T.C.
NECMETTİN ERBAKAN NİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY SİNİR AĞLARININ EĞİTİMİ İÇİN KELEBEK OPTİMİZASYONU ALGORİTMASININ İYİLEŞTİRİLMESİ

Büşra IRMAK

YÜKSEK LİSANS TEZİ

Bilgisayar Mühendisliği Anabilim Dalı

Ocak-2022
KONYA

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.

Büşra IRMAK

Tarih: 17.01.2022

ÖZET

YÜKSEK LİSANS TEZİ

YAPAY SİNİR AĞLARININ EĞİTİMİ İÇİN KELEBEK OPTİMİZASYONU ALGORİTMASININ İYİLEŞTİRİLMESİ

Büşra IRMAK

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Şaban GÜLCÜ

2022, 65 Sayfa

Jüri
Dr. Öğr. Üyesi Şaban GÜLCÜ
Prof. Dr. Sabri KOÇER
Dr. Öğr. Üyesi Semih YUMUŞAK

Kelebek optimizasyonu algoritması (KOA) meta sezgisel bir algoritmadır ve global optimum sorununu çözmek için tasarlanmıştır. KOA, kelebeklerin koku, görme, tat, dokunma ve duyma duyularını kullanarak yiyecek ve çiftleşme eşini bulmalarını rol model alarak tasarlanan bir algoritmadır. Bu duyular ayrıca bir yerden bir yere göç etmek, yırtıcıdan kaçmak ve uygun yerlere yumurta koymak için de yararlıdır. KOA, hiper arama alanında optimumu bulmak için bu davranışı taklit eder. Bu çalışmada çok katmanlı algılayıcıları (ÇKA) eğitmek için iyileştirilmiş kelebek optimizasyonu algoritması (İKOA) önerilmiştir. Çalışmada İKOA algoritmasını geliştirmek için kaostan yararlanılmıştır. Yerel ve global arama denklemleri arasındaki dengeyi kuran p anahtarı kaotik haritalarla her iterasyonda güncellenmiştir. Önerilen İKOA algoritması 13 kıyaslama fonksiyonu üzerinde test edildi ve KOA'dan daha iyi performans gösterdi. Önerilen İKOA algoritması ağırlık ve bias değerlerini optimuma getirmek için kullanılmıştır. İKOA ile eğitilerek geliştirilen ÇKA'ya İKOA-ÇKA ismi verilmiştir. İKOA-ÇKA algoritmasının başarısının doğruluğu literatürde sıkça kullanılan iris, meme kanseri, kalp, balon ve xor veri seti olmak üzere 5 farklı veri seti üzerinde test edilmiştir. Her bir veri seti için farklı ÇKA yapıları kullanılmıştır. İKOA-ÇKA, literatürdeki ÇKA'yı eğitmek için geliştirilen kelebek optimizasyonu algoritması tabanlı KOA-ÇKA, yarasa optimizasyonu algoritması tabanlı BAT-ÇKA, maddenin halleri optimizasyonu algoritması tabanlı SMS-ÇKA ve geri yayılım (GY) olmak üzere dört farklı algoritma ile karşılaştırıldı. Karşılaştırma sonuçlarında İKOA-ÇKA algoritmasının BAT-ÇKA, SMS-ÇKA ve GY algoritmalarını geride bıraktığı, KOA-ÇKA algoritması ile başa baş rekabet ettiği hatta bazı durumlarda öne geçtiği gözlemlenmiştir. Bunun nedeni ise İKOA-ÇKA algoritmasının üstün arama stratejisine ve yerel optimumu atlamadaki üstün yeteneğe sahip olmasıdır.

Anahtar Kelimeler: Çok katmanlı algılayıcılar, Kaos, Kelebek optimizasyonu algoritması, Meta sezgisel, Yapay sinir ağları, Yapay sinir ağları eğitimi.

ABSTRACT

MS THESIS

IMPROVEMENT OF BUTTERFLY OPTIMIZATION ALGORITHM FOR TRAINING OF ARTIFICIAL NEURAL NETWORKS

Büşra IRMAK

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTİN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN COMPUTER ENGINEERING

Advisor: Asst. Prof. Dr. Şaban GÜLCÜ

2022, 65 Pages

Jury
Asst. Prof. Dr. Şaban GÜLCÜ
Prof. Dr. Sabri KOÇER
Asst. Prof. Dr. Semih YUMUŞAK

The butterfly optimization algorithm (BOA) is a meta heuristic algorithm and is designed to solve the problem of the global optimum. The BOA is an algorithm designed as a role model for butterflies to find food and mating mates using their senses of smell, sight, taste, touch and hearing. These senses are also useful for migrating from place to place, escaping from a predator, and laying eggs in suitable places. The BOA mimics this behavior to find the optimum in the hyper search domain. In this study, an improved butterfly optimization algorithm (IBOA) is proposed to train multilayer perceptron (MLP). In the study, chaos was used to develop the IBOA algorithm. The p key, which balances the local and global search equations, is updated at each iteration with chaotic maps. The proposed IBOA algorithm was tested on 13 benchmark functions and it showed better performance than the BOA. The proposed IBOA algorithm is used to optimize the weight and bias values of MLPs. The new algorithm, MLP trained by IBOA was named IBOA-MLP. The performance of the IBOA-MLP algorithm was tested on 5 different data sets, which are frequently used in the literature, namely iris, breast cancer, heart, balloon and xor datasets. The different MLP structures were used for each data set. The IBOA-MLP was compared with four different algorithms in the literature, namely BOA-MLP based on the butterfly optimization algorithm, BAT-MLP based on the bat optimization algorithm, SMS-MLP based on the states of matter search algorithm and back propagation (BP), which were developed to train MLP. In the comparison results, it has been observed that the IBOA-MLP algorithm surpasses the BAT-MLP, SMS-MLP and BP algorithms, and even competes with the BOA-MLP algorithm head-to-head and even gets ahead in some cases. The reason for this is that the IBOA-MLP algorithm has a superior search strategy and superior ability to bypass the local optimum.

Keywords: Artificial neural networks, Butterfly optimization algorithm, Chaos, Meta heuristic, Multilayer perceptron, Training artificial neural networks.

ÖNSÖZ

Yüksek lisans eğitimine başladığım andan itibaren desteğini esirgemeyen, bilgi ve tecrübesi ile bana yol gösteren değerli danışmanım Dr. Öğr. Üyesi Şaban GÜLCÜ'ye, fikirleri ve yardımlarıyla katkı sağlayan hocam Arş. Gör. Dr. Murat KARAKOYUN'a teşekkürlerimi sunarım. Tez çalışmasının son halini alması aşamasında kıymetli fikirleri ile katkıda bulunan jüri üyeleri Sayın Prof. Dr. Sabri KOÇER'e ve Sayın Dr. Öğr. Üyesi Semih YUMUŞAK'a teşekkürlerimi sunarım.

Tez çalışmam boyunca bana maddi ve manevi desteklerini hiçbir zaman esirgemeyen aileme teşekkürlerimi sunarım.

Büşra IRMAK
KONYA-2022

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

ALO	:	Karınca optimizasyonu algoritmasını
APSO	:	Adaptif parçacık sürüsü optimizasyon algoritması
BAT	:	Yarasa optimizasyonu algoritması
CSA	:	Karga arama algoritması
ÇBPSO	:	Çok boyutlu parçacık sürü optimizasyonu
ÇKA	:	Çok katmanlı algılayıcı
DE	:	Diferansiyel evrim
DGO	:	Dinamik grup optimizasyonu
FNN	:	İleri beslemeli sinir ağları
GA	:	Genetik algoritma
GKA	:	Guguk kuşu arama
GSA	:	Yerçekimi arama algoritması
GWO	:	Gri kurt optimizasyonu
GY	:	Geri yayılım
HA	:	Harmoni arama
SMS	:	Maddenin halleri optimizasyonu algoritması
IAMO	:	Havyan göçü optimizasyon
İKOA	:	İyileştirilmiş kelebek optimizasyonu algoritması
KA	:	Böbrek-ilhamlı algoritma
KKO	:	Karınca koloni optimizasyon algoritması
KKO	:	Karınca koloni algoritması
KOA	:	Kelebek optimizasyonu algoritması
PBIL	:	Nüfusa dayalı artımlı öğrenme
PSO	:	Parçacık sürü optimizasyon algoritması
RTS	:	Reaktif tabu araması
SGHS	:	Küresel en iyi uyum arama
SMS	:	Maddenin geçiş halleri algoritması
TB	:	Tavlama benzetimi
WOA	:	Balina optimizasyonu algoritması
YAK	:	Yapay arı koloni algoritması
YSA	:	Yapay sinir ağları

ŞEKİLLER LİSTESİ

ÇİZELGELER LİSTESİ

1. GİRİŞ

Günümüzde bilgisayarlar ve bilgisayar sistemleri yaşamımızla iç içe geçmiş ve ayrılmaz bir parçası haline gelmiştir. Hemen hemen hayatımızın her alanında bilgisayara başvurulmaktadır. Bilgisayarlar, geçmiş yıllarda sadece hesap yaparken ya da veri transferleri gerçekleştirirken zaman içerisinde ilerleme kat edilmiş ve büyük miktardaki verileri çözümleyen, bu verileri kullanarak olaylar hakkında yorumlar yapabilen teknolojik makinelere dönüşmüşlerdir. Günümüzde de bu gelişim devam etmektedir ve bilgisayarlar hem olaylar hakkında karar verebilme hem de olaylar arasındaki ilişkiyi öğrenebilme yeteneklerini kazanmışlardır. Matematiksel olarak formülasyonu kurulamayan ve çözülmesi mümkün olmayan problemler bilgisayarlar tarafından çözülmeye başlanmıştır.

Bilgisayardaki bu ilerlemenin en önemli faktörlerinden biri de yapay zekâ alanıdır. Turing makineleriyle ile birlikte temeli atılmış ve yapay zekâ alanında en fazla araştırmaya sahip konulardan bir tanesi yapay sinir ağlarıdır (YSA) (H Ergezer 2003). İlk yapay sinir ağının modellenmesi çalışması Warren McCulloch ve Walter Pitts tarafından 1943 yılında ortaya atılmıştır.

YSA, insan beyninin hareketini rol model alarak geliştirilmiş kendi alanı içerisinde en önemli icatlardan biridir. YSA sınıflandırma, tanıma, tahmin ve optimizasyon gibi birçok alanda kullanılmaktadır. İnsan beyni taklit edilerek geliştirilen YSA birçok alanda olduğu gibi optimizasyon alanında da başarısını ispat etmiştir.

Günümüzde en güçlü öğrenme metotlarından olan YSA; bilinmeyen fonksiyonları tahmin etmek ve sınıflandırma yapmak için kullanılır. Bu işlemleri yaparken en çok kullanılan YSA yöntemi çok katmanlı algılayıcıdır (ÇKA).

YSA'nın doğru sonuçlar verebilmesi, başarılı sınıflandırmalar yapabilmesi bias ve ağırlık değerlerinin en uygun şekilde güncellenmesi ile sağlanmaktadır. Optimum değerlerle eğitilen YSA sınıflandırma değerlerini bulmada daha fazla doğru sonuca ulaşabilmektedir. Literatürde birçok araştırmacı çok katmanlı algılayıcıları eğitmek için algoritmalar önermişlerdir. Ancak, önerilen bu algoritmalardan olan klasik teknikler gerçek dünyada, optimizasyon problemlerini çözmede sık sık sorunlar ile karşılaşmaktadır. Büyük miktarda hesaplama süresine, büyük miktarda hafızaya ihtiyaç duyma eğilimindedirler. Daha da önemlisi yerel optimum içinde sıkışıp kalmakta ve kalitesiz çözümler üretmektedirler (Mirjalili 2015, Tang ve ark 2018). Bu zorlukların üstesinden gelmek için meta sezgisel algoritmalar YSA'ları eğitmek için kullanılmaya

başlanmıştır (Jaddi ve Abdullah 2018). Canlıların avlanma, üreme ve beslenme gibi biyolojik hareketleri rol model alınarak tasarlanan meta sezgisel algoritmalar, problemlere makul sürelerde çözüm önerisi üreterek optimum sonucu bulmayı hedeflemektedirler. Meta sezgisel algoritmalardan bir tanesi de sürü zekâsı tabanlı olan kelebek optimizasyonu algoritmasıdır (KOA). KOA algoritması Arora ve Singh (2019) tarafından doğadan ilham alınarak geliştirilmiştir.

KOA, kelebeklerin koku, görme, tat, dokunma ve duyma duyularını kullanarak yiyecek ve çiftleşme eşleri bulmalarını rol model alarak tasarlanan bir algoritmadır. Bu duyular ayrıca bir yerden bir yere göç etmek, yırtıcıdan kaçmak ve uygun yerlere yumurta bırakmak için de yararlıdır. Tüm bu duyular arasında koku, kelebeğin uzun mesafelerden bile genellikle nektar gibi yiyecek bulmasına yardımcı olan en önemli duyudur. Çalışmamızda, KOA algoritmasının performansını artırmak için iyileştirilmiş kelebek optimizasyonu algoritması (İKOA) geliştirildi.

Tez çalışması iki kısımdan oluşmaktadır. Tezin ilk kısmında, sürekli optimizasyon problemlerin bir alt dalı olan tek amaçlı optimizasyon kıyaslama problemlerinin çözümü için İKOA algoritması geliştirilmiştir. İKOA algoritması, kelebeklerin yukarıda bahsedilen avlanma, barınma, çiftleşme gibi davranışlarının güçlendirilmesi amaçlanarak KOA algoritmasına parametre analizi ve kaotik haritalar ekleyerek iyileştirilmiş bir algoritmadır. İKOA geliştirilirken Chebyshev, Circle, Gauss, Iterative, Logistic, Piecewise, Sine, Singer, Sinusoidal ve Tent haritası olmak üzere 10 adet kaotik harita kullanılmıştır. Haritalar KOA algoritmasının en önemli parametrelerinden biri olan p (anahtar olasılığı) değerini güncellemek ve optimize hale getirmek için kullanılmıştır. Geliştirilen İKOA algoritması, 13 tane kıyaslama fonksiyonu üzerinde test edilmiştir. Bu fonksiyonlar global optimizasyon üzerine çalışanlar tarafından iyi bilinmektedirler ve genellikle optimizasyon algoritmalarının testi için kullanılmaktadırlar. KOA algoritmanın sonuçları ve İKOA algoritmasının 10 farklı harita ile elde ettiği sonuçlar kıyaslanmıştır. Tent haritalı İKOA algoritmasının daha başarılı olduğu gösterilmiştir.

Tezin ikinci kısmında, YSA'nın eğitimi İKOA algoritması ile gerçekleştirilmiş ve önerilen bu yeni algoritma İKOA-ÇKA olarak adlandırılmıştır. İKOA algoritması YSA'nın optimum bias ve ağırlık değerlerini bulmaya çalışmaktadır. Deneysel çalışma kısmında UCI makine öğrenme deposundan alınan beş veri seti (xor, iris, kalp, balon, meme kanseri) kullanılmıştır. Elde edilen sonuçlar 4 farklı ÇKA algoritması ile kıyaslanmıştır. Kıyaslama sonucuna göre, İKOA-ÇKA algoritmasının iyi bir performans elde ettiği görülmüştür.

Bu tez çalışması 6 ana bölümden oluşmaktadır ve bu bölümler aşağıdaki şekilde organize edilmiştir.

Birinci bölümde, tez çalışmasının konusu hakkında bir giriş yapılmaktadır. Tez çalışmasının amacına ve bu tez çalışmasının literatüre katkılarına değinilmektedir.

İkinci bölümde, tez konusuyla ilgili literatürdeki mevcut çalışmalar incelenmektedir.

Üçüncü bölümde, tez çalışmasında kullanılan materyal ve yöntemler anlatılmaktadır. Öncelikle YSA hakkında detaylı bilgi verilmektedir. Daha sonra, tez çalışmasında geliştirilen İKOA algoritmasının temelini oluşturan KOA algoritması anlatılmaktadır. Devamında ise, deneysel çalışmalarda kullanılan kıyaslama fonksiyonları ve sınıflandırma veri setleri tanıtılmaktadır. Son olarak, tezde kullanılan istatistiksel performans metrikleri hakkında bilgi verilmektedir.

Dördüncü bölümde, tez çalışmasında geliştirilmiş olan İKOA algoritması ve YSA'nın İKOA algoritması ile eğitimi (İKOA-ÇKA) ayrıntılı bir şekilde anlatılmaktadır.

Beşinci bölümde, geliştirilen algoritmaların deneysel sonuçları sunulmaktadır. Öncelikle 10 farklı harita ile İKOA algoritmasının kıyaslama fonksiyonları üzerindeki deneysel sonuçları verilmektedir. Elde edilen sonuçlar ile KOA algoritmasının sonuçları algoritmaların başarısı ve çalışma süreleri açısından karşılaştırılmaktadır ve Tent haritalı İKOA algoritmasının daha başarılı olduğu gösterilmektedir. Bu bölümün ikinci kısmında ise Sine haritalı İKOA-ÇKA algoritmasının deneysel sonuçları sunulmaktadır. Deneyler sınıflandırma problemleri üzerinde gerçekleştirilmiştir ve literatürdeki algoritmaların sonuçları ile karşılaştırılmaktadır. Test örneklerinin çoğunda İKOA-ÇKA algoritmasının diğer algoritmalardan daha başarılı olduğu görülmektedir.

Son olarak altıncı bölümde, tez çalışmasında elde edilen genel sonuçlar özet olarak verilmektedir. Ayrıca, tez konusuyla alakalı ileride yapılabilecek çalışmalar hakkında öneriler verilmektedir.

2. KAYNAK ARAŞTIRMASI

Literatürde YSA'nın eğitimi için birçok meta sezgisel algoritma kullanılmıştır. Bunlardan bazıları bu bölümünde incelenmektedir.

Mirjalili (2015) yaptığı çalışmada çok katmanlı algılayıcı (ÇKA) eğitimi için yakın zamanda önerilen gri kurt optimizasyonu (GKO) algoritmasını kullanmıştır. Önerilen yöntemin testi için beş sınıflandırma ve sekiz standart veri kümesi kullanılmıştır. En iyi bilinen meta sezgisel eğiticilerle kıyaslanmış ve rekabetçi sonuçlar elde ettiği görülmüştür. Aynı zamanda, sınıflandırmada yüksek seviyede başarı elde ettiği de görülmüştür.

Kulluk ve ark (2012) yaptıkları çalışmada sınıflandırma problemleri için sıklıkla kullanılan ileri besleme (FF) tipli YSA'ların denetimli eğitimi için harmoni arama algoritmalarının uygulanmasını ele almışlardır. Çalışmada, kendinden uyarlamalı küresel en iyi uyum arama (SGHS) algoritmasına özel dikkat çekilmiş ve uyum arama algoritmasının beş farklı çeşidi incelenmiştir. YSA'ların veri gösterimine uygun bir yapı SGHS algoritmasına uyarlanmıştır. Teknik, YSA'yı altı kriter sınıflandırma problemi ve gerçek dünya problemi üzerine eğiterek ampirik olarak test etmiş ve doğrulamışlardır.

Ghaleini ve ark (2019) yaptıkları çalışmada istinat duvarlarının emniyet faktörlerini tahmin etmek için YSA'nın, yapay arı kolonisi (YAK) algoritması ile optimize edildiği bir model üretmişlerdir. Emniyet faktörlerinde daha yüksek düzeyde doğruluk ve performans tahmini almak için YSA'nın ağırlık ve bias değerleri YAK algoritması ile optimize edilmiştir. Üretilen model ile ağ performansının güçlendirildiği tespit edilmiştir.

Tang ve ark (2018) yaptıkları çalışmada ileri beslemeli bir sinir ağını eğitmek için yarı enlemesine yarı evrimsel olan dinamik grup optimizasyonu (DGO) algoritmasını kullanmışlardır. Tasarladıkları modele FNN-DGO adını vermişlerdir. DGO; sinir ağının eğitiminde, parametreleri optimum seviyeye getirerek ve ileri beslemeli sinir ağlarının yapısını daha güçlü olarak yapılandırarak bir optimizasyon rolü oynamaktadır. İki gerçek dünya problem ile test edilen ve diğer eğitim algoritmaları ile karşılaştırılan FNN-DGO'nun umut verici sonuçlar sergilediği görülmektedir.

Jaddi ve Abdullah (2018) çalışmasında, böbreklerin insan vücudunda gerçekleştirdiği filtrasyon, salgılama, emilim gibi davranışları rol model alınarak oluşturulan Böbrek-ilhamlı algoritma (KA) algoritmasının geliştirilmiş hali ile YSA parametreleri optimize edilmiştir. Geliştirilen KA algoritmasında minimum ve maksimum değerleri arasında değiştirilen α değeri ile keşif ve sömürü yetenekleri

güçlendirilmiş ve bu da YSA eğitiminde önemli etki yaratmıştır. Önerilen yöntem, kıyaslama sınıflandırılması ve zaman serisi tahmin problem olmak üzere iki probleme uygulanmış ve test edilmiştir. Test sonuçlarının umut verici olduğu görülmektedir.

yaptıkları çalışmada, bir yapay sinir ağının parametrelerini optimize etmek için kooperatif bir ikili-gerçek parçacık sürüsü optimizasyonu algoritması kullanmışlardır. Öncelikle iki nöron arasındaki bağlantıyı kontrol etmek için bağlantı anahtarına sahip bir sinir ağı tanıtılır. Önerilen sinir ağının gelişmiş yapısını ve optimum parametrelerini bulmak için kooperatif ikili-gerçek parçacık sürüsü optimizasyon algoritması kullanılmaktadır. Test sonuçları önerilen algoritmanın rekabetçi sonuçlar verdiğini göstermektedir.

Kiranyaz ve ark (2009) yaptıkları çalışmada, çok boyutlu bir parçacık sürü optimizasyonu (ÇBPSO) yardımı ile bir YSA'nın dizaynı için optimum ağ yapılarını geliştirerek oluşturulmuş bir teknik sunmuştur. Sürü özel boyutlu PSO yardımı ile parçacıkları boyutlar arası geçiş yapacak şekilde tekrar oluşturulur. Bunun sonucunda optimum sonucun bilinmediği arama uzayında parçacıklar hem boyutsal hem konumsal olarak arama işlemi gerçekleştirebilirler. Çok boyutlu parçacık sürüsü optimizasyonu işlemi sonunda yakınsanan optimum boyut, ağ parametrelerinin (bağlantılar, ağırlıklar, biaslar) olduğu eşsiz bir YSA konfigürasyonu işlemine karşılık gelir. Önerilen teknik en zorlu sentetik problemler üzerinde birden fazla rakip teknikle karşılaştırılmış ve ÇBPSO'nun üstün bir genelleme özelliğine sahip olduğu görülmüştür.

Zhang ve Suganthan (2016) yaptıkları çalışmada makine öğrenmesi, istatistik, bilgisayar görüşü gibi alanlarda araştırmacılardan büyük ilgi gören YSA'nın randomize olarak eğitimini ele almıştır. YSA eğitimi için birçok yöntem mevcuttur ancak bu yöntemlerin çoğunda yerel optimuma takılma, yavaş yakınsama gibi problemlerle karşılaşılmaktadır. Bu problemlerin çözümü için randomizasyona dayalı eğitim yöntemleri geliştirilmiştir. Bu çalışma sonraki çalışmalar için önemli bir saha çalışması sunmaktadır.

Ojha ve ark (2017) yaptıkları çalışmada son yirmi yılda popüler olan YSA'nın gelişimini araştırmışlardır. YSA ağırlıklarının optimizasyonu, ağ mimarisi, aktivasyon düğümleri, öğrenme parametreleri, öğrenme ortamı vb. gibi optimize araçları ile optimizasyon yapıldığından bahsetmişlerdir. Geri yayılım gibi gradyan iniş algoritmasının ileri beslemeli sinir ağlarının optimize etmek için yaygın olarak uygulandığından ve başarısının, FNN (ileri beslemeli sinir ağları)'nın uygulamasından sayısız gerçek dünya sorununa açık olduğundan söz etmişlerdir. Bununla birlikte, gradyan

tabanlı optimizasyon yöntemlerinin kısıtlamaları nedeniyle, evrimsel algoritmalar, sürü zekası vb. dahil üst sezgisel algoritmaları, belirli bir problem için genelleştirilmiş FNN elde etmeyi amaçlayan araştırmacılar tarafından geniş çapta araştırıldığı söylemiştir. Kısaca çalışmada FNN'ler için yirmi yıllık bir literatür taraması sunulmuştur.

Heidari ve ark (2020) yaptıkları çalışmada aslan karınca optimizasyonu algoritmasını (ALO) ÇKA ile hibritleyerek ALO-ÇKA algoritmasını sunmuşlardır. ALO bir meta sezgisel algoritmadır. ALO-ÇKA literatürde bulunan ve önemli yeri olan genetik algoritma (GA), nüfusa dayalı artımlı öğrenme (PBIL), parçacık sürü optimizasyonu (PSO) ve diferansiyel evrim (DE) algoritmalarıyla karşılaştırılmış ve diğer algoritmalardan daha iyi sonuç verdiği görülmüştür.

Karaboga ve ark (2007) yaptıkları çalışmada bir optimizasyon problemi olan yapay sinir ağlarını eğitmek için aramada iyi keşif ve sömürü yeteneklerine sahip yapay arı kolonisi (YAK) algoritmasını kullanmışlardır.

Makine öğrenmesindeki en zor süreçlerden biri olarak kabul edilen yapay sinir ağlarının eğitimi sürecinin araştırmacıların radarına girmesi ile birlikte birçok çalışma yapılmıştır. Aljarah ve ark (2018) yaptıkları çalışmada yakın zamanda önerilen sürü zekâsı tabanlı olan balina optimizasyonu algoritması (WOA) ile ileri beslemeli bir yapay sinir ağı eğitilmiştir. Önerilen model farklı zorluk seviyelerine sahip yirmi farklı test verisi ile denenmiş ve doğrulanmıştır. Sonuçlar önerilen yöntemin yerel optimuma takılma ve yakınsama hızı sorunlarını çözmede diğer optimizasyon algoritmalarından daha başarılı olduğunu göstermektedir.

Battiti ve Tecchiolli (1995) yaptıkları çalışmada alt sembolik sistemlerin eğitilmesi görevi, bir kombinatoryal optimizasyon problemi olarak ele alınmış ve reaktif tabu aramasının (RTS) buluşsal şeması ile çözülmüştür. RTS yöntemi, türevlenemez fonksiyonlara uygulanabilir, rastgele başlatmaya göre sağlamdır ve yerel minimumlardan sonra aramaya devam etmede etkilidir. Tekniğin ileri beslemeli ve geri beslemeli sistemler üzerinde üç testi sunulmuş ve başarısı kanıtlanmıştır.

Sexton ve ark (1999) yaptıkları çalışmada, yapay sinir ağının ağırlıklarını bulmak için genelde kullanılan geri bildirim ağının dışında bir yol kullanılmak istenmiş ve geri yayılımın tipik yerel çözüm seçiminin ötesine geçmek için, küresel olarak araştıracak alternatif bir eğitim tekniği olarak benzetilmiş tavlama önerilmiştir. Bu çalışmada, yedi test fonksiyonu üzerinde yoğun bir Monte Carlo çalışması yoluyla geri yayılım doğrudan bu küresel arama tekniğiyle karşılaştırılmış ve başarılı sonuçlar elde edilmiştir.

Mendes ve ark (2002) yaptıkları çalışmada grup etkileşiminin sosyal dinamiklerine dayanan gerçek değerli işlevler için bir optimizasyon paradigması olan parçacık sürüsü optimizasyon algoritması kullanılmıştır. Çalışmada söz edilen algoritmanın sinir ağlarının eğitimine uygulanması önerilmiştir. Sınıflandırma ve regresyon görevleri için karşılaştırmalı testler yapılmış ve başarılı sonuçlar elde edilmiştir.

Sürekli arama uzayları üzerinde evrimsel bir optimizasyon yöntemi olan DE algoritması son zamanlarda gerçek dünya optimizasyon problemlerine başarıyla uygulandığı ve yapay sinir ağı eğitimi için de önerildiği görülmüştür. Ek olarak, DE algoritması, sinir ağı ağırlıklarının eğitimi bağlamında kapsamlı bir şekilde çalışılmış ve diferansiyel evrimin yakınsama hızının maliyeti için küresel optimumu bulmada yararlı algoritma olduğu kanıtlanmış bir algoritmadır. Ilonen ve ark (2003) yaptıkları çalışmada, ileri beslemeli sinir ağları için aday bir küresel optimizasyon yöntemi olarak diferansiyel evrim analiz edilmiştir. Gradyan tabanlı yöntemlerle karşılaştırıldığında, öğrenme oranı veya çözüm kalitesi açısından belirgin bir avantaj sağlamıyor gibi görünüyor olsa da DE'nin daha çok, ulaşılan optimumun doğrulanmasında ve gradyan bilgisi sağlamayan düzenlileştirme terimlerinin ve geleneksel olmayan transfer fonksiyonlarının geliştirilmesinde kullanılabileceği sonucuna varılmıştır.

Blum ve Socha (2005) çalışmalarında, sağlık alanındaki yapılan çalışmalarda en sık kullanılan üç veri kümesini ele almıştır ve sınıflandırma işlemini başarılı bir şekilde gerçekleştirebilmek amacıyla karınca koloni algoritması (KKO) kullanmış ve sinir ağını eğitmiştir. Bu çalışma KKO algoritması optimizasyon tekniği olarak kullanılabilir ve başarısı kanıtlanmış bir algoritma olarak belirlenmiştir.

Zanchettin ve ark (2011) yaptıkları çalışmada GA, tabu arama algoritması ve tavlama benzetimi algoritmalarının birleşiminden meydana gelmiş hibrit algoritmayı YSA eğitimi için kullanmışlardır. Çalışmada kullanılan üç algoritmanın veri kümelerine ayrı ayrı uygulandığında elde edilen başarı oranının, hibrit algoritmanın başarısından daha düşük olduğu sonucu elde edilmiştir.

Zhang ve ark (2007) yaptıkları çalışmada adaptif parçacık sürüsü optimizasyon algoritmasını (APSO) ve geleneksel geri-yayılım algoritmasını kullanarak hibrit algoritma oluşturmuş ve bu hibrit algoritmayı YSA'nın eğitimi için kullanmışlardır ve çeşitli veri kümeleri kullanarak test etmişlerdir. Elde edilen sonuçlara göre oluşturulmuş hibrit algoritma her iki algoritmayı da performans ve yakınsama hızı olarak geride bırakmıştır.

Özbakir ve ark (2009) sınıflandırmada verileri arasındaki gizli bilgileri öğrenebilmek ve kural çıkarımı yapmak amacıyla KKO algoritması tabanlı yeni bir meta sezgisel algoritma geliştirmiş ve bu algoritma ile YSA eğitimi gerçekleştirmişlerdir. Veri kümeleri ile elde edilen deney sonuçları ele alındığında, önerilen yeni modelin doğru keşif yapabildiği ve başarılı olduğu sonucuna ulaşılmıştır.

Zamani ve Sadeghian (2010), yaptıkları çalışmada yapay sinir ağları eğitimi için PSO algoritmasını kullanmışlardır. Sınıflandırma yapmak amacıyla oluşturdukları sistem farklı veri kümeleri üzerinde test edilmiş ve başarı oranları ele alınarak incelenmiştir. Bu çalışma ile YSA eğitiminde yüksek başarılı sonuçlar elde edildiği gözlemlenmiştir.

Vazquez (2011) yaptığı çalışmada, YSA eğitimini guguk kuşu arama (GKA) algoritması kullanarak gerçekleştirmiş ve beş farklı veri seti üzerinde test etmiştir. Dört veri kümesinde istenilen başarı oranı yakalanmış ancak bir tanesinde istenilen sonuçlara ulaşılamamış ve başarısız olmuştur.

Mirjalili ve ark (2012), çalışmalarında yerçekimi arama algoritması (GSA) ve PSO algoritmasının birleşiminden oluşan bir hibrit algoritma geliştirmişlerdir. Geliştirilen bu yeni hibrit model YSA eğitimi için kullanılmıştır. Çalışma sonucunda ise elde edilen hibrit modelin yakınsama hızında ve doğruluk oranlarında yüksek seviyede başarı elde edildiği gözlemlenmiştir.

Kulluk (2013) yaptığı çalışmada, harmoni arama (HS) ve av arama (HuS) algoritmalarını birleştirerek yeni bir hibrit meta sezgisel algoritma elde etmiştir. Bu hibrit algoritma yapay sinir ağının eğitimi için kullanılmıştır. Eğitimin başarısı, sekiz farklı veri seti kullanılarak ve diğer meta sezgisel algoritmalar ile karşılaştırılarak gözlemlenmiştir. Çalışmada kullanılan hibrit model geleneksel, literatürde başarısı çokça konuşulan çoğu algoritmayı geride bırakarak başarısını kanıtlamıştır.

Pereira ve ark (2014) çalışmalarında ise, sinir ağı eğitimi için sosyal örümcek algoritması kullanmışlardır. Elde edilen algoritmayı sağlık verilerinin sınıflandırması amacı ile kullanmışlardır. Elde edilen bu algoritma yüksek başarı elde edemese de çoğu algoritma ile rekabetçi sonuçlar sergilemiş ve ortalama bir başarı seviyesini tutturmuştur.

Chen ve ark (2015) yaptıkları çalışmada YSA eğitimi için PSO ve GKA algoritmalarını başarılı bir şekilde birleştirerek hibrit bir model oluşturmuş ve eğitim için kullanmışlardır. Her iki algoritmanın başarılı yönleri ele alınarak birleştirilmiş ve geliştirilmiştir. Elde edilen algoritma fonksiyon tahmini ve sınıflandırma durumlarında kullanılmıştır. Başarı oranı incelenirken GKA ve PSO algoritmalarının salt hallerinin

başarı oranları alınmış ve yeni elde edilen hibrit modelin başarısı ile kıyaslanmıştır. Sonuç olarak hibrit modelin başarısı diğer iki algoritmayı geride bırakmıştır.

Sarangi ve ark (2014) YAK algoritmasını YSA eğitiminde kullanmışlardır. Testler beş veri seti üzerinde yapılmıştır. Çalışmada önerilen YAK algoritmasının sinir ağı eğitimi diğer algoritmalarla kıyaslandığında başarısının daha yüksek olduğu gözlemlenmiştir.

Al Nuaimi ve Abdullah (2017) PSO ve YAK algoritmasını birleştirerek yeni bir hibrit algoritma önermişlerdir. Önerilen bu hibrit meta sezgisel algoritma sinir ağının eğitimi için kullanılmıştır. Elde edilen sonuçlarda hibrit modelin diğer iki algoritmanın sonuçlarından daha iyi sonuçlar ettiği gözlemlenmiş ve hibrit model başarısını ispatlamıştır.

Gülcü (2021) yaptığı çalışmada, YSA eğitimi için gradyan yaklaşımların dezavantajları olduğunu açıklayarak meta sezgisel bir yaklaşım olan havyan göçü optimizasyon göçünü Levy uçuş özelliği ile geliştirmiş ve önermiştir. Önerilen hibrit algoritma, IAMO-MLP olarak adlandırmıştır. Geliştirilen algoritmanın yerel optimadan başarılı bir şekilde kaçtığı ve başlangıç konumlarının performansı etkilemediği gözlemlenmiştir.

Dash (2018) yapmış olduğu çalışmada, sinir ağı eğitimi için geliştirilmiş sıçrayan kurbağa algoritmasını kullanmıştır. Oluşturulan algoritma üç farklı veri setinde denenmiştir. Çalışmada elde edilen sonuçlar incelendiğinde başarı oranının ortalamanın üstünde olduğu gözlemlenmiştir.

Gullipalli (2021)yaptıkları çalışmada ise, YSA eğitimini GKA algoritması kullanarak yapmışlardır. Modifikasyonlar gerçekleştirilerek algoritma yakınsama kabiliyetini artırılması amaçlanmıştır. Eğitim yapılarak elde edilmiş ağ ile zaman serisi verileri üzerinde test yapılmıştır. Önerilen bu modelde elde edilen sonuçların başarı oranlarının yalın halinden daha yüksek seviyede olduğu gözlemlenmiştir.

Erdogan ve Gulcu (2021) yaptıkları çalışmada yapay sinir ağlarının eğitimi için kargaların fazla besinlerini depolama ve gerektiğinde depolama alanından geri alma davranışlarından ilham alan popülasyon tabanlı bir meta-sezgisel optimizasyon olan karga arama algoritması kullanılmıştır. Hibritlenen algoritmaya CSA-MLP adı verilmiştir. Deneysel sonuçlar karga arama algoritmasının çok katmanlı algılayıcı eğitiminde güvenilir bir yaklaşım olduğu görülmüştür.

3. MATERYAL VE YÖNTEM

Bu bölümde yapay sinir ağları, çok katmanlı algılayıcılar, kelebek optimizasyonu algoritması, deneysel çalışmalarda kullanılan kıyaslama fonksiyonları ve sınıflandırma veri setleri ve istatiksel performans metrikleri detaylı olarak anlatılmaktadır.

3.1. Yapay Sinir Ağları

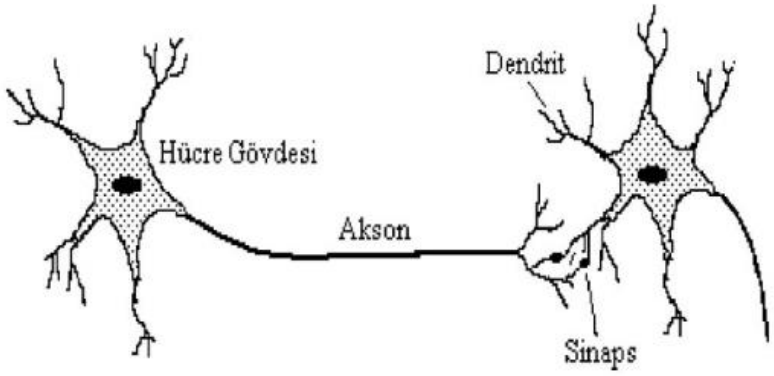
Yapay sinir ağları, insan beyninin sinir sisteminin çalışmasını model alarak geliştirilmiş bir bilgi işleme tekniğidir. Bir diğer ifadeyle, insan beynindeki nöronların birbiri ile kurdukları sinaptik bağların dijital platforma taşınmış halidir. Biyolojik sinir sistemi elemanları ve yapay sinir ağlarının görev karşılıkları Çizelge 3.1'de gösterilmektedir Koç ve ark (2004).

Çizelge 3.1. Biyolojik sinir sistemine ait elemanlar ve YSA'daki karşılığı

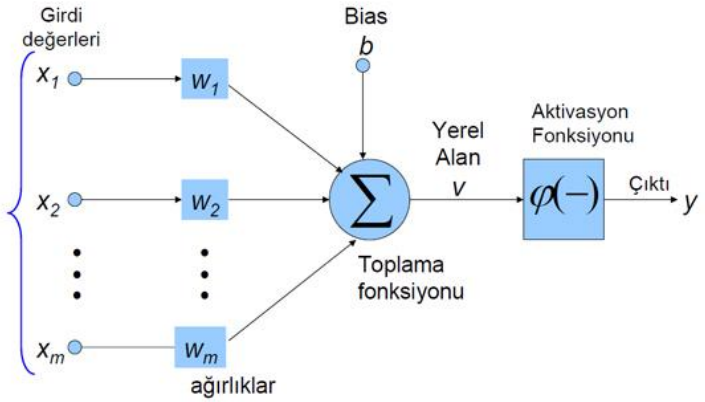
Sinir Sistemi	Yapay Sinir Ağı
Nöron	İşlem Elemanı
Dentrit	Toplama Fonksiyonu
Hücre Gövdesi	Aktivasyon Fonksiyonu
Akson	Eleman Çıkışı
Sinaps	Ağırlıklar

Şekil 3.1'de biyolojik sinir hücresinin genel yapısı gösterilmektedir. Şekil 3.1'de görüldüğü gibi uzantılardan kısa olanlara dentrit adı verilir ve dentritler binlerce dala dallanmıştır. Görevi giriş bilgilerini almaktır. Tek ve uzun olan uzantıya ise akson adı verilir ve görevi çıktı bilgilerini diğer sinir hücrelerine taşımaktır. Akson ve dentritin birleştiği yere sinaps adı verilir (Pandya ve Macy 1995). Şekil 3.1'den de anlaşıldığı gibi, dentritlerden gelen bilgi sinir hücresinde işlenmekte ve çıktı bilgileri akson üzerinden diğer sinir hücrelerine aktarılmaktadır.

İnsan sinir hücresi taklit edilerek yapay sinir hücresi geliştirilmiştir. Şekil 3.2'de bir yapay sinir hücresi gösterilmiştir. Nöronun net girdisini elde etmek için sinir hücresine giren girdiler ilgili bağlantı ağırlıkları ile çarpılırlar ve daha sonra birleştirme fonksiyonu ile birleştirilirler. Çıkan sonuç aktivasyon fonksiyonu ile işlenmekte ve böylece nöronun net çıktısı elde edilmektedir.



Şekil 3.1. Biyolojik sinir hücresi genel yapısı



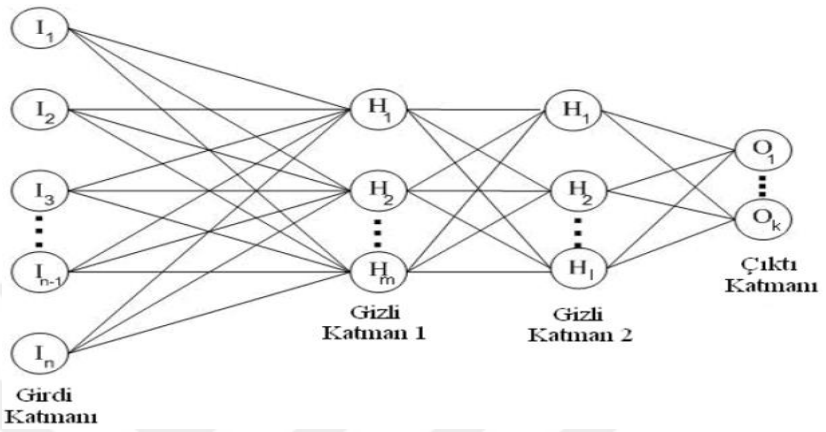
Şekil 3.2. Yapay sinir hücresi

Yapay sinir hücrelerin birleşmesi ve gruplandırılmasıyla YSA'lar oluşur. Bu birleşme katmanlardan oluşur ve bunun sonucunda YSA birbirine bağlı birden fazla katmandan meydana gelir. YSA girdi katmanı, gizli katman ve çıktı katmanı olmak üzere 3 katmandan oluşmaktadır. Fakat bazı durumlarda gizli katman sayısı birden fazla olabilmektedir. Birçok YSA modelinde işler sırayla gitmektedir. Kısaca, gizli katman kendinden önceki girdi katmanından verileri alır işler ve kendinden sonra olan çıktı katmanına yönlendirir. Şekil 3.3'de, iki gizli katmana sahip bir YSA yapısı gösterilmektedir. YSA'yı oluşturan üç katmanın özellikleri şu şekilde özetlenebilir:

• Giriş katmanı (input layer): Bilgi girişinin yapıldığı katmandır. Bu katmanda işlem yapılmaz, katmana gelen bilgi direk gizli katmanlara iletilir. Her düğümün yalnızca bir tane girdisi ve bir tane çıktısı vardır.

• Gizli katmanlar (hidden layers): Girdilere karşılık çıktıların matematiksel işlemlerle üretileceği katmanlardır. Giriş katmanı ile çıkış katmanının arasındaki iletişimi sağlar ve bilgiyi aktarırlar. Bir YSA içerisinde birden fazla gizli katman bulunabilir.

• Çıkış katmanı (output layer): Gizli katmanlarda üretilen sonucu alır ve sistemin dışına aktararak çıktıyı oluşturan katmandır.



Şekil 3.3. Yapay sinir ağları yapısı

YSA günümüzde çok fazla tercih edilen bir yapı haline gelmiştir. Çizelge 3.2'de geleneksel algoritmalar ile YSA karşılaştırması göstermektedir. Bu çizelgeye göre, YSA'nın avantajlarından dolayı tercih sebebi olduğu aşikârdır.

Çizelge 3.2. Geleneksel algoritmalar ile YSA'nın karşılaştırılması (Pirim 2006)

Geleneksel Algoritmalar	**Yapay Sinir Ağları**
Çıkışlar koyulan kurallara ve girişlerin uygulanması ile elde edilir.	Öğrenme sırasında giriş çıkış bilgileri verilerek, kurallar koyulur.
Bilgiler ve algoritmalar kesindir	Deneyimden yararlanır.
Hesaplama merkezi, eş zamanlı ve ardışıktır.	Hesaplama toplu, eş zamansız ve öğrenmeden sonra paraleldir
Bellek paketlenmiş ve hazır bilgi depolanmıştır.	Bellek ayrılmış ve ağa yayılmıştır.
Hata toleransı yoktur.	Hata toleransı vardır.
Nispeten hızlıdır.	Yavaş ve donanıma bağımlıdır.

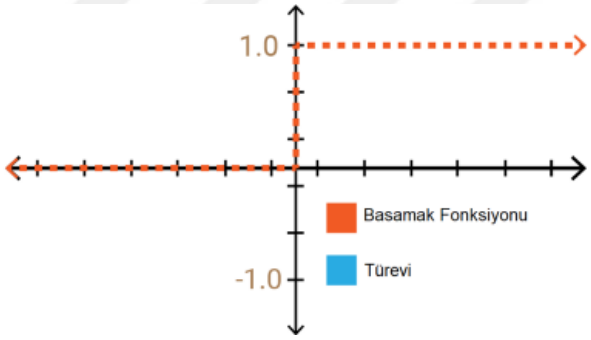
Yukarıda bahsedildiği gibi nöronun net çıktısını elde edebilmek için aktivasyon fonksiyonları kullanılır. Girdileri ve çıktıları eşleyen, birbirleri arasında diyalog kuran fonksiyon aktivasyon fonksiyonudur. Eğer aktivasyon fonksiyonu kullanılmazsa YSA'lar tek dereceli polinomlar olur. Aktivite edilmeyen YSA'lar linear olacağı için öğrenme kabiliyetleri oldukça düşük olacaktır. Sinir ağının doğrusal olmayan durumları da

öğrenmesini istiyorsak aktivasyon fonksiyonu kullanmak önem arz etmektedir. Literatürde birden fazla aktivasyon fonksiyonu bulunmaktadır. Hangi aktivasyon fonksiyonun seçileceği öğrenme kabiliyeti için oldukça önemlidir. Literatürde bulunan bazı aktivasyon fonksiyonları aşağıda kısaca anlatılmaktadır.

Basamak (Step) Fonksiyonu

Gelen net girdinin belirlenen bir eşik değerinin altında veya üstünde olmasına göre hücrenin çıktısı 1 veya 0 olur. Basamak aktivasyon fonksiyonunun formülü Denklem (3.1)'de gösterilmektedir. Şekil 3.4'de basamak aktivasyon fonksiyonunun grafiği gösterilmektedir.

$$f(x) = \begin{cases} 0 \; için \; x < 0 \\ 1 \; için \; x \geq 0 \end{cases} \tag{3.1}$$

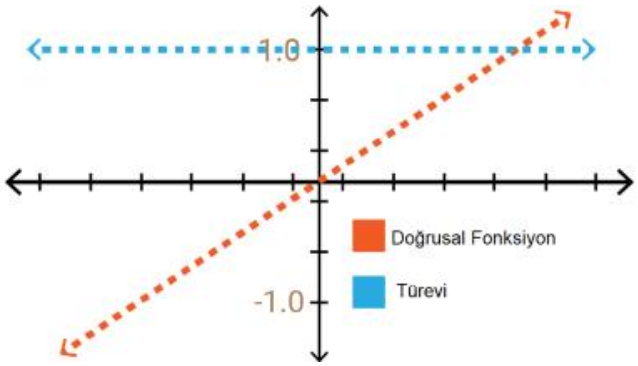


Şekil 3.4. Basamak fonksiyonu ve türevi

Doğrusal (Linear) Fonksiyon

Sonuç olarak bir dizi aktivasyon kodu üretir ve basamak fonksiyonundan farklı olmak üzere bu değerler 0 veya 1 gibi ikili değer değildir. Birkaç nöronu birbirine bağlamaya izin verir fakat bu fonksiyonun sorunu türevinin sabit olmasıdır. Buradaki olumsuzluk geri yayılım algoritması öğrenme yaparken türev fonksiyonunu kullanan bir algoritmadır ve türev alındığında lineer fonksiyonunda değer hep sabit olur. Denklem (3.2)'de doğrusal aktivasyon fonksiyonunun matematiksel gösterimi verilmektedir. Şekil 3.5'da doğrusal aktivasyon fonksiyonunun grafiği gösterilmektedir.

$$f(x) = x \tag{3.2}$$

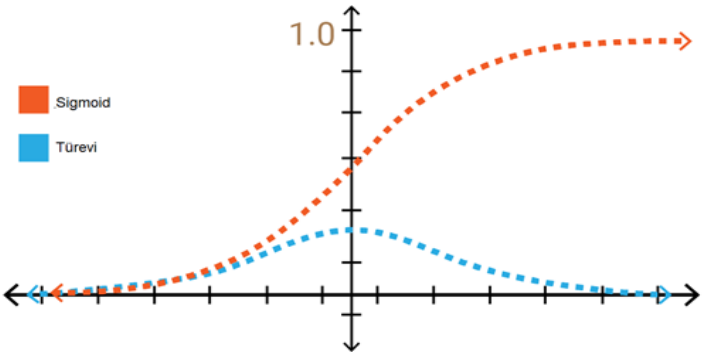


Şekil 3.5. Doğrusal (linear) fonksiyon ve türevi

Sigmoid Fonksiyonu

Doğada bulunan problemlerin çoğu doğrusal değildir. Sigmoid fonksiyonu da doğrusal olmayan türevlenebilir bir fonksiyondur. Basamak fonksiyonu gibi ikili değerler almaz, bunun dışına çıkabilir ve bu durum öğrenmenin gerçekleşeceğinin kanıtıdır. Bundan dolayı sigmoid fonksiyonu literatürde çok tercih edilen ve birçok problem için kullanılabilen bir aktivasyon fonksiyonudur. Ancak sigmoid fonksiyonunun da bir sorunu var. Grafiğe bakıldığında fonksiyonun uçlarına doğru y değerleri x'e çok az tepki vermektedir. Bu türev değerlerinin çok küçük olması ve 0'a yakınlaşması gibi sorunlar ortaya çıkarır. Buna gradyanların ölmesi/kaybolması (vanishing gradient) denir ve öğrenme olayı minimum düzeyde gerçekleşir. Denklem (3.3)'de sigmoid aktivasyon fonksiyonunun matematiksel gösterimi vardır. Alabileceği değer aralığı (0, 1) olarak sınırlandırılmıştır. Şekil 3.6'da sigmoid aktivasyon fonksiyonunun grafiği gösterilmektedir.

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.3}$$

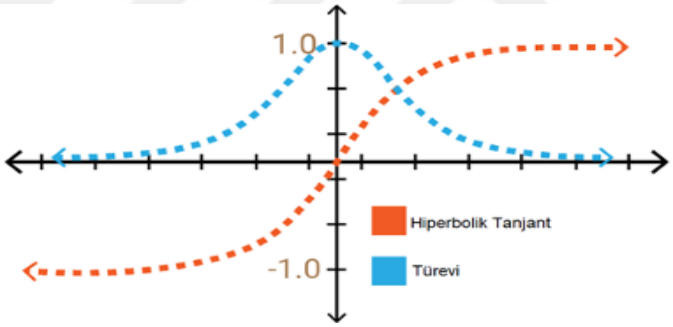


Şekil 3.6. Sigmoid fonksiyonu ve türevi

Hiperbolik Tanjant Fonksiyonu

Sigmoid fonksiyonuna çok benzer yapıya sahip bir fonksiyondur. Fakat bu fonksiyonda değer aralığı (-1, 1) şeklindedir. Sigmoid fonksiyonuna göre avantajı ise türevinin daha dik olması yani daha çok değer alabilmesidir. Bu durum daha hızlı öğrenmeyi sağlayacağı ve sınıflama işlemi için daha geniş aralığa sahip olmasından dolayı daha verimli olacağı anlamına gelmektedir. Ama yine fonksiyonun uçlarında gradyanların ölmesi problemi devam etmektedir. Denklem (3.4)'de hiperbolik tanjant aktivasyon fonksiyonunun matematiksel gösterimi verilmektedir. Alabileceği değer aralığı (-1, 1) olarak sınırlandırılmıştır. Şekil 3.7'de hiperbolik tanjant aktivasyon fonksiyonunun grafiği gösterilmektedir.

$$f(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{e^{x} + e^{-x}} \tag{3.4}$$



Şekil 3.7. Hiperbolik tanjant fonksiyonu ve türevi

Yapay Sinir Ağlarının Sistematik Çalışması

Karakaş (2020)YSA'nın çalışmasını sistematik olarak şu şekilde sıralamaktadır.

1. **Örneklerin toplanması:** Ağın girdi değerlerine karşılık çıktı değerlerini bulması için eğitim verilerinin ve test verilerinin toplanması gerekmektedir.
2. **Ağın topolojik yapısının belirlenmesi:** Ağın doğru tahmin yapabilmesi için ağın yapısının belirlenmesi gerekmektedir. Kaç adet girdiden, kaç ara katmandan oluşacak ve ara katmanlarda kaç hücre elemanı bulunacak ve kaç çıktı hücre elemanı elde edilecek bu parametreler oldukça önemlidir. Ağın topolojik yapısı kullanılacak verilere göre seçilmelidir.
3. **Öğrenme parametrelerinin belirlenmesi:** YSA'nın öğrenme katsayısı, aktivasyon fonksiyonu gibi önemli parametreler bu aşamada belirlenmektedir.

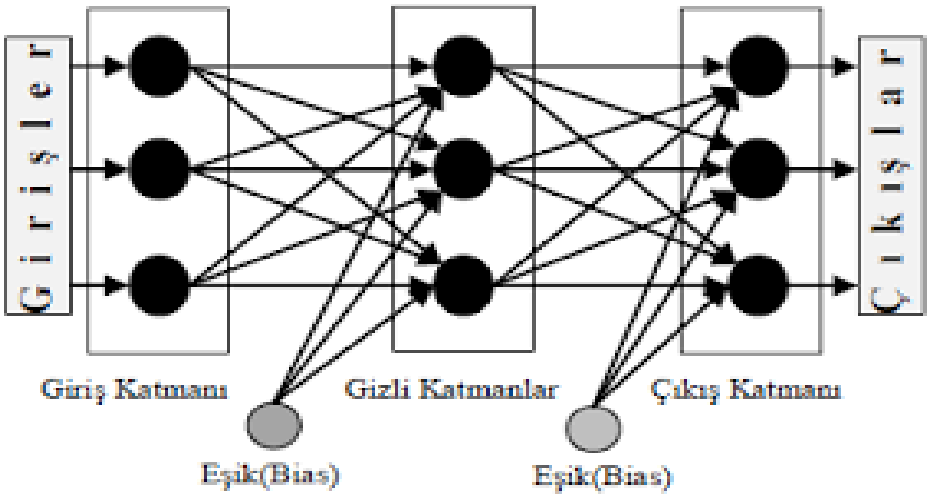
4. **Ağın başlangıç değerlerinin atanması:** Başlangıç ağırlık değerlerinin ve eşik değerlerinin atanması işlemi bu adımda gerçekleştirilmektedir.
5. **Eğitim verilerinden örneklerin seçilmesi ve ağa gösterilmesi:** Ağın öğrenmeye başlaması için gerekli olan adımdır.
6. **Öğrenme sırasında ileri hesaplama yapılması:** Ağa verilen girdi verileri için çıktı değerlerinin hesaplanması işlemi bu adımda gerçekleştirilmektedir.
7. **Elde edilen çıktı değerleri ile gerçek çıktı değerlerinin karşılaştırılması:** Bu adımda hata oranı elde edilerek ağın başarılı bir şekilde çalışıp çalışmadığı gözlemlenmelidir.
8. **Ağırlıkların değiştirilmesi:** Elde edilen çıktı değerleri ile beklenen çıktı değerleri karşılaştırılarak elde edilen hata oranını azaltmak için ağın ağırlık değerlerinin değiştirileceği adımdır. Ağırlıkların değiştirilip gerçeğe daha yakın sonuçlar elde edilmesi ağın başarısını yükseltmek için oldukça önemlidir ve ağın performansını arttırarak verilerin daha doğru tahmin edilmesine olanak sağlamaktadır.

YSA'nın öğrenme sürecini durdurmak için bazı kriterler belirlenmiştir. Bu kriterler iterasyon sayısı, belirlenen hata değerinin altına düşülmesi ve benzer değerler üretme gibi kıstaslar olabilir. Durdurma kriterlerinin belirlenmesi yapay sinir ağlarının öğrenme kabiliyeti için önemli bir adımdır. Durdurma kriteri her problem için farklıdır ancak parametreler genelde yukarıda bahsedildiği gibidir. Hatanın minimum olduğu ve o noktada kaldığı değer problemin optimum durdurma kriteri olarak seçilmelidir. Her öğrenme döngüsü tamamlandığında ağın genelleme yeteneği kazanıp kazanmadığı kontrol edilir. Eğer ağ yeteri kadar öğrendiyse ve yeni gelecek veriyi genelleme yaparak tahmin etme yeteneğine ulaştıysa öğrenme işleminin sona ermesi gerekir. Döngü tekrarlanıyor ancak ilerleyen iterasyonlarda ağın performansı artmıyorsa ağın öğrenme işleminin durdurulması gerekir (Karakaş 2020).

Yapay sinir ağları yapılarına göre ileri beslemeli YSA ve geri beslemeli YSA olmak üzere ikiye ayrılmaktadır: İleri beslemeli ağ ve geri beslemeli ağ.

Yapay sinir ağlarının en basit ve anlaşılır yapısı ileri beslemeli sinir ağlarıdır. İleri beslemeli sinir ağlarının akışı girdi katmanında çıktı katmanına doğru akar. İleri beslemeli ağda geriye doğru bir akış ve besleme asla gerçekleşmez. Girdiden alınan değerler işlenir ve ara katmana oradan da çıktıya gönderilir. Geri beslemeli sinir ağlarında ileri beslemenin yanı sıra geriye doğru bir bilgi akışı da vardır. Geri beslemeli ağ dinamiktir. Çıktılar oluşurken hem o anki hem de daha önceki girdilere bakılarak oluşturulur ve ağ böylelikle beslenir.

YSA'nın en temel çeşidi çok katmanlı algılayıcılardır (ÇKA). Çok katmanlı algılayıcı girdi katmanı ve çıktı katmanı arasında bir veya birden fazla gizli katmanın olduğu ileri beslemeli bir ağ yapısıdır. ÇKA'da öğrenme algoritması olarak genellikle geri yayılım algoritması kullanılır (Türkoğlu 2019). Şekil 3.8'de ÇKA'nın yapısı gösterilmektedir. Türkoğlu (2019) bir ÇKA'nın şu bileşenlerden oluştuğunu belirtmiştir: Yapay sinir hücresi ve katmanlar, birleştirme fonksiyonu, aktivasyon fonksiyonu, hata fonksiyonu, öğrenme algoritması.



Şekil 3.8. Çok katmanlı algılayıcı yapısı

Girdiler, birleştirme ve aktivasyon fonksiyonundan oluşan yapay sinir hücresi insan sinir sisteminin taklidi ile geliştirilmiştir. Yapay sinir hücresinde girdi değerleri (ağırlıklar) düğüm ağırlıkları ile çarpılır ve birleştirme fonksiyonuna gönderilir. Birleştirme fonksiyonundan dönen sonuç aktivasyon fonksiyonuna yollanır ve böylece yapay sinir hücresinin net çıktısı elde edilmiş olur.

Gelen girdilerin ağırlıklarını bileştirerek net girdiyi alan fonksiyona birleştirme fonksiyonu denir. Birleştirme fonksiyonundan gelen sonuçlar aktivasyon fonksiyonuna gönderilir ve çıktıya dönüştürülür. Literatürde sıkıştırma veya eşik değeri fonksiyonu olarak adlandırılmaktadır. Bunun sebebi çıktı sinyallerinin [0,1] veya [-1,1] aralığında sınırlanmasıdır (Herz ve ark 1998). Yukarıda aktivasyon fonksiyonlarından açıklayıcı bir şekilde bahsedilmiştir. Bu tez çalışmasında, aktivasyon fonksiyonu olarak sigmoid fonksiyonu seçilmiştir. Bunun sebebi türevinin alınabilir olması ve literatürde sık kullanımı ile başarısının kanıtlanmış olmasıdır.

ÇKA bileşenlerinden biri olan hata fonksiyonunda ise YSA'nın öğrenme temelli bir sistem olduğu bilinmektedir ve bu sistemdeki hatayı bulmak için bir amaç fonksiyonu tanımlanmaktadır.

Son olarak ÇKA'nın son bileşeni olan öğrenme algoritmasından bahsetmek gerekirse, ÇKA'lar çoğunlukla geri yayılım olarak adlandırdığımız algoritma ile öğrenimini gerçekleştirirler. Geri yayılım algoritmaları YSA için literatürde çok önemli bir yere sahiptir fakat her problemi çözebilir olarak düşünülmemelidir. Bununla birlikte problem çözmede üstün başarısı da göz ardı edilmemektedir.. Bundan dolayı problem tespitinin doğruluğu oldukça önem arz etmektedir. ÇKA'nın geri yayılım algoritması ile eğitilmesi üç aşamada gerçekleşir. Ağ girdisinin girdi katmanından çıktı katmanına doğru ilerlemesi, çıktı birimlerinde hatanın hesaplanması, geriye doğru yayımı ve geriye doğru yayılan hataya göre ağırlıkların değiştirilmesidir (Türkoğlu 2019). Ağın eğitimi tamamlandıktan sonra ÇKA ileri doğru çalışır.

3.2. Kelebek Optimizasyonu Algoritması

Linnaean Animal Kingdom sisteminde, kelebekler Lepidoptera sınıfında yer alır. Dünyada 18.000'den fazla kelebek türü vardır. Milyonlarca yıl hayatta kalmalarının nedeni ise duyularında yatmaktadır (Saccheri ve ark 1998).

Kelebekler yiyecek ve çiftleşme partneri bulmak için koku, görme, tat alma, dokunma ve işitme duyularını kullanırlar. Bu duyular ayrıca bir yerden başka bir yere göç etmede, avcıdan kaçmada ve uygun yerlere yumurtlamada yardımcı olur. Tüm bu duyular arasında koku, kelebeğin uzak mesafelerden bile yiyecek, genellikle nektar bulmasına yardımcı olan en önemli duyudur (Blair ve Launer 1997).

Kelebekler nektarın kaynağını bulmak için koku almada kullanılan duyu alıcılarını kullanırlar ve bu alıcılar kelebeğin anten, bacak, el parmakları vb. gibi vücut kısımlarına dağılmıştır. Bu alıcılar aslında kelebeğin vücut yüzeyindeki sinir hücreleridir ve kemoreseptörler olarak adlandırılır Pollard ve Yates (1994). Bu kemoreseptörler, güçlü bir genetik çizgiyi sürdürmek için kelebeğe en iyi çiftleşme ortağını bulma konusunda rehberlik eder. Erkek bir kelebek, dişi kelebeğin belirli reaksiyonlara neden olmak için yaydığı koku salgıları olan feromon aracılığıyla dişiyi tanımlayabilir (Arora ve Singh 2019). Bilimsel gözlemlere dayalı olarak, kelebeklerin kokunun kaynağını bulma konusunda çok doğru bir algıya sahip oldukları bulunmuştur (Raguso 2008). Ayrıca farklı kokuları ayırabilir ve yoğunluklarını hissedebilirler (Wyatt 2003).

Sürü zekâsı tabanlı olan kelebek optimizasyonu algoritması (KOA) global optimum sorununu çözmek için Arora ve Singh (2019) tarafından doğadan ilham alınarak geliştirilmiştir. KOA, esas olarak nektarın veya çiftleşme eşinin yerini belirlemek için koku duyularını kullanan kelebeklerin hareket etme stratejisine dayanmaktadır. Kelebekler, bir nektar/çiftleşme eşinin potansiyel yönünü belirlemek için yukarıda belirtildiği gibi duyu algılayıcısı ile kokuyu algılar ve analiz eder. KOA, arama alanında optimumu bulmak için bu davranışı taklit eder.

KOA, kelebeklerin koku, görme, tat, dokunma ve duyma duyularını kullanarak yiyecek ve çiftleşme eşini bulmalarını rol model alarak tasarlanan bir algoritmadır. Tüm bu duyular arasında koku, kelebeğin uzun mesafelerden bile genellikle nektar gibi yiyecek bulmasına yardımcı olan en önemli duyudur. Kelebekler, optimizasyon için KOA'nın arama aracılarıdır. Bir kelebek, uygunluğu ile ilişkili bir yoğunlukta koku üretecektir. Yani bir kelebek bir konumdan diğerine hareket ettikçe uygunluğu değişecektir.

Koku mesafe boyunca yayılır ve diğer kelebekler bunu hissedebilir ve kelebekler bu şekilde kişisel bilgilerini diğer kelebeklerle paylaşabilir ve kolektif bir sosyal bilgi ağı oluşturabilir. Bir kelebek başka bir kelebeğin kokusunu algılayabildiğinde, ona doğru hareket edecektir ve bu aşama önerilen algoritmada global arama olarak adlandırılmaktadır (Arora ve Singh 2019).

KOA aşağıda bulunan üç madde rol model alınarak geliştirilmiştir.

1. Tüm kelebeklerin, kelebeklerin birbirini çekmesini sağlayan bir koku yayması beklenir.

2. Her kelebek rastgele veya daha fazla koku yayan en iyi kelebeğe doğru hareket edecektir.

3. Bir kelebeğin uyaran yoğunluğu, objektif fonksiyonun değerinden etkilenir veya belirlenir.

Kelebeklerin davranışları rol model alınarak yapılan KOA'da bu üç aşamayı şöyle açıklanabilir:

1. Başlatma Aşaması: Parametreler belirlenir, algoritma için bir başlangıç popülasyonu üretilir. Başlangıç popülasyonu oluşturulurken kelebeklerin konumu, koku değerleri hesaplanarak rastgele olarak atanır.
2. İterasyon Aşaması: Bu aşama asıl işlemlerin yürütüldüğü, her bir kelebeğin KOA'ya özel parametrelerle en iyi sonuca gitmeye çalıştığı kısımdır. Her

yinelemede, arama uzayındaki tüm kelebekler yeni pozisyonlara taşınır ve daha sonra uygunluk değerleri değerlendirilir.

3. Son aşama: Durdurma kriterinin sağlandığı ve elde edilen optimum ya da optimuma en yakın sonucun raporlandığı kısımdır.

KOA modalitesini anlamak üç temel kavrama dayanır. Bu kavramlar: duyusal yöntem (c), uyaran yoğunluğu (I) ve güç üssüdür (α). Duyusal yöntem, duyusal enerji formunu ölçmek ve onu benzer şekillerde işlemek adına algılayıcılar tarafından kullanılan ham girdiyi ifade eder. Uyaran yoğunluğu olan I parametresi üssel bir değerle sınırlandırılmıştır. Bilim adamlarının daha önce yaptığı çalışmalara göre bunun nedeni, uyaran güçlendikçe böceklerin uyarana yoğun bir şekilde gittiği ve bu durumun sonunda daha az duyarlı hale gelmesidir. Bu durumu düzeltmek için α parametresi işin içine girmiştir. α parametresi modaliteye (KOA'da koku) bağımlı olan güç üssüdür. α=1 ise, koku emilimi olmadığı anlamına gelir. Yani belirli bir kelebek tarafından yayılan koku miktarı, diğer kelebekler tarafından aynı kapasitede algılanır. Bu bizi tek bir çözüme, genellikle optimuma yaklaştırır. α=0 ise herhangi bir kelebeğin yaydığı kokunun diğer kelebekler tarafından algılanamayacağı anlamına gelir. Bu da bize yerel aramayı sağlar. α ve c [0, 1] arasında rastgele bir sayıyı, f kokunun algılanan büyüklüğünü, I ise uyarıcı yoğunluğunu ifade eder. Algoritmada iki önemli aşama söz konusudur: yerel arama ve global arama. Global arama Denklem (3.6)'da yerel arama ise Denklem (3.7)'de gösterilmektedir. Yerel arama, global aramadan farklı olarak x_i^t kelebeği global en iyiye (g) doğru gitmez, arama uzayında rassal bir yürüyüş sergiler.

$$f = c \times I^a \tag{3.5}$$

Burada α ve c [0, 1] arasında rastgele bir sayıyı, *f* kokunun algılanan büyüklüğünü, *I* ise uyarıcı yoğunluğunu ifade eder.

$$X_i^{t+1} = X_i^t + (r^2 \times g^* - X_i^t) \times f_i \tag{3.6}$$

Burada *g** mevcut yinelemedeki tüm çözümler arasında bulunan mevcut en iyi çözümü, X_i^t ve X_k^t arama uzayındaki kelebekleri, r [0, 1] aralığında değer alan rasgeleliği sağlayan bir parametreyi, f_i ise kelebeğin hissedilen kokusunu temsil etmektedir.

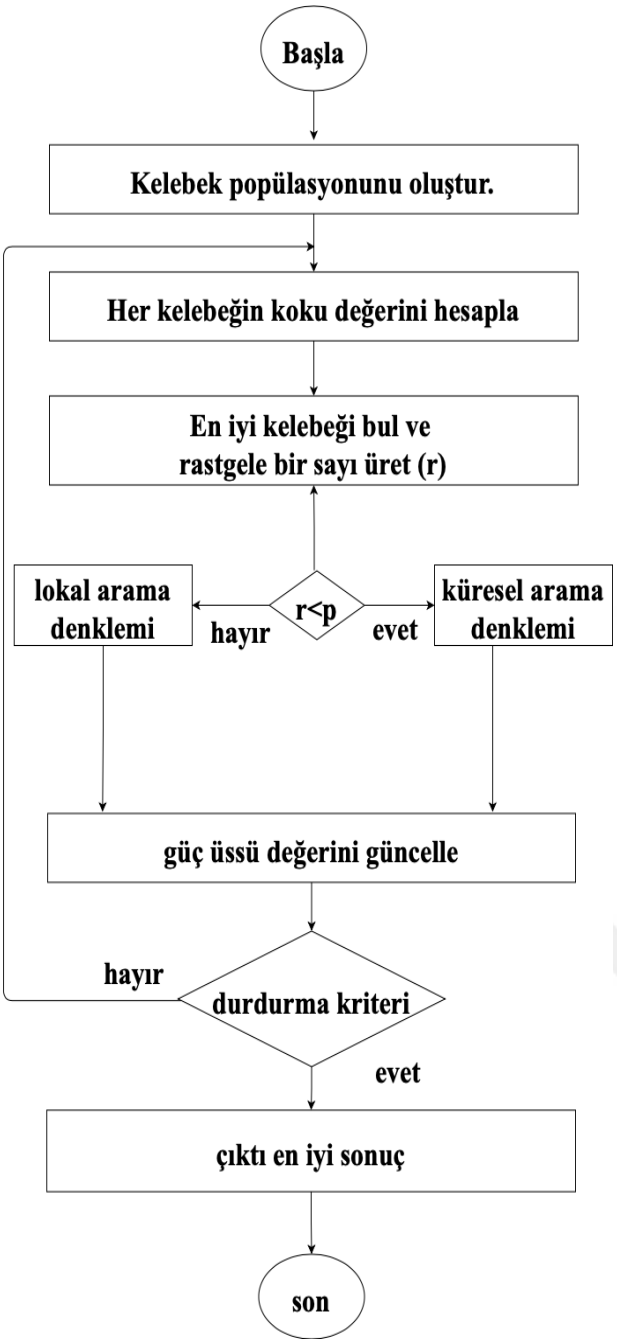
$$X_i^{t+1} = X_i^t + \left(r^2 \times X_j^t - X_k^t\right) \times f_i \quad (3.7)$$

Burada globalden farklı olarak x_i^t kelebeği global en iyiye (g^*) doğru gitmez, arama uzayında rassal bir yürüyüş sergiler.

Yiyecek ve çiftleşme partnerini kelebeklerle aramak, hem yerel hem de global ölçekte gerçekleşebilir. Dolayısıyla, genel global arama ile yoğun yerel arama arasında geçiş yapmak için KOA'da bir anahtarlama olasılığı p kullanılır. Anahtar olasılığı bir kelebeğin en iyi kelebeğe mi yoksa rastgele mi hareket edeceğine karar verir. KOA algoritmasının sözde kodu ve akış diyagramı Şekil 3.9 ve Şekil 3.10'da gösterilmektedir.

*Amaç fonksiyonu f(**x**), **x**=(x_1,x_2, …, x_{dim}), boyut hessaplanır*
Kelebeklerin başlangıç popülasyonu oluşturulur. $\mathbf{x}_i$= (i=1,2, …, n)
$\mathbf{x}_i$'deki uyaran yoğunluğu I_i , f($\mathbf{x}_i$) ile belirlenir.
Sensör modalitesi c, güç üssü α ve anahtar olasılığı p belirlenir.
While** durdurma kriterleri karşılanmıyorsa **do
    for each** popülasyondaki o anki kelebek, bf **do
    bf'ye ait koku değeri denklem (3.5) kullanılarak hesaplanır.
    end for
    En iyi kelemek bulunur. g_{best}
    for each** popülasyondaki o anki kelebek, bf **do
        [0,1] arasında random sayı r üretilir.
        ***if** r<p ise*
        Denklem (3.6) kullanılır.
        else
        Denklem (3.7) kullanılır.
        En if
    end for
    değerler güncellenir
end while
En iyi sonuç çıktı olarak verilir.

Şekil 3.9. KOA algoritmasının sözde kodu (Arora ve Singh 2019)



Şekil 3.10. KOA algoritmasının akış diyagramı

3.3. Kıyaslama Fonksiyonları

Bu alt bölümde, deneysel çalışmalarda kullanılan kıyaslama fonksiyonları anlatılmaktadır. Çizelge 3.3'de bu 13 kıyaslama fonksiyonların formülleri, boyutları, global minimum değerleri ve arama aralığı gösterilmektedir. Bu fonksiyonlar literatürde yaygın olarak kullanılan çok boyutlu problemlerdir. f_1– f_5 fonksiyonları tek modlu fonksiyonlardır. f_6, bir minimumu olan ve süreksiz olan adım fonksiyonudur. f_7

fonksiyonu, gürültülü ikinci dereceden bir fonksiyondur. f_8– f_{13}, çok modlu test fonksiyonlarıdır. Bu fonksiyonlar için yerel minimumların sayısı problem boyutlarına bağlı olarak üstel olarak artmaktadır. Bu fonksiyonlar birçok optimizasyon problemi için en zor problem sınıfına aittirler. f_1– f_{13} fonksiyonlarının isimleri sırasıyla şu şekildedir: sphere, schwefel 2.22, schwefel 1.2, schwefel 2.21, rosenbrock, step, quartic with noise, schwefel, rastrigin, ackley, griewank, penalized1, penalized2. Faklı zorluk derecesine sahip bu 13 kıyaslama fonksiyonu İKOA algoritmasının performansını test etmek ve diğer çalışmalarla karşılaştırmak için deneylerde kullanılmıştır. Kıyaslama fonksiyonlarının tümünde D=30'dur. f_8 hariç hepsinin global minimum değeri sıfırdır. Faklı zorluk derecesine sahip olduklarından dolayı arama aralıkları farklı alınmıştır ve Çizelge 3.3'de gösterilmektedir.

Çizelge 3.3. Test fonksiyonları, boyutlar, global minimum ve arama aralığı

Fonksiyon	**D**	f_{min}	**Aralık**
$f_1 = \sum_{i=1}^{D} x_i^2$	30	0	[-100, 100]
$f_2 = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	0	[-10, 10]
$f_3 = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	30	0	[-100, 100]
$f_4 = max_i\{\lvert x_i \rvert, 1 \le i \le D\}$	30	0	[-100, 100]
$f_5 = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	0	[-30, 30]
$f_6 = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	30	0	[-100, 100]
$f_7 = \sum_{i=1}^{D} i x_i^4 + random[0,1)$	30	0	[-1.28, 1.28]
$f_8 = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	-418.9829*D	[-500, 500]
$f_9 = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	0	[-5.12, 5.12]
$f_{10} = -20exp\left(-0.2 \times \sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)\right) + 20 + e$	30	0	[-32, 32]
$f_{11} = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	0	[-600, 600]
$f_{12} = \frac{\pi}{D}\left\{10\sin^2(\pi y_i) + \sum_{i=1}^{D-1}(y_i - 1)^2 [1 + 10\sin^2(\pi y_{i+1})] + (y_D - 1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a < x_i < a \\ k(-x_i - a)^m, & x_i < a \end{cases}$	30	0	[-50, 50]
$f_{13} = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{D}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)] + (x_D - 1)^2[1 + \sin^2(2\pi x_D)]\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	30	0	[-50, 50]

3.4. Sınıflandırma Veri Setleri

İKOA-ÇKA algoritmasının performansını test etmek ve diğer çalışmalarla karşılaştırmak için deneylerde farklı eğitim/test örnekleri ve farklı zorluk seviyelerine

sahip 5 sınıflandırma veri seti kullanılmıştır. Bu 5 veri seti şunlardır: xor, balon, kalp, meme kanseri ve iris. Balon, kalp, meme kanseri ve iris veri setleri UCI veri havuzundan alınmıştır ve yaygın olarak makine öğrenmesi algoritmalarının testi için kullanılmaktadır. Çizelge 3.4'de veri setlerinin özellikleri gösterilmektedir. 8 eğitim ve 8 test örneği, 3 nitelik ve 2 sınıf ile xor veri kümesinin en kolay problem olduğu görülmektedir. Balon veri seti 4 özellik, 20 eğitim örneği, 20 test örneği ve 2 sınıfa sahiptir. Iris veri setinin 4 özelliği, 150 eğitim örneği, 150 test örneği ve 3 sınıfı sayısı bulunmaktadır. Meme kanseri veri setinde 9 özellik, 599 eğitim örneği, 100 test örneği ve 2 sınıf bulunmaktadır. Kalp veri setinde ise 22 özellik ve 80 eğitim örneği, 187 test örneği ve 2 sınıf olduğunu görmekteyiz. Bu eğitim ve test veri kümeleri www.seyedalimirjalili.com adresinden alınmıştır ve Mirjalili (2015) çalışmasında da kullanılmıştır.

Çizelge 3.4. Sınıflandırılma veri setleri

Veri seti	Nitelik sayısı	Eğitim örneği sayısı	Test örneği sayısı	Sınıf sayısı
Xor	3	8	8	2
Balon	4	20	20	2
Iris	4	150	150	3
Meme kanseri	9	599	100	2
Kalp	22	80	187	2

3.5. İstatiksel Performans Metrikleri

Deneysel çalışmalar esnasında, algoritmaların başarısını ölçmek için kullanılan istatiksel performans metrikleri alt başlıklarda bu bölüm içerisinde anlatılacaktadır.

3.5.1. Ortalama karesel hata (MSE)

İstatistikte, bir tahmin edicinin ortalama kare hatası (MSE) veya ortalama kare sapması, hataların karelerinin ortalamasını verir. Yani, tahmini değerler ile gerçek değer arasındaki ortalama kare farkını ölçer. MSE, kare hata kaybının beklenen değerine karşılık gelen bir risk fonksiyonudur. MSE formülü Denklem (3.9)'da gösterilmektedir. ÇKA'nın çıktı katmanında birden fazla nöron olabileceği için bu tez çalışmasında kullanılan MSE formülü Denklem (3.10)'da gösterilmektedir. Basitçe, ortalama kare hatası bir regresyon eğrisinin bir dizi noktaya ne kadar yakın olduğunu söyler. MSE; bir makine öğrenmesi modelinin, tahminleyicinin performansını ölçer. Her zaman pozitif

değerlidir ve MSE değeri sıfıra yakın olan tahminleyicinin daha iyi bir performans gösterdiği söylenebilir.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(g_i - t_i)^2 \quad (3.9)$$

Burada N örneklem sayısını ifade etmektedir. g_i ve t_i ise i. örneklem için sırasıyla gerçek değeri ve tahmin değerini ifade etmektedir.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K}(g_j^i - t_j^i)^2 \quad (3.10)$$

Burada N örneklem sayısını, K ÇKA'nın çıktı katmanındaki nöron sayısını ifade etmektedir. g_j^i ve t_j^i ise i. örneklem için çıktı katmanındaki j. nöronun sırasıyla gerçek değeri ve tahmin değerini ifade etmektedir.

3.5.2. Friedman Test

Friedman testi, bağımlı gruplar arasında normallik varsayımı sağlanmadığı durumlarda ortalama karşılaştırması yapabilmek amacı ile kullanılan bir istatistiksel analiz tekniğidir. Friedman testini kullanabilmek için bir adet grup (ya da blok) değişkeni ve bir adet de ortalama karşılaştırması yapılabilecek ölçüm değişkeni bulunmalıdır. Ölçüm yapılacak olan değişken, sürekli bir veri ya da sıralayıcı ölçeğe sahip bir veri olmalıdır.

Friedman testi; normal dağılıma uygun olmayan ölçümler için, istatistik analizi uygulamalarında k-adet (k>2) eşleştirilmiş veya bağımlı grup arasındaki ortalamaları karşılaştırmak için kullanılmaktadır. Testin genel yapısına bakıldığında, Wilcoxon eşleştirilmiş işaretli sıra testinin genelleştirilmiş hali olduğu söylenebilir. Wilcoxon testinde bağımlı veya eşleştirilmiş birimler arasında yalnızca iki grup arasında ortalama karşılaştırması yapılırken, bu test kapsamında en az üç grup arasında ortalama karşılaştırması yapılmaktadır.

Friedman testinin uygulanabilmesi için aşağıdaki temel varsayımların sağlanmış olması gereklidir:

- Grup içerisinde gözlemlerin bağımsız olması
- Bloklar arası gözlem gruplarının eşleştirilmiş ya da bağımlı bir yapıda olması

- Ortalama karşılaştırması yapılacak olan ölçüm değişkeninin sürekli veya sıralayıcı ölçekli bir yapıda olması
- Ortalama karşılaştırması yapılacak verilerin normal dağılıma uygun olmaması
- Bağımlı veya eşleştirilmiş verileri içeren blok sayısının ikiden fazla (k>2) olması

3.5.3. Duyarlılık (Sensitivity)

Duyarlılık, bir koşulun varlığını bildiren bir testin doğruluğunu matematiksel olarak tanımlar. Duyarlılık (gerçek pozitif oran), bu testte pozitif sonuç alan duruma sahip olanların oranını ifade eder. Duyarlılık formülü Denklem (3.11)'de gösterilmektedir.

$$duyarlılık = \frac{DP}{DP + YN} \tag{3.11}$$

Burada DP doğru pozitifi ve YN ise yanlış negatifi temsil etmektedir.

3.5.4. Özgüllük (Specificity)

Özgüllük, bir koşulun yokluğunu bildiren bir testin doğruluğunu matematiksel olarak tanımlar. Özgüllük (gerçek negatif oran), bu testte negatif sonuç alan koşulu taşımayanların oranını ifade eder. Özgüllük formülü Denklem (3.12)'de gösterilmektedir.

$$özgüllük = \frac{DN}{DN + YP} \tag{3.12}$$

Burada DN doğru negatifi ve YP ise yanlış pozitifi temsil etmektedir.

3.5.5. Kesinlik (Precision)

Kesinlik, değeri pozitif olarak tahmin ettiğimiz değerlerin gerçekten kaç adedinin pozitif olduğunu göstermektedir. Kesinlik formülü Denklem (3.13)'de gösterilmektedir. Kesinlik değeri özellikle yanlış pozitif tahminlemenin maliyeti yüksek olduğu durumlarda çok önemlidir.

$$kesinlik = \frac{DP}{DP + YP} \tag{3.13}$$

Burada DP doğru pozitifi ve YP ise yanlış pozitifi temsil etmektedir.

3.5.6. F1-Score

F1-Score değeri bize kesinlik ve duyarlılık değerlerinin harmonik ortalamasını göstermektedir. F1-Score formülü Denklem (3.14)'de gösterilmektedir. Basit bir ortalama yerine harmonik ortalama olmasının sebebi; uç durumları da göz ardı etmememiz gerektiğidir. Eğer basit bir ortalama hesaplaması olsaydı kesinlik değeri 1 ve duyarlılık değeri 0 olan bir modelin F1-Score değeri 0,5 olarak gelecektir ve bu bizi yanıltacaktır.

$$f1 - score = 2 * \frac{kesinlik * \ duyarlılık}{kesinlik + duyarlılık} \tag{3.14}$$

4. ÖNERİLEN YÖNTEM

4.1. İyileştirilmiş Kelebek Optimizasyonu Algoritması

Bölüm 3'de ele alınan kelebek optimizasyonu algoritmasının sorunları araştırıldı. Lokale takılma ve erken yakınsama sorunları ile karşılaştığı görüldü. Literatürdeki bu sorunları çözen parametre analizleri, hibrit modelleri, farklı arama stratejileri araştırıldı ve başarısı ve çıktı sonuçları göz önünde bulundurularak kaotik haritaların kullanımı kararlaştırıldı. Optimizasyon alanında kullanılan çok çeşitli kaotik harita vardır. Ancak çalışmada literatürde en sık kullanılan 10 kaotik harita seçildi ve kullanıldı.

KOA algoritmasının Denklem (3.6) ve (3.7)'de gösterilen lokale ve globale yaklaştıran denklem olarak adlandırdığımız denklemler arasındaki dengeyi kuran hangisinin seçileceğine karar veren, KOA algoritmasının yapıtaşı olan değer p anahtarı parametresidir. Karar verme mekanizmasında random bir sayı oluşturulur ve bu sayının p anahtarında büyük veya küçük olmak durumu kontrol edilerek Denklem (3.6) ve (3.7)'nin kullanılacağı kararı verilir. KOA algoritmasının yalın halinde bu parametrenin 0.8 olarak alınmasının optimize değer olarak belirlendiği görülmüştür.

Çalışmada İKOA'ya kaotik haritalar uygulanırken p anahtarını optimize etmek amacı güdülmüştür. p anahtarının algoritmanın yalın halinde kullanılan optimum değer olarak değinilen 0.8 değerinden başlatılmasına karar verildi.10 kaotik harita kıyaslama fonksiyonları bölümünde bahsedilen 13 farklı fonksiyon üzerinde denendi ve başarısı analiz edildi. İKOA'nın sonuçlarında iyileşme olduğu görüldü. Bunun üzerine YSA öğrenimi için kullanılmaya karar verildi. Kullanılan haritalar aşağıda detaylı bir şekilde formülize edilerek anlatılacaktır.

Kaotik haritalardaki, kaotik kelimesinin kökeni evrenin düzene girmeden önce içinde bulunduğu, biçimden ve düzenden yoksun, uyumsuz ve karmakarışık olma anlamına gelen kaos kelimesinden gelmektedir. Buradaki haritalar kelimesi ise kullanılan algoritmalardaki kaos olarak nitelendirilebilecek bir davranışı kullanarak bazı parametrelerle eşleştirme veya ilişkilendirme anlamına gelir. Dolayısıyla kaotik haritalar, lineer olmayan sistemlerde karmaşık ve dinamik davranış gösteren haritalardır Pecora ve Carroll (1990) Son on yıldan beri kaotik haritalar, optimizasyon algoritmalarının arama uzayını daha dinamik ve küresel olarak keşfetmelerine yardımcı olan dinamik davranışları nedeniyle optimizasyon alanında geniş çapta takdir görmüştür. Davranışı

yalnızca başlangıç koşullarında tahmin edilebilir, ancak daha sonra rastgele davranış gösterir.

Chebyshev haritası: Chebyshev haritasının formülü Denklem (4.1)'de gösterilmektedir.

$$x_{k+1} = \cos(k\cos^{-1}(x_k)) \tag{4.1}$$

Circle haritası: Circle haritasının formülü Denklem (4.2)'de gösterilmektedir.

$$x_{k+1} = x_k + 0.2 - (0.5 - 2\pi)\sin(2\pi x_k) mod(1) \tag{4.2}$$

Gauss haritası: Gauss haritasının formülü Denklem(4.3)'de gösterilmektedir.

$$x_{k+1} = \begin{cases} 0 & x_k = 0 \\ \dfrac{1}{x_k mod(1)} & değilse \end{cases} \tag{4.3}$$

Bu harita aynı zamanda (0, 1) aralığında kaotik diziler üretir.

Iterative haritası: Iterative haritasının formülü Denklem (4.4)'de gösterilmektedir.

$$x_{k+1} = sin\left(\frac{0.7\pi}{x_k}\right) \tag{4.4}$$

Logistic haritası: Logistic haritanın formülü Denklem (4.5)'de gösterilmektedir.

$$x_{k+1} = 4x_k(1 - x_k) \tag{4.5}$$

Piecewise haritası: Piecewise haritasının formülü Denklem (4.6)'da gösterilmektedir.

$$x_{k+1} = \begin{cases} \frac{x_k}{\Pi} & 0 \le x_k \le \Pi \\ \frac{x_k - \Pi}{0.5 - \Pi} & \Pi \le x_k \le 0.5 \\ \frac{1 - \Pi - x_k}{0.5 - \Pi} & 0.5 \le x_k \le 1 - \Pi \\ \frac{1 - x_k}{\Pi} & 1 - \Pi \le x_k \le 1 \end{cases} \tag{4.6}$$

Burada $0<\Pi\leq 0.5$.

Sine haritası: Sine haritasının formülü Denklem (4.7)'de gösterilmektedir.

$$x_{k+1} = \frac{\alpha}{4} sin(\pi x_k) \tag{4.7}$$

Singer haritası: Singer haritasının formülü Denklem (4.8)'de gösterilmektedir.

$$x_{k+1} = 1.07(7.86x_k - 23.31x_k^2 + 28.75x_k^3 + 13.302875x_k^4) \tag{4.8}$$

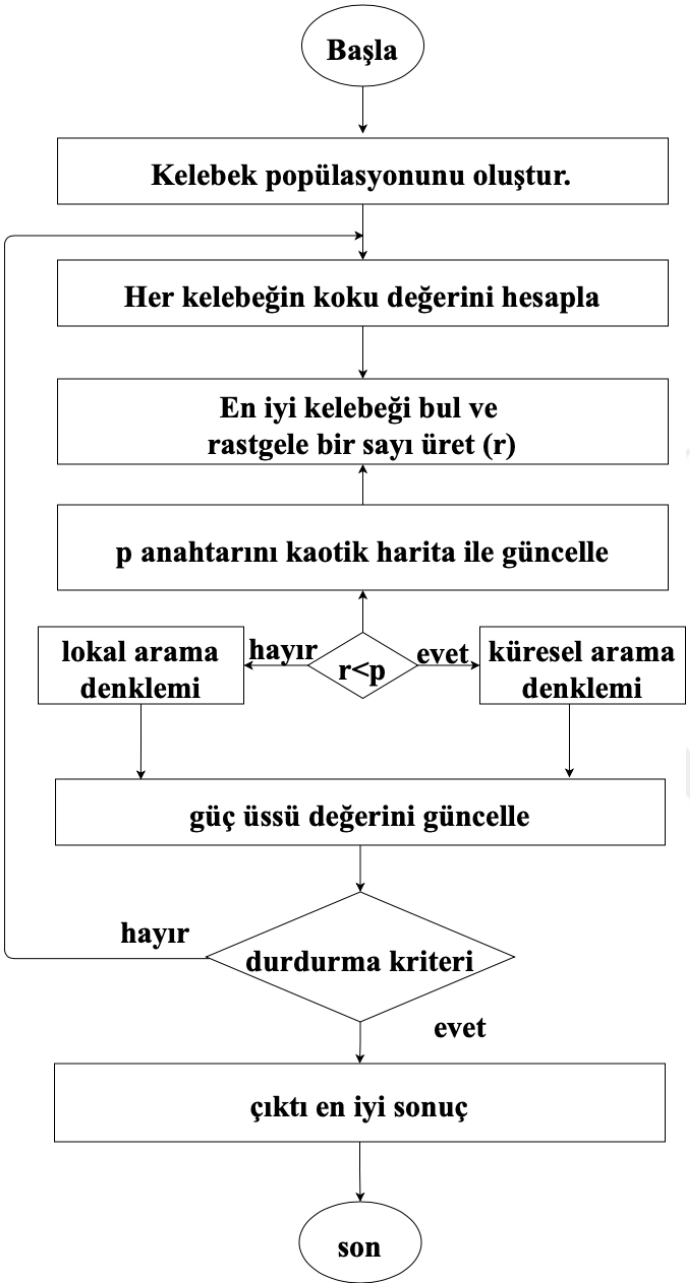
Sinusoidal haritası: Sinusoidal haritasının formülü Denklem (4.9)'da gösterilmektedir.

$$x_{k+1} = \sin(\pi x_k) \tag{4.9}$$

Tent haritası: Tent haritasının formülü Denklem (4.10)'da gösterilmektedir.

$$x_{k+1} = \begin{cases} \dfrac{x_k}{0.7} & x_k < 0.7 \\ \dfrac{10}{3}(1 - x_k) & x_k \geq 0.7 \end{cases} \tag{4.10}$$

İKOA algoritmasının başında kelebek popülasyonu rastgele oluşturulur. Popülasyonun her üyesi $x_{i,j}$ olarak temsil edilebilir. Burada i kelebeği j boyutu temsil etmektedir. İKOA için sabit bir parametre kombinasyonu kullanılmaktadır. Popülasyon büyüklüğü n 50'dir, modüler modalite(c) 0,01 ve güç üssü(α), yinelemeler boyunca 0,1'dir. Öncelikle kelebeklerin modaliteyi anlamak için koku duyularına ihtiyaçları vardır. Burada denklem (3.5) kullanılarak koku değeri hesaplanır. Ardından KOA'nın global ve yerel arama yeteneklerini kontrol eden anahtar olasılığı p, kaotik bir sayı ile değiştirilir. Bunun üzerine P anahtarının değeri sabit olan 0.8 değerinden çıkıp her iterasyonda kaosdan yararlanarak güncellenmiş böylece çeşitlilik oluşturularak gelişim sağlanmıştır. İKOA algoritmasının akış diyagramı ve sözde kodu Şekil 4.1 ve Şekil 4.2'de gösterilmektedir.



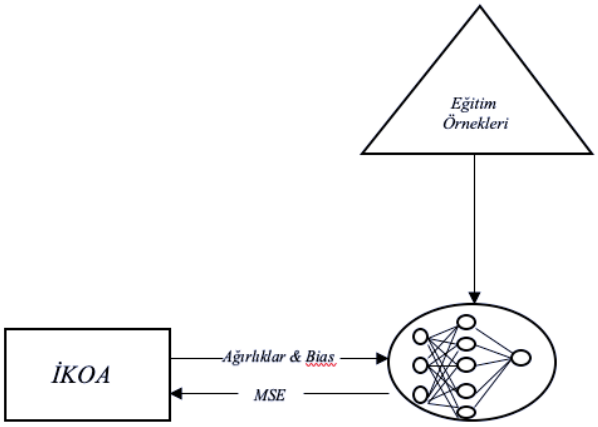
Şekil 4.1. İKOA algoritmasının akış diyagramı

*Amaç fonksiyonu f(**x**), **x**=(x_1,x_2, …, x_{dim}), boyut hessaplanır*
Kelebeklerin başlangıç popülasyonu oluşturulur. $\mathbf{x}_i$= (i=1,2, …, n)
$\mathbf{x}_i$'deki uyaran yoğunluğu I_i , f($\mathbf{x}_i$) ile belirlenir.
Sensör modalitesi c, güç üssü α ve anahtar olasılığı p belirlenir.
While** durdurma kriterleri karşılanmıyorsa **do
    for each** popülasyondaki o anki kelebek, bf **do
    bf'ye ait koku değeri denklem (3.5) kullanılarak hesaplanır.
    end for
    En iyi kelemek bulunur.(g_{best})
    for each** popülasyondaki o anki kelebek, bf **do
        p anahtarı kaotik haritalarla güncellenir.
        [0,1] arasında random sayı r üretilir.
        ***if** r<p ise*
        Denklem (3.6) kullanılır.
        else
        Denklem (3.7) kullanılır.
        En if
    end for
    değerler güncellenir
end while

Şekil 4.2. İKOA algoritmasının sözde kodu

4.2. YSA'nın İKOA ile Eğitimi

YSA girdi ve çıktılardan öğrenir. Bunun için YSA'daki ağırlıklar ve bias değerleri giriş ve çıkışlara göre güncellenir (Kiranyaz ve ark 2009). YSA'nın doğru sonuçlar vermesi ve başarılı sınıflandırmalar yapabilmesi, bias ve ağırlık değerlerinin en uygun şekilde güncellenmesi ile sağlanmaktadır. Literatürde, araştırmacılar Çok Katmanlı Algılayıcıyı (ÇKA) eğitmek için algoritmalar önermişlerdir. Ancak klasik teknikler bu optimizasyon probleminin çözümünde sıklıkla problemlerle karşılaşmaktadır. Büyük miktarda bilgi işlem süresine, büyük miktarda belleğe ihtiyaç duyma eğilimindedirler. Daha da önemlisi yerel optimuma takılıp kalitesiz çözümler üretme problemi ile karşı karşıyalardır. Bu zorlukların üstesinden gelmek için, ÇKA'yı eğitmek için meta-sezgisel algoritmalar kullanılmıştır (H Ergezer 2003). Bu tez çalışmasında, ağırlık ve bias değerlerini optimize etmek için hibrit İKOA-ÇKA algoritması geliştirilmiştir. ÇKA eğitiminde bir meta-sezgisel algoritma olan İKOA algoritması ilk kez kullanılmıştır. Ağırlıkların ve biasların optimizasyon süreci Şekil 4.3'de gösterilmektedir.



Şekil 4.3. İKOA-ÇKA'da ağırlıkların ve biasın optimizasyon süreci

ÇKA'daki ağırlık ve bias değerlerini optimize etmek için, ağırlık ve biasların öncelikle İKOA-ÇKA algoritmasında temsil edilmeleri gerekir. Bu amaçla, ağırlıklar ve biaslardan oluşan bir temsil vektörü kullanılır. Bu vektör Şekil 4.3'de ÇKA yapısına göre Denklem (4.11)'de gösterilmiştir.

$$Vector = \left(w_{i,j} \sqcup w_{j,k} \sqcup v_{0,j} \sqcup v_{0,k}\right) \tag{4.11}$$

Burada $w_{i,j}$ giriş katmanı ile gizli katman arasındaki ağırlıkların değerlerini, $w_{j,k}$ gizli katman ile çıkış katmanı arasındaki ağırlıkların değerlerini temsil etmektedir. $v_{0,j}$ giriş katmanı ile gizli katman arasındaki bias değerlerini, $v_{0,k}$ gizli katman ile çıkış katmanı arasındaki bias değerlerini temsil etmektedir. $\sqcup$ gösterimi iki kümenin birleşimini temsil etmektedir.

İKOA-ÇKA algoritmasının başında kelebek popülasyonu rastgele oluşturulur. Her kelebek farklı bir ÇKA'yı, yani Denklem (4.11)'deki temsil vektörünü temsil eder. Daha sonra, eğitim veri seti kullanılarak her bir kelebeğin kokusu hesaplanır ve popülasyondaki en iyi kelebek bulunur. Sonrasında her bir kelebeğin konumu (ÇKA'daki ağırlık ve bias değerleri) Denklem (3.6) ve Denklem (3.7) kullanılarak güncellenir. Denklemin seçilmesinde p anahtarı etkilidir. p anahtar değeri küçüldükçe Denklem (3.6)'nın, p anahtar değeri büyüdükçe ise Denklem (3.7)'nin seçilme olasılığı daha çok artar. Burada önemli husus iki denklem arasındaki dengenin sağlanması için p anahtarı değerinin optimize edilmesidir. Optimize için kaotik haritalar kullanılmış ve eğitim süreci başlatılmıştır. Bu süreç, Bölüm 3'de anlatılan, sonlandırma kriterleri karşılanana kadar devam eder.

5. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Bu bölümde tez çalışmasında geliştirilmiş olan İKOA ve İKOA-ÇKA algoritmalarının deneysel sonuçları sunulmaktadır ve elde edilen sonuçların yorumları verilmektedir. Deneylerde kullanılan donanım ve yazılımların özellikleri şu şekildedir: Microsoft Windows 10, Intel i5-3470 3.20 GHz, 6 GB bellek. Algoritmalar MATLAB R2018a'da kodlandı ve çalıştırıldı. Bu tez çalışmasındaki tüm istatistiksel analizler Microsoft Excel 2013 yazılımı ile yapılmıştır.

5.1. Kıyaslama Fonksiyonlarının Sonuçları

Bu bölümde, İKOA algoritmasının deneysel sonuçları sunulmaktadır. İKOA algoritmasının performansı Çizelge 3.3'de formülleri verilen 13 adet kıyaslama fonksiyonu üzerinde test edilmiştir. Bu fonksiyonların formülleri ve özellikleri Materyal ve Yöntem bölümünde verilmiştir. Bu fonksiyonlar global optimizasyon üzerine çalışanlar tarafından iyi bilinmektedirler ve optimizasyon algoritmalarının testi ve analizi için sıklıkla kullanılmaktadırlar. Bu 13 fonksiyonun global optimum değerleri ve arama aralığı Çizelge 3.3'de verilmektedir. Deneylerde her bir test fonksiyonun boyutu 30 olarak ayarlanmıştır. 10 farklı haritaya sahip İKOA algoritması ve KOA algoritması karşılaştırılmıştır. Algoritmaları adilce karşılaştırmak için, her bir algoritma her bir çalıştırmada aynı sayıda uygunluk fonksiyonu çağırmasını (FEs) gerçekleştirmiştir: f_1, f_6, f_{12} ve f_{13} için 75000 FEs; f_2 ve f_{11} için 100000 FEs; f_7, f_8 ve f_9 için 150000 FEs; f_3, f_4 ve f_5 için 250000 FEs. Algoritmalarda popülasyon boyutu 50 olarak ayarlanmıştır. Algoritmalar her bir fonksiyon üzerinde bağımsız olarak 30 kez çalıştırılmıştır.

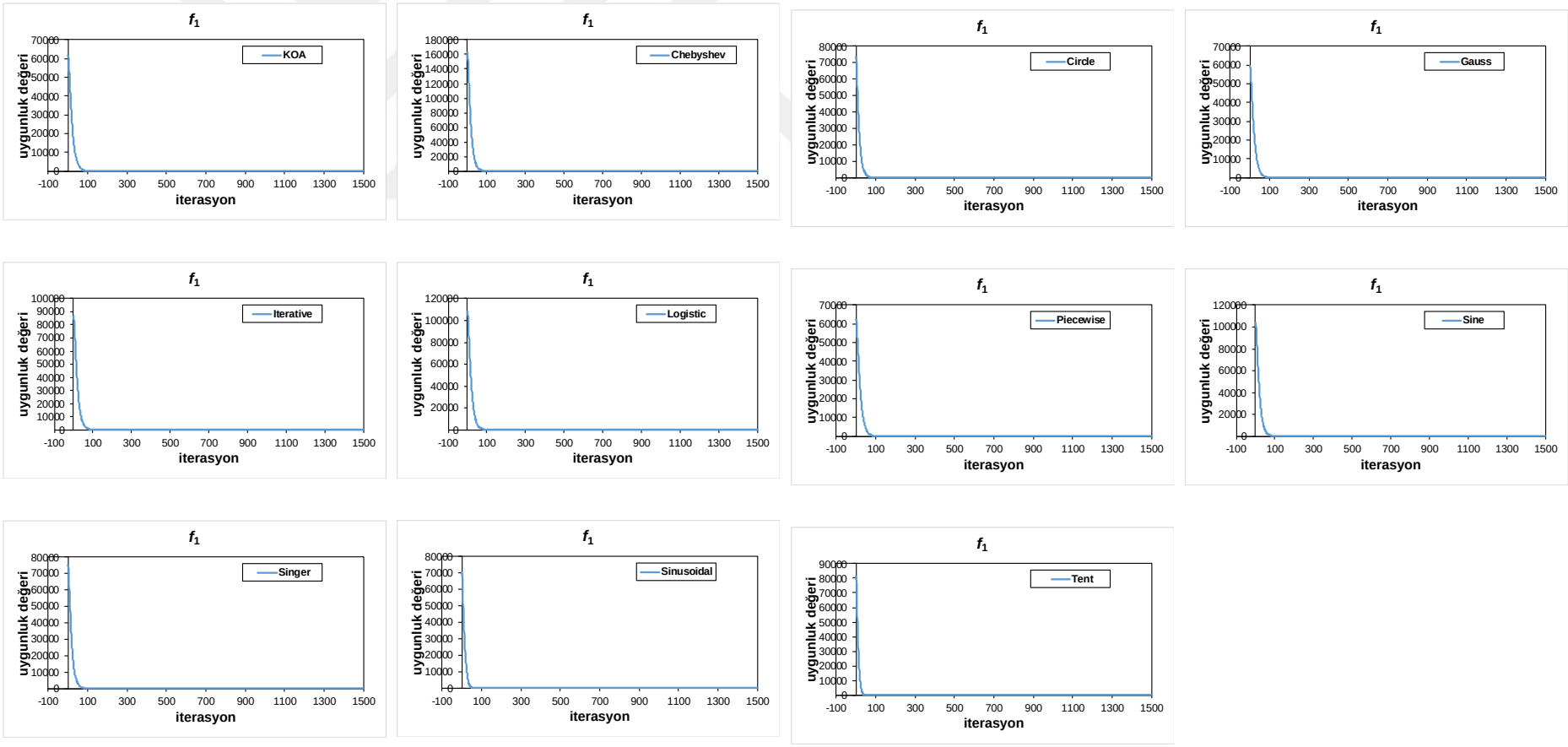
Çizelge 5.1'de 10 farklı haritaya sahip İKOA algoritmasının ve KOA algoritmasının ortalama sonuçları gösterilmektedir. Çizelgede en iyi sonuçlar koyu olarak gösterilmektedir. Sonuçlar incelendiğinde Tent haritalı İKOA algoritmasının diğer yöntemlere göre daha iyi sonuçlar elde ettiği görülmektedir. Çizelge 5.2'de algoritmaların ortalama çalışma süreleri saniye cinsinden gösterilmektedir. Şekil 5.1'de f_1 fonksiyonu için algoritmaların yakınsama grafikleri gösterilmektedir. Grafikler incelendiğinde algoritmaların birbirine rekabetçi sonuçlar gösterdiği görülmektedir. f_2- f_{13} fonksiyonlar için oluşturulan yakınsama grafikleri EKLER bölümünde sunulmaktadır.

Çizelge 5.1. Fonksiyonların 30 çalıştırma sonuçlarının ortalaması

	Chebyshev	**Circle**	**Gauss**	**İterative**	**Logistic**	**Piecewise**	**Sine**	**Singer**	**Sinusodial**	**Tent**	**KOA**
f_1	2,32E-12	1,39E-12	6,56E-05	3,43E-12	3,37E-12	9,03E-05	3,58E-12	2,81E-12	1,61E-07	**7,11E-28**	3,61E-12
f_2	8,86E+13	3,68E-10	4,04E+03	5,24E-10	3,74E-10	3,31E+00	3,87E-10	4,67E-10	2,53E-08	**2,20E-15**	5,31E-10
f_3	3,52E-17	3,50E-17	4,34E-10	5,88E-17	5,81E-17	4,38E-10	7,09E-17	6,45E-17	3,70E-17	**4,26E-28**	5,13E-17
f_4	2,77E-13	2,14E-13	1,17E-04	2,83E-13	2,84E-13	1,22E-04	2,76E-13	2,64E-13	1,08E-10	**3,33E-15**	2,78E-13
f_5	2,87E+01	2,87E+01	**2,69E+01**	2,87E+01	2,86E+01	2,70E+01	2,86E+01	2,86E+01	2,82E+01	2,90E+01	2,86E+01
f_6	1,89E+00	3,27E+00	3,48E-01	1,80E+00	2,07E+00	3,41E-01	2,10E+00	2,38E+00	**1,11E-01**	7,41E+00	1,94E+00
f_7	1,01E-03	8,26E-04	1,15E-02	9,84E-04	1,08E-03	1,25E-02	1,05E-03	9,53E-04	6,57E-04	**1,92E-04**	9,69E-04
f_8	-6,03E+03	-6,54E+03	-5,21E+03	-5,41E+03	-5,50E+03	-5,30E+03	-5,40E+03	-5,80E+03	**-8,66E+03**	-5,33E+02	-5,32E+03
f_9	7,74E+01	4,60E+00	1,57E+02	6,67E+01	6,91E+01	1,69E+02	7,61E+01	3,96E+01	2,51E+00	**0,00E+00**	5,89E+01
f_{10}	4,60E-14	1,61E-10	7,56E-02	4,74E-13	3,14E-15	8,30E-02	**1,95E-15**	1,32E-13	2,70E-04	7,64E-15	1,31E-11
f_{11}	4,42E-15	**0,00E+00**	9,53E-04	**0,00E+00**	**0,00E+00**	1,17E-03	2,96E-17	**0,00E+00**	6,37E-08	**0,00E+00**	**0,00E+00**
f_{12}	1,23E-01	3,24E-01	4,11E-01	1,04E-01	1,70E-01	3,34E-01	1,67E-01	6,17E-02	**6,33E-03**	1,61E+00	1,72E-01
f_{13}	1,50E+00	1,43E+00	8,73E-01	1,51E+00	1,69E+00	6,85E-01	1,69E+00	6,66E-01	**1,44E-01**	3,00E+00	1,82E+00

Çizelge 5.2. Ortalaması süre (saniye)

	Chebyshev	**Circle**	**Gauss**	**Iterative**	**Logistic**	**Piecewise**	**Sine**	**Singer**	**Sinusodial**	**Tent**	**KOA**
f_1	7,3	7,4	8,8	8,9	6,9	8,1	8,5	8,7	10,9	8,1	7,8
f_2	9,8	10,3	12,2	12,0	9,6	11,3	12,1	12,5	11,0	11,5	10,8
f_3	64,9	65,4	70,1	69,8	61,4	67,8	66,4	67,5	70,2	69,6	65,6
f_4	25,2	24,8	29,3	28,3	23,0	26,9	28,9	29,8	29,1	29,5	25,2
f_5	30,0	30,2	35,1	33,9	27,5	32,0	34,4	35,1	33,6	35,2	30,3
f_6	7,6	7,7	9,1	8,5	7,0	8,8	8,9	9,1	10,8	10,4	8,0
f_7	16,5	17,3	20,0	19,4	16,0	19,1	19,7	20,3	18,4	19,0	18,2
f_8	17,1	17,2	20,0	19,7	16,2	18,7	20,2	20,2	19,2	19,4	18,0
f_9	16,3	16,6	19,8	18,9	16,0	18,4	19,3	19,6	18,8	19,7	16,9
f_{10}	8,3	8,4	10,0	9,5	8,4	9,4	9,6	9,9	10,0	12,5	8,8
f_{11}	11,7	12,3	14,1	13,1	11,8	13,3	14,1	14,5	13,3	13,7	12,8
f_{12}	13,3	13,4	14,4	14,1	12,6	14,0	14,8	14,9	14,4	19,3	13,9
f_{13}	13,1	13,3	14,4	14,0	12,5	14,2	14,8	14,9	14,3	14,8	13,8



Şekil 5.1. f1 için yakınsama grafiği

Çizelge 5.1'deki kıyaslama fonksiyonlarına ilişkin algoritmaların sonuçlarına göre, Tent haritalı İKOA algoritmasının kıyaslama fonksiyonlarının çoğunda diğer algoritmalardan daha iyi performans gösterdiği açıktır. Tent haritalı İKOA algoritması, 7 kıyaslama fonksiyonunda en iyi sonuçları elde etmiştir. Aynı zamanda bir kıyaslama fonksiyonunda en iyi üçüncü sonuca sahiptir. Çizelge 5.1'e göre, önerilen İKOA algoritmasının klasik KOA algoritmasından daha iyi performansa sahip olduğu açıktır.

5.2. Sınıflandırma Veri Setlerinin Sonuçları

İKOA-ÇKA algoritmasını test etmek ve başarısını ölçmek için xor, balon, iris, meme kanseri ve kalp olmak üzere 5 farklı veri seti kullanılmıştır. Bu veri setlerin özellikleri Materyal ve Yöntem bölümünde verilmiştir. İKOA-ÇKA'nın başarısını doğrulamak için, İKOA-ÇKA algoritması literatürdeki kelebek optimizasyonu algoritması (Arora ve Singh 2019) tabanlı KOA-ÇKA, yarasa optimizasyonu algoritması (Yang 2010) tabanlı BAT-ÇKA, maddenin halleri optimizasyonu algoritması (Cuevas ve ark 2014) tabanlı SMS-ÇKA ve (Hecht-Nielsen 1992) GY algoritmaları ile karşılaştırılmıştır. Tüm veri setleri için, bias ve ağırlıkların başlangıç değerleri [-10, 10] aralığında rastgele üretilmiştir. Karşılaştırma için kullanılan algoritmaların eğitim/test sayısı, sınıf sayısı, ÇKA yapısı gibi parametreler Çizelge 5.3'de gösterilmektedir.

Bu tez çalışmasında, en uygun gizli katman nöron sayısı bulunmaya odaklanılmamıştır. Literatürde gizli katman nöron sayısı genellikle $(2 \times n) + 1$ formülü ile hesaplanır. Bu nedenle çalışmamızda, gizli katman nöron sayısını bu formül ile elde ettik. Formülde n girdi katmanındaki nöron sayısıdır. Çizelge 5.3'de veri setlerinde kullanılan ÇKA yapısı gösterilmektedir.

ÇKA eğitimi için İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA, ve GY algoritmalarının ortak bir amacı vardır. Bu amaç bias ve ağırlık değerlerini optimuma getirmektir. Öngörülen istatistiksel sonuçlar, son yinelemede algoritmalar tarafından elde edilen en iyi MSE'lerin ortalaması ve standart sapmasıdır. Son yinelemede MSE'nin daha düşük ortalama ve standart sapması daha iyi performansı gösterir (Mirjalili 2015).

Çizelge 5.3'ü incelediğimizde 8 eğitim/test örneği, 3 özellik ve 2 sınıf ile xor veri kümesinin yapay sinir ağları başarısı için en kolay veri seti olduğu görülmüştür. Balon veri setini incelediğimizde 4 özellik 20 eğitim/test örneği ve 2 sınıf ile zorluk açısından ikinci sıradadır. Iris veri setine bakınca 4 özellik, 150 eğitim/test örneği ve 3 sınıf özelliği bulunmaktadır. Meme kanseri veri setine baktığımızda 9 özellik, 599 eğitim, 100 test

örneği ve 2 sınıf bulunmaktadır. Kalp veri setinde ise 22 özellik ve 80/187 test/eğitim örneği olduğunu görmekteyiz. Görüldüğü gibi, İKOA-ÇKA algoritmasının başarısını doğrulamak için farklı eğitim/test örnekleri ve farklı zorluk seviyelerine sahip veri setleri seçilmiştir. Aktivasyon fonksiyonu olarak ise literatürde en çok kullanılan sigmoid fonksiyonu seçilmiştir.

Çizelge 5.3. Sınıflandırma veri kümelerinin özellikleri

Veri Setleri	**Özellik Sayısı**	**Eğitim**	**Test**	**Sınıf Sayısı**	**ÇKA Yapısı**	**Vektör boyutu**
Xor	3	8	8	2	3-7-1	36
Balon	4	20	20	2	4-9-1	55
Iris	4	150	150	3	4-9-3	75
Meme Kanseri	9	599	100	2	9-19-1	210
Kalp	22	80	187	2	22-45-1	1081

Çizelge 5.4'de İKOA-ÇKA algoritmasının MSE sonuçları literatürdeki KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY olmak üzere 4 algoritmanın MSE sonuçları ile karşılaştırılmıştır. KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmalarının sonuçları (Irmak ve Gülcü) yayınından alınmıştır. Karşılaştırma yapılırken 10 harita içerisinden en başarılı sonuçlar üreten Sine haritası seçilmiş aşağıdaki sonuçlar çizelgesi hazırlanmıştır. Çizelge 5.4 incelendiğinde İKOA-ÇKA algoritmasının BAT-ÇKA, SMS-ÇKA ve GY algoritmalarını geride bıraktığı, KOA-ÇKA algoritmasını ise xor ve iris veri setlerinde geride bıraktığı ve diğer veri setlerinde ise rekabetçi sonuçlar sergilediği gözlemlenmektedir. Koyu renkli sonuçlar en iyi sonucu göstermektedir.

Çizelge 5.4. Eğitim verilerine ilişkin MSE sonuçlarının ortalama ve standart sapması

Veri Setleri		İKOA-ÇKA	KOA-ÇKA	BAT-ÇKA	SMS-ÇKA	GY
Xor	Ortalama	**5,24E-03**	7,83E-03	1,27E-01	1,33E-01	1,41E-01
	Standart sapma	**7,95E-03**	9,62E-03	6,22E-02	3,39E-02	1,65E-01
Balon	Ortalama	2,89E-09	**2,79E-09**	1,07E-02	1,11E-02	1,00E-01
	Standart sapma	8,54E-09	**6,90E-09**	2,83E-02	1,31E-02	1,51E-01
Iris	Ortalama	**4,71E-02**	4,74E-02	2,10E-01	2,58E-01	5,38E-02
	Standart sapma	**7,91E-03**	8,15E-03	1,38E-01	5,12E-02	1,30E-01
Meme kanseri	Ortalama	1,82E-03	**1,78E-03**	6,83E-03	2,27E-02	2,88E-02
	Standart sapma	8,02E-05	**8,25E-05**	7,92E-03	4,33E-03	1,21E-01
Kap	Ortalama	1,19E-01	**1,15E-01**	1,51E-01	1,18E-01	2,98E-01
	Standart sapma	6,37E-03	**4,75E-03**	3,21E-02	3,13E-02	1,32E-01

Çizelge 5.5'de İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY test verilerindeki ortalama sınıflandırma doğruluğu gösterilmektedir. Koyu renkle gösterilen değerler en iyi sonucu ifade etmektedir. Çizelge 5.5'deki sonuçlar incelendiğinde İKOA-ÇKA'nın başarılı olduğu görülmektedir. Çizelge 5.6'da algoritmaların ortalama çalışma süreleri görülmektedir. Çizelge 5.6 incelendiğinde en hızlı algoritmanı GY algoritması olduğu gözlemlenmektedir.

Çizelge 5.5. Test verilerinde ortalama sınıflandırma doğruluğu

Veri Setleri	İKOA-ÇKA	KOA-ÇKA	BAT-ÇKA	SMS-ÇKA	GY
Xor	**100,00%**	**100,00%**	85,00%	83,75%	84,17%
Balom	**100,00%**	**100,00%**	98,83%	99,17%	90,00%
Iris	**97,18%**	**97,18%**	82,91%	86,78%	90,82%
Meme Kanseri	**99,43%**	99,37%	96,13%	85,67%	93,03%
Kalp	**74,83%**	74,46%	71,35%	68,57%	61,66%

Çizelge 5.6. Algoritmaların ortalama çalışma süreleri (saniye)

Dataset	İKOA-ÇKA	KOA-ÇKA	BAT-ÇKA	SMS-ÇKA	GY
Xor	8,9	9,2	4,7	5,5	**1,6**
Balon	37,9	37,0	19,4	20,2	**1,7**
Iris	1338,7	1408,6	775,2	881,6	**9,6**
Meme Kanseri	5035,1	5101,9	2607,3	2830,3	**29,5**
Kalp	1686,6	1479,0	937,0	892,4	**16,5**

Çizelge 5.7. Xor veri seti için duyarlılık, özgüllük, kesinlik ve F1-score sonuçları

Algoritma	Duyarlılık	Özgüllük	Kesinlik	F_1-Score
İKOA-ÇKA	**100,00%**	**100,00%**	**100,00%**	**1,0000**
KOA-ÇKA	**100,00%**	**100,00%**	**100,00%**	**1,0000**
BAT-ÇKA	82,50%	87,50%	89,44%	0,8435
SMS-ÇKA	81,70%	86,70%	88,90%	0,8349
GY	84,17%	84,17%	85,94%	0,8221

Çizelge 5.7'de xor veri seti için duyarlılık, özgüllük, kesinlik ve F1-score değerlerinin sonuçları gösterilmektedir. Xor veri seti kümesinde 3 giriş, 8 eğitim/test örneği ve tek bir çıkış değeri vardır. Bu veri setinin İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmaları ile istatistiksel olarak elde edilen sonuçları Çizelge 5.7'de gösterilmektedir. Sonuçlar incelendiğinde İKOA-ÇKA algoritmasının BAT-ÇKA, SMS-ÇKA ve GY algoritmalarını geride bıraktığı, KOA-ÇKA algoritması ile aynı başarı oranı sergilediği gözlemlenmektedir.

Balon veri setini 4 özellik, 18 eğitim/test örneği ve 2 sınıf oluşturmaktadır. ÇKA yapısını 4 giriş nöronu, 9 gizli katman nöronu ve 1 çıkış nöronu oluşturmaktadır. İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmalarının bu veri seti üzerindeki duyarlılık, özgüllük, kesinlik ve F1-score değerleri Çizelge 5.8'de gösterilmektedir. Çizelge 5.8 incelendiğinde ve İKOA-ÇKA algoritmasının BAT-ÇKA, SMS-ÇKA ve GY algoritmalarını geride bıraktığı, KOA-ÇKA algoritması ile aynı başarı oranı sergilediği gözlemlenmektedir. Burada dikkat edilmesi gereken bir diğer husus GY algoritmasının doğruluk değerinin düşük olmasıdır.

Çizelge 5.8. Balon veri seti için duyarlılık, özgüllük, kesinlik ve F1-score sonuçları

Algoritma	Duyarlılık	Özgüllük	Kesinlik	F_1-Score
İKOA-ÇKA	**100,00%**	**100,00%**	**100,00%**	**1,0000**
KOA-ÇKA	**100,00%**	**100,00%**	**100,00%**	**1,0000**
BAT-ÇKA	99,17%	98,61%	98,30%	0,9859
SMS-ÇKA	99,20%	99,20%	98,80%	0,9894
GY	75,00%	100,00%	86,67%	0,7838

Iris veri seti 4 özellik, 150 eğitim/test örneği ve 3 sınıftan oluşmaktadır. ÇKA 4 giriş nöronu, 9 gizli katman nöronu ve 3 çıkış nöronundan oluşmaktadır. 75 boyuta

sahiptir. İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmalarının bu veri seti üzerindeki duyarlılık, özgüllük, kesinlik ve F1-score değerleri Çizelge 5.9'da gösterilmektedir. Çizelge 5.9 incelendiğinde İKOA-ÇKA'nın BAT-ÇKA, SMS-ÇKA ve GY algoritmasını geride bıraktığı KOA-ÇKA algoritmasına ise rekabetçi sonuçlar gösterdiği gözlemlenmektedir.

Çizelge 5.9. İris veri seti için duyarlılık, özgüllük, kesinlik ve F1-score sonuçları

Algoritma	Duyarlılık	Özgüllük	Kesinlik	F_1-Score
İKOA-ÇKA	97,18%	98,59%	97,22%	0,9718
KOA-ÇKA	**97,38%**	**98,69%**	**97,42%**	**0,9738**
BAT-ÇKA	82,91%	91,46%	78,07%	0,7883
SMS-ÇKA	82,10%	91,10%	81,00%	0,8002
BP	90,82%	95,41%	88,43%	0,8899

Literatürde önemli bir yere sahip olan bir başka zorlu veri seti meme kanseri veri setidir. Bu veri seti 9 özellik, 599 eğitim örneği, 100 test örneği ve 2 sınıftan oluşmaktadır. ÇKA 9 giriş nöronu, 19 gizli katman nöronu ve 1 çıkış nöronundan oluşmaktadır. Vektör uzunluğu 210 boyuta sahiptir. İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmalarının bu veri seti üzerindeki duyarlılık, özgüllük, kesinlik ve F1-score değerleri Çizelge 5.10'da gösterilmektedir. Çizelge 5.10 incelendiğinde İKOA-ÇKA'nın tüm algoritmaları geride bıraktığı ve başarılı olduğu gözlemlenmektedir.

Çizelge 5.10. Meme Kanseri veri seti için duyarlılık, özgüllük, kesinlik ve F1-score sonuçları

Algoritma	Duyarlılık	Özgüllük	Kesinlik	F_1-Score
İKOA-ÇKA	**99,05%**	**99,54%**	**98,33%**	**0,9866**
KOA-ÇKA	**99,05%**	99,45%	98,02%	0,9850
BAT-ÇKA	87,46%	98,44%	93,29%	0,8922
SMS-ÇKA	50,20%	95,10%	83,70%	0,5828
GY	85,87%	94,94%	81,16%	0,8255

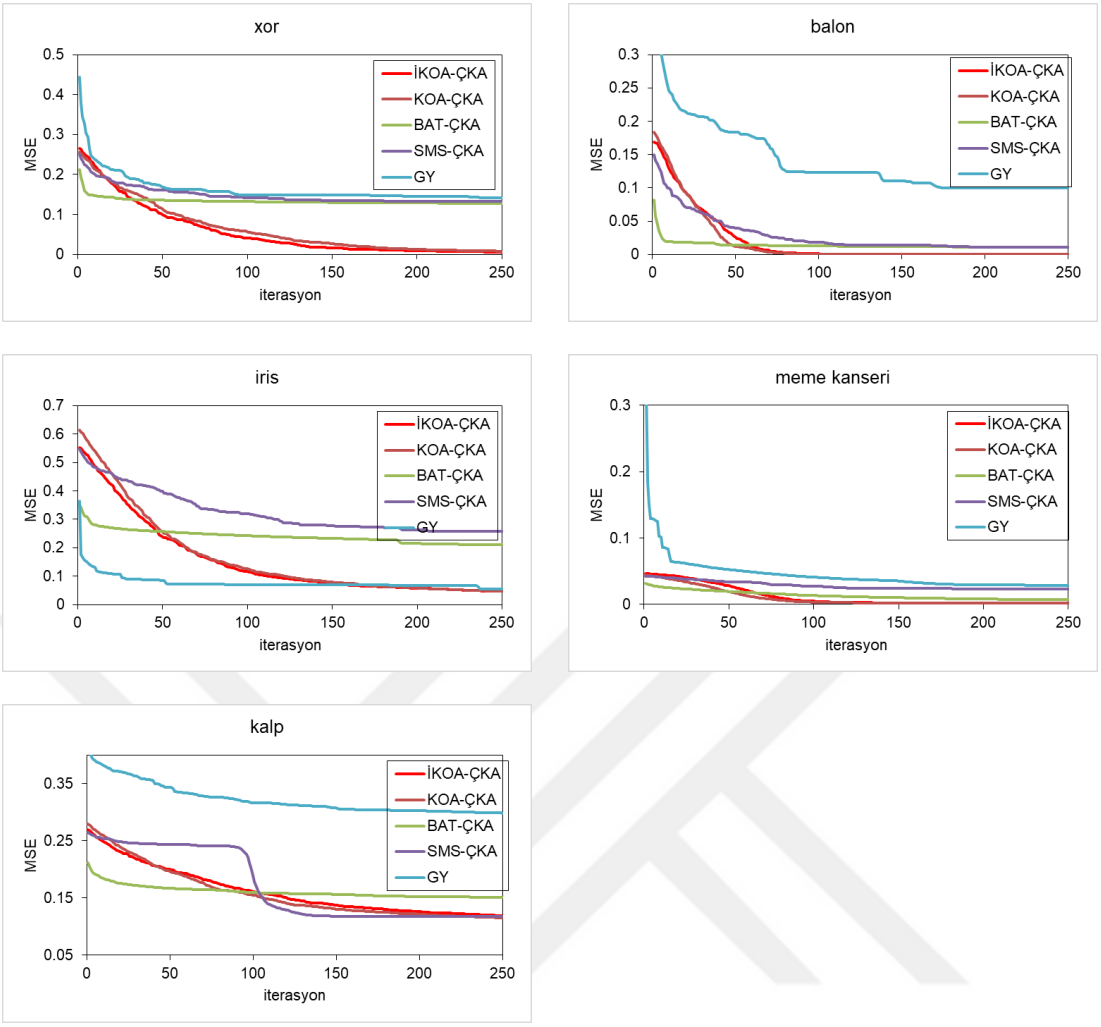
Son veri setimiz kalp veri setidir. Bu veri seti 22 özellik, 80 eğitim örneği, 187 test örneği ve 2 sınıftan oluşmaktadır. ÇKA 22 giriş nöronu, 45 gizli katman nöronu ve 1 çıkış nöronundan oluşmaktadır. Vektör uzunluğu 1081 boyuta sahiptir. İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmalarının bu veri seti üzerindeki duyarlılık, özgüllük, kesinlik ve F1-score değerleri Çizelge 5.11'de gösterilmektedir.

Çizelge 5.11 incelendiğinde, İKOA-ÇKA'nın tüm algoritmaları geride bıraktığı ve başarılı olduğu gözlemlenmektedir. Ayrıca geri yayılım algoritmasının bu veri setindeki düşük başarısı dikkat çekicidir.

Çizelge 5.11. Kalp veri seti için duyarlılık, özgüllük, kesinlik ve F1-score sonuçları

Algoritma	Duyarlılık	Özgüllük	Kesinlik	F_1-Score
İKOA-ÇKA	**74,90%**	**74,00%**	**97,09%**	**0,8446**
KOA-ÇKA	74,71%	71,56%	96,79%	0,8422
BAT-ÇKA	71,32%	71,78%	96,73%	0,8180
SMS-ÇKA	68,39%	70,67%	96,41%	0,7980
GY	62,00%	57,78%	88,48%	0,6965

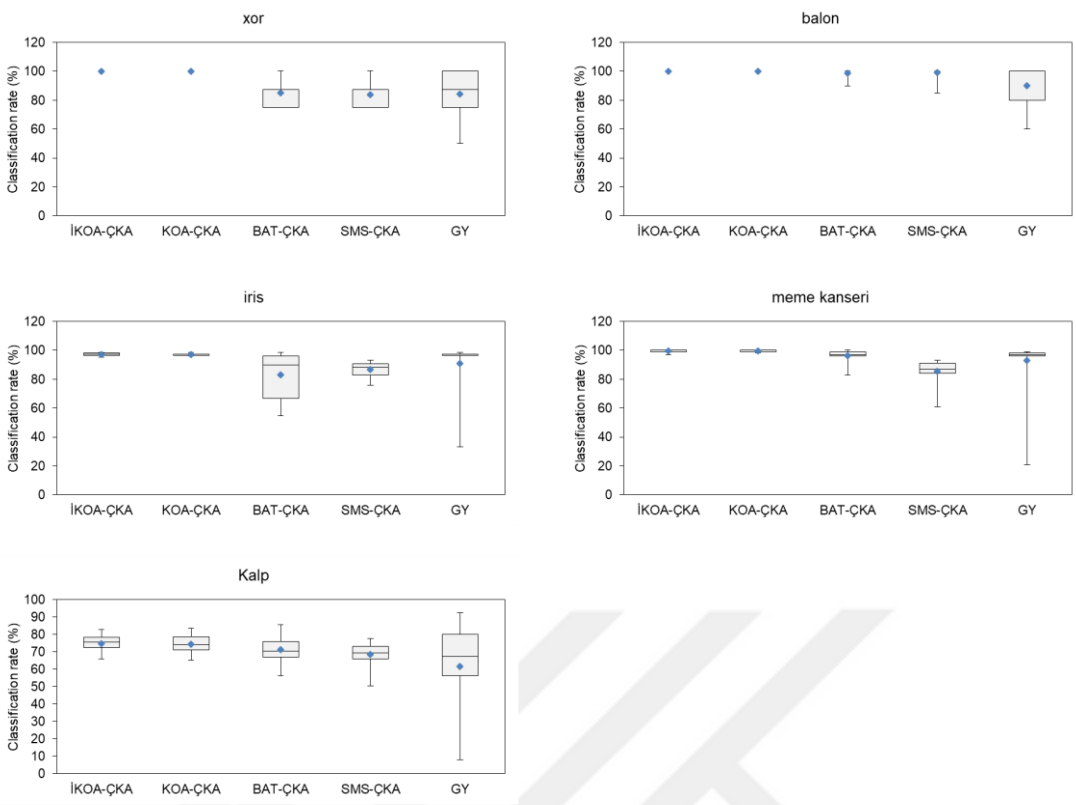
Şekil 5.2' de algoritmaların yakınsama grafikleri gözlemlenmektedir. Yakınsama grafikleri incelendiğinde, xor ve balon veri setlerinde İKOA-ÇKA ve KOA-ÇKA algoritmalarının daha iyi sonuçlar verdiği görülmektedir. İris veri setinde İKOA-ÇKA, KOA-ÇKA ve GY algoritmasının daha iyi sonuç verdiği görülmektedir. Meme kanseri veri setinde tüm algoritmaların rekabetçi sonuç gösterdiği fakat İKOA-ÇKA, KOA-ÇKA ve BAT-ÇKA algoritmaların global optimumu aramada daha ısrarcı olduğu gözükmektedir. Kalp veri setinde ise İKOA-ÇKA ve KOA-ÇKA'nın daha iyi sonuçlar verdiği görülmektedir. Genel olarak değerlendirildiğinde, İKOA-ÇKA algoritmasının lokal optimum noktalarına takılmadığı global optimumu aramada ısrarcı olduğu söylenebilir.



Şekil 5.2. Algoritmaların yakınsama grafikleri

Şekil 5.3'de İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmalarının, sınıflandırma doğruluğu sonuçlarının kutu grafikleri görülmektedir. Burada İKOA-ÇKA'nın başarılı olduğu görülmektedir.

Çizelge 5.12'de İKOA-ÇKA, KOA-ÇKA, BAT-ÇKA, SMS-ÇKA ve GY algoritmalarının Friedman testi sonuçları sunulmaktadır. En iyi sonucun İKOA-ÇKA algoritmasına ait olduğu görülmektedir. Dikkat çekici bir diğer husus ise literatürde yer edinmiş SMS-ÇKA ve GY algoritmalarının bu kadar düşük başarı sergilemesidir. Çizelgedeki Friedman test sonucuna göre, İKOA-ÇKA algoritması 4,7 sıralama puanı ile diğer algoritmalardan daha üst sıralarda yer almaktadır. KOA-ÇKA algoritması 4,3 sıralama puanı ile ikinci sıralamaya sahiptir. BAT-ÇKA algoritması 2,4 sıralama puanı ile üçüncü sıralamaya sahiptir. SMS-ÇKA algoritması ve GY algoritması 1,8 sıralama puanı ile dördüncü sıralamaya sahiptir.



Şekil 5.3. Algoritmaların kutu grafikleri

Çizelge 5.12. Algoritmaların sınıflandırma oranına göre ortalama sıralaması (Friedman testi)

Algoritma	**Sıralama**
İKOA-ÇKA	**4,7**
KOA-ÇKA	4,3
BAT-ÇKA	2,4
SMS-ÇKA	1,8
GY	1,8

6. SONUÇLAR VE ÖNERİLER

6.1 Sonuçlar

Yapay sinir ağlarının çalışma mekanizmasına göre ağırlıkların güncellendiği değerler öğrenme başarıları ile doğrudan etkilidirler. Bu çalışmada YSA'yı eğitmek ve ağırlıklar ve biasın optimum değere getirilmesi için ilk defa önerilen İKOA algoritması ele alınmıştır. İKOA algoritması arama stratejisi ve yerel optimumdan kaçma yeteneği üstün olan KOA algoritmasının kaotik haritalardan yararlanarak geliştirilmiş halidir. İKOA geliştirilirken, KOA algoritmasının yerel arama ve global arama denklemleri arasındaki dengeyi kuran p anahtarı parametresi kaotik haritalarla güncellenmiş ve bu iki denklem arasındaki seçim dengesi iyileştirilmiştir. İKOA algoritmasının başarısı 13 kıyaslama fonksiyonu üzerinde doğrulandı. Sonrasında, geliştirilen İKOA algoritması YSA'daki bias ve ağırlık değerlerini optimum noktasına getirebilmek için kullanıldı ve YSA'nın öğrenme yeteneğinin artırılması amaçlandı. Elde edilen İKOA-ÇKA algoritmasının başarısı literatürde önemli olan iris, balon, kalp, meme kanseri ve xor veri setleri üzerinde test edildi. Elde edilen sonuçlar yine literatürde önemli, BAT-ÇKA, SMS-ÇKA ve GY olmak üzere dört algoritma ile karşılaştırıldı. Karşılaştırma sonuçlarında İKOA-ÇKA algoritmasının BAT-ÇKA, SMS-ÇKA ve GY algoritmalarını geride bıraktığı, KOA-ÇKA algoritması ile başa baş rekabet ettiği ve bazı durumlarda öne geçtiği gözlemlenmiştir. İKOA algoritmasının bias ve ağırlık değerlerini bulmada başarılı olduğu sonucuna varılmıştır. Bundan dolayı ÇKA eğitimi için uygunluğu ispatlanmıştır. Ayrıca kullanılan veri setleri üzerinde elde edilen değerlerin sınıflandırma doğruluğu oranı da çok yüksektir. Bunun en büyük nedeni, başarısız olan algoritmaların aksine İKOA algoritmasının üstün arama ve keşif yeteneğidir.

6.2 Öneriler

Literatürde YSA eğitimi için birçok algoritma denenmiştir. Ancak bias ve ağırlık değerlerini optimuma en yakın noktaya getiren algoritmaların meta sezgisel algoritmalar olduğu görülmüştür. Geliştirmek için ele alınan KOA algoritması bir meta sezgisel algoritmadır ve arama stratejisi, yerel optimumdan kaçma, üstün sömürü ve keşif yeteneği bakımından gelişmiş bir algoritmadır. Bundan dolayı YSA eğitimi için son derece kullanışlı ve başarılıdır, literatürde başarısını ispat etmiş birçok algoritmayı geride bırakmayı başarmıştır. KOA algoritması gelişmeye ve geliştirilmeye açık bir algoritmadır. İlerleyen zamanda hibrit modeller üzerinde çalışılması, farklı arama

stratejileri kullanarak performans artırımı, parametre analizleri kullanılarak popülasyonun daha iyi dağılımının sağlanması gibi bias ve ağırlık değerlerini optimuma getirmeye yönelik çalışmalar yapılması önerilmektedir.

7. KAYNAKLAR

Al Nuaimi ZNAM, Abdullah R, 2017. Neural network training using hybrid particlemove artificial bee colony algorithm for pattern classification. Journal of Information and Communication Technology, 16, 2, 314-34.

Aljarah I, Faris H, Mirjalili S, 2018. Optimizing connection weights in neural networks using the whale optimization algorithm. Soft Computing, 22, 1, 1-15.

Arora S, Singh S, 2019. Butterfly optimization algorithm: a novel approach for global optimization. Soft Computing, 23, 3, 715-34.

Battiti R, Tecchiolli G, 1995. Training neural nets with the reactive tabu search. IEEE transactions on neural networks, 6, 5, 1185-200.

Blair RB, Launer AE, 1997. Butterfly diversity and human land use: Species assemblages along an urban grandient. Biological conservation, 80, 1, 113-25.

Blum C, Socha K. Training feed-forward neural networks with ant colony optimization: An application to pattern classification. Fifth International Conference on Hybrid Intelligent Systems (HIS'05), 6 pp.

Chen J-F, Do QH, Hsieh H-N, 2015. Training artificial neural networks by a hybrid PSO-CS algorithm. Algorithms, 8, 2, 292-308.

Cuevas E, Echavarría A, Ramírez-Ortegón MA, 2014. An optimization algorithm inspired by the States of Matter that improves the balance between exploration and exploitation. Applied intelligence, 40, 2, 256-72.

Dash R, 2018. Performance analysis of a higher order neural network with an improved shuffled frog leaping algorithm for currency exchange rate prediction. Applied Soft Computing, 67, 215-31.

Erdogan F, Gulcu S, 2021. Training of the Artificial Neural Networks using Crow Search Algorithm. International Journal of Intelligent Systems and Applications in Engineering, 9, 3, 101-8.

Ghaleini EN, Koopialipoor M, Momenzadeh M, Sarafraz ME, Mohamad ET, Gordan B, 2019. A combination of artificial bee colony and neural network for approximating the safety factor of retaining walls. Engineering with Computers, 35, 2, 647-58.

Gullipalli TR, 2021. An Improved under Sampling Approaches for Concept Drift and Class Imbalance Data Streams using Improved Cuckoo Search Algorithm. Turkish Journal of Computer and Mathematics Education (Turcomat), 12, 2, 2267–75-–75.

Gülcü Ş, 2021. An Improved Animal Migration Optimization Algorithm to Train the Feed-Forward Artificial Neural Networks. Arabian Journal for Science and Engineering, 1-25.

H Ergezer MD, E Özdemir, 2003. Yapay sinir ağları ve tanıma sistemleri. Pivolka.

Hecht-Nielsen R, 1992. Theory of the backpropagation neural network. In: Neural networks for perception. Eds: Elsevier, p. 65-93.

Heidari AA, Faris H, Mirjalili S, Aljarah I, Mafarja M, 2020. Ant lion optimizer: theory, literature review, and application in multi-layer perceptron neural networks. In: Nature-Inspired Optimizers. Eds: Springer, p. 23-46.

Herz F, Ungar L, Zhang J, Wachob D, Salganicoff M, (1998). System and method for scheduling broadcast of and access to video programs and other data using customer profiles, Google Patents.

Ilonen J, Kamarainen J-K, Lampinen J, 2003. Differential evolution training algorithm for feed-forward neural networks. Neural Processing Letters, 17, 1, 93-105.

Irmak B, Gülcü Ş, Training of the Feed-Forward Artificial Neural Networks using Butterfly Optimization Algorithm. MANAS Journal of Engineering, 9, 2, 160-8.

Jaddi NS, Abdullah S, 2018. Optimization of neural network using kidney-inspired algorithm with control of filtration rate and chaotic map for real-world rainfall forecasting. Engineering Applications of Artificial Intelligence, 67, 246-59.

Karaboga D, Akay B, Ozturk C. Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks. International conference on modeling decisions for artificial intelligence, 318-29.

Karakaş B, 2020. Sincap arama algoritması ile yapay sinir ağlarının eğitimi.

Kiranyaz S, Ince T, Yildirim A, Gabbouj M, 2009. Evolutionary artificial neural networks by multi-dimensional particle swarm optimization. Neural networks, 22, 10, 1448-62.

Koç ML, Balas CE, Arslan A, 2004. Taş dolgu dalgakıranların yapay sinir ağları ile ön tasarımı. Teknik Dergi, 15, 74.

Kulluk S, 2013. A novel hybrid algorithm combining hunting search with harmony search algorithm for training neural networks. Journal of the Operational Research Society, 64, 5, 748-61.

Kulluk S, Ozbakir L, Baykasoglu A, 2012. Training neural networks with harmony search algorithms for classification problems. Engineering Applications of Artificial Intelligence, 25, 1, 11-9.

Mendes R, Cortez P, Rocha M, Neves J. Particle swarms for feedforward neural network training. Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290), 1895-9.

Mirjalili S, 2015. How effective is the Grey Wolf optimizer in training multi-layer perceptrons. Applied Intelligence, 43, 1, 150-61.

Mirjalili S, Hashim SZM, Sardroudi HM, 2012. Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. Applied Mathematics and Computation, 218, 22, 11125-37.

Ojha VK, Abraham A, Snášel V, 2017. Metaheuristic design of feedforward neural networks: A review of two decades of research. Engineering Applications of Artificial Intelligence, 60, 97-116.

Özbakir L, Baykasoğlu A, Kulluk S, Yapıcı H, 2009. TACO-miner: An ant colony based algorithm for rule extraction from trained neural networks. Expert Systems with Applications, 36, 10, 12295-305.

Pandya AS, Macy RB, 1995. Pattern recognition with neural networks in C++, CRC press, p.

Pecora LM, Carroll TL, 1990. Synchronization in chaotic systems. Physical review letters, 64, 8, 821.

Pereira LA, Rodrigues D, Ribeiro PB, Papa JP, Weber SA. Social-spider optimization-based artificial neural networks training and its applications for Parkinson's disease identification. 2014 IEEE 27th international symposium on computer-based medical systems, 14-7.

Pirim AGH, 2006. Yapay zeka. Journal of Yaşar University, 1, 1, 81-93.

Pollard E, Yates TJ, 1994. Monitoring butterflies for ecology and conservation: the British butterfly monitoring scheme, Springer Science & Business Media, p.

Raguso RA, 2008. Wake up and smell the roses: the ecology and evolution of floral scent. Annual review of ecology, evolution, and systematics, 39, 549-69.

Saccheri I, Kuussaari M, Kankare M, Vikman P, Fortelius W, Hanski I, 1998. Inbreeding and extinction in a butterfly metapopulation. Nature, 392, 6675, 491-4.

Sarangi PP, Sahu A, Panda M, 2014. Training a feed-forward neural network using artificial bee colony with back-propagation algorithm. In: Intelligent computing, networking, and informatics. Eds: Springer, p. 511-9.

Sexton RS, Dorsey RE, Johnson JD, 1999. Beyond backpropagation: using simulated annealing for training neural networks. Journal of Organizational and End User Computing (JOEUC), 11, 3, 3-10.

Tang R, Fong S, Deb S, Vasilakos AV, Millham RC, 2018. Dynamic group optimisation algorithm for training feed-forward neural networks. Neurocomputing, 314, 1-19.

Türkoğlu B, 2019. Yapay sinir ağlarının yapay alg algoritması ile eğitimi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü.

Vazquez RA. Training spiking neural models using cuckoo search algorithm. 2011 IEEE Congress of Evolutionary Computation (CEC), 679-86.

Wyatt TD, 2003. Pheromones and animal behaviour: communication by smell and taste, Cambridge University Press, p.

Yang X-S, 2010. A new metaheuristic bat-inspired algorithm. In: Nature inspired cooperative strategies for optimization (NICSO 2010). Eds: Springer, p. 65-74.

Zamani M, Sadeghian A, 2010. A variation of particle swarm optimization for training of artificial neural networks. In: Computational Intelligence and Modern Heuristics. Eds: IntechOpen, p.

Zanchettin C, Ludermir TB, Almeida LM, 2011. Hybrid training method for MLP: optimization of architecture and training. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 41, 4, 1097-109.

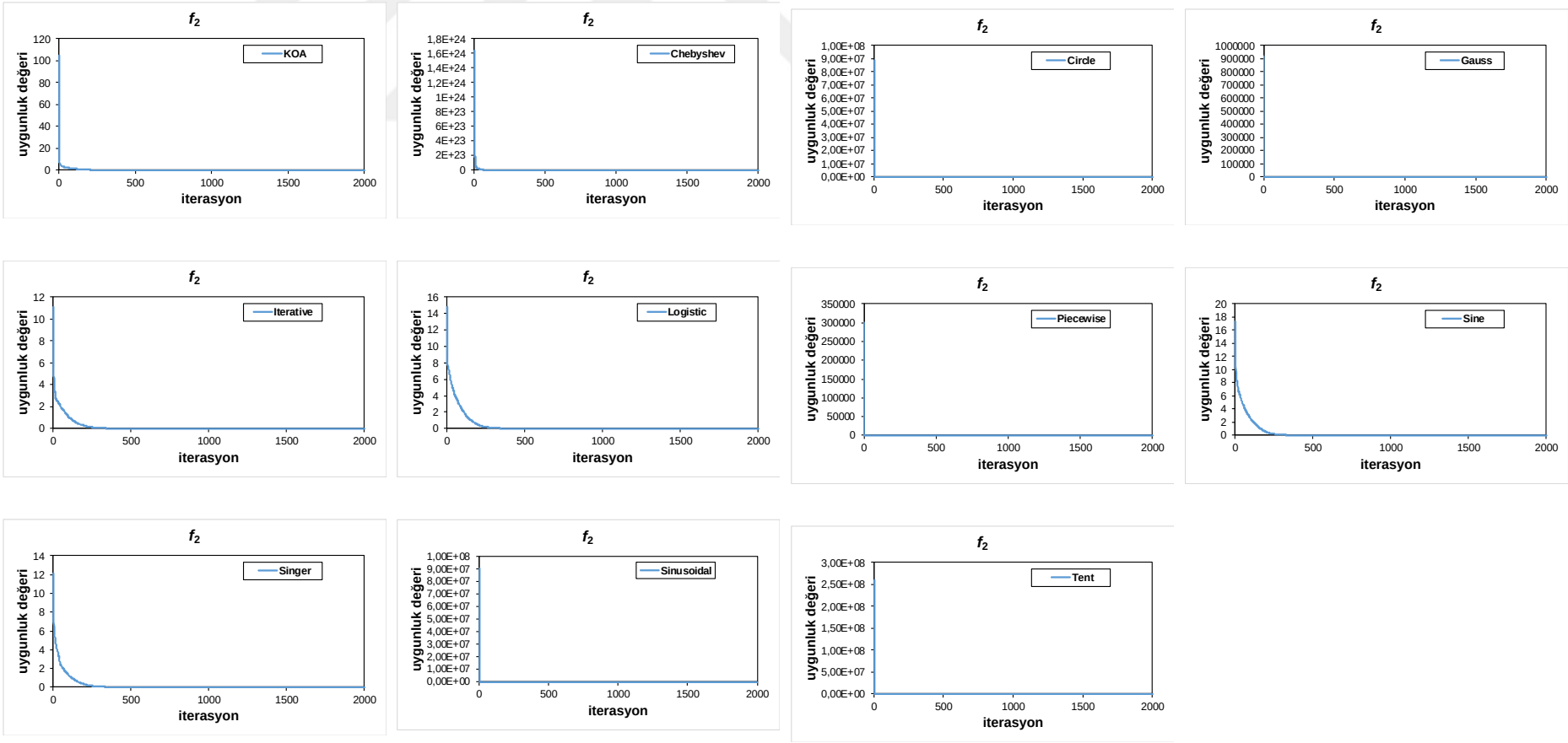
Zhang J-R, Zhang J, Lok T-M, Lyu MR, 2007. A hybrid particle swarm optimization–back-propagation algorithm for feedforward neural network training. Applied mathematics and computation, 185, 2, 1026-37.

Zhang L, Suganthan PN, 2016. A survey of randomized algorithms for training neural networks. Information Sciences, 364, 146-55.

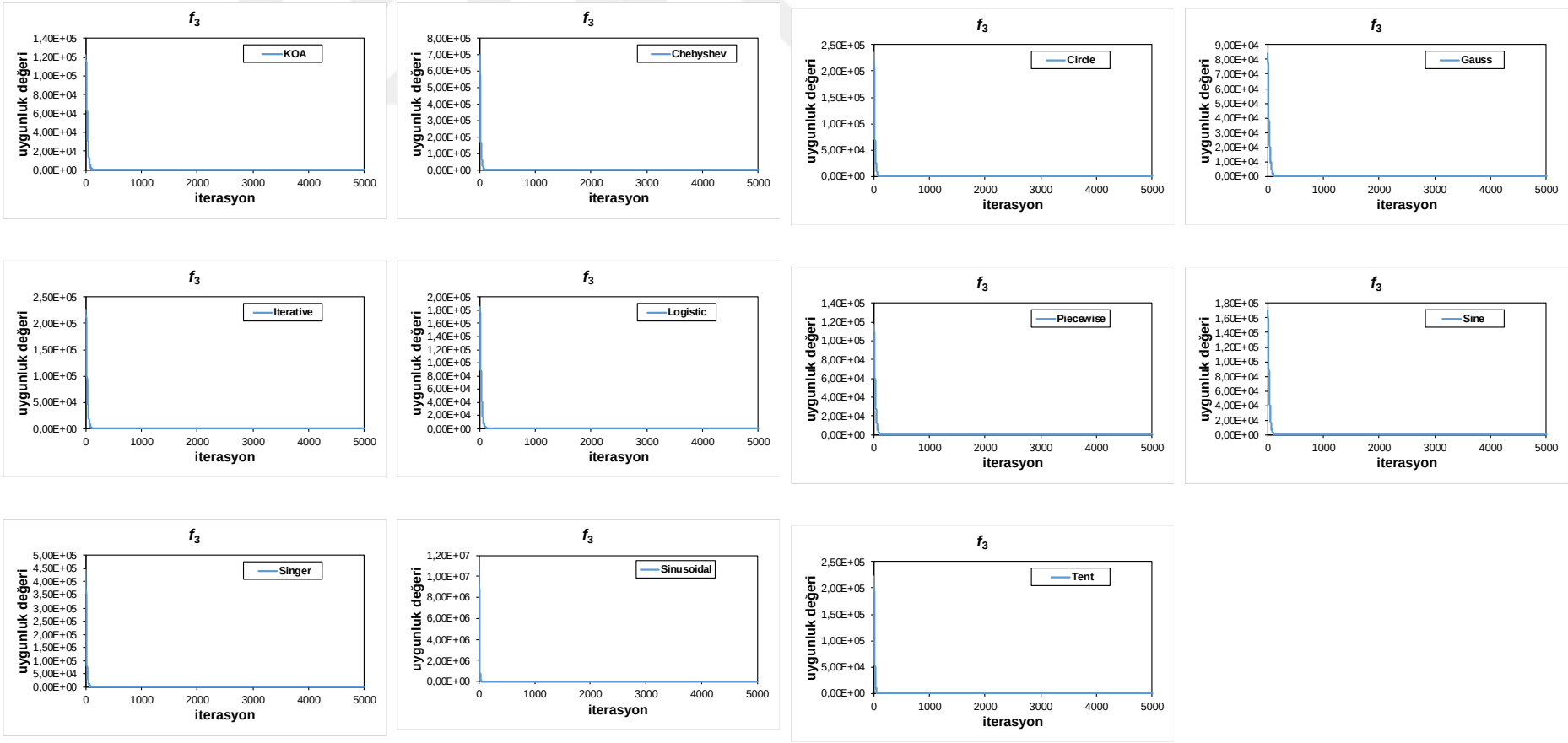
Zhao L, Qian F, 2011. Tuning the structure and parameters of a neural network using cooperative binary-real particle swarm optimization. Expert Systems with Applications, 38, 5, 4972-7.

EKLER

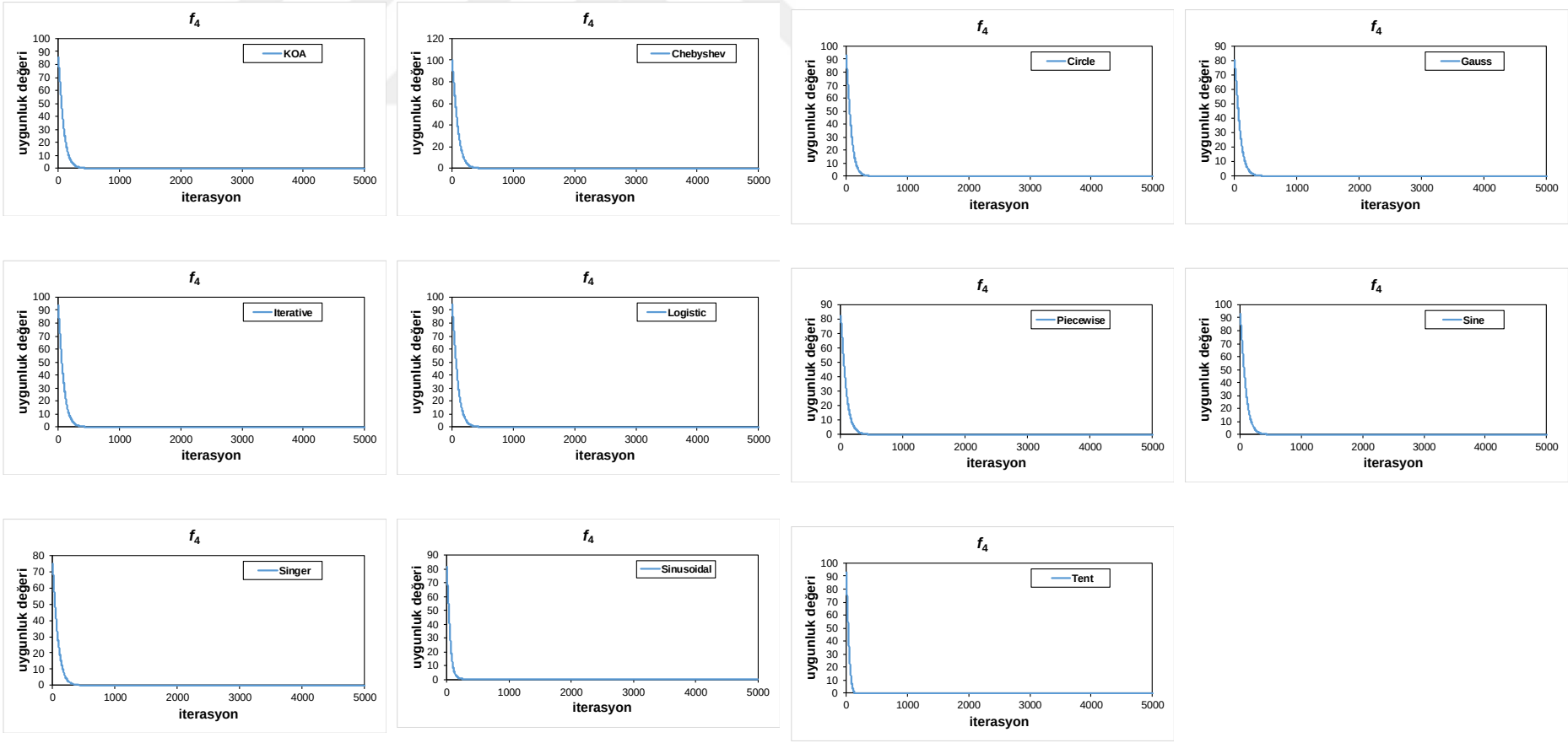
EK-1 Test fonksiyonlarının yakınsama grafikleri.



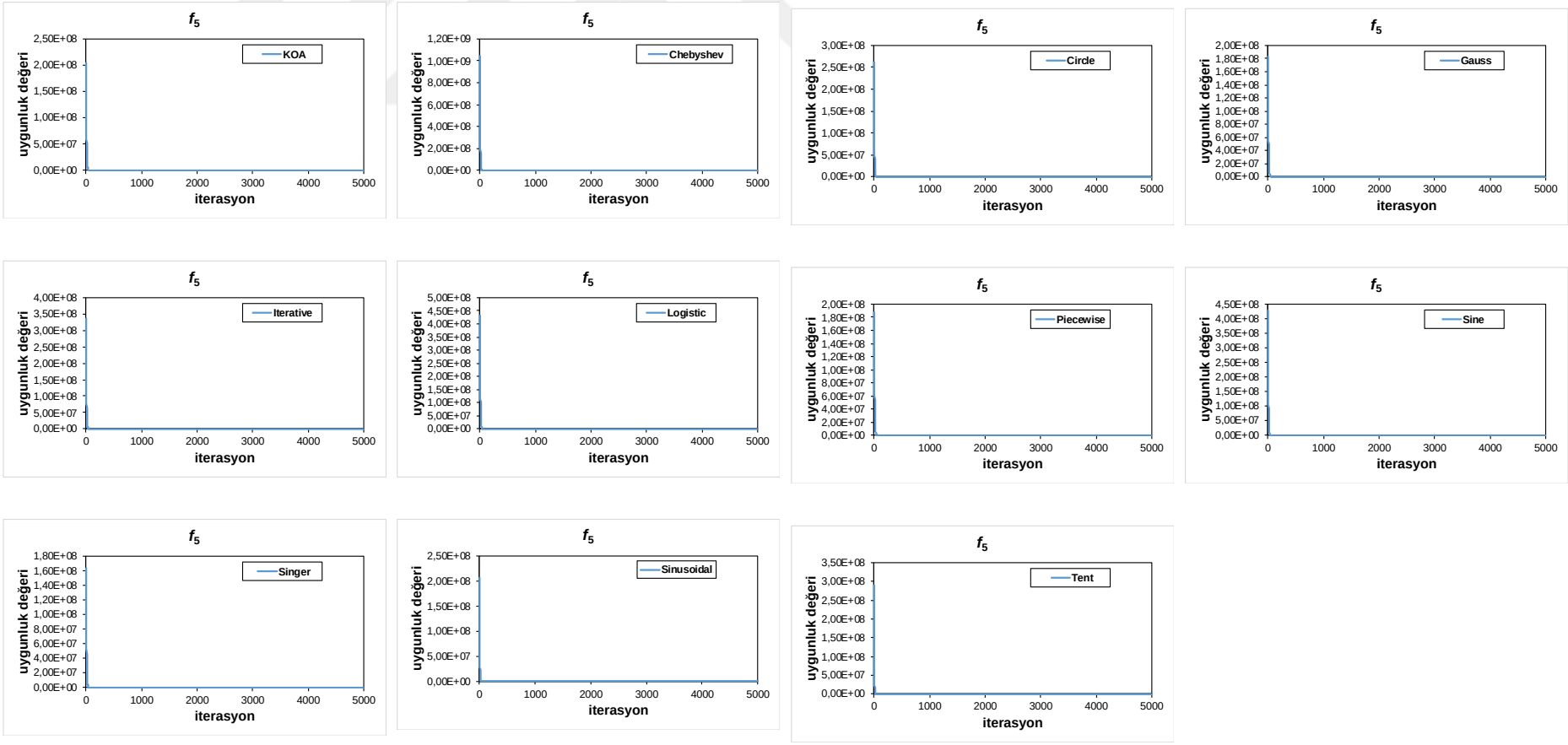
Şekil EK-1.2. f2 için yakınsama grafiği



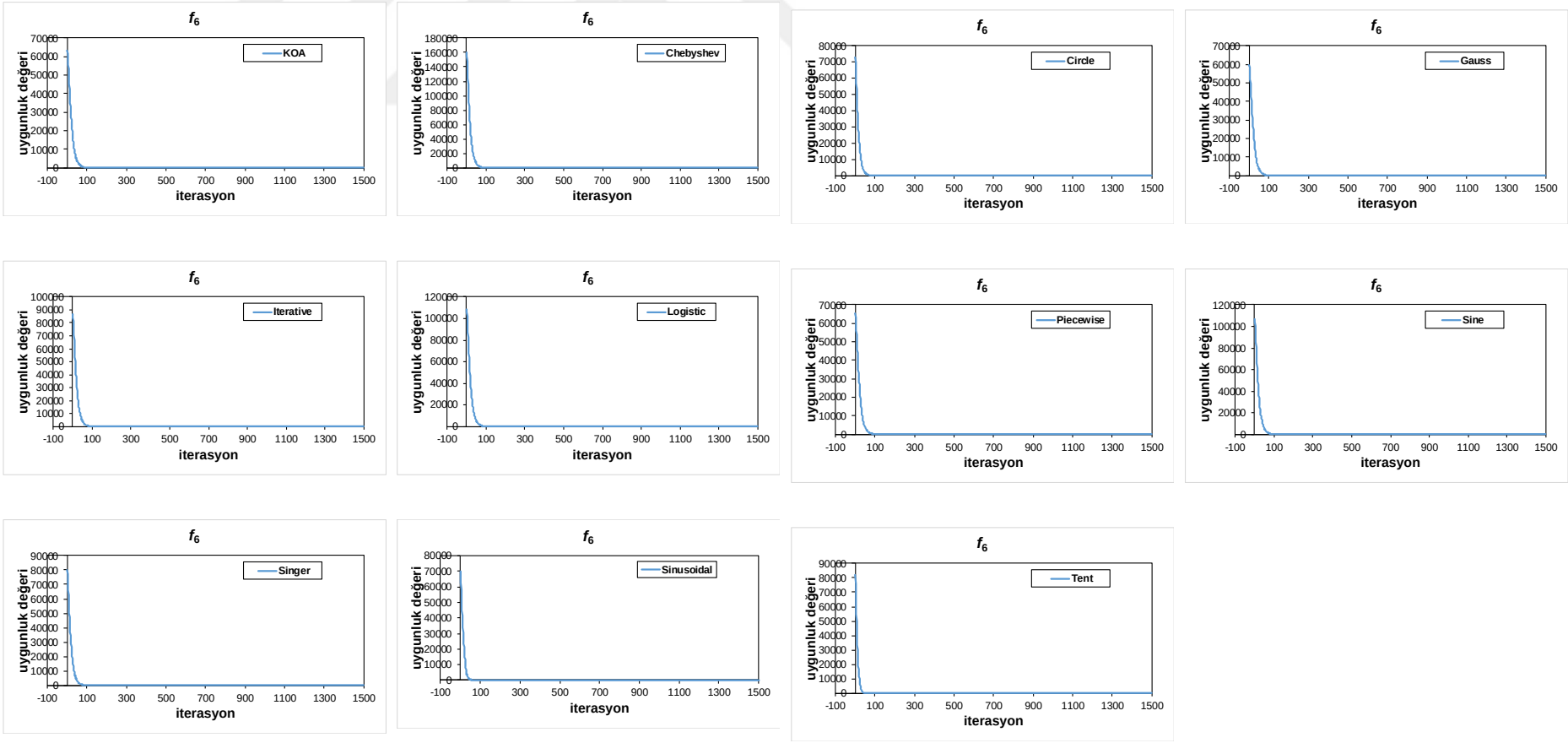
Şekil EK-1.3. f3 için yakınsama grafiği



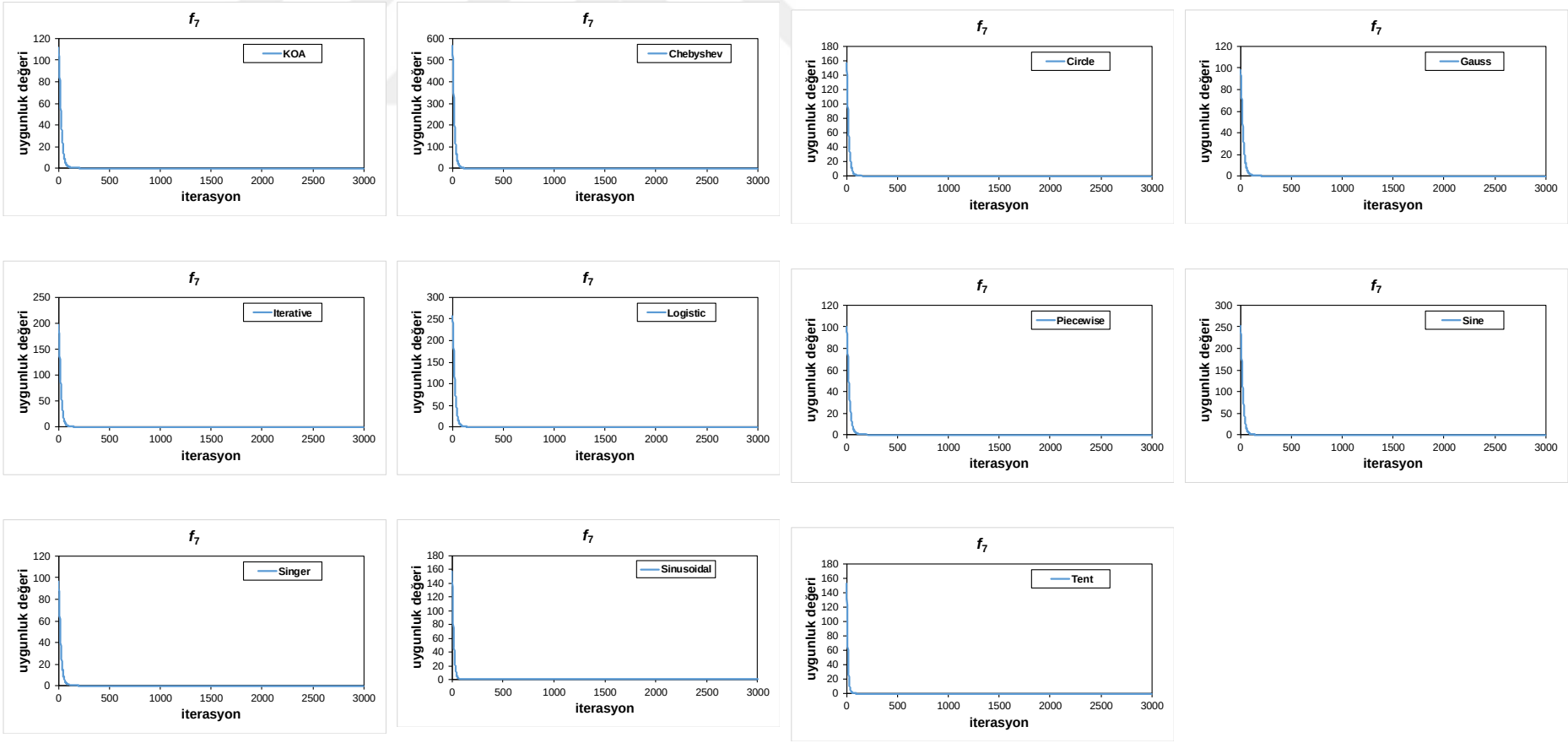
Şekil EK-1.4. f4 için yakınsama grafiği



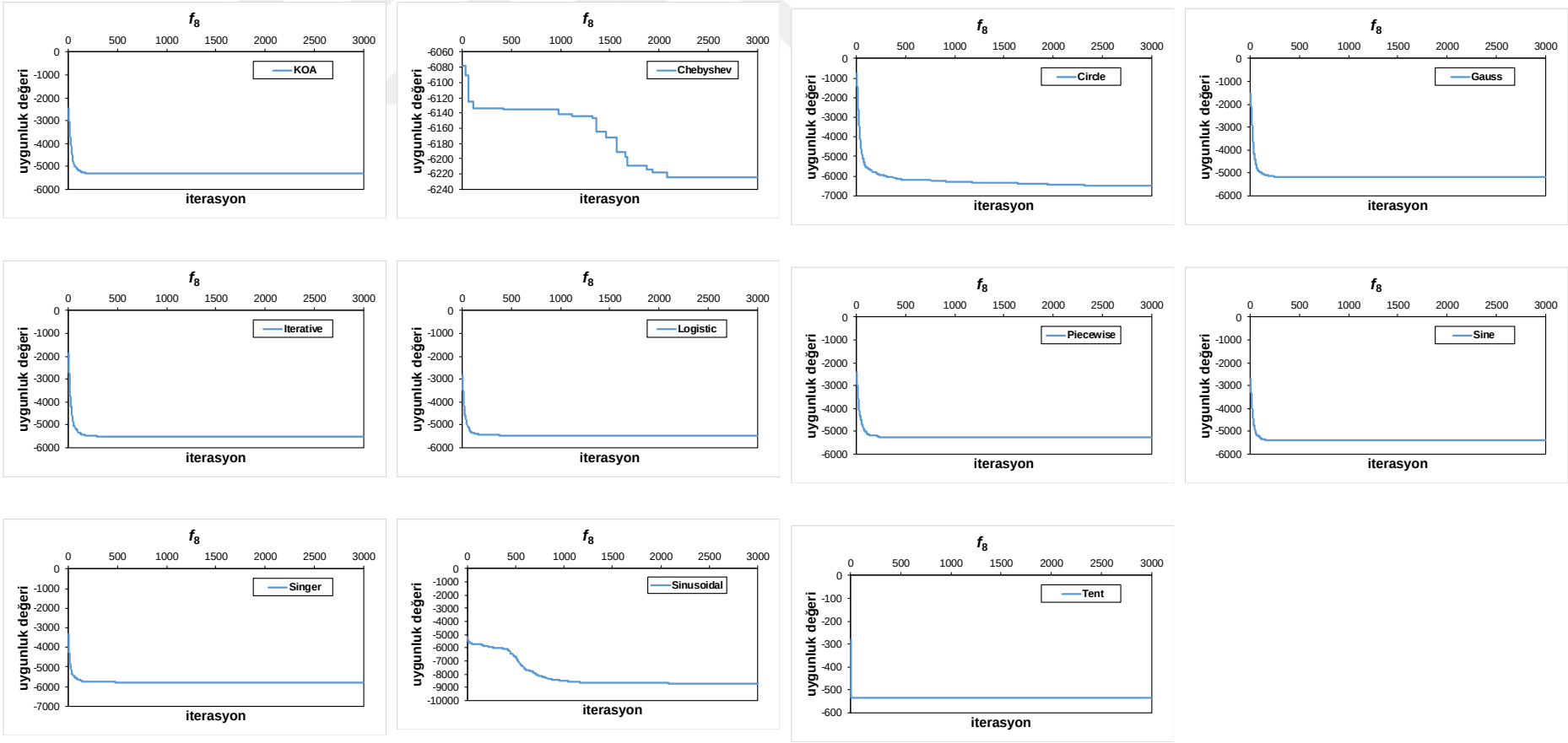
Şekil EK-1.5. f5 için yakınsama grafiği



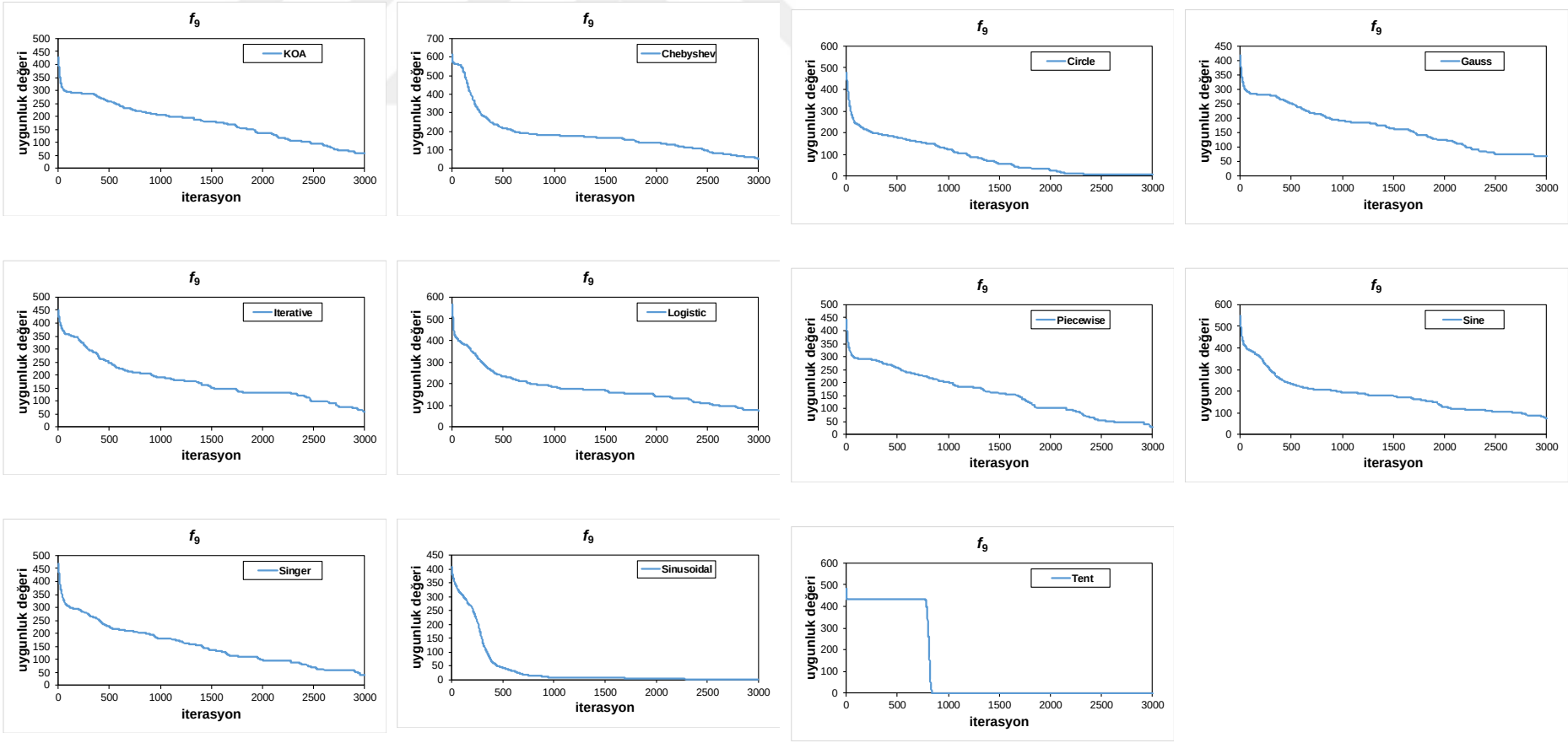
Şekil EK-1.6. f6 için yakınsama grafiği



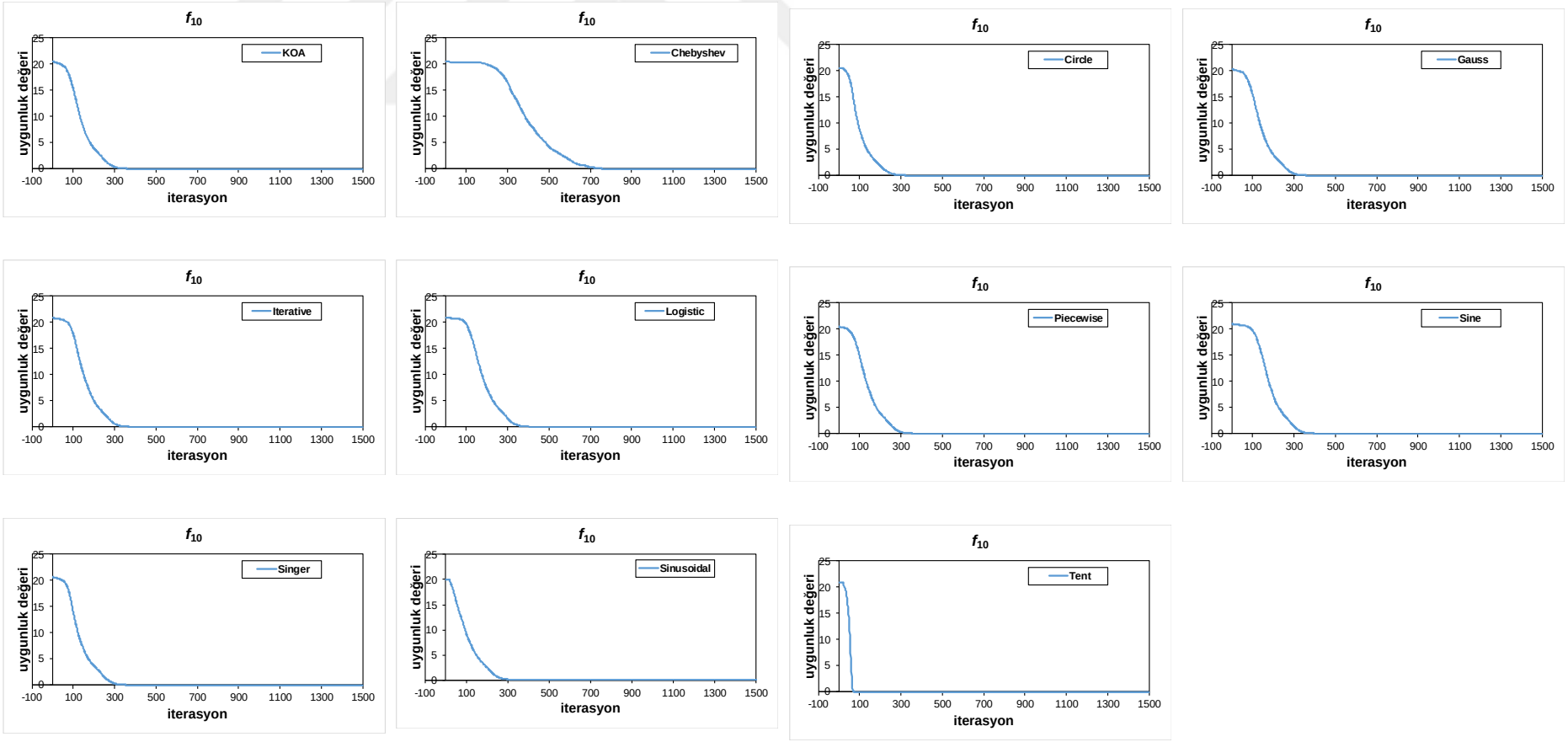
Şekil EK-1.7. f7 için yakınsama grafiği



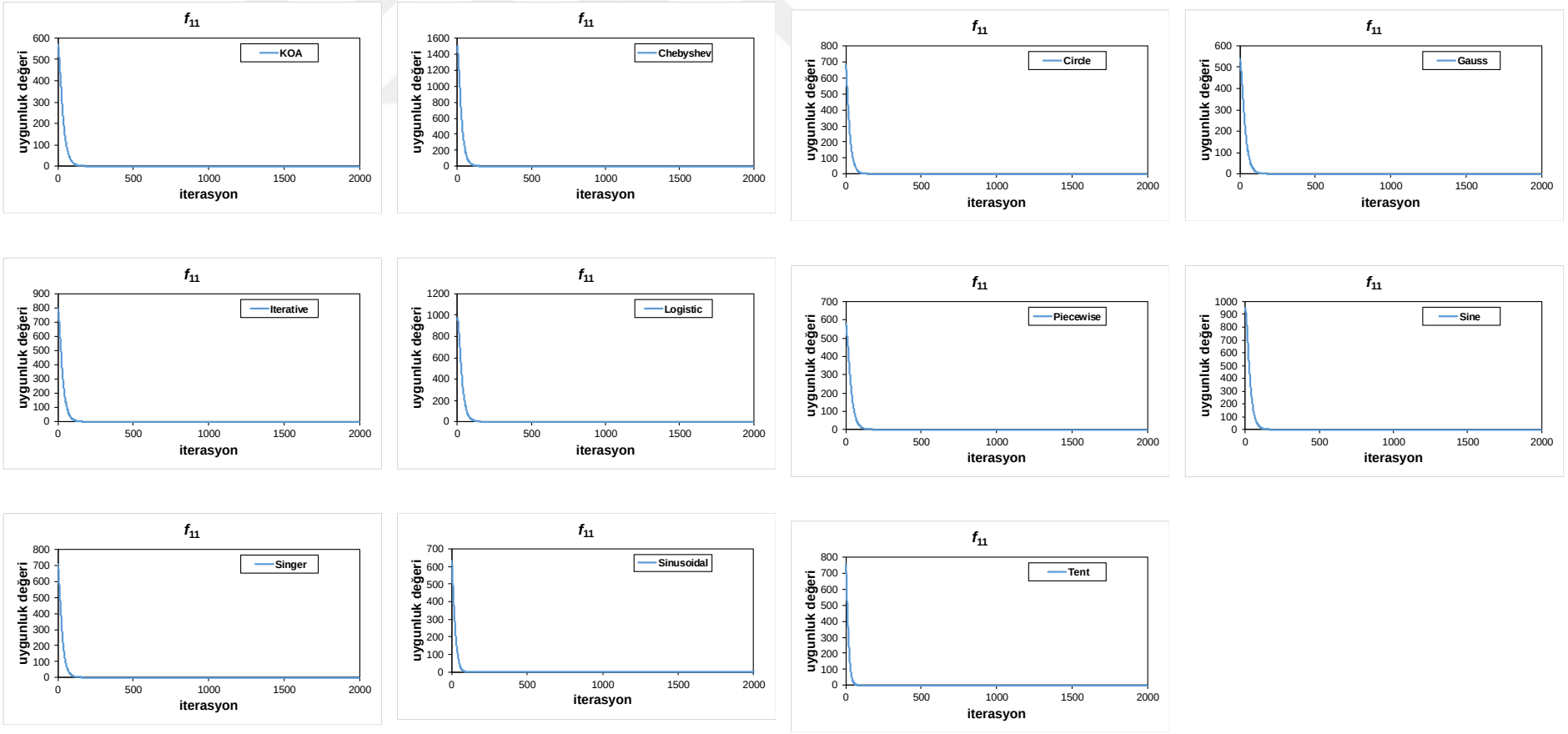
Şekil EK-1.8. f8 için yakınsama grafiği



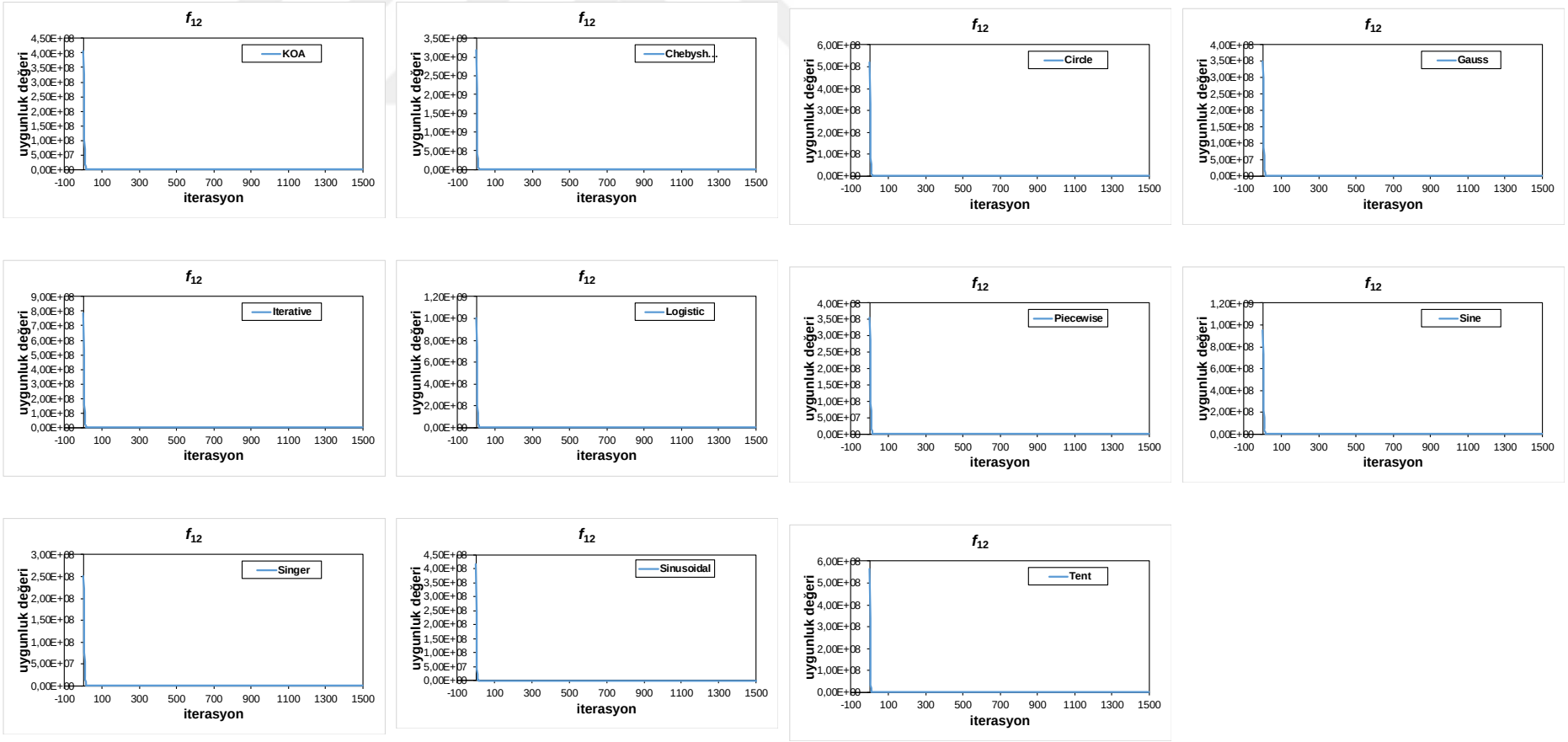
Şekil EK-1.9. f9 için yakınsama grafiği



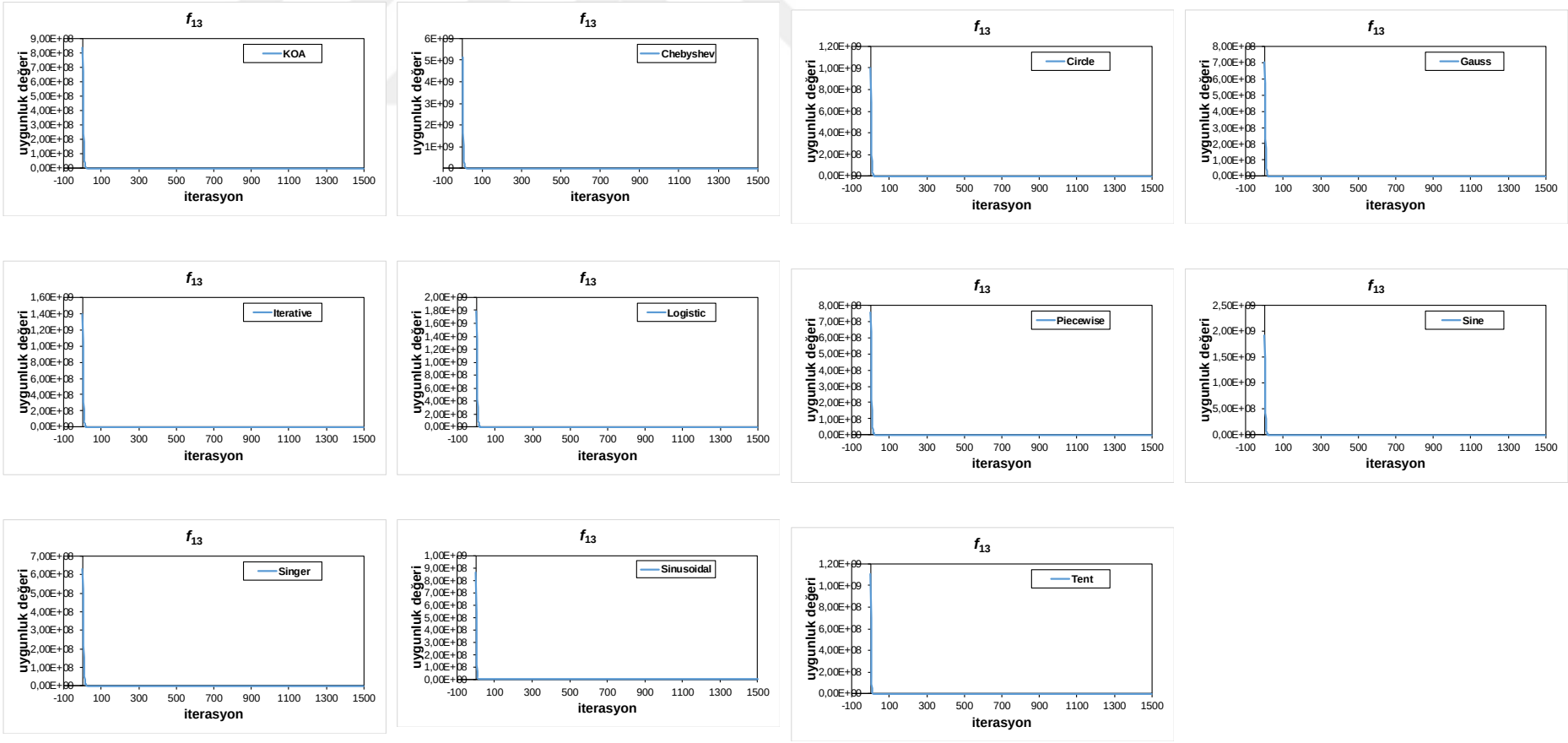
Şekil EK-1.10. f10 için yakınsama grafiği



Şekil EK-1.11. f11 için yakınsama grafiği



Şekil EK-1.12. f12 için yakınsama grafiği



Şekil EK-1.13. f13 için yakınsama grafiği

ÖZGEÇMİŞ