bir dikdörtgen şekilli mikroşerit antendir. Dikdörtgen şekilli mikroşerit antenin rezonans frekansını belirlemede etkili olan dikdörtgen iletken tabakanın uzunluğu 'L', genişliği 'W', dielektrik alt tabakanın kalınlığı 'h' ve yalıtkan tabakanın dielektrik katsayısı '$\varepsilon_r$' meta sezgisel optimizasyon algoritmaları ile iyileştirilmiştir.

Bu uygulamada Parçacık Sürü Optimizasyonu, Guguk Kuşu Arama algoritmasının farklı rezonans frekanslarında dikdörtgen şekilli mikroşerit anten için hesaplamaları yapılmıştır. Optimizasyon algoritmasının maliyet fonksiyonu için iletim hattı modelinden faydalanılmıştır. Optimizasyonu yapılması istenen mikroşerit antenin dielektrik malzemesi için seramik malzeme olan $Al_2O_3$ düşünülmüştür. Bu malzemenin dielektrik sabiti 9.8' dir. Alt tabaka kalınlığı 1.6 mm olarak belirlenmiştir.

İstenen ve optimize edilmiş parametreler sonucunda hesaplanan sonuçların hatasını en aza indirecek bir uygunluk fonksiyonu oluşturulmuştur. Denklem 4.2' de uygunluk fonksiyonu gösterilmektedir.

$$Uygunluk\ Değeri = w_1 \left| \frac{f_r^i - f_r^h}{f_r^i} \right| \tag{4.2}$$

Bu fonksiyonda $w_1$ ağırlık katsayısı olup uygulama için 4 olarak seçilmiştir. $f_r^h$ algoritma ile hesaplanan rezonans frekansı, $f_r^i$ istenen rezonans frekansıdır.

Dikdörtgen şekilli mikroşerit anten için belirlenen uygunluk fonksiyonuna Parçacık Sürü Optimizasyonu ve Guguk Kuşu algoritmasının uygulanması sonucu elde edilen fiziksel anten karakteristikleri belirlenmiştir. Bu anten karakteristikleri belirlenen 5 rezonans frekans değerine göre belirlenmiştir. Optimizasyon algoritmalarının rastgeleliğinden dolayı, her bir algoritmanın her bir frekans değerinde onar kez çalıştırılması sonrasında elde edilen değerlerin en iyilerine ulaşılmıştır. Uygulamalarda

iki algoritma içinde toplum sayıları 25 ve 100 iterasyon olarak seçilmiştir. Parçacık Sürü Optimizasyonunda sosyal bileşen 0.5, bilişsel bileşen 1 ve atalet momenti 0.7 seçilmiştir. Guguk Kuşu algoritmasında yumurtlama çapı parametresi 0.25 olarak seçilmiştir.

Çizelge 4.2' de PSO ve GKAA optimizasyon algoritmaları kullanılarak 5 farklı frekans değerinde dikdörtgen şekilli mikroşerit antenin yama uzunluğu ve genişliği değerleri gösterilmiştir.

**Çizelge 4.2:** Dikdörtgen Şekilli Mikroşerit Anten Optimizasyon Sonuçları

| $f_r$ (GHz) | $\varepsilon_r$ | h (mm) | $W_{PSO}$ | $L_{PSO}$ | $W_{GKAA}$ | $L_{GKAA}$ |
|---|---|---|---|---|---|---|
| 1.8 | 9.8 | 1.6 | 29.2034 | 22.0337 | 28.4024 | 22.3038 |
| 2.4 | 9.8 | 1.6 | 21.4520 | 31.4999 | 21.3030 | 31.5055 |
| 3.0 | 9.8 | 1.6 | 17.0937 | 28.1284 | 17.2290 | 28.7381 |
| 5.8 | 9.8 | 1.6 | 8.5043 | 28.6482 | 8.4764 | 29.2046 |
| 6.2 | 9.8 | 1.6 | 7.8800 | 31.3284 | 7.8467 | 31.4947 |

## 4.2. Selçuklu Yıldızı Şekilli Mikroşerit Yama Anten Tasarımı

Selçuklu Yıldızı şekilli mikroşerit anten teorik hesaplamalarla, ANSYS HFSS programı ile simüle edilmiştir. Farklı malzemelerin kullanıldığı Selçuklu şekilli mikroşerit anten çalışması ile 1342 anten simülasyonu veri seti oluşturulmuştur. Bu veri seti Selçuklu Yıldızı şekilli mikroşerit antenin boyut parametreleri ve rezonans frekansı, geri dönüş kaybı, bant genişliği, anten öz direnci, duran dalga oranı gibi önemli mikroşerit anten parametrelerinden meydana gelmiştir. Veri seti oluşturulurken, yamaya ait tasarım parametreleri, belirli aralıklarla taramaya tabii tutulmuş ve rasgele frekanslarda kabul edilebilir dönüş kaybı değerine sahip tasarım parametreleri belirlenmiştir. Örnek olarak; yama yarıçapı $R_1$ değeri, 2 ile 8mm boyutları arasında 0.5mm' lik adımlarla taratılmıştır. Sonrasında seçilen her bir yarıçap değeri için, yama merkezi, yama yarıçapının orta noktası ve yıldızın köşe noktası olmak üzere üç farklı besleme konumu için taramalar tekrarlanmıştır. Son olarak da literatürdekine benzer şekilde, taban boyutlarını yama yarıçapı ve dielektrik kalınlığına bağlayan K parametresinin 1 ile 8 arasındaki taramaları 1' lik adımlarla tekrarlanmıştır. Bu ifade Denklem 4.3' de verildiği gibi düzenlenmiştir. Yapılan tüm bu taramalar sonucunda optimum simulasyon tasarımları elde edilmiştir.

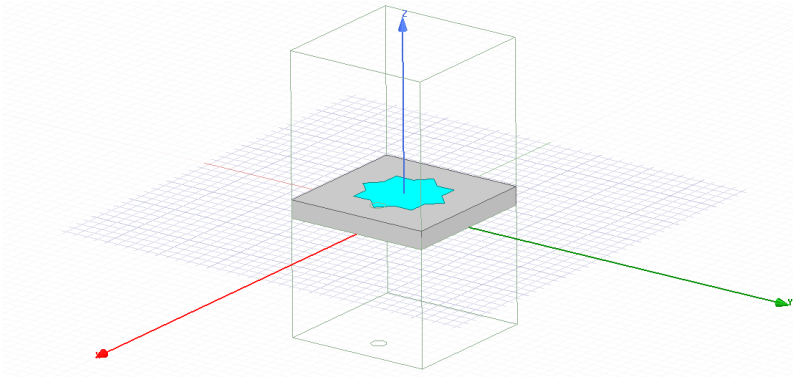
$$L_{gnd} = 2 \times R_1 + K \times h \tag{4.3}$$

Çizelge 4.3’ te Selçuklu Yıldızı şekilli mikroşerit anten için tasarım parametreleri gösterilmiştir.

**Çizelge 4.3:** Selçuklu Yıldızı Şekilli Mikroşerit Anten Simülasyon Tasarım Parametreleri

| No | Malzeme_TasarımNo | Ɛr | $h_1$ (mm) | $x_f$ (mm) | $y_f$ (mm) | L (mm) | $R_1$ (mm) | K |
|---|---|---|---|---|---|---|---|---|
| **1** | 5870_no75 | 2.33 | 0.254 | 4.00 | 0 | 16.76 | 8.0 | 3 |
| **2** | 5870_no180 | 2.33 | 0.254 | 0.00 | 0 | 18.03 | 8.0 | 8 |
| **3** | 5880lz_no69 | 2.00 | 4.750 | 3.50 | 0 | 21.75 | 3.5 | 3 |
| **4** | 5880lz_no130 | 2.00 | 4.750 | 0.00 | 0 | 20.75 | 8.0 | 1 |
| **5** | 6010_no110 | 10.20 | 2.540 | 0.00 | 0 | 23.70 | 5.5 | 5 |
| **6** | DE104_no156 | 4.37 | 1.550 | 4.00 | 0 | 23.75 | 8.0 | 5 |
| **7** | FR4_no195 | 4.60 | 1.600 | 0.00 | 0 | 22.40 | 8.0 | 4 |
| **8** | FR4_no207 | 4.60 | 1.600 | 0.00 | 0 | 19.80 | 7.5 | 3 |
| **9** | FR4_no217 | 4.40 | 1.600 | 7.50 | 0 | 26.20 | 7.5 | 7 |
| **10** | HRFR-4UV_no4 | 4.60 | 1.600 | 3.50 | 0 | 19.80 | 3.5 | 8 |
| **11** | HRFR-4UV_no16 | 4.60 | 1.600 | 1.50 | 0 | 18.80 | 3.0 | 8 |
| **12** | HRFR-4UV_no32 | 4.60 | 1.600 | 0.00 | 0 | 21.80 | 4.5 | 8 |
| **13** | HRFR-4UV_no64 | 4.60 | 1.600 | 3.80 | 0 | 26.20 | 7.5 | 7 |
| **14** | HRFR-4UV_no71 | 4.60 | 1.600 | 4.50 | 0 | 20.20 | 4.5 | 7 |
| **15** | HRFR-4UV_no78 | 4.60 | 1.600 | 8.00 | 0 | 27.20 | 8.0 | 7 |
| **16** | HRFR-4UV_no82 | 4.60 | 1.600 | 3.50 | 0 | 16.60 | 3.5 | 6 |
| **17** | HRFR-4UV_no87 | 4.60 | 1.600 | 6.00 | 0 | 21.60 | 6.0 | 6 |
| **18** | HRFR-4UV_no103 | 4.60 | 1.600 | 3.80 | 0 | 24.60 | 7.5 | 6 |
| **19** | HRFR-4UV_no107 | 4.60 | 1.600 | 0.00 | 0 | 15.60 | 3.0 | 6 |
| **20** | HRFR-4UV_no111 | 4.60 | 1.600 | 0.00 | 0 | 19.60 | 5.0 | 6 |
| **21** | HRFR-4UV_no128 | 4.60 | 1.600 | 0.00 | 0 | 22.00 | 7.0 | 5 |
| **22** | HRFR-4UV_no230 | 4.60 | 1.600 | 6.00 | 0 | 16.80 | 6.0 | 3 |
| **23** | HRFR-4UV_no284 | 4.60 | 1.600 | 4.00 | 0 | 17.60 | 8.0 | 1 |

Şekil 4.3’ te ANSYS HFSS programı ile Selçuklu Yıldızı şekilli mikroşerit anten tasarımı gösterilmiştir.



**Şekil 4.3** Selçuklu Yıldızı Şekilli Mikroşerit Anten Tasarımı

### 4.3. Selçuklu Yıldızı Mikroşerit Anteni için Optimizasyon Algoritmaları ile Yapay Sinir Ağı Modeli

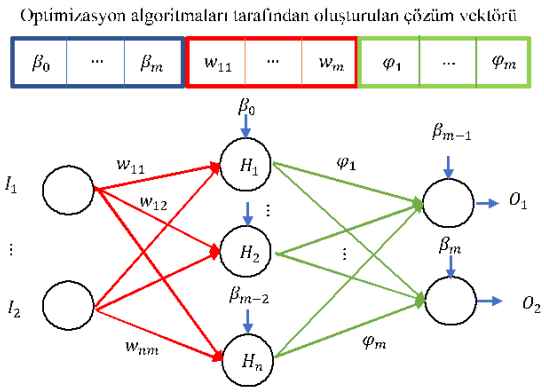
Bir Yapay Sinir Ağı kendisine gösterilen bir girdi setine karşılık gelebilecek bir çıktı seti belirlemektir. Toplam 12 parametreden oluşan bu veri setini, tekrarlanan veri sütunları çıkartıldıktan sonra, kalan 7 sütunluk haliyle, optimizasyon algoritmaları ile hibrit kullanılan Yapay Sinir Ağı modelinde çalıştırmak için bir ön işlemden geçirmek gerekmektedir.

İlk olarak veri seti rastgele biçimde %70 eğitim ve %30 test verisi olarak ayrılmıştır. 7 parametreden oluşan simülasyon verisinde farklı kombinasyonlar ile sinir ağı eğitilmeye çalışılmıştır. Tüm veri setini farklı ölçeklere sahip olan özelliklerini ortadan kaldırmak için minimum–maksimum normalizasyon tekniği kullanılarak giriş ve çıkış değerleri -1 ve 1 arasında normalize edilmiştir.

Çok katmanı yapay sinir ağı eğitiminde optimizasyon algoritmaları kullanılmıştır. Kullanılan optimizasyon algoritmalarının popülasyonunu belirlemek için ağın ağırlık ve biaslarından oluşan matris uzunluğu dikkate alınmıştır.

Optimizasyon algoritmalarının maliyet fonksiyonu olarak çok katmanlı sinir ağının ortalama kare hatası (MSE) kullanılmıştır. Ortalama kare hatasını minimize ederek ağ modeline uygun ağırlık ve bias vektörünün optimum sonuçları üretmesi sağlanmıştır.

YSA modeline uygun olarak ağırlık ve bias vektörü Şekil 4.4’ te gösterilmiştir.




**Şekil 4.4** Çözüm vektörünün YSA' ya yerleştirilmesi(Aljarah ve ark., 2018)

Optimizasyon algoritmasının çözüm vektörünü oluşturabilmesi için YSA modelindeki ağırlıklardan oluşturulan vektör uzunluğunun bilinmesi gerekir. Bu vektörün uzunluğunun hesaplandığı formül Denklem 4.3' te gösterilmiştir.

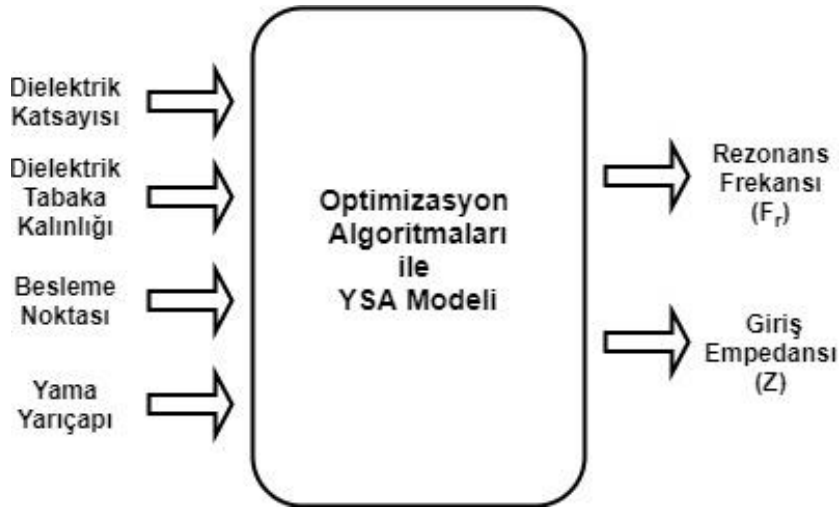
$$Vektör\ Uzunluğu = (i+1)xh + (h+1)xh + \cdots + o\ x\ (h+1) \tag{4.3}$$

Burada i ağın giriş sayısını, h gizli katmandaki nöron sayısını ve o çıkış sayısını temsil etmektedir.

**Çizelge 4.4:** Selçuklu Yıldızı mikroşerit anten veri seti için oluşan YSA modeli

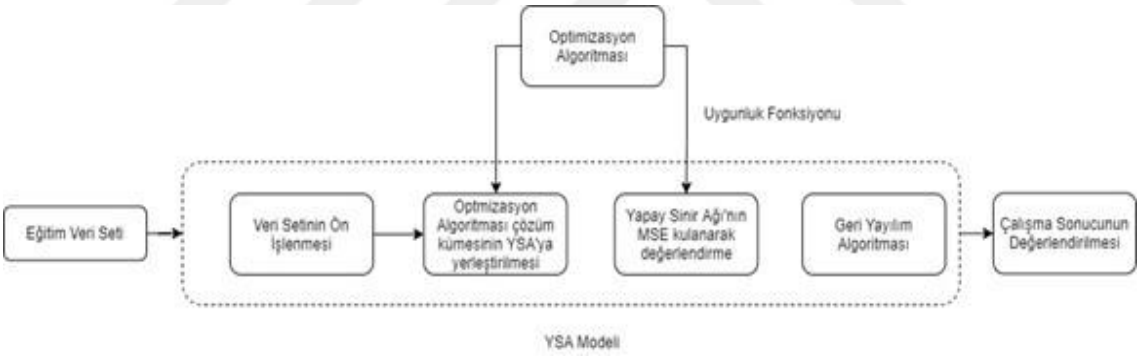
| Veri Seti | Özellik | Ağ Yapısı (Giriş-Gizli-Çıkış) | Ağırlık ve Bias Vektör Boyutu |
|---|---|---|---|
| Selçuklu Yıldızı Mikroşerit Anten | 6 | 4-10-10-2 | 182 |

İlk aşamada Selçuklu Yıldızı mikroşerit anteni tasarlanmış ve YSA' nın eğitim ve test veri seti oluşturulmuştur. İkinci adımda 4 giriş ve 2 çıkış olarak veri seti ön işlemeden geçirilerek çok katmanlı ağa sunulmuştur. Üçüncü adımda, optimizasyon algoritması kullanılarak çok katmanlı ağdaki ağırlık ve bias değerleri optimize edilmiştir. Yapay Sinir Ağının eğitiminden önce, optimizasyon algoritması ile maliyet fonksiyonu olarak Ortalama Kareler Hatası (MSE) kullanılmıştır. Daha sonra sinir ağı, yerel minimumları bulan Levenberg-Marquardt geri yayılım algoritması ile yeniden eğitilmiştir. Son adımda ise test veri setinde eğitilen modelin performansı incelenir. Oluşturulan YSA modelinin giriş ve çıkış parametreleri Şekil 4.5' te gösterilmiştir.




**Şekil 4.5** Optimizasyon Algoritmaları ile Oluşturulan YSA Modeli

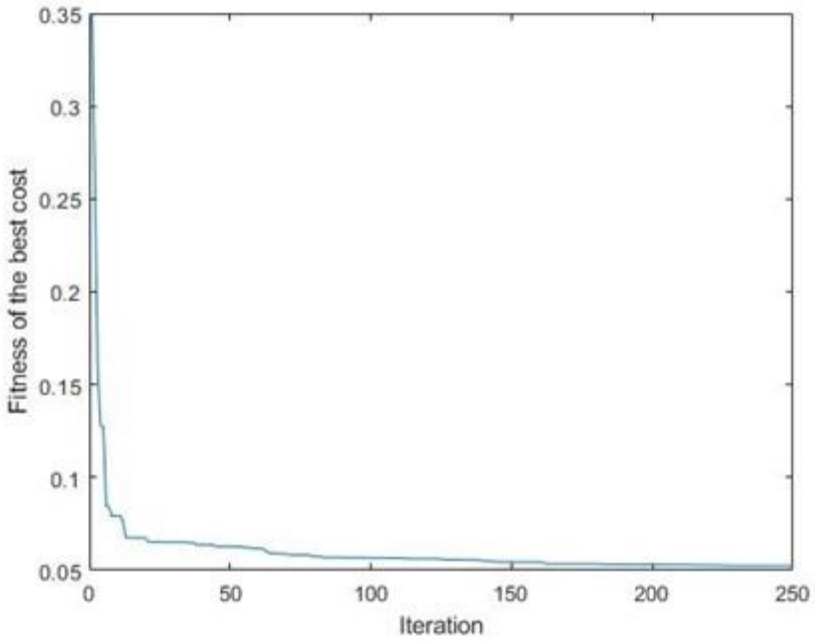
Ağın eğitimi için önerilen en iyi optimizasyon algoritmasını bulmak ve başarısını gözlemlemek için Genetik Algoritma, Parçacık Sürü Optimizasyonu, Guguk Kuşu Arama Algoritması, Gri Kurt Optimizasyonu, Ateş Böceği Algoritması, Yabani Ot Algoritması ve Balina Algoritması ile YSA eğitilerek karşılaştırılmıştır.



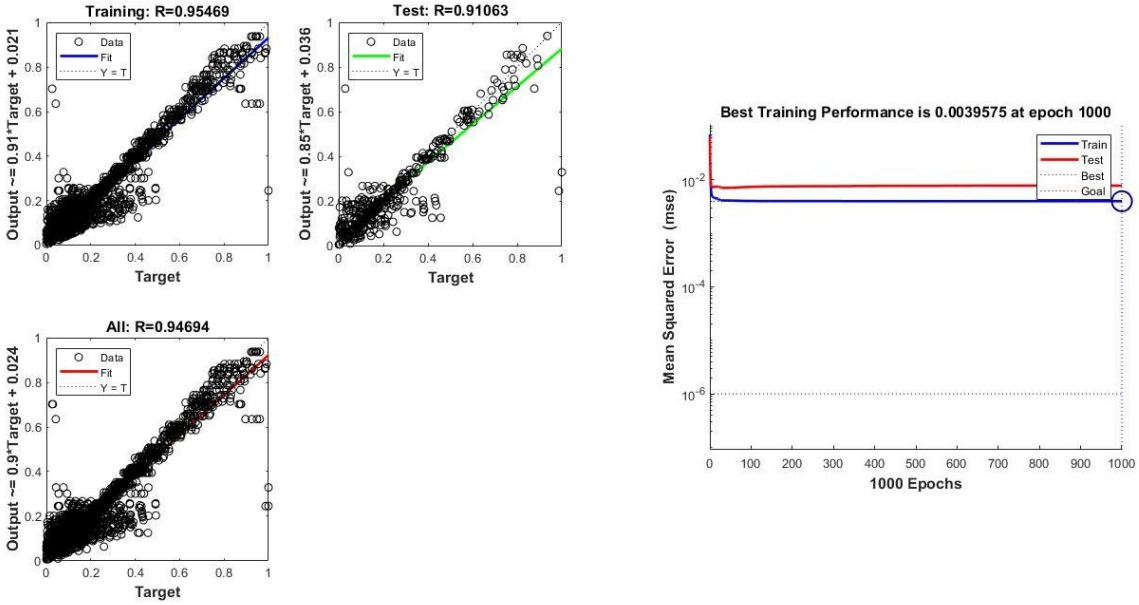

**Şekil 4.6** Önerilen YSA Modelinin Akış Şeması

Şekil 4.6' da optimizasyon algoritması ile oluşturulan YSA modelinin akış şeması gösterilmiştir.

Genetik Algoritma ile YSA modelindeki en iyi optimizasyon algoritması yakınsama eğrisi, regresyon grafiği ve performans grafiği Şekil 4.7 ve 4.8' de gösterilmiştir.

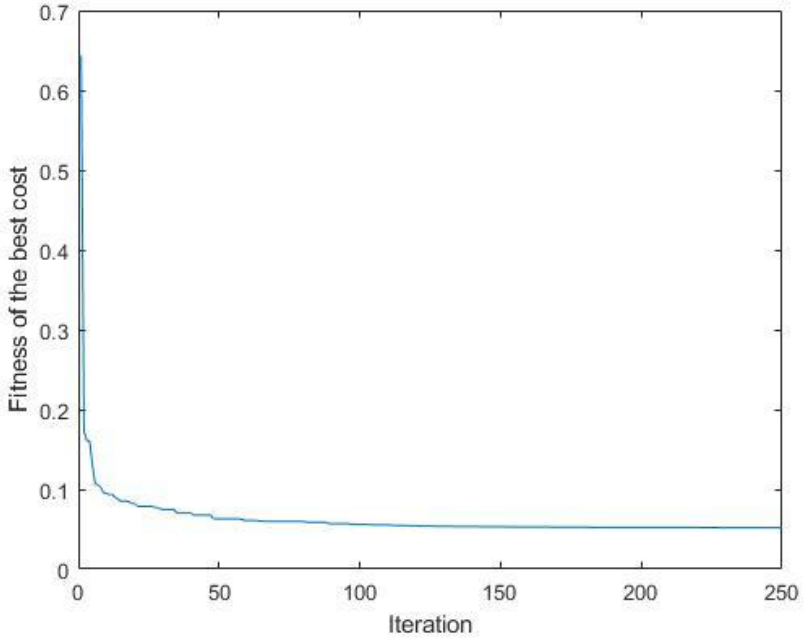



**Şekil 4.7** Genetik Algoritması Yakınsama Grafiği

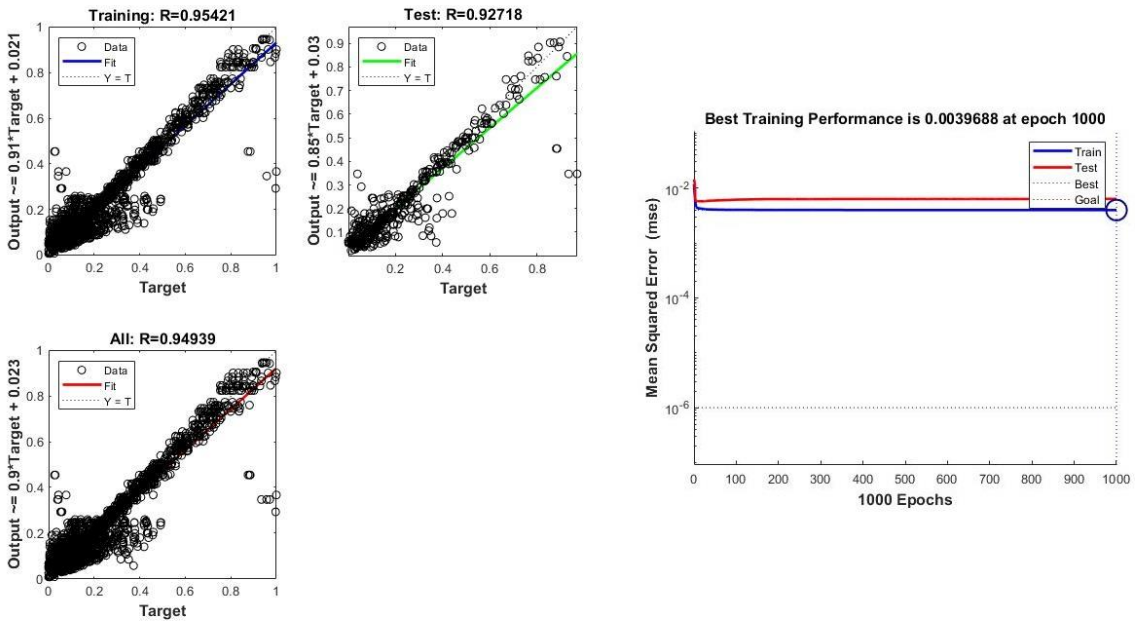



**Şekil 4.8** YSA Regresyon ve Performans Grafiği

Parçacık Sürü Optimizasyonu ile YSA modelindeki en iyi optimizasyon algoritması yakınsama eğrisi, regresyon grafiği ve performans grafiği Şekil 4.9 ve 4.10' da gösterilmiştir.

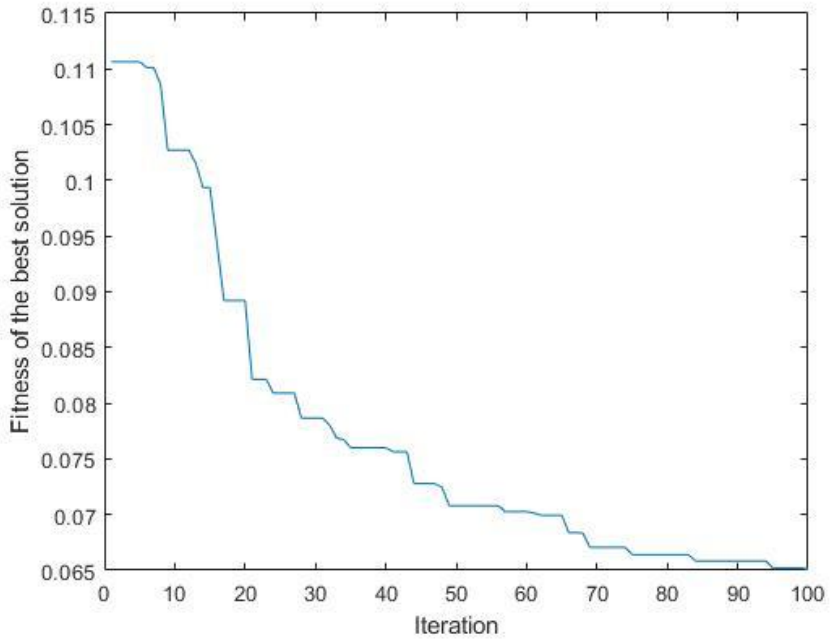



**Şekil 4.9** Parçacık Sürü Optimizasyonu Yakınsama Grafiği

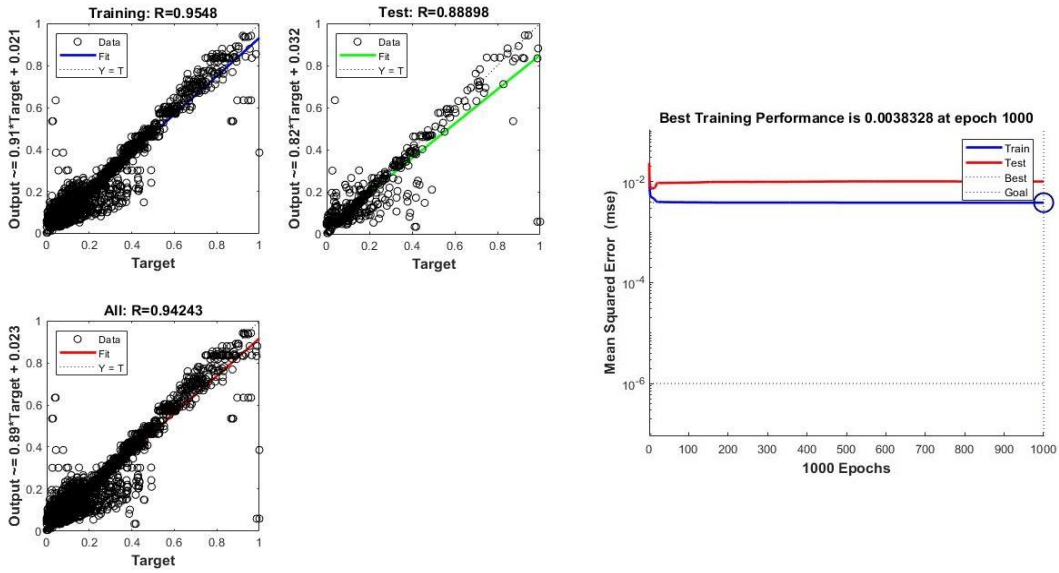



**Şekil 4.10** YSA Regresyon ve Performans Grafiği

Guguk Kuşu Arama Algoritması ile YSA modelindeki en iyi optimizasyon algoritması yakınsama eğrisi, regresyon grafiği ve performans grafiği Şekil 4.11 ve 4.12' de gösterilmiştir.

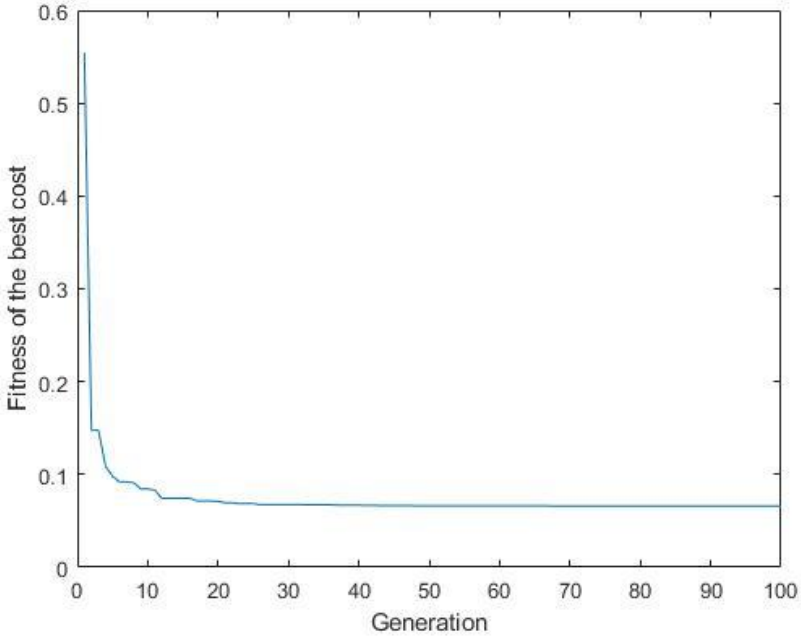



**Şekil 4.11** Guguk Kuşu Arama Algoritması Yakınsama Eğrisi

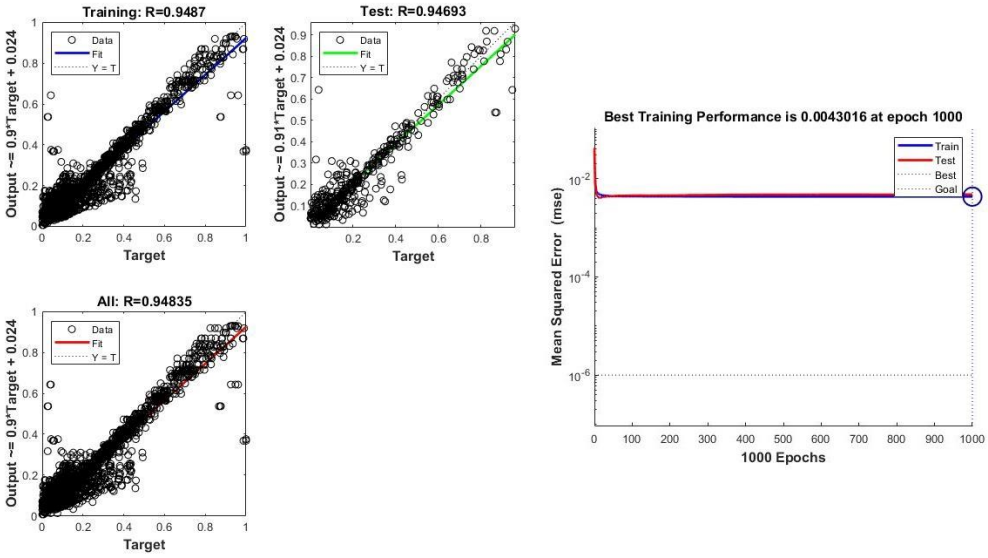



**Şekil 4.12** YSA Regresyon ve Performans Grafiği

Ateş Böceği Algoritması ile YSA modelindeki en iyi optimizasyon algoritması yakınsama eğrisi, regresyon grafiği ve performans grafiği Şekil 4.13 ve 4.14' te gösterilmiştir.

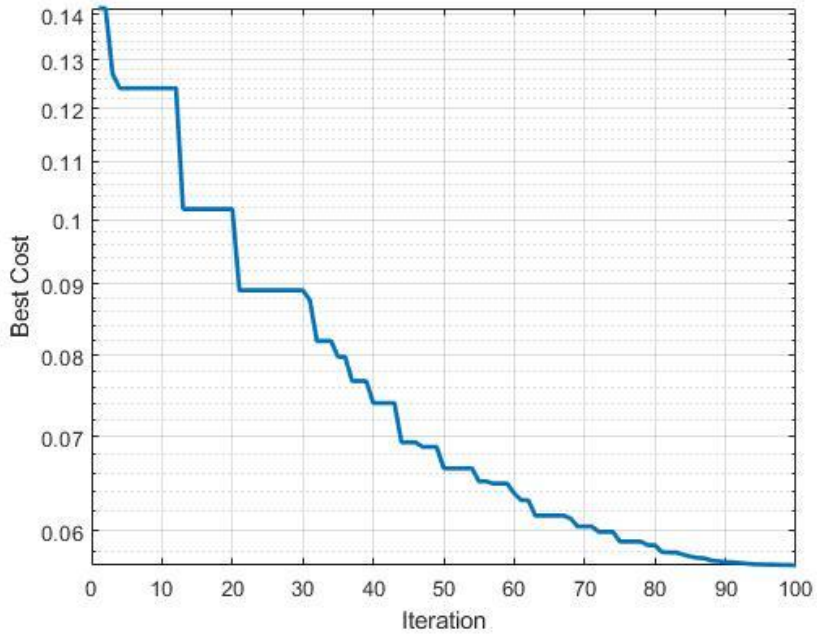



**Şekil 4.13** Ateşböceği Algoritması Yakınsama Eğrisi

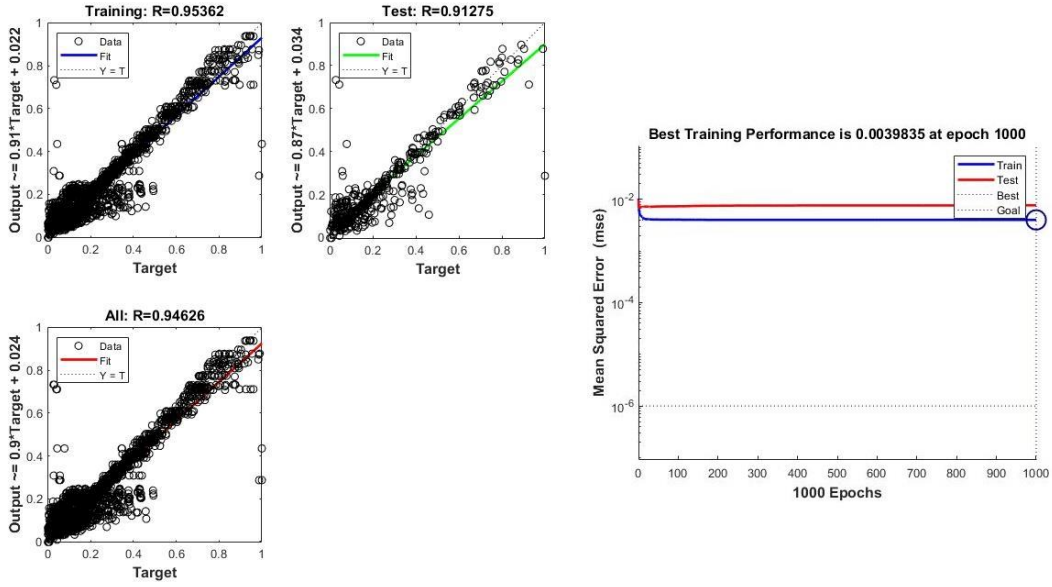



**Şekil 4.14** YSA Regresyon ve Performans Grafiği

Yabani Ot Algoritması ile YSA modelindeki en iyi optimizasyon algoritması yakınsama eğrisi, regresyon grafiği ve performans grafiği Şekil 4.15 ve 4.16' da gösterilmiştir.

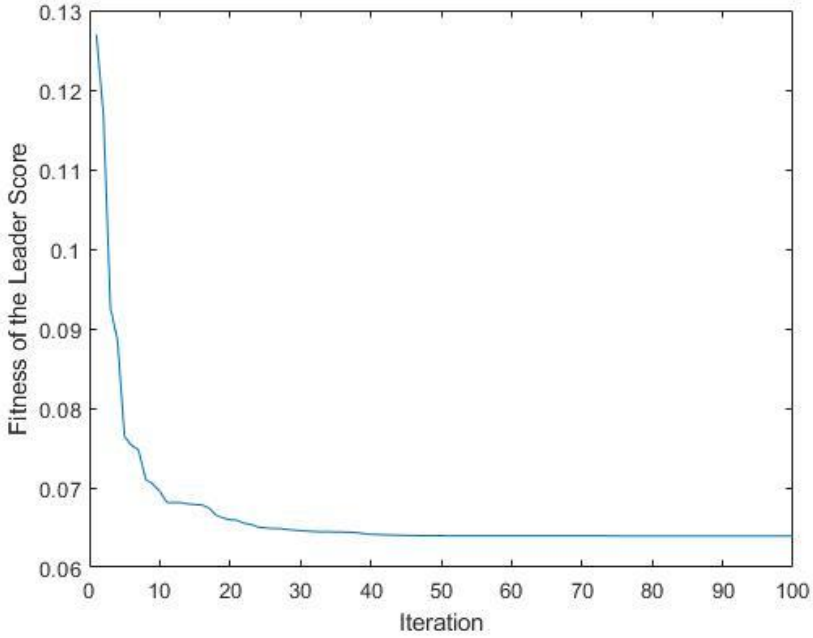



**Şekil 4.15** Yabani Ot Algoritması Yakınsama Eğrisi

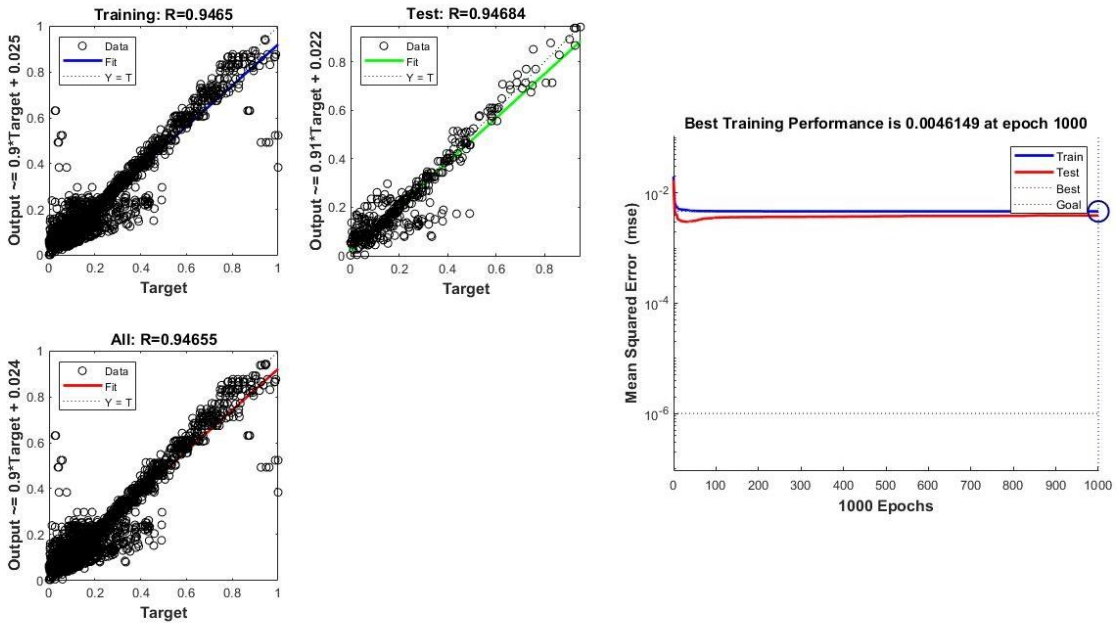



**Şekil 4.16** YSA Regresyon ve Performans Grafiği

Balina Algoritması ile YSA modelindeki en iyi optimizasyon algoritması yakınsama eğrisi, regresyon grafiği ve performans grafiği Şekil 4.17 ve 4.18' de gösterilmiştir.

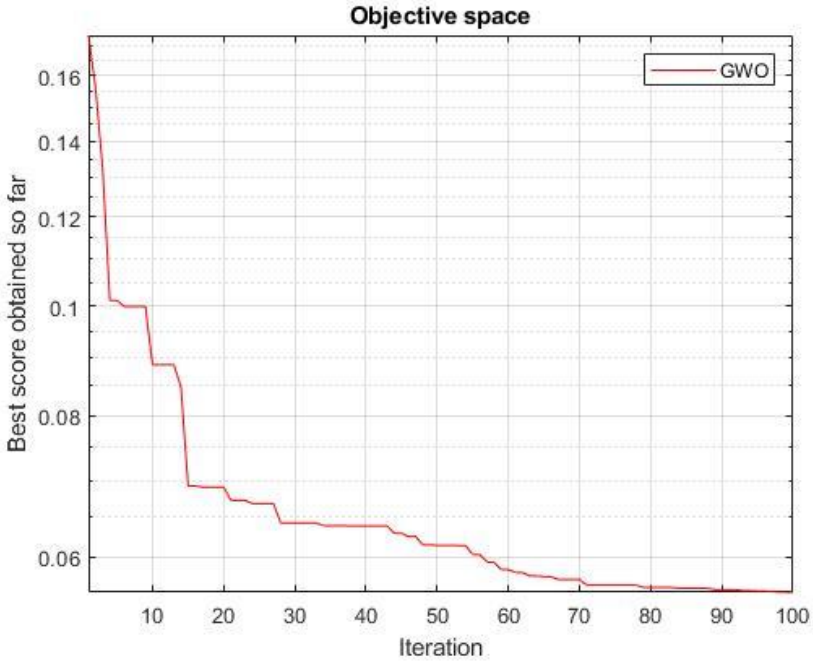



**Şekil 4.17** Balina Optimizasyonu Yakınsama Eğrisi

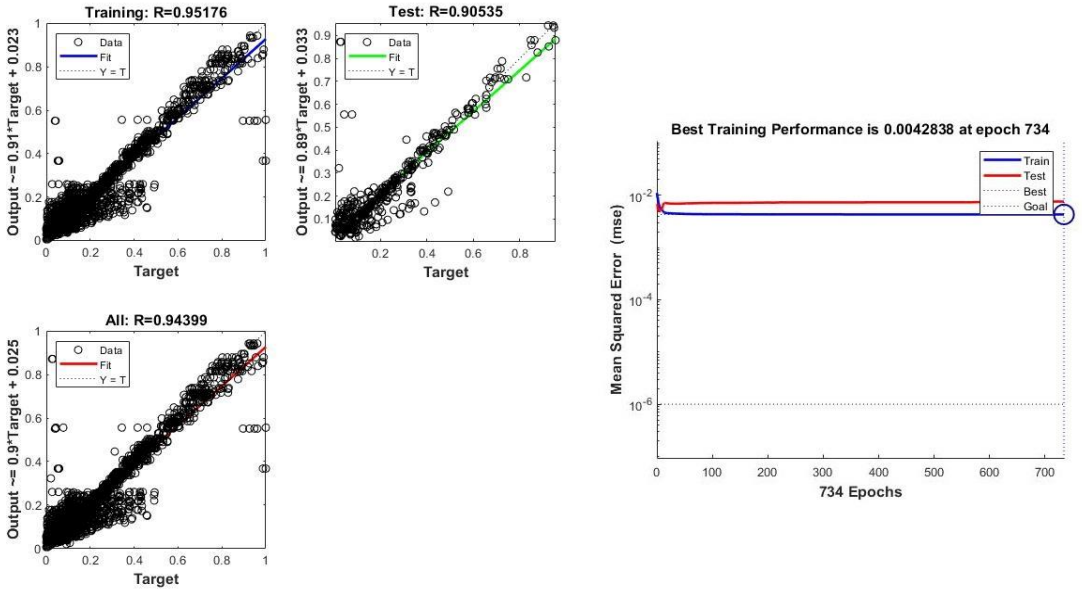



**Şekil 4.18** YSA Regresyon ve Performans Grafiği

Gri Kurt Optimizasyonu ile YSA modelindeki en iyi optimizasyon algoritması yakınsama eğrisi, regresyon grafiği ve performans grafiği Şekil 4.19 ve 4.20' de gösterilmiştir.




**Şekil 4.19** Gri Kurt Optimizasyonu Yakınsama Eğrisi




**Şekil 4.20** YSA Regresyon ve Performans Grafiği

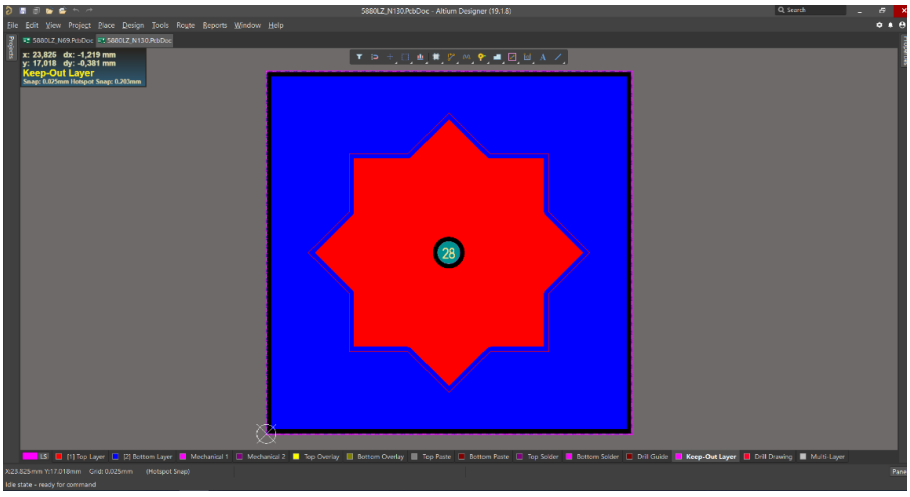
Çizelge 4.5' te 7 farklı optimizasyon algoritması kullanılarak eğitilmiş YSA modelinin performans hatası ve regresyon cevabı karşılaştırılmıştır. Elde edilen sonuçlarda en iyi test verisi değerleri Ateş böceği algoritması ile elde edilmiştir. Diğer algoritmalarında oldukça iyi sonuçlar elde ettiği gözlemlenmiştir.

**Çizelge 4.5:** Optimizasyon Algoritması ile Oluşturulan YSA Modelinin Sonuç Tablosu

| YSA Eğitiminde Kullanılan Optimizasyon Algoritması | Yakınsama Değeri | Performans Hatası | Eğitim Regresyon (%) | Test Regresyon (%) | Tüm Regresyon (%) |
|---|---|---|---|---|---|
| Genetik Algoritma | 0.0529 | 0.0039575 | 95.469 | 91.063 | 94.694 |
| Parçacık Sürü Optimizasyonu | 0.0479 | 0.0039688 | 95.421 | 92.718 | 94.939 |
| Guguk Kuşu Algoritması | 0.0625 | 0.0038328 | 95.480 | 88.898 | 94.243 |
| Ateş Böceği Algoritması | 0.0562 | 0.0043016 | 94.870 | 94.693 | 94.835 |
| Yabani Ot Algoritması | 0.0268 | 0.0039835 | 95.362 | 91.275 | 94.626 |
| Balina Algoritması | 0.0624 | 0.0046149 | 94.650 | 94.684 | 94.655 |
| Gri Kurt Optimizasyonu | 0.0345 | 0.0042838 | 95.176 | 90.535 | 94.399 |

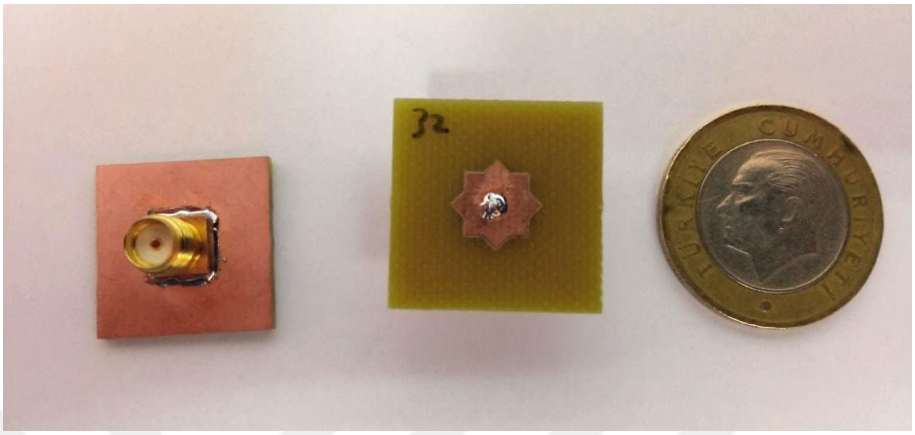
## 4.4. Selçuklu Yıldızı Mikroşerit Anten Üretimi ve Ölçüm Sonuçları

Selçuklu Yıldızı mikroşerit anten tasarımlarında farklı boyutlarda üretilmek üzere Rogers Duroid 5880LZ, Rogers Duroid 5870, Rogers Duroid 6010, FR-4, DE104 ve HRFR-4UV malzemeleri kullanılmıştır. Tasarlanan antenlerde ölçüm yapılması için 50Ω' luk SMA konektör kullanıldığı için, empedans uyumsuzluğu ortadan kaldırmak amacıyla hattın giriş empedansı 50Ω olacak şekilde ayarlanmıştır. Selçuklu Yıldızı mikroşerit antenlerin tasarımında ilk olarak anten çizimi için Altium Designer yazılımı ile baskı devre çizimi gerçekleştirilmiştir. Şekil 4.21' de mikroşerit anten baskı devresi için gerekli kısımlar boyanmış baskı devre çizimi gösterilmiştir.



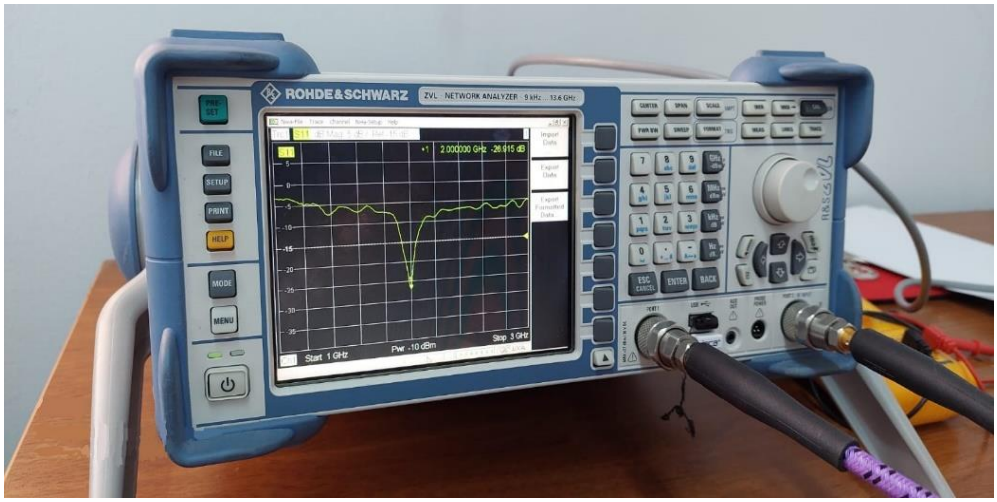

**Şekil 4.21** Selçuklu Yıldızı mikroşerit anten Baskı Devre Şeması

Mikroşerit anten çiziminden sonra Rogers Duroid 5880LZ, Rogers Duroid 5870, Rogers Duroid 6010, FR-4 ve DE104 malzemeleri tasarlanan ölçülere göre kesilmiştir. Selçuklu Yıldızı mikroşerit antenin laboratuvar ortamında gerçekleştirilmiş hali Şekil 4.22' de görüldüğü gibi çizimden plakalara aktarılmıştır.



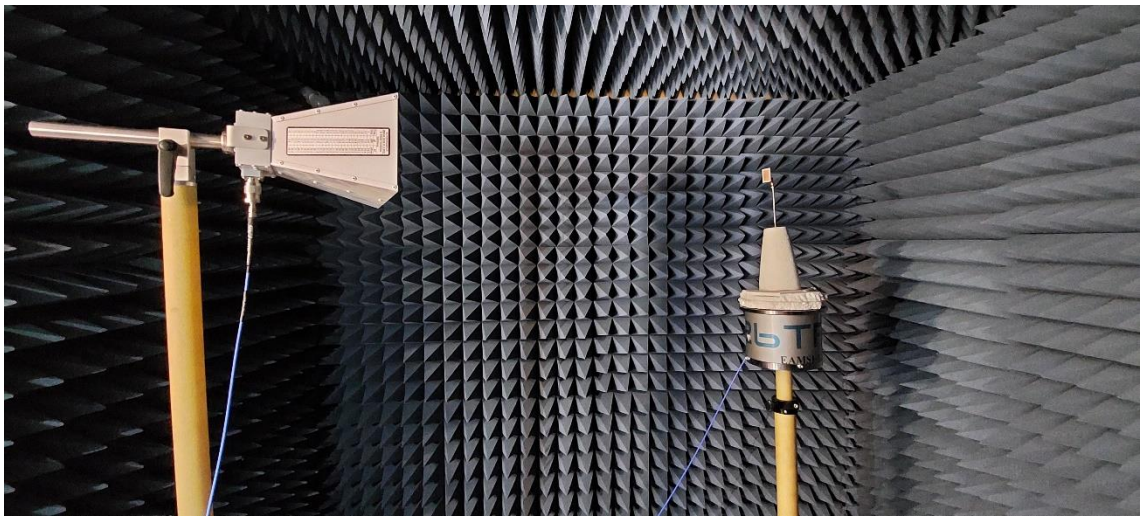

**Şekil 4.22** Selçuklu Yıldızı Mikroşerit Anten

Mikroşerit anten ölçüm sonuçlarının yapılabilmesi için besleme noktalarından delinmiş ve konektörleri bağlanarak hazır hale getirilmiştir. Böylelikle Selçuklu Yıldızı mikroşerit anten tasarımının son aşaması yapılmıştır. Mikroşerit antenlerin ölçümü Rohde Schwarz ZVL-13 Vektör Network Analizör ile normal oda şartlarında gerçekleştirilmiştir.



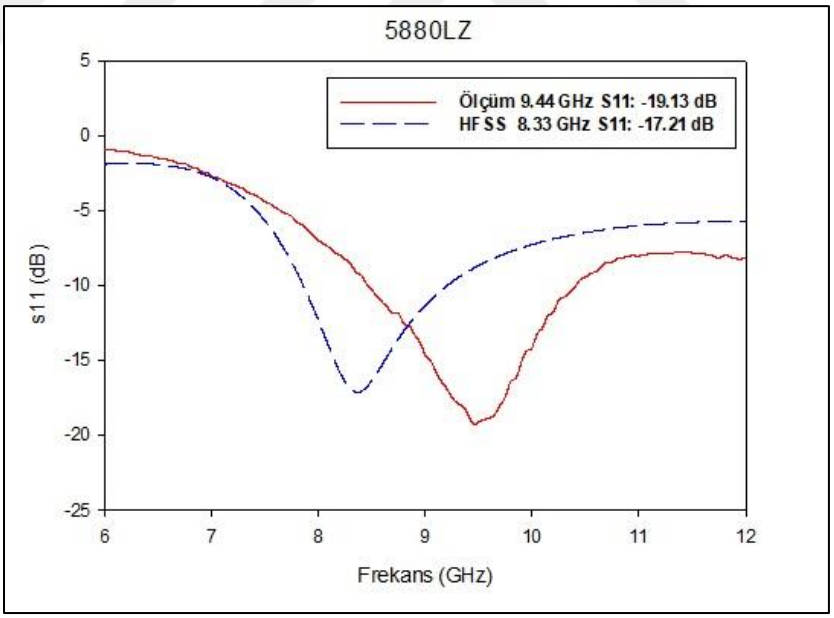

**Şekil 4.23** Rohde Schwarz Sinyal Jeneratörü

Üretilen tüm antenlerin ölçüm sonuçları ile simülasyon sonucu karşılaştırılmıştır. Ayrıca kazanç değerleri yansımasız oda ortamda Vektor Network Analizör (VNA) 'de ölçülmüştür.

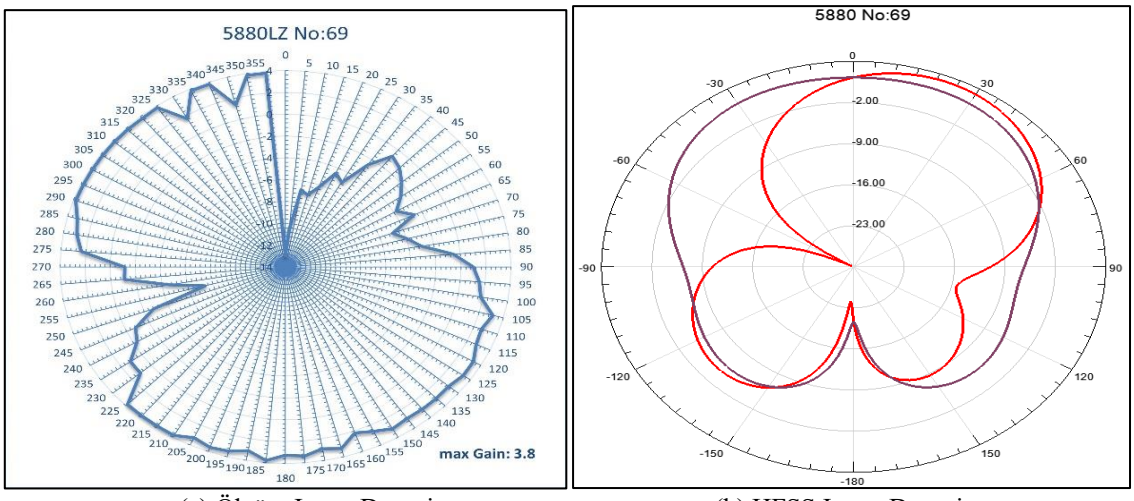


**Şekil 4.24** Mikroşerit Antenin Kazanç Değerinin Ölçülmesi

5880LZ malzemesi ile tasarlanan bir antenin ölçüm ve simülasyon frekansları Şekil 4.25' te gösterilmiştir. Bu antenin rezonans frekansı 9.44 GHz ve $S_{11}$ değeri -19.13 dB olarak ölçülmüştür. Giriş empedansı değeri 51.820 Ω, duran dalga oranı 1.345 olarak ölçülmüştür. Bant genişliği %20.63 olarak hesaplanmıştır.

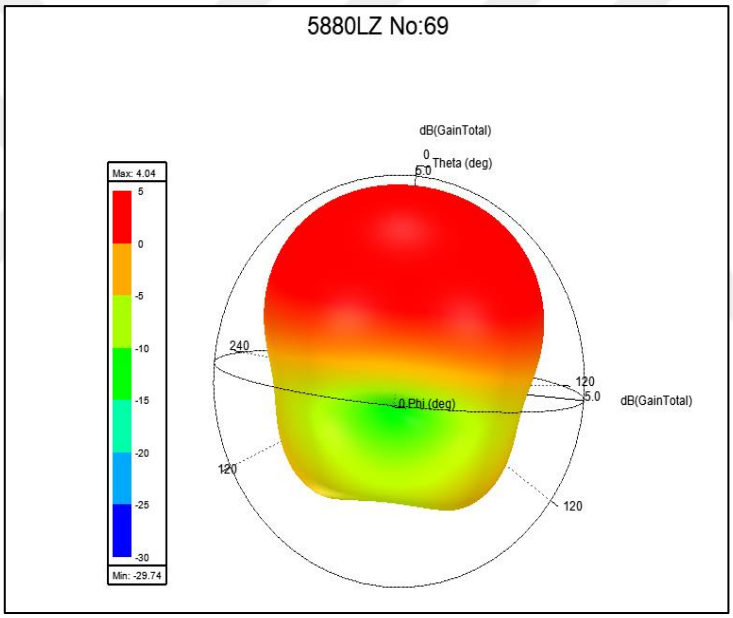



**Şekil 4.25** 5880LZ $S_{11}$ Parametresi Grafiği



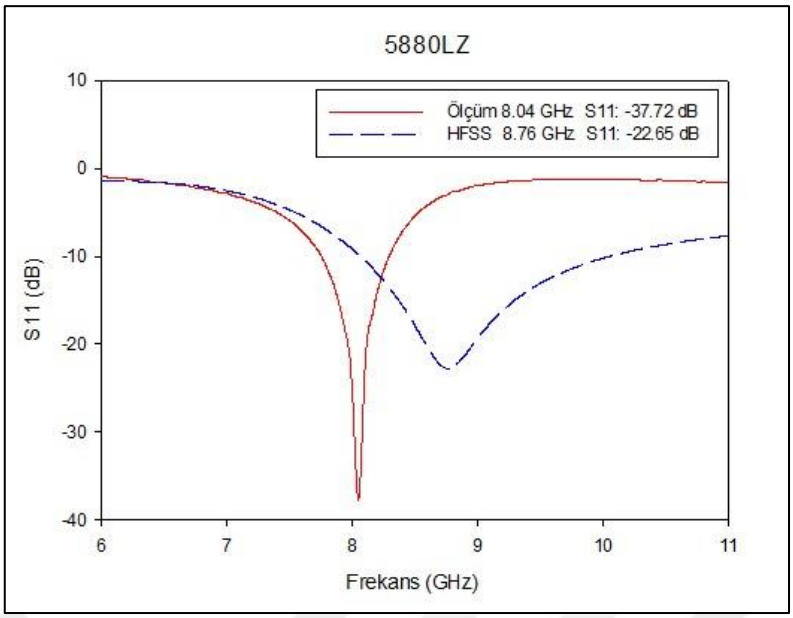

(a) Ölçüm Işıma Deseni (b) HFSS Işıma Deseni

**Şekil 4.26** 5880LZ Kazanç Grafiği

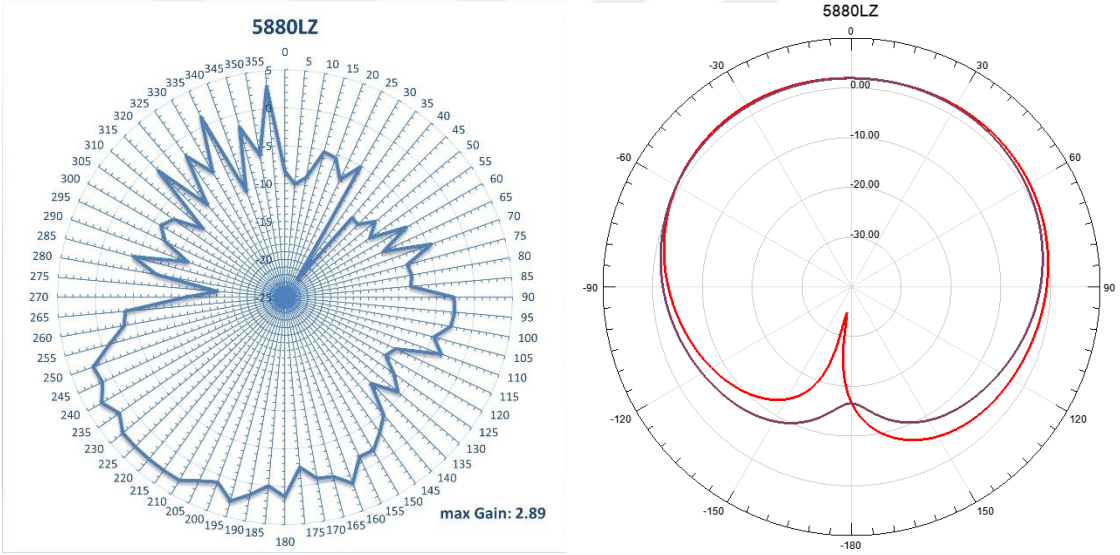



**Şekil 4.27** 5880LZ No:69 Kazanç (dB) cinsinden 3 Boyutlu Kutupsal Grafiği

5880LZ malzemesi ile tasarlanan bir antenin ölçüm ve simülasyon frekansları Şekil 4.28' de gösterilmiştir. Bu antenin rezonans frekansı 8.76GHz ve $S_{11}$ değeri -37.72 dB olarak ölçülmüştür. Giriş empedansı değeri 49.13 Ω, duran dalga oranı 1.148 olarak ölçülmüştür. Bant genişliği %6.83 olarak hesaplanmıştır.

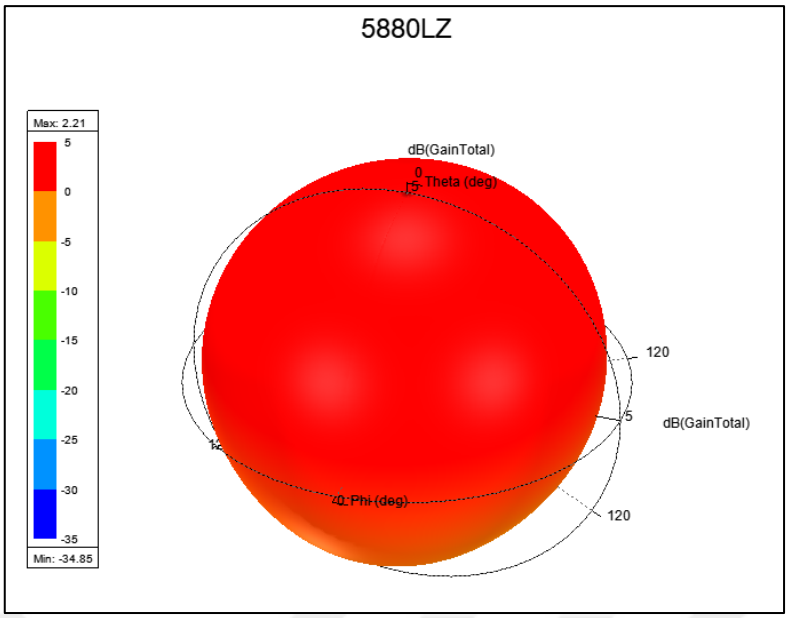



**Şekil 4.28** 5880LZ No:130 $S_{11}$ Parametresi Grafiği



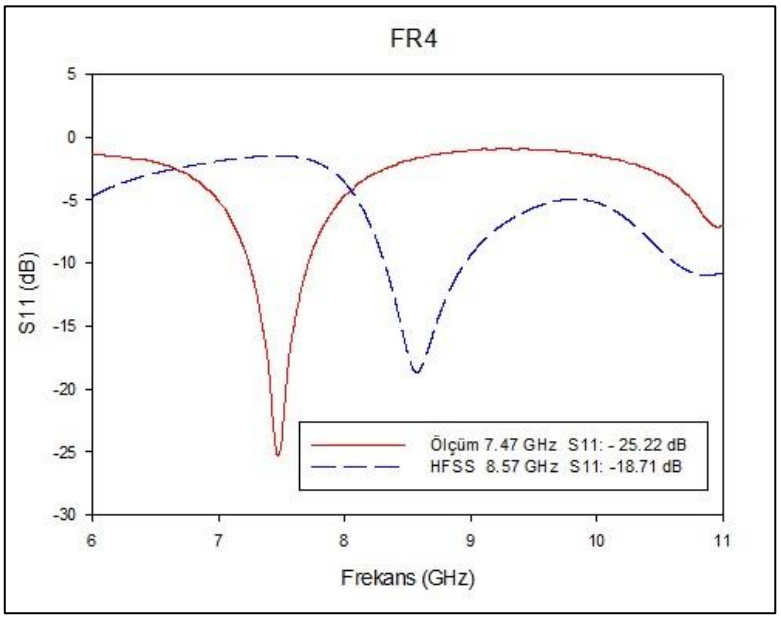

(a) Ölçüm Işıma Deseni (b) HFSS Işıma Deseni

**Şekil 4.29** 5880LZ No:130 Kazanç Grafiği

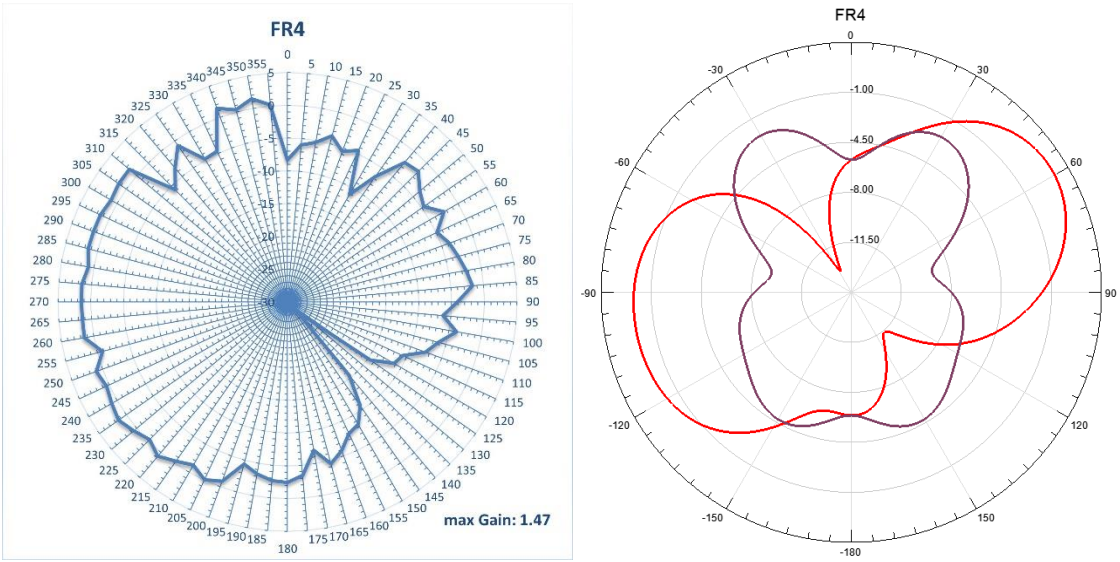



**Şekil 4.30** 5880LZ No:130 Kazanç Grafiği

FR-4 malzemesi ile tasarlanan bir antenin ölçüm ve simülasyon frekansları Şekil 4.31' da gösterilmiştir. Bu antenin rezonans frekansı 7.47 GHz ve $S_{11}$ değeri -25.22 dB olarak ölçülmüştür. Giriş empedansı değeri 47.30 **Ω**, duran dalga oranı 1.017 olarak ölçülmüştür. Bant genişliği %6.04 olarak hesaplanmıştır.

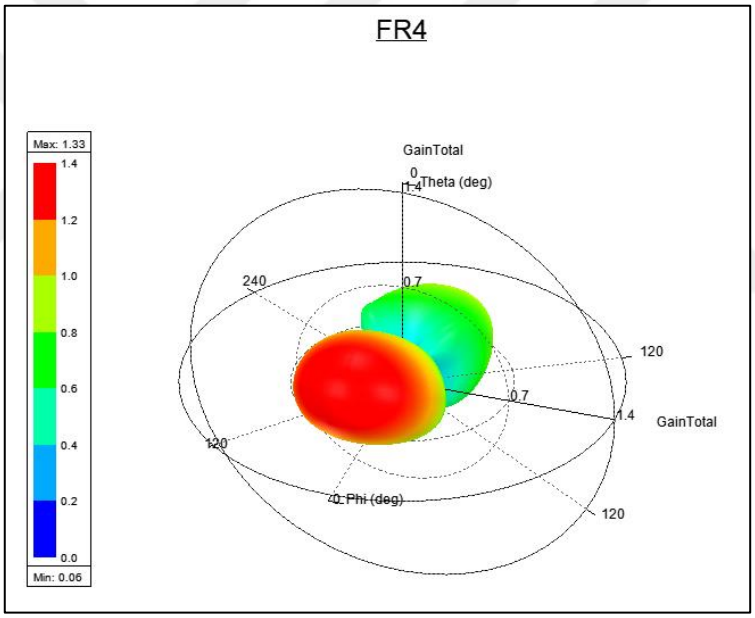



**Şekil 4.31** FR-4 No:217 $S_{11}$ Parametresi Grafiği




(a) Ölçüm Işıma Deseni (b) HFSS Işıma Deseni

**Şekil 4.32** FR-4 No:217 Kazanç Grafiği




**Şekil 4.33** HFSS FR4 No:217 Kazanç Grafiği

**Şekil 4.34** Üretilen SYMŞA Antenler

Tez çalışması kapsamında 0.5GHz – 13.6GHz frekansları arasında simüle edilen 1342 anten içinden beklenen özellikleri karşılayacak 23 adet Selçuklu Yıldızı mikroşerit antenin simülasyon ve ölçüm sonuçları Çizelge 4.6’ da gösterilmiştir. Antenler frekans, geri dönüş kaybı ve giriş empedans parametreleri açısından uyumluluk göstermiştir.

**Çizelge 4.6:** 23 Adet Selçuklu Yıldızı mikroşerit anten veri seti

| | | HFSS Sonuçları | | | | Ölçüm Sonuçları | | | |
|---|---|---|---|---|---|---|---|---|---|
| **No** | **Malzeme_TasarımNo** | **$f_r$ (GHz)** | **$S_{11}$(dB)** | **Z(Ω)** | **VSWR** | **$f_r$ (GHz)** | **$S_{11}$(dB)** | **Z (Ω)** | **VSWR** |
| **1** | 5870_no75 | 7.67 | -7.02 | 56.792 | 1.265 | 7.10 | -22.20 | 53.156 | 1.135 |
| **2** | 5870_no180 | 0.64 | -16.18 | 52.535 | 1.368 | 1.90 | -17.51 | 35.300 | 1.036 |
| **3** | 5880lz_no69 | 8.33 | -17.21 | 51.930 | 1.422 | 9.44 | -19.23 | 51.820 | 1.345 |
| **4** | 5880lz_no130 | 8.76 | -22.65 | 50.320 | 1.439 | 8.04 | -37.72 | 49.130 | 1.148 |
| **5** | 6010_no110 | 7.50 | -22.68 | 40.840 | - | 9.18 | -19.45 | 50.043 | 1.462 |
| **6** | DE104_no156 | 4.90 | -11.23 | 41.270 | 1.756 | 5.33 | -20.76 | 48.822 | 1.214 |
| **7** | FR4_no195 | 9.95 | -10.65 | 43.583 | 1.655 | 8.44 | -11.54 | 42.370 | 1.547 |
| **8** | FR4_no207 | 10.65 | -11.90 | 49.035 | 1.469 | 10.65 | -11.65 | 47.300 | 1.340 |
| **9** | FR4_no217 | 8.57 | -18.71 | 39.100 | 1.860 | 4.12 | 18.34 | 40.330 | 1.017 |
| **10** | HRFR-4UV_no4 | 8.89 | -19.38 | 51.624 | 1.409 | 9.18 | -32.25 | 49.137 | 1.042 |
| **11** | HRFR-4UV_no16 | 2.23 | -16.27 | 53.537 | 1.380 | 1.49 | -24.25 | 43.560 | 1.353 |
| **12** | HRFR-4UV_no32 | 1.60 | -12.93 | 45.972 | 1.583 | 1.97 | -21.20 | 57.270 | 1.461 |
| **13** | HRFR-4UV_no64 | 5.76 | -7.43 | 38.330 | 1.892 | 6.11 | -15.94 | 43.320 | 1.389 |
| **14** | HRFR-4UV_no71 | 1.42 | -14.96 | 50.475 | 1.435 | 1.97 | -24.50 | 51.029 | 1.040 |
| **15** | HRFR-4UV_no78 | 5.33 | -13.21 | 37.850 | 1.922 | 5.77 | -19.75 | 54.950 | 1.043 |
| **16** | HRFR-4UV_no82 | 1.80 | -20.77 | 60.041 | 1.202 | 1.97 | -13.21 | 47.950 | 1.047 |
| **17** | HRFR-4UV_no87 | 1.08 | -13.51 | 47.265 | 1.535 | 2.10 | -17.29 | 56.122 | 1.446 |
| **18** | HRFR-4UV_no103 | 5.76 | -7.49 | 40.541 | 1.785 | 5.77 | -13.15 | 75.922 | 1.601 |
| **19** | HRFR-4UV_no107 | 2.38 | -22.18 | 61.954 | 1.169 | 2.44 | -24.26 | 52.195 | 1.401 |
| **20** | HRFR-4UV_no111 | 1.46 | -14.81 | 50.060 | 1.444 | 1.90 | -12.23 | 36.760 | 1.070 |
| **21** | HRFR-4UV_no128 | 1.06 | -12.52 | 44.545 | 1.620 | 2.10 | -23.71 | 51.020 | 1.203 |
| **22** | HRFR-4UV_no230 | 1.06 | -20.34 | 59.352 | 1.213 | 2.10 | -12.59 | 53.356 | 2.000 |
| **23** | HRFR-4UV_no284 | 14.16 | -41.84 | 55.458 | 1.298 | 10.70 | -17.40 | 53.846 | 1.032 |

# 5. SONUÇLAR VE ÖNERİLER

Bu çalışmada, Selçuklu Yıldızı mikroşerit anten üremi için kullanılan en yaygın malzemelerden anten tasarımları simülasyon sonuçları elde edilmiştir. Tasarımları gerçekleştirilen antenlerin Yapay Sinir Ağları ve optimizasyon algoritmaları ile optimum tasarımı gerçekleştirilebilmesi için yeterli sayıda veriye ulaşılmıştır. Toplamda 1342 adet farklı anten tasarımı simülasyon ortamındaki denemeler sonucunda elde edilmiş olup bu tasarımlardan mümkün olan en fazla sayıda anten üretimi ise laboratuvar ortamında üretilmiştir. Üretilen antenlerin ölçümleri de yine laboratuvar ortamında gerçekleştirilerek bu sonuçlar, simülasyon sonuçlarıyla karşılaştırılmıştır. Mikroşerit anten parametreleri ile oluşturulan veri seti, yapay zekâ yöntemleri kullanılarak analiz edilmiştir.

Yeni oluşturulan ağda geri yayılım algoritması ile bir meta-sezgisel algoritmayı birleştirerek sinir ağı eğitimi için yeni bir algoritma geliştirilmiştir. Bu yeni hibrit eğitim algoritmasında, yerel ve global arama eş zamanlı olarak yapılmıştır. Oluşturulan model farklı optimizasyon algoritmaları ile denenmiş ve sonuçları karşılaştırılmıştır.

Tasarımı gerçekleştirilen mikroşerit antenlerin üretimindeki zorluklar ve hatalar gözlemlenmiştir. Besleme noktalarının değişmesi ile mikroşerit antenin rezonans frekansı, bant genişliği, geri dönüş kaybı ve kazancı değişim göstermiştir. Selçuklu Yıldızı mikroşerit anten için yama boyutlarındaki ve dielektrik tabaka malzemesinin değişmesi anten ölçüm sonuçlarında değişime neden olmaktadır.

Yine kullanılan taban malzemelerinin bir kısmının yüksek standartlarda üretilmemiş olması ve katalog bilgilerinde yer alan dielektrik sabiti ve kayıp tanjantından daha farklı değerlere sahip olması da üretilen antenlerin ölçüm sonuçlarıyla simülasyon sonuçları arasında farklılık yaşanmasına sebep olan faktörlerden biridir. Her malzemenin gerçek dielektrik parametrelerini hızlı ve güvenilir bir şekilde ölçebileceğimiz bir düzenek laboratuvarımızda mevcut olmadığı için katalog bilgilerinden faydalanılarak tasarımlar gerçekleştirilmiştir. Ayrıca antenleri beslemede kullanılan SMA tipi konektörlerin de belli bir standarta sahip olmaması, düşük kalitede olanların montaj esnasında kolayca hasar görebilmesi ve yaşanan stok problemleri sebebiyle her antende aynı tip SMA konektörün kullanılamaması gibi sebepler de yine simülasyon ile ölçüm sonuçlarının arasındaki farkın gerekçelerinden biridir.

Bir başka gerekçe ise, antenlerin bir kısmının profesyonel olarak PCB makinesi ile üretilirken kalan antenler el ile laboratuvarımızda üretilmiştir. El işçiliğindeki farklılıklar ve mm' lik boyut hassasiyetinin yakalanamaması olarak sıralanabilir.

Önerilen YSA modeli ile Selçuklu yıldızı mikroşerit anten tasarımı için bir formül elde edilmesi yönünde çalışmalar ilerletilmesi hedeflenmektedir. Bu sayede kolay üretimi ile rahatlıkla geniş bant uygulamalarında yer alabilecek Selçuklu Yıldızı mikroşerit anten için kendine has nümerik formülü elde edilip tasarlandığı rezonans frekansında ve yüksek bant genişliğine sahip antenlerin çalıştırılması kolay hale gelebilecektir.

Boyut ve tasarım olarak başarısını kanıtlayan bu yeni anten şeklinin, nümerik formülizasyonunun da tamamlanmasıyla literatürde kendine sıklıkla yer bulacağı açıktır. Ayrıca estetik bir görüntüsü olması sebebiyle de görünür uygulamalarda rahatlıkla kullanımı tercih edilebilecektir.

**KAYNAKLAR**

Abbassi, P. k., Badra, N. M., Allam, A. M. M. A. ve El-Rafei, A., 2019, WiFi Antenna Design and Modeling using Artificial Neural Networks, *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, 270-274.

Akbulut, İ., 2008, Parçacık sürü optimizasyonu ile anten tasarımı, *Bilişim Ensititüsü*.

Aljarah, I., Faris, H. ve Mirjalili, S. J. S. C., 2018, Optimizing connection weights in neural networks using the whale optimization algorithm, 22 (1), 1-15.

Aoad, A. ve Tuereli, M. S. U., 2018, Design, simulation, and fabrication of a small size of a new spiral shaped of circular microstrip patch antenna, *Microwave and Optical Technology Letters*, 60 (12), 2912-2918.

Balanis, C. A., 2016, Antenna theory: analysis and design, John wiley & sons, p.

Bayraktar, Ö., Uzer, D., Gültekin, S. S. ve Top, R. J. M. T. P., 2019, Usage of t-resonator method at determination of dielectric constant of fabric materials for wearable antenna designs, 18, 1796-1802.

Bhattacharyya, A., Garg, R. J. I. t. o. a. ve propagation, 1985, Spectral domain analysis of wall admittances for circular and annular microstrip patches and the effect of surface waves, 33 (10), 1067-1073.

Bisht, S., Saini, S., Prakash, V., Nautiyal, B. J. I. J. o. E. T. ve Engineering, A., 2014, Study the various feeding techniques of microstrip antenna using design and simulation using CST microwave studio, 4 (9), 318-324.

Deschamps, G. A., 1953, Microstrip microwave antennas, *Proceedings of the Third Symposium on the USAF Antenna Research and Development Program, Oct*, 18-22.

Deshmukh, A. A., Kudoo, C., Shah, M. ve Chaudhary, V., 2018, Novel Design Of Phi-shape Microstrip Antenna For Circular Polarized Response, *8th International Conference on Advances in Computing & Communications (Icacc-2018)*, 143, 74-79.

Gandomi, A. H., Yang, X.-S., Talatahari, S. ve Alavi, A. H., 2013, Metaheuristic applications in structures and infrastructures, Newnes, p.

Giri, R., Chowdhury, A., Ghosh, A., Das, S., Abraham, A. ve Snasel, V., 2010, A modified invasive weed optimization algorithm for training of feed-forward neural networks, *2010 IEEE International Conference on Systems, Man and Cybernetics*, 3166-3173.

Goldberg, D. E. ve Holland, J. H., 1988, Genetic algorithms and machine learning.

Gollapudi, S. V., Pattnaik, S. S., Bajpai, O., Devi, S., Vidya Sagar, C., Pradyumna, P. K., Bakwad, K. J. I. J. o. R., Manufacturing, M. C. A. E. C. s. b. t. C. f. A., Packaging

of Microwave, O. ve Boulder, D. E. a. t. U. o. C. a., 2008, Bacterial foraging optimization technique to calculate resonant frequency of rectangular microstrip antenna, 18 (4), 383-388.

Guney, K., Sarikaya, N. J. I. T. o. A. ve Propagation, 2007, A hybrid method based on combining artificial neural network and fuzzy inference system for simultaneous computation of resonant frequencies of rectangular, circular, and triangular microstrip antennas, 55 (3), 659-668.

Haykin, S. S., 2009, Neural networks and learning machines/Simon Haykin, New York: Prentice Hall.

He, X. ve Xu, S., 2010, Process neural networks: Theory and applications, Springer Science & Business Media, p.

Howell, J. Q., 1975, Microstrip antennas, *IEEE Transactions on Antennas Propagation*
23, 90-93.

Imeci, S., Yaldiz, E. ve Unlersen, M. J. b. o., 2018, An Empirical Formulation for Estimation of the Seljuk Star Patch Antenna Dimensions for a Given Frequency, 82.

James, J. R., Hall, P. S. ve Wood, C., 1986, Microstrip antenna: theory and design, 12, Iet, p.

Kara, M. J. M. ve Letters, O. T., 1996, Closed-form expressions for the resonant frequency of rectangular microstrip antenna elements with thick substrates, 12 (3), 131-136.

Karaboğa, D. J. N. A. Y., 2014, Yapay Zeka Optimizasyon Algoritmalari, 245.

Karaboga, N., Guney, K. ve Akdagli, A. J. I. j. o. e., 1999, A new effective patch radius expression obtained by using a modified tabu search algorithm for the resonant frequency of electrically thick circular microstrip antennae, 86 (7), 825-835.

Karimkashi, S., Kishk, A. A. J. I. t. o. a. ve propagation, 2010, Invasive weed optimization and its features in electromagnetics, 58 (4), 1269-1278.

Kennedy, J. ve Eberhart, R., 1995, Particle swarm optimization, *Proceedings of ICNN'95-International Conference on Neural Networks*, 1942-1948.

Koç, İ., Nureddin, R. ve Kahramanlı, H., 2018, Türkiye'de enerji talebini tahmin etmek için doğrusal form kullanarak GSA (Yerçekimi Arama Algoritması) ve IWO (Yabani Ot Optimizasyon Algoritması) tekniklerinin uygulanması.

Kumar, G., Gupta, K. J. I. T. o. A. ve Propagation, 1984, Broad-band microstrip antennas using additional resonators gap-coupled to the radiating edges, 32 (12), 1375-1379.

Kumar, G. ve Ray, K. P., 2003, Broadband microstrip antennas, Artech house, p.

Lo, Y., Solomon, D., Richards, W. J. I. t. o. A. ve Propagation, 1979, Theory and experiment on microstrip antennas, 27 (2), 137-145.

Lo, Y. ve Richards, W. J. E. L., 1981, Perturbation approach to design of circularly polarised microstrip antennas, 17, 383-385.

Luzon, M. A. ve Gerasta, O. J., 2018, Slotted Circular Polarized Rectangular Microstrip Patch Antenna with Enhanced Bandwidth for Wireless Communication in 2.45GHz, *2018 Ieee 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (Hnicem)*.

Mallahzadeh, A. R. R., Oraizi, H. ve Davoodi-Rad, Z. J. P. i. E. R., 2008, Application of the invasive weed optimization technique for antenna configurations, 79, 137-150.

Mehrabian, A. R. ve Lucas, C. J. E. i., 2006, A novel numerical optimization algorithm inspired from weed colonization, 1 (4), 355-366.

Mirjalili, S., Mirjalili, S. M. ve Lewis, A. J. A. i. e. s., 2014, Grey wolf optimizer, 69, 46-61.

Mirjalili, S. ve Lewis, A. J. A. i. e. s., 2016, The whale optimization algorithm, 95, 51-67.

Munson, R. J. I. T. o. A., 1974, Conformal microstrip antennas and microstrip phased arrays, *IEEE Transactions on Antennas*
*propagation*, 22 (1), 74-78.

Okkan, U., Köse, O., Özsoy, M. ve Uysal, H., 2017, Yabani ot ve diferansiyel evrim algoritmalarının aylık kavramsal bir yağış-akış modeli kalibrasyonu üzerinden performanslarının irdelenmesi.

Öksüz, H. B., 2019, Cok Serbestlik Dereceli Bir Hidrolik Robot Manipülatörün Deneysel Modellenmesi ve Optimal Kontrolü, *Konya Teknik Üniversitesi*, 112.

Rao, R. V., 2019, Jaya: an advanced optimization algorithm and its engineering applications.

Rathod, S. M., Awale, R. N., Ray, K. P. ve Kakatkar, S. S., 2019, Directivity Enhancement of a Circular Microstrip Antenna with Shorting Post, *Iete Journal of Research*.

Sharma, N. ve Sharma, V., 2018, A design of Microstrip Patch Antenna using hybrid fractal slot for wideband applications, *Ain Shams Engineering Journal*, 9 (4), 2491-2497.

Singh, J., Singh, G., Kaur, S. ve Sohi, B., 2015, Performance analysis of different neural network models for parameters estimation of coaxial fed 2.4 GHz E-shaped

Microstrip patch antenna, *2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS)*, 1-5.

Singh, U. ve Rattan, M. J. P. I. E. R., 2014, Design of linear and circular antenna arrays using cuckoo optimization algorithm, 46, 1-11.

Sivia, J. S., Pharwaha, A. P. S. ve Kamal, T. S. J. P. C. S., 2016, Neurocomputational models for parameter estimation of circular microstrip patch antennas, 85, 393-400.

Thakare, V. V., Singhal, P. J. C. ve Systems, 2011, Artificial Intelligence in the estimation of patch dimensions of Rectangular Microstrip Antennas, 2 (04), 330.

Uzer, D. ve Gultekin, S. S., 2012, A New Seljuk Star Shape Microstrip Antenna Design, *Proceedings of Progress in Electromagnetics Research Symposium (Piers 2012)*, 574-577.

Uzer, D., 2016, Geniş band mikro şerit yama anten tasarımları için uygun yöntemlerin araştırılması, *Selçuk Üniversitesi Fen Bilimleri Enstitüsü*.

Uzer, D., Gültekin, S. S., Top, R., Uğurlu, E., Dündar, Ö. J. I. J. o. A. M., Electronics ve Computers, 2016, A Comparison of Different Patch Geometry Effects on Bandwidth, (Special Issue-1), 421-423.

Vilović, I., Burum, N. ve Brailo, M., 2013, Microstrip antenna design using neural networks optimized by PSO, *ICECom 2013*, 1-4.

Yang, X.-S., 2010a, Firefly algorithm, Levy flights and global optimization, In: Research and development in intelligent systems XXVI, Eds: Springer, p. 209-218.

Yang, X.-S., 2010b, Nature-inspired metaheuristic algorithms, Luniver press, p.

Yang, X.-S., Deb, S. J. I. J. o. M. M. ve Optimisation, N., 2010, Engineering optimisation by cuckoo search, 1 (4), 330-343.

Yang, X.-S., 2013, Cuckoo search and firefly algorithm: theory and applications, Springer, p.

Yang, X.-S., 2017, Nature-inspired algorithms and applied optimization, Springer, p.

Yang, X.-S., 2018, Optimization techniques and applications with examples, John Wiley & Sons, p.

Yang, X. ve Deb, S., 2009, Cuckoo Search via Lévy flights, *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, 210-214.

Yilmaz, A. E., Kuzuoglu, M. J. M. ve Letters, O. T., 2007, Calculation of optimized parameters of rectangular microstrip patch antenna using particle swarm optimization, 49 (12), 2905-2907.

Yohandri, Jumiah, Y. ve Sumantyo, J. T. S., 2018, Design of a C- Band Circular Polarization Microstrip Antenna, *2nd International Conference on Mathematics, Science, Education and Technology*, 335.

Zaharis, Z. D., Skeberis, C., Xenos, T. D., Lazaridis, P. I. ve Cosmas, J. J. I. T. o. B., 2013, Design of a novel antenna array beamformer using neural networks trained by modified adaptive dispersion invasive weed optimization based data, 59 (3), 455-460.

Zhang, P. F., Liu, S. Z., Chen, R. R. ve Huang, X. L., 2016, A Reconfigurable Microstrip Patch Antenna with Frequency and Circular Polarization Diversities, *Chinese Journal of Electronics*, 25 (2), 379-383.

# EKLER

## EK-1 Optimizasyon Algoritması ve Geri Yayılım Algoritması ile Eğitilen Yapay Sinir Ağları MATLAB Kodları

```
clc
tic
 close all
 clear all
% rng default

%% Veri Secimi
%Datasets
 data_name = 'D1';     % Selçuk Yildizi Anten Verileri 1
% data_name = 'D2';    % Selçuk Yildizi Anten Verileri 2
%% Algoritma Secimi
%    algorithm_name = 'F1';       % Genetic Algorithm
%    algorithm_name = 'F2';       % Particle Swarm Optimization
%    algorithm_name = 'F3';       % Cuckoo Search Algorithm
%    algorithm_name = 'F4';       % Grey Wolf Optimizer
%    algorithm_name = 'F5';       % Firefly Algorithm
%    algorithm_name = 'F6';       % Invasive Weed Optimization
%    algorithm_name = 'F7';       % Whale Optimization Algorithm
%% Input and target data file
[data,input,output,l] = Get_Data_Name(data_name);
input = input';
output = output';
%% k fold cross validation
% No_of_folds = 10;
% [test_data,train_data] = KFoldCrossValidation(data,No_of_folds);

%% Data divide
[inputTrain,inputTest,outputTrain,outputTest_r] =
divide(input,output);
%% Normalization
inputTrain = normalization(inputTrain);
inputTest = normalization(inputTest);
outputTrain = normalization(outputTrain);
outputTest = normalization(outputTest_r);

hiddenlayer   = [10 10];

net = feedforwardnet(hiddenlayer,'trainlm');
% net.divideParam.trainRatio = 0.7;  % training set [%]
net.divideParam.valRatio   = 0; % validation set [%]
% net.divideParam.testRatio  = 0.15; % test set [%]
net.trainParam.epochs=1000;
net.trainParam.lr=0.01;
net.trainParam.goal=0.000001;
% net.trainParam.show=50;
% net.trainParam.max_fail=500;
net.layers{1}.transferFcn  = 'tansig';
net.layers{2}.transferFcn  = 'tansig';
net.layers{3}.transferFcn  = 'purelin';
% net.input.processFcns = {'removeconstantrows','mapminmax'};
% net.output.processFcns = {'removeconstantrows','mapminmax'};
net.performFcn = 'mse';
net = configure(net,inputTrain,outputTrain);
net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
      'plotregression', 'plotfit'};
```

```
y = net(inputTrain);
mse = perform(net,outputTrain,y);
alloutputs = sim(net,inputTrain);
h = @(u) myfunc(u,net,inputTrain,outputTrain);
% %% Vector Length Calculation
m = length (inputs(:,1));
o = length (outputs(:,1));
a = length (hiddenlayer);
p1=0;
if a>1
for i=1:a-1
p = (hiddenlayer(1,i)+1)*hiddenlayer(1,i+1);
p1= p1 +p;
end
Nw=(m+1)*hiddenlayer(1)+o*(hiddenlayer(a)+1)+p1; %Nw - number of
weights
else
Nw=(m+1)*hiddenlayer(1)+o*(hiddenlayer(a)+1);
end

%% Optimization
[x,fmin] = optimization_func(algorithm_name,h,wb);
net = setwb(net, x');
%% train
net = train(net,inputTrain,outputTrain);
% % k fold
% %h = @(u) myfunc(u,l,net,inputs,targets);

%% YSA Sonuc
% net = setwb(net, x');
% [b,IW,LW] = separatewb(net,x');
% alpha = 0.1;
% test_input  = test_data(:,1:4)';
% test_output = test_data(:,5:6)';
%
% test_inputs  = normalization(test_input);
% test_outputs = normalization(test_output);


y_test = net(inputTest);

% Calculating the mean squared error
mse_result = perform(net,outputTest,y_test);
testoutput = denormalization(y_test,outputTest_r);
mse_result_real = perform(net,testoutput,outputTest_r);
% figure, plotconfusion(outputTest,y_test,'hybrid')
toc;
```

## EK-2 Genetik Algoritma MATLAB Kodları

```
function [BestChrom] = ga(h,wn,lb,ub)
Problem.obj = h;
Problem.nVar = wn;
M = 100; % number of chromosomes (cadinate solutions)
N = Problem.nVar;  % number of genes (variables)
MaxGen = 50;
Pc = 0.85;
Pm = 0.01;
Er = 0.05;
% Weigth Bounds
% for j=1:Problem.nVar
%     Lb(1,j)= lb;
%     Ub(1,j)= ub;
% end
visualization = 1; % set to 0 if you do not want the convergence curve
[BestChrom]  = GeneticAlgorithm (M , N, MaxGen , Pc, Pm , Er , lb ,
ub , Problem.obj , visualization);
disp('The best chromosome found: ')
BestChrom.Gene
disp('The best fitness value: ')
BestChrom.Fitness
end
function [BestChrom]  = GeneticAlgorithm (M , N, MaxGen , Pc, Pm ,
Er , Lb, Ub, obj, visualization)
cgcurve = zeros(1 , MaxGen);
%%  Initialization
[ population ] = initialization(M, N,Lb,Ub);
for i = 1 : M
    population.Chromosomes(i).fitness =
obj( population.Chromosomes(i).Gene(:) );
end
g = 1;
disp(['Generation #' , num2str(g)]);
[max_val , indx] = sort([ population.Chromosomes(:).fitness ] ,
'descend');
cgcurve(g) = population.Chromosomes(indx(1)).fitness;
%% Main loop
for g = 2 : MaxGen
    disp(['Generation #' , num2str(g)]);
    % Calcualte the fitness values
    for i = 1 : M
        population.Chromosomes(i).fitness =
obj( population.Chromosomes(i).Gene(:) );
    end

    for k = 1: 2: M
        % Selection
        [ parent1, parent2] = selection(population);

        % Crossover
        [child1 , child2] = crossover(parent1 , parent2, Pc,
'single');

        % Mutation
        [child1] = mutation(child1, Pm);
        [child2] = mutation(child2, Pm);

        newPopulation.Chromosomes(k).Gene = child1.Gene;
        newPopulation.Chromosomes(k+1).Gene = child2.Gene;
```

```
    end

    for i = 1 : M
        newPopulation.Chromosomes(i).fitness =
obj( newPopulation.Chromosomes(i).Gene(:) );
    end
    % Elitism
    [ newPopulation ] = elitism(population, newPopulation, Er);

    cgcurve(g) = newPopulation.Chromosomes(1).fitness;

    population = newPopulation; % replace the previous population with
the newly made
end

BestChrom.Gene    = population.Chromosomes(1).Gene;
BestChrom.Fitness = population.Chromosomes(1).fitness;
if visualization == 1
    plot( 1 : MaxGen , cgcurve);
    xlabel('Generation');
    ylabel('Fitness of the best elite')
end
end
%% Initialization
function [ population ] = initialization(M, N,Lb,Ub)
for i = 1 : M
    for j = 1 : N
        population.Chromosomes(i).Gene(j) = Lb+(Ub-
Lb).*rand(size(Lb));
    end
end
end
%% Selection
function [parent1, parent2] = selection(population)
M = length(population.Chromosomes(:));
if any([population.Chromosomes(:).fitness] < 0 )
    % Fitness scaling in case of negative values scaled(f) = a * f + b
    a = 1;
    b = abs( min(  [population.Chromosomes(:).fitness] )  );
    Scaled_fitness = a *  [population.Chromosomes(:).fitness] + b;

    normalized_fitness = [Scaled_fitness] ./ sum([Scaled_fitness]);
else
    normalized_fitness = [population.Chromosomes(:).fitness] ./
sum([population.Chromosomes(:).fitness]);
end
%normalized_fitness = [population.Chromosomes(:).fitness] ./
sum([population.Chromosomes(:).fitness]);
[sorted_fitness_values , sorted_idx] = sort(normalized_fitness ,
'descend');
for i = 1 : length(population.Chromosomes)
    temp_population.Chromosomes(i).Gene =
population.Chromosomes(sorted_idx(i)).Gene;
    temp_population.Chromosomes(i).fitness =
population.Chromosomes(sorted_idx(i)).fitness;
    temp_population.Chromosomes(i).normalized_fitness =
normalized_fitness(sorted_idx(i));
end
cumsum = zeros(1 , M);
for i = 1 : M
```

```
    for j = i : M
        cumsum(i) = cumsum(i) +
temp_population.Chromosomes(j).normalized_fitness;
    end
end
R = rand(); % in [0,1]
parent1_idx = M;
for i = 1: length(cumsum)
    if R > cumsum(i)
        parent1_idx = i - 1;
        break;
    end
end
parent2_idx = parent1_idx;
while_loop_stop = 0; % to break the while loop in rare cases where we
keep getting the same index
while parent2_idx == parent1_idx
    while_loop_stop = while_loop_stop + 1;
    R = rand(); % in [0,1]
    if while_loop_stop > 20
        break;
    end
    for i = 1: length(cumsum)
        if R > cumsum(i)
            parent2_idx = i - 1;
            break;
        end
    end
end
parent1 = temp_population.Chromosomes(parent1_idx);
parent2 = temp_population.Chromosomes(parent2_idx);
end
%% Crossover
function [child1 , child2] = crossover(parent1 , parent2, Pc,
crossoverName)
switch crossoverName
    case 'single'
        Gene_no = length(parent1.Gene);
        ub = Gene_no - 1;
        lb = 1;
        Cross_P = round (  (ub - lb) *rand() + lb  );

        Part1 = parent1.Gene(1:Cross_P);
        Part2 = parent2.Gene(Cross_P + 1 : Gene_no);
        child1.Gene = [Part1, Part2];

        Part1 = parent2.Gene(1:Cross_P);
        Part2 = parent1.Gene(Cross_P + 1 : Gene_no);
        child2.Gene = [Part1, Part2];


    case 'double'
        Gene_no = length(parent1);

        ub = length(parent1.Gene) - 1;
        lb = 1;
        Cross_P1 = round (  (ub - lb) *rand() + lb  );

        Cross_P2 = Cross_P1;
```

```
        while Cross_P2 == Cross_P1
            Cross_P2 = round (  (ub - lb) *rand() + lb  );
        end

        if Cross_P1 > Cross_P2
            temp =  Cross_P1;
            Cross_P1 =  Cross_P2;
            Cross_P2 = temp;
        end
        Part1 = parent1.Gene(1:Cross_P1);
        Part2 = parent2.Gene(Cross_P1 + 1 :Cross_P2);
        Part3 = parent1.Gene(Cross_P2+1:end);

        child1.Gene = [Part1 , Part2 , Part3];


        Part1 = parent2.Gene(1:Cross_P1);
        Part2 = parent1.Gene(Cross_P1 + 1 :Cross_P2);
        Part3 = parent2.Gene(Cross_P2+1:end);

        child2.Gene = [Part1 , Part2 , Part3];
end
R1 = rand();
if R1 <= Pc
    child1 = child1;
else
    child1 = parent1;
end
R2 = rand();
if R2 <= Pc
    child2 = child2;
else
    child2 = parent2;
end
end
%% Mutation
function [child] = mutation(child, Pm)
Gene_no = length(child.Gene);
for k = 1: Gene_no
    R = rand();
    if R < Pm
        child.Gene(k) = ~ child.Gene(k);
    end
end
end
%% Elitism
function [ newPopulation2 ] = elitism(population , newPopulation, Er)
M = length(population.Chromosomes); % number of individuals
Elite_no = round(M * Er);
[max_val , indx] = sort([ population.Chromosomes(:).fitness ] );

% The elites from the previous population
for k = 1 : Elite_no
    newPopulation2.Chromosomes(k).Gene  =
population.Chromosomes(indx(k)).Gene;
    newPopulation2.Chromosomes(k).fitness  =
population.Chromosomes(indx(k)).fitness;
end
% The rest from the new population
for k = Elite_no + 1 :  M
```

```
    newPopulation2.Chromosomes(k).Gene  =
newPopulation.Chromosomes(k).Gene;
    newPopulation2.Chromosomes(k).fitness  =
newPopulation.Chromosomes(k).fitness;
end
end
```

## EK-3 Parçacık Sürü Optimizasyonu MATLAB Kodları

```
function BestSol = pso(h,wn,lb,ub)
CostFunction=h;         % Cost Function
nVar = wn;              % Number of Decision Variables
VarSize=[1 nVar];       % Size of Decision Variables Matrix
VarMin= min(lb);        % Lower Bound of Variables
VarMax= max(ub);        % Upper Bound of Variables
visualization = 1;      % set to 0 if you do not want the convergence
curve
%% PSO Parameters
MaxIt=100;         % Maximum Number of Iterations
nPop=50;          % Population Size (Swarm Size)
% PSO Parameters
% w=1;              % Inertia Weight
% wdamp=0.99;       % Inertia Weight Damping Ratio
% c1=1.5;           % Personal Learning Coefficient
% c2=2.0;           % Global Learning Coefficient
% If you would like to use Constriction Coefficients for PSO,
% uncomment the following block and comment the above set of
parameters.
% % Constriction Coefficients
phi1=2.05;
phi2=2.05;
phi=phi1+phi2;
chi=2/(phi-2+sqrt(phi^2-4*phi));
w=chi;            % Inertia Weight
wdamp=1;          % Inertia Weight Damping Ratio
c1=chi*phi1;      % Personal Learning Coefficient
c2=chi*phi2;      % Global Learning Coefficient
% Velocity Limits
VelMax=0.1*(VarMax-VarMin);
VelMin=-VelMax;
%% Initialization
empty_particle.Position=[];
empty_particle.Cost=[];
empty_particle.Velocity=[];
empty_particle.Best.Position=[];
empty_particle.Best.Cost=[];
particle=repmat(empty_particle,nPop,1);
GlobalBest.Cost=inf;
for i=1:nPop

    % Initialize Position
    particle(i).Position=unifrnd(VarMin,VarMax,VarSize);

    % Initialize Velocity
    particle(i).Velocity=zeros(VarSize);

    % Evaluation
    particle(i).Cost=CostFunction(particle(i).Position);

    % Update Personal Best
    particle(i).Best.Position=particle(i).Position;
    particle(i).Best.Cost=particle(i).Cost;

    % Update Global Best
    if particle(i).Best.Cost<GlobalBest.Cost

        GlobalBest=particle(i).Best;
```

```
    end

end
BestCost=zeros(MaxIt,1);
%% PSO Main Loop
for it=1:MaxIt

    for i=1:nPop

        % Update Velocity
        particle(i).Velocity = w*particle(i).Velocity ...
            +c1*rand(VarSize).*(particle(i).Best.Position-
particle(i).Position) ...
            +c2*rand(VarSize).*(GlobalBest.Position-
particle(i).Position);

        % Apply Velocity Limits
        particle(i).Velocity = max(particle(i).Velocity,VelMin);
        particle(i).Velocity = min(particle(i).Velocity,VelMax);

        % Update Position
        particle(i).Position = particle(i).Position +
particle(i).Velocity;

        % Velocity Mirror Effect
        IsOutside=(particle(i).Position<VarMin |
particle(i).Position>VarMax);
        particle(i).Velocity(IsOutside)=-
particle(i).Velocity(IsOutside);

        % Apply Position Limits
        particle(i).Position = max(particle(i).Position,VarMin);
        particle(i).Position = min(particle(i).Position,VarMax);

        % Evaluation
        particle(i).Cost = CostFunction(particle(i).Position);

        % Update Personal Best
        if particle(i).Cost<particle(i).Best.Cost

            particle(i).Best.Position=particle(i).Position;
            particle(i).Best.Cost=particle(i).Cost;

            % Update Global Best
            if particle(i).Best.Cost<GlobalBest.Cost

                GlobalBest=particle(i).Best;

            end

        end

    end

    BestCost(it)=GlobalBest.Cost;
```

```
    disp(['Iteration ' num2str(it) ': Mean Square Error = '
num2str(BestCost(it))]);
    if visualization == 1
    plot( 1 : MaxIt , BestCost);
    xlabel('Iteration');
    ylabel('Fitness of the best cost')
    end
    w=w*wdamp;

end
BestSol = GlobalBest;

% %% Results
% figure;
% %plot(BestCost,'LineWidth',2);
% semilogy(BestCost,'LineWidth',2);
% xlabel('Iteration');
% ylabel('Best Cost');
% grid on;

end
```

## EK-4 Guguk Kuşu Arama Algoritması MATLAB Kodları

```
function [bestnest,fmin]=cuckoo_search(h,wn,lb,ub)
% Number of nests (or different solutions)
n=100;
% Discovery rate of alien eggs/solutions
pa=0.25;
%% Change this if you want to get better results
N_IterTotal=100;
%% Simple bounds of the search domain
% Lower bounds
Lb=lb*ones(1,wn);
% Upper bounds
Ub=ub*ones(1,wn);
% Random initial solutions
for i=1:n
nest(i,:)=Lb+(Ub-Lb).*rand(size(Lb));
end
% Get the current best
fitness=10^10*ones(n,1);
[fmin,bestnest,nest,fitness]=get_best_nest(nest,nest,fitness,h);
BestCost=zeros(N_IterTotal,1);
N_iter=0;
%% Starting iterations
for iter=1:N_IterTotal
    % Generate new solutions (but keep the current best)
     new_nest=get_cuckoos(nest,bestnest,Lb,Ub);
     [fnew,best,nest,fitness]=get_best_nest(nest,new_nest,fitness,h);
    % Update the counter
      N_iter=N_iter+1;
    % Discovery and randomization
      new_nest=empty_nests(nest,Lb,Ub,pa) ;

    % Evaluate this set of solutions
      [fnew,best,nest,fitness]=get_best_nest(nest,new_nest,fitness,h);
    % Update the counter again
      N_iter=N_iter+1;

    % Find the best objective so far
    if fnew<fmin
        fmin=fnew;
        bestnest=best;
    end
    BestCost(iter)=fmin;
    disp(['Iteration ' num2str(iter) ': Mean Square Error = '
num2str(fmin)]);
    % Çözümleri çizdirme
    plot( 1 : N_IterTotal,BestCost);
    xlabel('Iteration');
    ylabel('Fitness of the best solution')
    %plot(N_IterTotal,fmin)
end

%% End of iterations
%% Post-optimization processing
%% Display all the nests
disp(strcat('Total number of iterations=',num2str(N_iter/2)));
%% --------------- All subfunctions are list below ------------------
%% Get cuckoos by ramdom walk
function nest=get_cuckoos(nest,best,Lb,Ub)
% Levy flights
```

```
n=size(nest,1);
% Levy exponent and coefficient
% For details, see equation (2.21), Page 16 (chapter 2) of the book
% X. S. Yang, Nature-Inspired Metaheuristic Algorithms, 2nd Edition,
Luniver Press, (2010).
beta=3/2;
sigma=(gamma(1+beta)*sin(pi*beta/2)/(gamma((1+beta)/2)*beta*2^((beta-
1)/2)))^(1/beta);
for j=1:n
    s=nest(j,:);
    % This is a simple way of implementing Levy flights
    % For standard random walks, use step=1;
    %% Levy flights by Mantegna's algorithm
    u=randn(size(s))*sigma;
    v=randn(size(s));
    step=u./abs(v).^(1/beta);

    % In the next equation, the difference factor (s-best) means that
    % when the solution is the best solution, it remains unchanged.
    stepsize=0.01*step.*(s-best);
    % Here the factor 0.01 comes from the fact that L/100 should the
typical
    % step size of walks/flights where L is the typical lenghtscale;
    % otherwise, Levy flights may become too aggresive/efficient,
    % which makes new solutions (even) jump out side of the design
domain
    % (and thus wasting evaluations).
    % Now the actual random walks or flights
    s=s+stepsize.*randn(size(s));
   % Apply simple bounds/limits
   nest(j,:)=simplebounds(s,Lb,Ub);
end
%% Find the current best nest
function
[fmin,best,nest,fitness]=get_best_nest(nest,newnest,fitness,h)
% Evaluating all new solutions
for j=1:size(nest,1)
    fnew=h(newnest(j,:));
    if fnew<=fitness(j)
       fitness(j)=fnew;
       nest(j,:)=newnest(j,:);
    end
end
% Find the current best
[fmin,K]=min(fitness) ;
best=nest(K,:);
%% Replace some nests by constructing new solutions/nests
function new_nest=empty_nests(nest,Lb,Ub,pa)
% A fraction of worse nests are discovered with a probability pa
n=size(nest,1);
% Discovered or not -- a status vector
K=rand(size(nest))>pa;
% In the real world, if a cuckoo's egg is very similar to a host's
eggs, then
% this cuckoo's egg is less likely to be discovered, thus the fitness
should
% be related to the difference in solutions.  Therefore, it is a good
idea
% to do a random walk in a biased way with some random step sizes.
%% New solution by biased/selective random walks
stepsize=rand*(nest(randperm(n),:)-nest(randperm(n),:));
```

```
new_nest=nest+stepsize.*K;
for j=1:size(new_nest,1)
    s=new_nest(j,:);
  new_nest(j,:)=simplebounds(s,Lb,Ub);
end
% Application of simple constraints
function s=simplebounds(s,Lb,Ub)
  % Apply the lower bound
  ns_tmp=s;
  I=ns_tmp<Lb;
  ns_tmp(I)=Lb(I);

  % Apply the upper bounds
  J=ns_tmp>Ub;
  ns_tmp(J)=Ub(J);
  % Update this new move
  s=ns_tmp;
```

## EK-5 Gri Kurt Optimizasyonu MATLAB Kodları

```
function [Best_score,Best_pos] = gwo(h,wn,lb,ub)
fobj = h;
SearchAgents_no=50;
Max_iter=100;
% initialize alpha, beta, and delta_pos
Best_pos=zeros(1,wn);
Best_score=inf; %change this to -inf for maximization problems

Beta_pos=zeros(1,wn);
Beta_score=inf; %change this to -inf for maximization problems

Delta_pos=zeros(1,wn);
Delta_score=inf; %change this to -inf for maximization problems

%Initialize the positions of search agents
Positions=initialization(SearchAgents_no,wn,ub,lb);

Convergence_curve=zeros(1,Max_iter);

l=0;% Loop counter

% Main loop
while l<Max_iter
    for i=1:size(Positions,1)

        % Calculate objective function for each search agent
        fitness=fobj(Positions(i,:));

        % Update Alpha, Beta, and Delta
        if fitness<Best_score
            Best_score=fitness; % Update alpha
            Best_pos=Positions(i,:);
        end

        if fitness>Best_score && fitness<Beta_score
            Beta_score=fitness; % Update beta
            Beta_pos=Positions(i,:);
        end

        if fitness>Best_score && fitness>Beta_score &&
fitness<Delta_score
            Delta_score=fitness; % Update delta
            Delta_pos=Positions(i,:);
        end
    end


    a=2-l*((2)/Max_iter); % a decreases linearly fron 2 to 0

    % Update the Position of search agents including omegas
    for i=1:size(Positions,1)
        for j=1:size(Positions,2)

            r1=rand(); % r1 is a random number in [0,1]
            r2=rand(); % r2 is a random number in [0,1]
```

```
            A1=2*a*r1-a; % Equation (3.3)
            C1=2*r2; % Equation (3.4)

            D_alpha=abs(C1*Best_pos(j)-Positions(i,j)); % Equation
(3.5)-part 1
            X1=Best_pos(j)-A1*D_alpha; % Equation (3.6)-part 1

            r1=rand();
            r2=rand();

            A2=2*a*r1-a; % Equation (3.3)
            C2=2*r2; % Equation (3.4)

            D_beta=abs(C2*Beta_pos(j)-Positions(i,j)); % Equation
(3.5)-part 2
            X2=Beta_pos(j)-A2*D_beta; % Equation (3.6)-part 2

            r1=rand();
            r2=rand();

            A3=2*a*r1-a; % Equation (3.3)
            C3=2*r2; % Equation (3.4)

            D_delta=abs(C3*Delta_pos(j)-Positions(i,j)); % Equation
(3.5)-part 3
            X3=Delta_pos(j)-A3*D_delta; % Equation (3.5)-part 3

            Positions(i,j)=(X1+X2+X3)/3;% Equation (3.7)

        end

        % Return back the search agents that go beyond the boundaries
of the search space
        Flag4ub=Positions(i,:)>ub;
        Flag4lb=Positions(i,:)<lb;

Positions(i,:)=(Positions(i,:).*(~(Flag4ub+Flag4lb)))+ub.*Flag4ub+lb.*
Flag4lb;
    end
    l=l+1;
    Convergence_curve(l)=Best_score;
    disp(['Iteration ' num2str(l) ': Best Cost = '
num2str(Convergence_curve(l))]);
end
    display(['The best solution obtained by GWO is : ',
num2str(Best_pos)]);
    display(['The best optimal value of the objective funciton found
by GWO is : ', num2str(Best_score)]);
    semilogy((Convergence_curve),'Color','r')
    hold on
    %semilogy(PSO_cg_curve,'Color','b')
    title('Objective space')
    xlabel('Iteration');
    ylabel('Best score obtained so far');

    axis tight
    grid on
    box on
    legend('GWO')
```

```
end
function Positions=initialization(SearchAgents_no,wn,ub,lb)

Boundary_no= size(ub,2); % numnber of boundaries

% If the boundaries of all variables are equal and user enter a signle
% number for both ub and lb
if Boundary_no==1
    Positions=rand(SearchAgents_no,wn).*(ub-lb)+lb;
end

% If each variable has a different lb and ub
if Boundary_no>1
    for i=1:wn
        ub_i=ub(i);
        lb_i=lb(i);
        Positions(:,i)=rand(SearchAgents_no,1).*(ub_i-lb_i)+lb_i;
    end
end
end
```

## EK-6 Ateş Böceği Algoritması MATLAB Kodları

```
function [nbest,fbest]=firefly(h,wn,lb,ub)

Lb=lb*ones(1,wn);
Ub=ub*ones(1,wn);

% Initial random guess
u0=Lb+(Ub-Lb).*rand(1,wn);
% Check input parameters (otherwise set as default values)
if nargin<5, para=[50 100 0.25 0.20 1]; end % parameters [n
N_iteration alpha betamin gamma]
if nargin<4, Ub=[]; end
if nargin<3, Lb=[]; end
if nargin<2, disp('Usuage: FA_mincon(@cost,u0,Lb,Ub,para)');
end

% n = number of fireflies
% MaxGeneration = number of pseudo time steps
% ------------------------------------------------
% alpha=0.25;      % Randomness 0--1 (highly random)
% betamn=0.20;     % minimum value of beta
% gamma=1;         % Absorption coefficient
% ------------------------------------------------
n=para(1);  MaxGeneration=para(2);
alpha=para(3); betamin=para(4); gamma=para(5);
visualization = 1;
% Total number of function evaluations


% Check if the upper bound & lower bound are the same size
if length(Lb) ~=length(Ub)
    disp('Simple bounds/limits are improper!');
    return
end

% Initial values of an array
zn=ones(n,1)*10^100;
% ------------------------------------------------
% generating the initial locations of n fireflies
[ns,Lightn]=init_ffa(n,wn,Lb,Ub,u0);
BestCost=zeros(MaxGeneration,1);
% Iterations or pseudo time marching
for k=1:MaxGeneration    %%%%% start iterations

% This line of reducing alpha is optional
 alpha=alpha_new(alpha,MaxGeneration);

% Evaluate new solutions (for all n fireflies)
for i=1:n
   zn(i)=h(ns(i,:));
   Lightn(i)=zn(i);
end

% Ranking fireflies by their light intensity/objectives
[Lightn,Index]=sort(zn);
ns_tmp=ns;
for i=1:n
 ns(i,:)=ns_tmp(Index(i),:);
end
```

```
%% Find the current best
nso=ns; Lighto=Lightn;
nbest=ns(1,:); Lightbest=Lightn(1);

% For output only
fbest=Lightbest;
BestCost(k)=fbest;
% Move all fireflies to the better locations
[ns]=ffa_move(n,wn,ns,Lightn,nso,Lighto,nbest,...
      Lightbest,alpha,betamin,gamma,Lb,Ub);
 disp(['Iteration ' num2str(k) ': Mean Square Error = '
num2str(fbest)]);
    if visualization == 1
    plot( 1:MaxGeneration , BestCost);
    xlabel('Generation');
    ylabel('Fitness of the best cost')
    end
end   %%%%% end of iterations

% ----------------------------------------------------
% ----- All the subfunctions are listed here ------------
% The initial locations of n fireflies
function [ns,Lightn]=init_ffa(n,d,Lb,Ub,u0)
  % if there are bounds/limits,
if isempty(Lb)>0
   for i=1:n
   ns(i,:)=Lb+(Ub-Lb).*rand(1,d);
   end
else
   % generate solutions around the random guess
   for i=1:n
   ns(i,:)=u0+randn(1,d);
   end
end

% initial value before function evaluations
Lightn=ones(n,1)*10^100;

% Move all fireflies toward brighter ones
function [ns]=ffa_move(n,wn,ns,Lightn,nso,Lighto,...
             nbest,Lightbest,alpha,betamin,gamma,Lb,Ub)
% Scaling of the system
scale=abs(Ub-Lb);

% Updating fireflies
for i=1:n
% The attractiveness parameter beta=exp(-gamma*r)
   for j=1:n
      r=sqrt(sum((ns(i,:)-ns(j,:)).^2));
      % Update moves
if Lightn(i)>Lighto(j) % Brighter and more attractive
   beta0=1; beta=(beta0-betamin)*exp(-gamma*r.^2)+betamin;
   tmpf=alpha.*(rand(1,wn)-0.5).*scale;
   ns(i,:)=ns(i,:).*(1-beta)+nso(j,:).*beta+tmpf;
      end
   end % end for j

end % end for i
```

```
% Check if the updated solutions/locations are within limits
[ns]=findlimits(n,ns,Lb,Ub);

% This function is optional, as it is not in the original FA
% The idea to reduce randomness is to increase the convergence,
% however, if you reduce randomness too quickly, then premature
% convergence can occur. So use with care.
function alpha=alpha_new(alpha,NGen)
% alpha_n=alpha_0(1-delta)^NGen=10^(-4);
% alpha_0=0.9
delta=1-(10^(-4)/0.9)^(1/NGen);
alpha=(1-delta)*alpha;

% Make sure the fireflies are within the bounds/limits
function [ns]=findlimits(n,ns,Lb,Ub)
for i=1:n
     % Apply the lower bound
  ns_tmp=ns(i,:);
  I=ns_tmp<Lb;
  ns_tmp(I)=Lb(I);

  % Apply the upper bounds
  J=ns_tmp>Ub;
  ns_tmp(J)=Ub(J);
  % Update this new move
  ns(i,:)=ns_tmp;
end
```

## EK-7 Yabani Ot Algoritması MATLAB Kodları

```
function [BestSol] = iwo(h,wn,lb,ub)
CostFunction = @(x) h(x);  % Objective Function
nVar = wn;            % Number of Decision Variables
VarSize = [1 nVar]; % Decision Variables Matrix Size
VarMin = lb;        % Lower Bound of Decision Variables
VarMax = ub;         % Upper Bound of Decision Variables
%% IWO Parameters
MaxIt = 100;     % Maximum Number of Iterations
nPop0 = 10;      % Initial Population Size
nPop = 50;       % Maximum Population Size
Smin = 0;        % Minimum Number of Seeds
Smax = 5;        % Maximum Number of Seeds
Exponent = 2;            % Variance Reduction Exponent
sigma_initial = 0.5;     % Initial Value of Standard Deviation
sigma_final = 0.001;     % Final Value of Standard Deviation
%% Initialization
% Empty Plant Structure
empty_plant.Position = [];
empty_plant.Cost = [];
pop = repmat(empty_plant, nPop0, 1);     % Initial Population Array
for i = 1:numel(pop)

    % Initialize Position
    pop(i).Position = unifrnd(VarMin, VarMax, VarSize);

    % Evaluation
    pop(i).Cost = CostFunction(pop(i).Position);

end
% Initialize Best Cost History
BestCosts = zeros(MaxIt, 1);
%% IWO Main Loop
for it = 1:MaxIt

    % Update Standard Deviation
    sigma = ((MaxIt - it)/(MaxIt - 1))^Exponent * (sigma_initial -
sigma_final) + sigma_final;

    % Get Best and Worst Cost Values
    Costs = [pop.Cost];
    BestCost = min(Costs);
    WorstCost = max(Costs);

    % Initialize Offsprings Population
    newpop = [];

    % Reproduction
    for i = 1:numel(pop)

        ratio = (pop(i).Cost - WorstCost)/(BestCost - WorstCost);
        S = floor(Smin + (Smax - Smin)*ratio);

        for j = 1:S

            % Initialize Offspring
            newsol = empty_plant;
```

```
            % Generate Random Location
            newsol.Position = pop(i).Position + sigma *
randn(VarSize);

            % Apply Lower/Upper Bounds
            newsol.Position = max(newsol.Position, VarMin);
            newsol.Position = min(newsol.Position, VarMax);

            % Evaluate Offsring
            newsol.Cost = CostFunction(newsol.Position);

            % Add Offpsring to the Population
            newpop = [newpop
                      newsol];  %#ok

        end

    end

    % Merge Populations
    pop = [pop
           newpop];

    % Sort Population
    [~, SortOrder]=sort([pop.Cost]);
    pop = pop(SortOrder);
    % Competitive Exclusion (Delete Extra Members)
    if numel(pop)>nPop
        pop = pop(1:nPop);
    end

    % Store Best Solution Ever Found
    BestSol = pop(1);

    % Store Best Cost History
    BestCosts(it) = BestSol.Cost;

    % Display Iteration Information
    disp(['Iteration ' num2str(it) ': Best Cost = '
num2str(BestCosts(it))]);

end
%% Results
figure;
% plot(BestCosts,'LineWidth',2);
semilogy(BestCosts,'LineWidth',2);
xlabel('Iteration');
ylabel('Best Cost');
grid on;
```

## EK-8 Balina Optimizasyonu Algoritması MATLAB Kodları

```
function [Leader_score,Leader_pos]=woa(h,wn,lb,ub)
fobj = h;
dim = wn;
SearchAgents_no=50; % Number of search agents
Max_iter=100; % Maximum numbef of iterations
visualization = 1;
% initialize position vector and score for the leader
Leader_pos=zeros(1,dim);
Leader_score=inf; %change this to -inf for maximization problems

%Initialize the positions of search agents
Positions=initialization(SearchAgents_no,dim,ub,lb);
Convergence_curve=zeros(1,Max_iter);
t=0;% Loop counter
% Main loop
while t<Max_iter
    for i=1:size(Positions,1)

        % Return back the search agents that go beyond the boundaries
of the search space
        Flag4ub=Positions(i,:)>ub;
        Flag4lb=Positions(i,:)<lb;

Positions(i,:)=(Positions(i,:).*(~(Flag4ub+Flag4lb)))+ub.*Flag4ub+lb.*
Flag4lb;

        % Calculate objective function for each search agent
        fitness=fobj(Positions(i,:));

        % Update the leader
        if fitness<Leader_score % Change this to > for maximization
problem
            Leader_score=fitness; % Update alpha
            Leader_pos=Positions(i,:);
        end

    end

    a=2-t*((2)/Max_iter); % a decreases linearly fron 2 to 0 in Eq.
(2.3)

    % a2 linearly dicreases from -1 to -2 to calculate t in Eq. (3.12)
    a2=-1+t*((-1)/Max_iter);

    % Update the Position of search agents
    for i=1:size(Positions,1)
        r1=rand(); % r1 is a random number in [0,1]
        r2=rand(); % r2 is a random number in [0,1]

        A=2*a*r1-a;  % Eq. (2.3) in the paper
        C=2*r2;      % Eq. (2.4) in the paper


        b=1;               %  parameters in Eq. (2.5)
        l=(a2-1)*rand+1;   %  parameters in Eq. (2.5)

        p = rand();        % p in Eq. (2.6)
```

```
        for j=1:size(Positions,2)

            if p<0.5
                if abs(A)>=1
                    rand_leader_index =
floor(SearchAgents_no*rand()+1);
                    X_rand = Positions(rand_leader_index, :);
                    D_X_rand=abs(C*X_rand(j)-Positions(i,j)); % Eq.
(2.7)
                    Positions(i,j)=X_rand(j)-A*D_X_rand;      % Eq.
(2.8)

                elseif abs(A)<1
                    D_Leader=abs(C*Leader_pos(j)-Positions(i,j)); %
Eq. (2.1)
                    Positions(i,j)=Leader_pos(j)-A*D_Leader;      %
Eq. (2.2)
                end

            elseif p>=0.5

                distance2Leader=abs(Leader_pos(j)-Positions(i,j));
                % Eq. (2.5)

Positions(i,j)=distance2Leader*exp(b.*l).*cos(l.*2*pi)+Leader_pos(j);

            end

        end
    end
    t=t+1;
    Convergence_curve(t)=Leader_score;
    disp(['Iteration ' num2str(t) ': Leader Score = '
num2str(Convergence_curve(t))]);
end
    if visualization == 1
    plot( 1 : Max_iter , Convergence_curve);
    xlabel('Iteration');
    ylabel('Fitness of the Leader Score')
    end
end
function Positions=initialization(SearchAgents_no,dim,ub,lb)
Boundary_no= size(ub,2); % numnber of boundaries
% If the boundaries of all variables are equal and user enter a signle
% number for both ub and lb
if Boundary_no==1
    Positions=rand(SearchAgents_no,dim).*(ub-lb)+lb;
end
% If each variable has a different lb and ub
if Boundary_no>1
    for i=1:dim
        ub_i=ub(i);
        lb_i=lb(i);
        Positions(:,i)=rand(SearchAgents_no,1).*(ub_i-lb_i)+lb_i;
    end
end
end
```