Fırat Üniversitesi Müh. Bil. Dergisi
34(1), 85-104, 2022

*Araştırma Makalesi*
https://doi.org/10.35234/fumbd.969335

# Küresel Optimizasyon için Gauss Kaotik Haritası ile Kartal Optimizasyonu

**Salih Berkan AYDEMİR[1*]**
[1] Bilgisayar Mühendisliği, Mühendislik-Mimarlık Fakültesi, Amasya Üniversitesi, Amasya, Türkiye
[*1] salih.aydemir@amasya.edu.tr

**(Geliş/Received: 09/07/2021; Kabul/Accepted: 05/10/2021)**

**Öz:** Bu çalışmada, Gauss kaotik haritası, kartal (Aquila) optimizasyon algoritmasındaki rastgele değişkenlerin yerine kullanılmaktadır. Kaotik haritaların tekrar edilememezlik özelliği ile küresel optimum noktaya yakınsama durumu incelenir. Gauss kaotik haritası, çözüm uzayının farklı noktalarını ele alıp, algoritmanın yerel optimum noktada takılmasını önler. Önerilen kaotik kartal optimizasyonu 13 kalite test fonksiyonu üzerinde test edilir. 13 test fonksiyonu içerisinde, 12 test fonksiyonunda yeni Gauss tabanlı kaotik kartal optimizasyonunun klasik kartal optimizasyonuna göre daha iyi yakınsama gösterdiği görülür. Ek olarak önerilen kaotik tabanlı kartal optimizasyonu ile üç test fonksiyonunda, küresel minimum noktaya yakınsar. Önerilen algoritma ve klasik algoritmanın yakınsama eğrileri, grafikler halinde özetlenir. Ayrıca önerilen kaotik tabanlı yöntem yapay arı kolonisi algoritması, diferansiyel evrim algoritması ve parçacık sürü optimizasyon algoritması ile karşılaştırılır. Deney sonuçları ise Gauss kaotik haritası tabanlı kartal algoritmasının diğer yöntemlerden daha erken iterasyonlarda yakınsadığı görülür.

**Anahtar kelimeler:** Meta-sezgisel Algoritmalar, Kalite Test Fonksiyonları, Optimizasyon, Kaotik Haritalar, Kaos Teori.

## Aquila Optimization with Gaussian Chaotic Map for Global Optimization

**Abstract:** In this study, the Gaussian chaotic map is replaced with random variables in the eagle (Aquila) optimization algorithm. The non-repeatability of chaotic maps and the convergence to the global optimum point are examined. The Gaussian chaotic map handles different points of the solution space and prevents the algorithm from getting stuck at the local optimum. The proposed chaotic eagle optimization is tested on 13 quality test functions. Among the 13 test functions, it is seen that the new Gaussian-based chaotic eagle optimization in 12 test functions converges better than the classical eagle optimization. In addition, with the proposed chaotic-based eagle optimization, it converges to the global minimum in the three test functions. The proposed algorithm and the convergence curves of the classical algorithm are summarized in graphs. In addition, the proposed chaotic-based method is compared with artificial bee colony algorithm, differential evolution algorithm and particle swarm optimization algorithm. The experimental results show that the eagle algorithm based on the Gaussian chaotic map converges in earlier iterations than the other methods.

**Key words:** Meta-heuristic Algorithms, Benchmark Functions, Optimization, Chaotic Maps, Chaos Theory.

## 1. Giriş

Son yıllarda, doğadan esinlenerek önerilen birçok optimizasyon algoritması bulunmaktadır. Bu tip algoritmalara meta-sezgisel optimizasyon algoritmaları (MOA) adı verilir [1]. Canlıların, besine yaklaşma, besin arama ve en uygun besin için yer değiştirme stratejileri meta-sezgisel algoritmaların temelini oluşturur. Amaç, bir çözüm uzayı içerisinde en ideal çözüm kümesine ulaşabilmektir.

MOA'lar, türev tabanlı optimizasyon algoritmalarının aksine, türev bilgisi gerekmeksizin yakınsama yaparlar [1]. Metasezgisel algoritmaların en önemli özelliklerinde biri de yerel optimum noktaya takılmadan, küresel minimum noktaya yakınsama davranışı içinde olmalarıdır [2]. Diğer yandan MOA'lar kısıtlı ve kısıtsız olmak üzere, günlük hayattaki birçok gerçek dünya problemine de uygulanabilirler. Kısıtlı olan problemlerde, ilgili kısıtlar göz önüne alınarak amaç fonksiyonunun optimum değerine yakınsaması test edilmektedir [3]. Kısıtsız olan MOA problemlerinde ise amaç sadece amaç fonksiyonunun optimum değerine yakınsamasını test etmektir [4]. Kimya mühendisliğinde proses tasarımı ve sentez problemleri, karışık tamsayılı doğrusal olmayan kısıtlı optimizasyon problemi olarak tanımlanabilir [5]. Kaynaklı kiriş tasarımı yapılabilmektedir. Kaynaklı kiriş tasarım probleminin temel amacı, kaynaklı kirişi minimum maliyetle tasarlamaktır. Bu problem beş kısıtlama içerir ve kaynaklı bir kiriş geliştirmek için dört değişken kullanılır [6]. Ayrıca MOA'lar medical görüntü işleme ve tıp alalında da sıkça kullanılmaktadır [7-9].

* Sorumlu yazar: salih.aydemir@amasya.edu.tr. Yazarların ORCID Numarası: [1] 0000-0003-0069-3479

MOA, iki temel safhadan oluşurlar. Bunlar sömürü ve keşif safhalarıdır [10]. Keşif ve sömürü safhası, çözüm uzayı içerisinde rastgele değerler ele alınarak belirlenir. Dolayısıyla bütün çözüm uzayının keşfedilmesi sağlanır. Keşif safhasında en kaliteli besin keşfedilip, sömürü safhasında ise keşfedilen besinin sömürülmesi söz konusudur. Sömürü ve keşif safhaları arasındaki denge, küresel optimuma ulaşmak için çok önemlidir [10]. Eğer keşif aşaması ön planda olursa, en uygun besin keşfedilmiş ancak sömürülememiş olur. Diğer yandan sömürü aşaması ön planda olursa, küresel optimum keşfedilmediği için yerel optimum sömürülebilir. Bu iki safhanın sonucunda küresel optimum noktaya ulaşılması hedeflenmektedir. Bir optimizasyon algoritmasında en büyük problemlerden biri de algoritmanın yerel optimum noktaya takılıp kalmasıdır. Böylece algoritma küresel optimum noktaya yakınsayamayacak ve ideal çözüm vektörü de yerel optimum değer için hesaplanacaktır. MOA'da sömürü ve keşif safhalarında rastgele parametrelerin kullanılması problemin yerel minimum noktada takılmasını çözebilmektedir. Bu kapsamda MOA, dört ana başlık altında incelenebilir: Sürü tabanlı, fizik ve matematiksel tabanlı, insan temelli ve evrimsel algoritmalar. Sürü tabanlı algoritmaların temel özellikleri kendi kendilerine örgütlenmeleri ve iş bölümü yapmalarıdır. Bir sistemin bileşenlerini dönüştürme yeteneği olarak herhangi bir dış yardım olmaksızın uygun bir forma dönüştürürler [11]. Parçacık Sürü Optimizasyonu (PSO) [12], Karga Arama Algoritması (KAA) [13], Yusufçuk Böceği Algoritması (YBA) [14], Yapay Arı Kolonisi algoritması (YAK)[15], Guguk Kuşu Algoritması (GKA)[16], Balina Optimizasyon Algoritması (BOA) [17] ve Gri Kurt Optimizasyon (GKO) [18], Harris Şahinleri Algoritması (HŞA) [19], Kartal (Aquila) Optimizasyon Algoritması (KOA) [20] sürü tabanlı bazı optimizasyon algoritmalarının örnekleridir. Fizik ve matematiksel problemlere dayalı optimizasyon algoritmaları ise matematiksel denklemler ve fiziksel bazı kurallar hesaba katılarak oluşturulmuştur. Çoklu Evren Algoritması (ÇEA) [21], Yüklü Sistem Araması (YSA) [22], Yerçekimsel Arama Algoritması (YAO) [23] fiziksel tabanlı optimizasyon algoritmalarına örnek olarak gösterilebilir. Diğer yandan, bazı matematiksel denklemler ve bu denklemlerin modellenmesi ile oluşturulan optimizasyon yöntemlerine, Sin-Cos Algoritması (SCA) [24] ve Aritmetik Optimizasyon [25] algoritmaları örnek olarak gösterilebilir. Doğada biyolojik evrimin süreci boyunca çeşitli, evrimsel optimizasyon algoritmaları önerilmiştir. Genetik Algoritmalar (GA) [26], Evrim Stratejisi (ES) [27] ve Diferansiyel Evrim (DE) [28] bu algoritmalar içerisinde yer almaktadır. İnsan temelli optimizasyon algoritmaları, insan davranışı ve işbirliğinden esinlenerek önerilen yöntemlerdir. Emperyalist Rekabetçi Algoritma (ERA) [29], Öğrenmeye Dayalı Öğretim algoritması (ÖDÖA) [30] ve politik optimizasyonu (PO) [31] gibi algoritmalar, insan temelli optimizasyon algoritmalarına örnektir.

Günümüze kadar bir çok optimizasyon algoritması önerilmiş ve hala daha önerilmeye de devam etmektedir. Bunun sebebi bir her optimizasyon problemi için küresel optimum noktayı garanti eden bir yöntemin olmamasıdır. Bu durum ise No Free-Lunch (NFL) teoriye göre ispatlanmıştır [32]. NFL'ye göre önerilen bazı yöntemler daha hızlı çalışabilir veya belli kalite test fonksiyonlarında daha iyi yakınsama gösterebilirler. Ancak tek bir optimizasyon yöntemi hiçbir zaman bütün problemleri çözememektedir. Bu nedenle erken iterasyonda optimum noktaya yakınsama, yerel optimuma takılmama, MOA'da rastgeleliğin ele alınması gibi özellikler literatürde önerilen optimizasyon algoritmalarını birbirinden ayırmaktadır.

Bu çalışmada KOA'da rastgele parametreler Gauss/Mouse kaotik harita ile düzenlenir. Kaotik haritaların tekrar edilememezlik ve ergodik yapıları [33] sayesinde önerilen Gauss kartal optimizasyon algoritması (GKOA), KOA'dan daha iyi yakınsamaya sahip olduğu kalite test fonksiyonlarıyla gösterilmiştir. GKOA'nın, KOA'ya göre üstün olan tarafları aşağıda listelenmiştir.

- Gauss/Mouse kaotik haritası, KOA'nın yapısında yer alan rastgele sayılar ile değiştirildiğinden çözüm uzayındaki farklı noktalara atlanabilir.
- Gauss/Mouse kaotik haritasının tekrar edilememezlik özelliği ile geniş bir çözüm uzayı sunar.
- GKOA, küresel optimuma yakınsamada, bir çok kalite test fonksiyonuna göre KOA'dan daha iyi yakınsama kabiliyetine sahiptir.

## 2. İlgili Çalışmalar

Günümüze kadar bir çok optimizasyon algoritması kaotik haritalar ile birlikte kullanılmıştır. Özellikle kaotik haritaların rastgele değerlerin yerine kullanılması, optimizasyon algoritmalarının geliştirilmesinde sıkça kullanılmaktadır. Bu bölümde, literatürde yer alan, kaotik haritaların diğer optimizasyon yöntemleriyle kullandığı çalışmalar bahsedilir. Kaur ve Arora'nın yaptığı çalışmada, BOA kaotik haritalar ile birleştirilmiş ve 20 farklı kalite test fonksiyonu üzerinde test edilmiştir [34]. Bütün rastgele değerler farklı kombinasyonlarda ele alındığında, 10 farklı senaryoya göre rastgele değerler, kaotik haritalar ile değiştirilmiştir. Wilcoxon's rank-sum test sonuçlarına göre de BOA ve kaotik BOA karşılaştırılmıştır. Önerilen kaotik BOA'nın, BOA'ya göre daha erken iterasyonda yakınsadığı görülmüştür. Diğer bir çalışmada altın bölgesi arama algoritması (ABA) [35] kaotik haritalar ile birlikte kullanılmıştır [36]. Önerilen kaotik ABA algoritması benzer şekilde 20 farklı kalite test

fonksiyonu test edilmiş ve literatürdeki diğer bazı algoritmalar ile karşılaştırılmıştır. Kohli ve Arora, sınırlandırılmış optimizasyon problemleri için kaotik GKO önermişlerdir [37]. Kaotik GKO'de 10 farklı kaotik harita ve 13 kalite test fonksiyonu kullanılmıştır. Kaotik GKO'nun klasik GKO'ya göre daha hızlı yakınsadığı görülmüştür. Diğer bir çalışmada, SCA 5 farklı kaotik harita ile 10 kalite test fonksiyonu üzerinde test edilmiştir [38]. Wilcoxon's rank-sum test sonuçlarına göre SCA ve kaotik SCA'nın arasındaki farklılık incelenmiştir [42]. Kuşların sürü hareketlerinde esinlenerek önerilen kuş sürüsü algoritmasında rastgele değerlerin kontrolü için kaotik haritalar kullanılmıştır. Ayrıca bütün rastgele değerler kaotik haritalar ile yer değiştirilmiş ve çalışmada 8 farklı kombinasyon oluşturulmuştur. Önerilen kaotik yöntem mühendislik problemleri üzerinde test edilmiştir [39].

Literatürdeki optimizasyon algoritmaları ile önerilen bütün kaotik yapılar, klasik optimizasyon yöntemlerine göre geniş bir çözüm uzayının taranmasını sağlar ve bu sayede yerel optimuma takılmadan, küresel optimuma yakınsama ihtimalini arttırır. Dolayısıyla önerilen kaotik tabanlı optimizasyon algoritmaları, klasik yöntemlerine göre daha iyi yakınsama kabiliyetine sahiptir. Yukarıda bahsedilen bütün çalışmalar göz önüne alındığında, kaotik haritaların optimizasyon yöntemleri üzerinde uygulanabilen etkili bir strateji olduğu söylenebilir. Bu çalışmada ise yakın zamanda önerilen KOA, Gauss kaotik haritası ile birlikte kullanılmıştır.

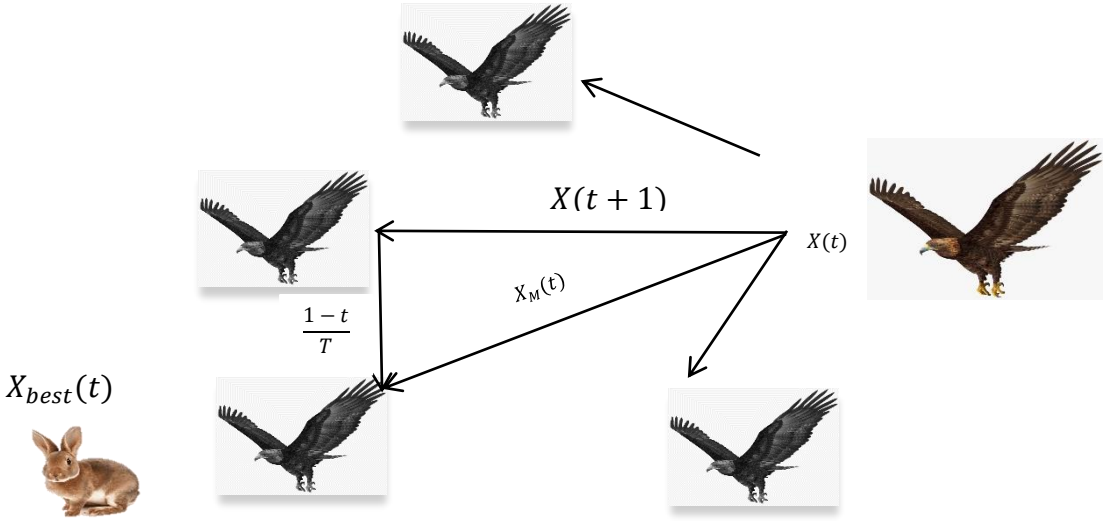
## 3. Kartal Optimizasyon Algoritması

Kuzey Yarımküre'de Aquila, en popüler yırtıcı kuşlardan biridir. Aquila tipik olarak koyu kahverengidir ve boynunun arkasında daha açık altın-kahverengi tüyleri bulunur. Bu grubun genç Aquila'sı çoğunlukla kuyrukta beyaz renge sahiptir ve genellikle kanatlarında küçük beyaz lekeler vardır. Aquila, sağlam ayakları ve büyük, keskin pençeleriyle birleşen hızını ve çevikliğini, başta tavşanlar olmak üzere dağ sıçanları sincaplar gibi çeşitli avları yakalamak için kullanır [40]. Aquila, alanı 200 km2'ye kadar çıkabilen bölgeleri korur. Yuvalarını dağlarda veya diğer yüksek kesimli yerlerde oluştururlar. Esas olarak, Aquila'nın dört farklı avlanma yöntemi kullanıldığı kabul edilmektedir ve çoğu Aquila'nın avlanma yöntemleri arasında akıllıca ve hızlı bir şekilde değişiklik yapma yeteneğine sahiptir. Aquila'nın avlanma yöntemleri şu şekildedir[20]. İlk yöntem olan dikey eğimli yüksek uçma, Aquila'nın yerden yüksek bir seviyeye yükseldiği uçuş sırasında kuşları avlamak için kullanılır. Avını keşfettikten sonra Aquila, yaklaştıkça hızla yükselen uzun, düşük açılı bir süzülmeye girer [41]. İkinci yöntem olan yerden düşük bir seviyede yükselerek kısa süzülme saldırısı ile kontur uçuşu, Aquila tarafından en çok kullanılan yöntem olarak kabul edilmektedir [42]. Üçüncü yöntem, yavaş bir alçalma saldırısı ile alçak bir uçuştur. Bu durumda Aquila yere iner ve bir sonraki saldırıda aşamalı olarak avına saldırır [43]. Dördüncü yöntem, Aquila'nın karada yürüdüğü ve avını çekmeye çalıştığı avı yakalamaktır [44].

Keşif ve sömürü aşamaları bir metasezgisel algoritmada iki ana unsurdur. Keşif safhasında en kaliteli besin keşfedilmektedir. Sömürü safhasında ise keşfedilen besinin en uygun şekilde sömürülmesi söz konusudur. Diğer bir deyişle, keşif aşaması, çözüm uzayını olabildiğince geniş, rastgele ve küresel olarak arama sürecini ifade eder. Sömürü aşaması, algoritmanın keşif aşamasıyla elde edilen alanda daha doğru arama yapma yetkinliğini ifade eder ve kesinlik arttıkça rastgeleliği azalır.

Aquila'nın avını yakalama stratejileri göz önüne alınarak zekice ve hızlı bir şekilde avına yaklaşması, optimizasyonun ana konusunu oluşturmaktadır. KOA, $t \leq \left(\frac{2}{3}\right) * T$ koşuluna göre keşif aşamasından sömürü aşamasına geçmektedir. Eğer $t \leq \left(\frac{2}{3}\right) * T$ ise keşif aşaması aktiftir. Aksi durumda keşif aşaması aktiftir. Burada $t$ o anki iterasyon sayısıdır. $T$ ise maximum iterasyon sayısıdır. KOA'nın Aquila'dan esinlenerek matematiksel modellere dönüştürülmesi aşağıda adım adım anlatılmıştır.

**Adım 1 Genişletilmiş keşif yeteneği ($X_1$):** Aquila, hem av bölgesini tanır hem de dikey eğim ile yüksek uçarak en iyi avlanma alanını seçer. KAO, avın bulunduğu arama alanının alanını belirlemek için yüksekten uçarak geniş çapta keşifler yapar. Aquila'nın davranışı matematiksel olarak aşağıdaki gibi modellenebilir.




**Şekil 1.** Aquila'nın süzülerek avına yaklaşma davranışı

Şekil-1'de verilen Aquila'nın davranışı matematiksel olarak aşağıdaki gibi modellenebilir.

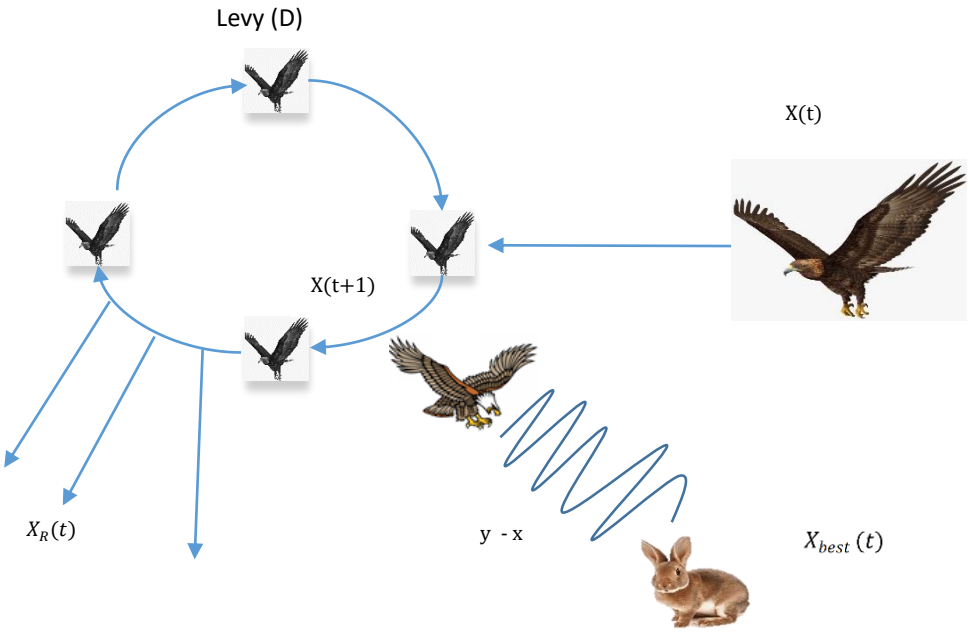
$$X_1(t+1) = X_{best}(t) \times \left(\frac{1-t}{T}\right) + (X_M(t) - X_{best}(t) \times rand) \quad (1)$$

Burada $X_1(t+1)$, $t$'nin bir sonraki iterasyonunun çözümüdür. $X_1$ ilk aday çözümdür. $X_{best}(t)$, $t$'ninci iterasyona kadar elde edilen en iyi çözümdür. $\left(\frac{1-t}{T}\right)$, iterasyon sayısı aracılığıyla geniş bir keşif kontrolü sağlar. $X_M(t)$, $t$'ninci iterasyonda mevcut çözümlerin ortalama değerini ifade eder. rand ise 0 ve 1 arasında rastgele bir sayı üretir.

$$X_M(t) = \frac{1}{N}\sum_{i=1}^{N} X_i(t), \text{ Her } j=1,2,\ldots,Boy. \quad (2)$$

Burada Boy, problemin boyutunu, $N$ ise aday çözümlerin sayısı temsil eder.

**Adım 2 Daraltılmış keşif yeteneği ($X_2$):** Yüksek uçuş ile av bölgesi bulunduğunda, Aquila hedef avın üstünde çember çizer, hedefe inmeye hazırlanır ve ardından avına saldırır. Bu metot kısa kayma saldırısı ile kontur uçuşu olarak adlandırılır.




**Şekil 2.** Aquila'nın kısa kayma saldırısı ile kontur uçuşu

Şekil-2'ün matematiksel modeli denklem-3 ile gösterilmiştir.

$$X_2(t+1) = X_{best}(t) \times Levy(D) + X_R(t) + (y - x) \times rand \tag{3}$$

Burada $X_2(t+1)$, t'nin bir sonraki iterasyonunun çözümü ve $X_2$, ikinci arama metodu olarak ifade edilir. $D$ uzayın boyutu ve $Levy(D)$ levy flight dağılım fonksiyonu olarak adlandırılır ve denklem-4 ile gösterilir. $X_R(t)$, $t$'ninci iterasyonda [1 $N$] aralığındaki rastgele değerdir.

$$Levy(D) = s \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}} \tag{4}$$

Burada s değeri 0.01 olarak sabit alınır. $u$ ve $v$, 0 ve 1 arasındaki rastgele sayılardır. $\sigma$ ise denklem 5 ile hesaplanır.

$$\left( \frac{\Gamma(1+\beta) \times sine(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{\frac{\beta-1}{2}}} \right) \tag{5}$$

$\beta$ sabit değeri genellikle 1.5 olarak kabul edilir. x ve y aramadaki spiral şekli temsil eder ve aşağıdaki gibi hesaplanır.

$$y = r \times \cos(\theta) \text{ ve } x = r \times \sin(\theta), \tag{6}$$
$$r = r_1 + \mathrm{U} \times D_1 \text{ ve } \theta = -\omega \times D_1 + \theta_1,\ \theta_1 = \frac{3\pi}{2}$$

$r_1$, arama döngüsü sayısı 1 ile 20 arasındaki sabit bir sayıdır. U, 0.00565 sayısına sabitlenmiştir. $D_1$, 1 ile arama uzayının boyutu arasındaki bir tam sayıdır ve $\omega$ sayısı 0.005'e sabitlenmiştir.

**Adım 3 Genişletilmiş sömürü yeteneği ($X_3$):** Av alanı doğru bir şekilde belirlendiğinde ve Aquila iniş ve saldırı için hazır olduğunda, Aquila av reaksiyonunu keşfi için bir ön saldırı ile dikey olarak alçalır. Bu davranışa yavaş alçalma saldırısı ile alçak uçuş denir. Aquila'nın bu davranışı denklem 7 ile modellenir.

(7)

$$X_3(t+1) = (X_{best}(t) - X_M(t)) \times \alpha - rand + ((\text{UB-LB}) \times \text{rand} + \text{LB}) \times \delta$$

Burada $X_3(t+1)$, *t*'nin bir sonraki iterasyonunun çözümü ve $X_3$, üçüncü arama metodu olarak ifade edilir. rand değeri 0 ve 1 arasında rastgele bir değer üretir. $\alpha$ ve $\delta$, sömürü ayarlama parametreleridir ve 0.1 değerine sabitlenmiştir. LB verilen problemin alt sınırı, UB ise verilen problemin üst sınırıdır.

**Adım 4 Daraltılmış sömürü yeteneği ($X_4$):** Bu metotta, Aquila avına yaklaştığında, onun stokastik hareketlerine göre karada saldırır. Bu yönteme yürüyüş ve av yakalama adı verilir. Bu davranışın matematiksel formu denklem 8'de verilmiştir.

$$X_4(t+1) = QF \times X_{best}(t) - (G_1 \times X(t) \times rand) - G_2 \times Levy(D) + rand \times G_1 \quad (8)$$

Burada $X_4(t+1)$, *t*'nin bir sonraki iterasyonunun çözümü ve $X_4$, dördüncü arama metodu olarak ifade edilir. QF, arama stratejilerini dengelemek için kullanılan bir kalite fonksiyonunu ifade eder. QF denklem 9 ile ifade edilir. G1, kaçış sırasında avı izlemek için kullanılan Aquila çeşitli hareketlerini gösterir. $G_2$, 2'den 0'a azalan değerleri alır ve denklem 10 ile gösterilir. $G_2$, ilk konumdan son konuma kadar kaçış sırasında avı takip etmek için kullanılan Aquila'nın uçuş eğimini gösterir ve denklem 11 ile gösterilir.

$$QF(t) = t^{\frac{2\times rand()-1}{(1-T)^2}} \quad (9)$$

$$G_1 = 2 \times rand() - 1 \quad (10)$$

$$G_2 = 2 \times (1 - \frac{t}{T}) \quad (11)$$

QF(t), *t*'ninci iterasyondaki kalite fonksiyonu olmak üzere, *t* ve *T* sırasıyla mevcut iterasyon ve en son iterasyon sayısıdır. KOA, Aquilaların avına yaklaşım tarzları ve avlanma stratejilerinden esinlenerek önerilen bir algoritma olduğundan sömürü ve keşif safhaları arasındaki denge, genişletilmiş ve daraltılmış olarak 2'ye ayrılmaktadır.

## 4. Kaotik Kartal Optimizasyonu

Bu çalışmada, KOA içerisindeki rastgele değerler Gauss kaotik haritası ile değiştirilir. Kaotik haritaların kendine has tekrar edilememezlik özelliği bu yapıları optimizasyon algoritmalarında etkili bir şekilde kullanılabilir hale getirmiştir. Matematiksel olarak kaos, doğrusal olmayan ve dinamik sistemlere dayanan basit bir deterministik rastgeleliktir [45]. Kaotik haritalar genellikle dinamik ve doğrusal olmayan sistemlerin çalışmasında oluşur [46]. Optimizasyon algoritmalarında, rastgele değerlere bağlı olarak genellikle arama alanı içinde bir arama stratejisi uygulanır. Kaotik haritaların avantajı, rastgele değerler yerine kaotik değerlerin kullanılmasıdır. Kaotik haritalara dayalı optimizasyon algoritmalarında, kaosun tekrarlanmaması ve ergodikliği nedeniyle stokastik aramalardan daha genel arama stratejisi sergilerler [33]. Bu çalışmada Chebyshev, Circle, Gauss/Mouse, Iterative, Logistic, Piecewise, Sine, Singer, Sinusoidal ve Tent haritasi olmak üzere 10 farklı kaotik harita kullanılmıştır. Ancak çalışmanın sadeliği ve kısalığı açısından sadece en iyi sonuç veren kaotik harita kullanılmıştır (Gauss/Mouse).

Kaosun doğası rastgele ve öngörülemez görünse de, aynı zamanda kendi içinde bir uyum içerir. Ayrıca, rastgelelik kaotik haritalarla ayarlandığı için yerel optimum tuzağından kaçınabilir. Bu çalışmada kullanılan kaotik harita denklem 12 ile ifade edilmektedir.

$$x_{k+1} = \begin{cases} 1, & x_k = 0 \\ \dfrac{1}{mod(x_k, 1)}, & d.d. \end{cases} \quad (12)$$

**ALGORİTMA 1: KAOTİK AQUİLA OPTİMİZASYONU**

**Başlangıç Aşaması:**
Popülasyonun başlangıç değerleri: X
KKO'nun başlangıç parametreleri: $\alpha, \delta$
Rastgele değerlere Gauss kaotik haritası atanır.
**while**(iterasyon < max_iterasyon) **do**
Amaç fonksiyonu hesapla
$X_{best}(t) =$ Amaç fonksiyonuna (Fitness function) göre elde edilen en iyi çözümü belirle
**for** (i=1,2,…,N)
Şu an ki çözümün ortalama değerini $X_M(t)$ güncellenir
$G_1$, $G_2$ ve Levy(D) gibi parametreleri güncellenir
**if** $t \leq \frac{2}{3}T$ **then**
Gauss kaotik harita vektörü güncellenir.
**if** $Gauss_{rand} \leq 0.5$ **then**
**Adım 1: Genişletilmiş keşif ($X_1$)**
Denklem 1 kullanılarak mevcut çözüm güncellenir.
**if** Fitness($X_1(t+1)$) < Fitness(X(t)) **then**
X(t) =($X_1(t+1)$)
**if** Fitness($X_1(t+1)$) < Fitness($X_{best}(t)$) **then**
$X_{best}(t) = X_1(t+1)$
**end if**
**end if**
**else**
**Adım 2: Daraltılmış keşif ($X_2$)**
Denklem 3 kullanılarak mevcut çözüm güncellenir.
**if** Fitness($X_2(t+1)$) < Fitness(X(t)) **then**
X(t) =($X_2(t+1)$)
**if** Fitness($X_2(t+1)$) < Fitness($X_{best}(t)$) **then**
$X_{best}(t) = X_2(t+1)$
**end if**
**end if**
**end if // Kaotik haritanın sonu**
**else // sömürü veya keşif seçimi**
**if** $Gauss_{rand} \leq 0.5$ **then**
**Adım 1: Genişletilmiş sömürü ($X_3$)**
Denklem 7 kullanılarak mevcut çözüm güncellenir.
**if** Fitness($X_3(t+1)$) < Fitness(X(t)) **then**
X(t) =($X_3(t+1)$)
**if** Fitness($X_3(t+1)$) < Fitness($X_{best}(t)$) **then**
$X_{best}(t) = X_3(t+1)$
**end if**
**end if**
**else**
**Adım 2: Daraltılmış sömürü ($X_4$)**
Denklem 8 kullanılarak mevcut çözüm güncellenir.
**if** Fitness($X_4(t+1)$) < Fitness(X(t)) **then**
X(t) =($X_4(t+1)$)
**if** Fitness($X_4(t+1)$) < Fitness($X_{best}(t)$) **then**
$X_{best}(t) = X_4(t+1)$
**end if**
**end if**
**end if**
**end if**
**end for**
**end while**
**return en iyi çözüm ($X_{best}$)**

Denklem 12'de görüldüğü gibi eğer mevcut konum 0 ise bir sonraki konum 1'e sabitlenir. Denklem 12, algoritma -1'de görüldüğü gibi, lokal ve global aramayı birbirinden ayırmak için kullanılmıştır. Diğer durumlarda ise mod işlemi uygulanır. Kaotik haritaların özellikleri dikkate alındığında bu çalışmada önerilen kaotik tabanlı

optimizasyona kaotik kartal optimizasyonu adı verilir (KKO). Kaotik haritaların ergodik yapıları sayesinde, çözüm uzayı daha hızlı taranır ve bu durum erken iterasyonlarda yakınsamayı sağlar. Diğer yandan çözüm uzayı genişler ve bu durum da yerel optimum noktaya takılmadan küresel optimum noktaya yakınsama ihtimalini arttırır. Önerilen KKO yöntemin algoritması, Algoritma-1'de verilmiştir.

## 5. Deney Sonuçları ve Analizler

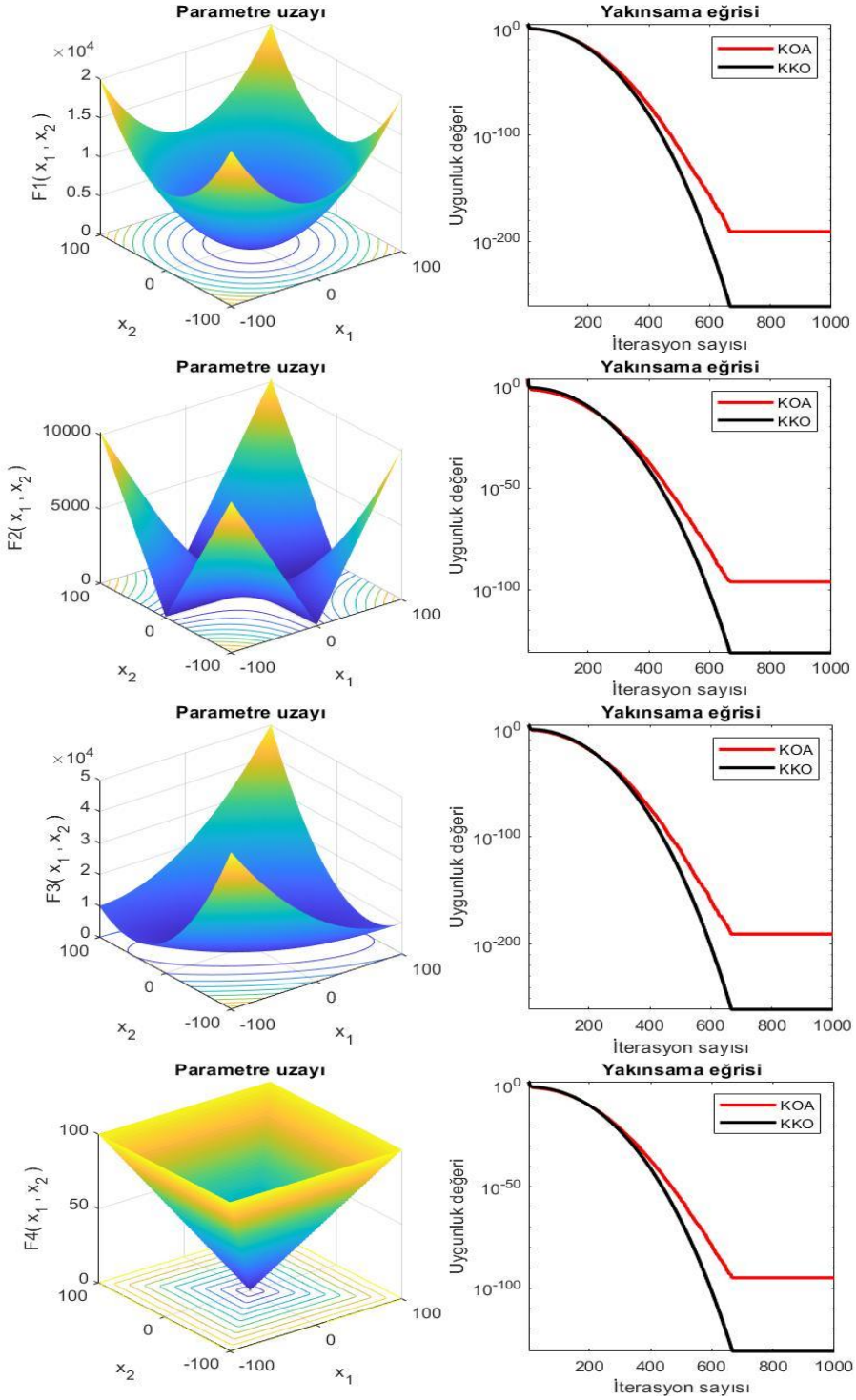
Bu çalışmada KKO algoritması, tek modlu ve çok modlu olmak üzere13 kalite test fonksiyonu için klasik KOA ile karşılaştırılmıştır. Algoritmalar 1000 iterasyonda, 50 farklı çalıştırmanın ortalamaları ile grafik haline getirilmiştir. Algoritmalar, R2020b matlab programı üzerinde ve i7-4700MQ CPU, 2.40GHz 8,00 GB RAM, 64 bit işletim sistemine sahip bilgisayar ile kodlanmıştır. Çalışmada kullanılan kalite test fonksiyonları Tablo-1'de verilmiştir.

**Tablo 1.** Kalite test fonksiyonları

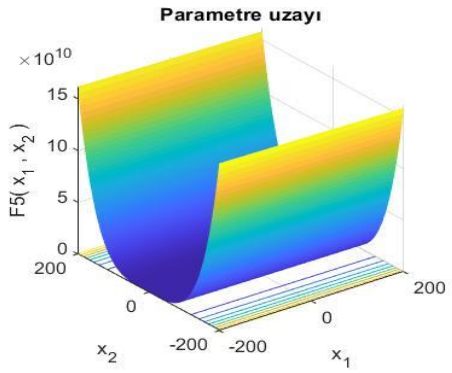
| Fonk.. | Matematiksel denklemi | Boyut | Aralık | $f_{min}$ |
|---|---|---|---|---|
| F1 | $f(x)=\sum_{i=1}^{n} x_i^2$ | 10 | [-100, 100] | 0 |
| F2 | $f(x)=\sum_{i=0}^{n}\lvert x_i\rvert+\prod_{i=0}^{n}\lvert x_i\rvert$ | 10 | [-10, 10] | 0 |
| F3 | $f(x)=\sum_{i=1}^{d}\left(\sum_{j=1}^{i} x_j\right)^2$ | 10 | [-100, 100] | 0 |
| F4 | $f(x)=\max_i\{\lvert x_i\rvert, 1\le i\le n\}$ | 10 | [-100, 100] | 0 |
| F5 | $f(x)=\sum_{i=1}^{n-1}[100(x_i^2-x_{i+1})^2+(1-x_i)^2]$ | 10 | [-30, 30] | 0 |
| F6 | $f(x)=\sum_{i=1}^{n}(x_i+0.5)^2$ | 10 | [-100, 100] | 0 |
| F7 | $f(x)=\sum_{i=0}^{n} ix_i^4+random[0,1)$ | 10 | [-1.28, 1.28] | 0 |
| F8 | $f(x)=\sum_{i=1}^{n} x_i^2-10\cos(2\pi x_i)+10$ | 10 | [-5.12, 5.12] | 0 |
| F9 | $f(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$ | 10 | [-32, 32] | 0 |
| F10 | $f(x)=1+\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right)$ | 10 | [-600, 600] | 0 |
| F11 | $f(x)=\frac{\pi}{n}10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10sin^2(\pi y_{i+1})+\sum_{i=1}^{n} u(x_i,10,100,4)]$ | | [-50, 50] | 0 |
| F12 | $f(x)=0.1\left(sin^2(3\pi x)+\sum_{i=1}^{n}(x_i-1)^2[1+sin^2(3\pi x_i+1)]+(x_n-1)^2(1+sin^2(2\pi x_n))\right)+\sum_{i=1}^{n} u(x_i,5,100,4)$ | 10 | [-50, 50] | 0 |
| F13 | $f(x)=\sum_{i=1}^{11}[a_i-\frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}]^2$ | 4 | [-5, 5] | 0.0003 |

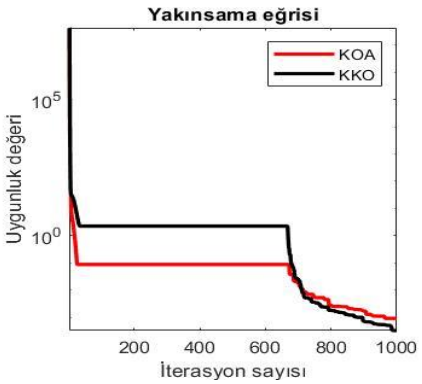
Tablo-1'de $y_i=1+\frac{x_i+1}{4}$ ve $u(x_i,a,k,m)=\begin{cases} K(x_i-a)^m & x_i>a \\ 0 & -a\le x_i\ge a \\ K(-x_i-a)^m & -a\le x_i \end{cases}$ olarak ifade edilir. Boyutlar ve çözüm uzayı aralıkları, karşılaştırmada adillik olması açısından klasik KOA yöntemiyle aynı seçilmiştir [20]. Bu çalışmada önerilen yöntemi test etmek için hem tek modlu hem de çok modlu kalite test fonksiyonları kullanılmıştır. Tek modlu test fonksiyonları tek yerel minimum noktaya sahipken, çok modlu test fonksiyonları birden fazla yerel min. noktaya sahiptir. F1-F7 fonksiyonları tek modlu, F8-F13 fonksiyonları ise çok modlu kalite test fonksiyonlarıdır. Bütün test fonksiyonları için Şekil-1,2 ve 3'de 2D parametre uzayları ve yakınsaklık analizleri verilmektedir. 2D parametre uzayları netlik açısından farklı aralıklarda alınmıştır. KOA ve KKO yöntemleri için, 1000 iterasyon sonucunda uygunluk değeri (fitness function value) açısından yakınsaklık analizleri yorumlanabilir. Öncelikle Şekil-1 göz önüne alındığında, KOA ve KKO optimizasyon algoritmalarının benzer yakınsama gösterdikleri görülebilir. Ancak F1, F2, F3 ve F4 kalite test fonksiyonları için KKO'nın daha düşük uygunluk değerlerine yakınsadığı görülebilir. Şekil-2'de, F5-F10 fonksiyonlarının yakınsaklık analizleri verilmiştir (Sıralamalar soldan sağa F5-6-7-8-9-10 şeklindedir). F5 ve F6 incelenirse her iki yönteminde etkili yakınsama kabiliyetine sahip olduğu söylenebilir. Ancak 700. iterasyondan sonra KKO'nın yerel min. değerine daha yakın değere ulaştığı görülebilir. Diğer yandan, F7 ve F9'da, KOA ve KKO benzer sonuçlara sahiptir. Ancak

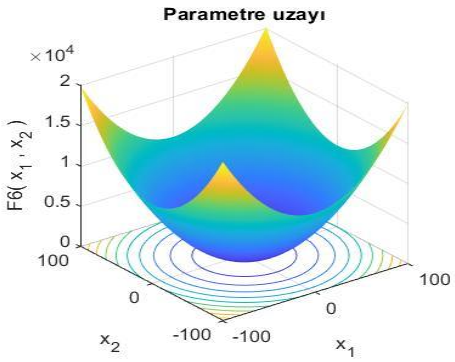
F8'de KKO'nın açık bir şekilde küresel min. değere daha yakın olduğu görülebilir. KKO, özellikle F10'da düşük iterasyon sayısında erken yakınsama özelliğine sahiptir.

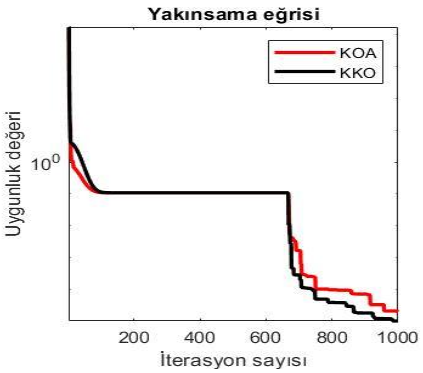


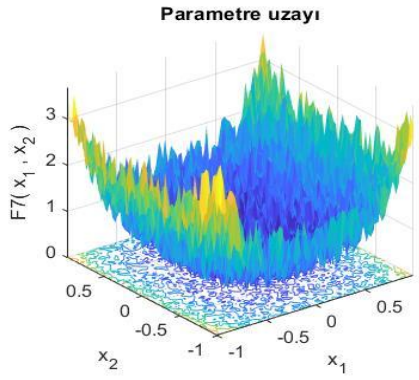

**Şekil 3.** F1-F4 kalite test fonksiyonlarının 2D parametre uzayı ve yakınsaklık analizleri

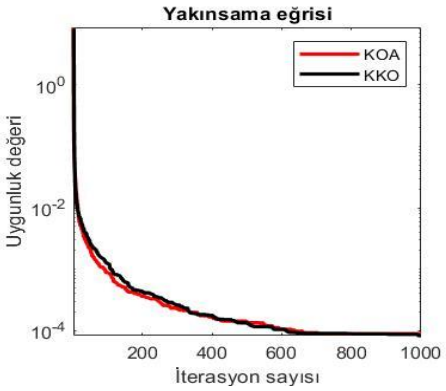

Parametre uzayı
×10^10
15
10
5
0
F5( x1 , x2 )
200
0
-200
x2
-200
0
200
x1

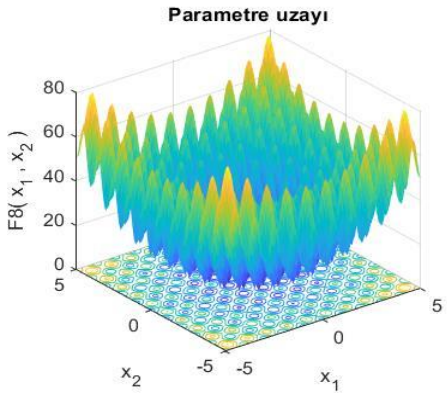

Yakınsama eğrisi
KOA
KKO
10^5
10^0
Uygunluk değeri
200
400
600
800
1000
İterasyon sayısı

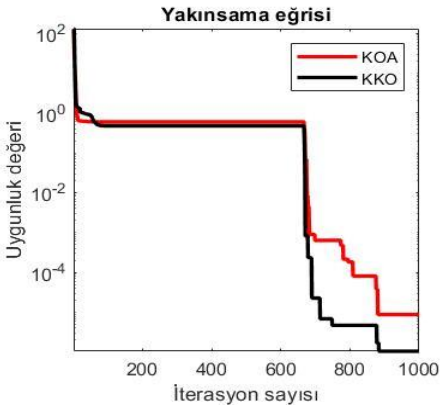

Parametre uzayı
×10^4
2
1.5
1
0.5
0
F6( x1 , x2 )
100
0
-100
x2
-100
0
100
x1

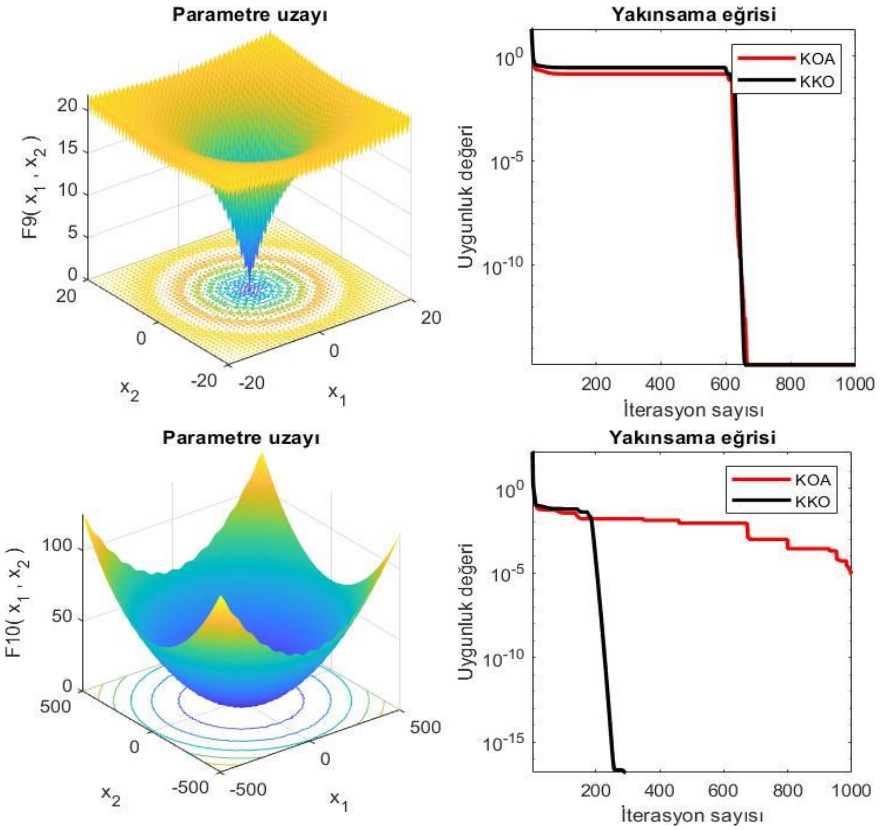

Yakınsama eğrisi
KOA
KKO
10^0
Uygunluk değeri
200
400
600
800
1000
İterasyon sayısı


Parametre uzayı
3
2
1
0
F7( x1 , x2 )
0.5
0
-0.5
x2
-1
-1
-0.5
0
0.5
x1

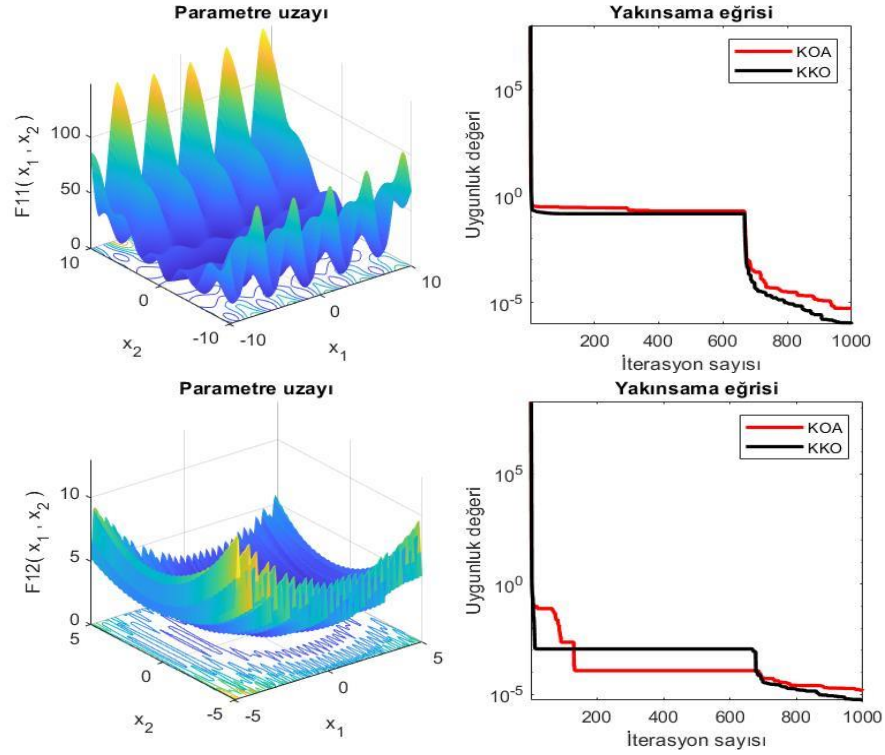

Yakınsama eğrisi
KOA
KKO
10^0
10^-2
10^-4
Uygunluk değeri
200
400
600
800
1000
İterasyon sayısı

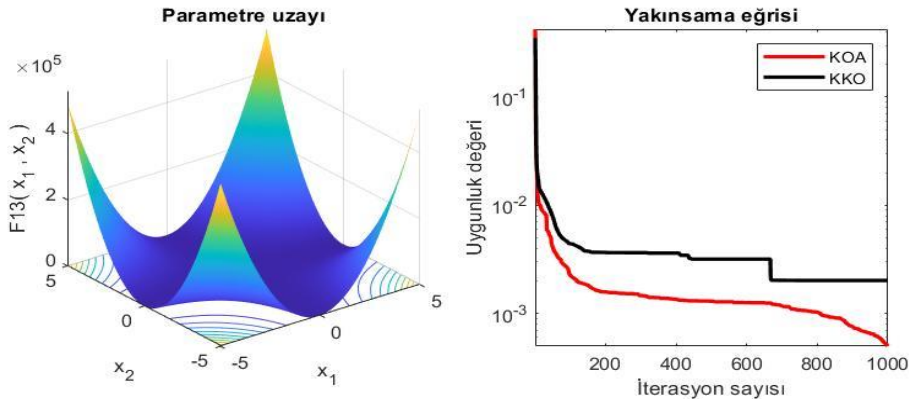

Parametre uzayı
80
60
40
20
0
F8( x1 , x2 )
5
0
-5
x2
-5
0
5
x1


Yakınsama eğrisi
KOA
KKO
10^2
10^0
10^-2
10^-4
Uygunluk değeri
200
400
600
800
1000
İterasyon sayısı




**Şekil 4.** F5-F10 kalite test fonksiyonlarının 2D parametre uzayı ve yakınsaklık analizleri






**Şekil 5.** F11- F13 kalite test fonksiyonunun 2D parametre uzayı ve yakınsaklık analizi

Son olarak, Şekil-3'de F11 ve F12 için KKO daha iyi sonuç verirken, F13'de klasik KOA yönteminin daha iyi sonuç verdiği görülebilir. Bütün kalite test fonksiyonları ele alınırsa, F9 ve F13 hariç, bütün test fonksiyonlarında önerilen KKO algoritması daha iyi sonuç vermektedir. Bir metasezgisel optimizasyon algoritmasında sömürü ve keşif arasındaki denge oldukça önemlidir. Bu çalışmada sömürü ve keşif arasındaki denge; rastgele sayılar yerine, Gauss/Mouse kaotik haritası ile kontrol edilmiştir. Kaotik yapıların tekrar edilmemezlik ve ergodik özelliği Şekil 5'de görselleştirilmiştir. Dolayısıyla Gauss/Mouse kaotik haritası ile arama uzayının farklı noktalarının taranması sağlanmıştır.


**Şekil 6.** Ergodik (b) ve ergodik olmayan (a) sistemlerin davranışları [47]

**Tablo 2.** KOA ve KKO için kalite test fonksiyonlarının sonuçları

| Opt. Alg. | KOA | | | KKO | | |
|---|---|---|---|---|---|---|
| Fonk | Ortalama | En iyi | Standart Sapma | Ortalama | En iyi | Standart Sapma |
| F1 | 2.65E-198 | 6.11E-294 | 0.00E+000 | **8.07E-262** | 0.00E+000 | 0.00E+000 |
| F2 | 6.15E-097 | 1.46E-147 | 4.23E-096 | **7.62E-132** | 3.40E-275 | 3.13E-131 |
| F3 | 1.35E-190 | 6.45E-293 | 0.00E+000 | **1.48E-261** | 0.00E+000 | 0.00E+000 |
| F4 | 1.63E-096 | 2.34E-150 | 1.14E-095 | **4.77E-132** | 4.89E-277 | 2.80E-131 |
| F5 | 1.41E-003 | 1.40E-006 | 3.36E-003 | **3.28E-004** | 2.45E-008 | 7.23E-004 |
| F6 | 4.78E-005 | 8.80E-008 | 1.08E-004 | **1.48E-005** | 7.82E-009 | 2.36E-005 |
| F7 | 8.58E-005 | 1.81E-006 | 7.05E-005 | **6.67E-005** | 2.95E-006 | 8.19E-005 |
| F8 | 1.45E-005 | 0.00E+000 | 9.85E-005 | **5.01E-007** | 0.00E+000 | 2.62E-006 |
| F9 | **9.59E-016** | 8.88E-016 | 4.97E-016 | 1.74E-015 | 8.88E-016 | 1.51E-015 |
| F10 | 6.38E-006 | 0.00E+000 | 4.47E-005 | **0.00E+000** | 0.00E+000 | 0.00E+000 |
| F11 | 4.03E-006 | 1.63E-008 | 700E-006 | **1.09E-006** | 5.99E-010 | 1.82E-006 |
| F12 | 8.81E-006 | 2.07E-007 | 1.04E-005 | **5.31E-006** | 2.06E-008 | 7.88E-006 |
| F13 | **4.64E-004** | 3.19E-004 | 8.84E-005 | 1.50E-003 | 4.61E-004 | 4.07E-004 |

Tablo-2'de, KOA ve KKO yöntemlerinin, 1000 iterasyonda, 50 kez bağımsız çalıştırma sonucunda elde edilen ortalama, en iyi ve standart sapma değerleri verilmektedir. Çok küçük sayılarda (< 1.00E-150) standart

sapma değerleri 0.00E+000 olarak yuvarlanmaktadır. Tablo-2'de F9 ve F13 fonksiyonları hariç diğer bütün durumlarda KKO algoritmasının daha iyi sonuç verdiği görülebilir. F9'da ise Şekil-2'den de yorumlanabildiği gibi KOA ve KKO birbirine çok yakın sonuçlar üretmektedir. Diğer yandan KOA'dan farklı olarak, KKO algoritması F1 ve F3'ün en iyi değerlerinde küresel minimum noktaya yakınsadığı görülmektedir.

### 5.1. İstatistiksel test

MOA'ların performansını değerlendirmek için istatistiksel testler yapılmalıdır [48-50]. Özellikle algoritmaları ortalama, standart sapma ve en iyi sonuçlara göre kıyaslama yeterli değildir [51]. Önerilen yeni bir algoritmanın klasik algoritmaya kıyasla önemli bir gelişme gösterdiğini kanıtlamak için istatistiksel bir test gereklidir. Algoritmaların sonuçlarının istatistiksel olarak anlamlı bir şekilde birbirinden farklı olup olmadığına karar vermek için birçok istatistiksel test kullanılmaktadır. Bu çalışmada ise optimizasyon algoritmaları üzerine sıkça kullanılan Wilcoxon işaretli-sıralı test kullanılmıştır [52].

**Tablo 3.** KOA ve KKO için Wilcoxon işaretli-sıralı test sonuçları

| Algoritmalar | KKO ve KOA karşılaştırılması | |
|---|---|---|
| Kalite test fonksiyonları | p değeri | Önem |
| F1 | 1.1300E-07 | + |
| F2 | 4.4985E-04 | + |
| F3 | 1.1155E-06 | + |
| F4 | 6.9928E-08 | + |
| F5 | 2.1900E-02 | + |
| F6 | 2.0800E-02 | + |
| F7 | 1.9700E-02 | + |
| F8 | 4.6526E-04 | + |
| F9 | 8.7500E-01 | - |
| F10 | 1.6219E-04 | + |
| F11 | 1.5200E-02 | + |
| F12 | 2.9000E-03 | + |
| F13 | 1.0872E-09 | + |
| Toplam | 12/1 | |

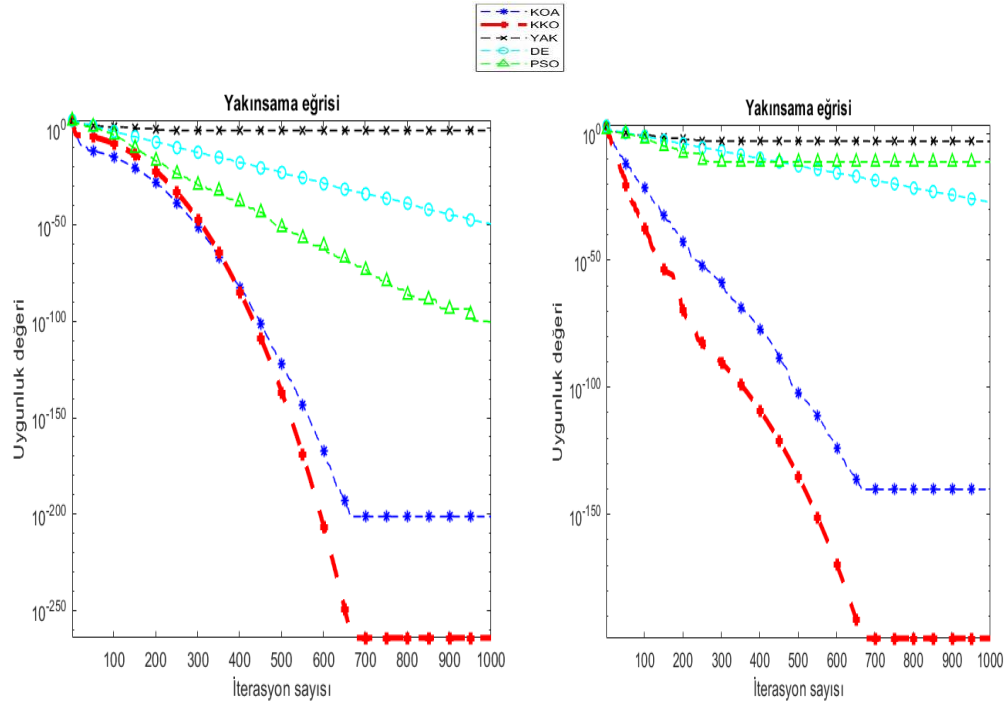
Wilcoxon işaretli-sıralı test %5 anlamlılık düzeyinde 50 kez bağımsız çalıştırmanın sonucunda, son iterasyondaki değerler dikkate alınarak hesaplanmıştır. Bu istatistiksel testte p değeri KOA ve KKO algoritmaları arasındaki önem değerini temsil eder. Eğer $p<0.05$ ise iki yöntem %95 ihtimal ve %5 anlamlılık düzeyinde birbirinden farklıdır denilebilir. Aksi taktirde, $p>0.05$ ise karşılaştırılan iki yöntem arasında, istatistiksel olarak önemli bir ölçüde fark olmadığı söylenebilir. Tablo-3'de KOA ve KKO'nın her bir kıyaslama test fonksiyonu için Wilcoxon işaretli-sıralı teste göre karşılaştırmaları sonucunda elde edilen p değerleri ve istatistiksel olarak anlamlı olup olmadıkları verilmektedir. "+/- " ifadesinde "+" istatistiksel olarak anlamlı olduğunu, "-" ise karşılaştırılan iki yöntem arasında önemli bir farklılık olmadığını, dolayısıyla istatistiksel olarak anlamlı olmadığını göstermektedir. Tablo-3 incelendiğinde, sadece F9 kıyaslama fonksiyonunda iki yöntem arasındaki belirgin bir fark olmadığı görülebilir. Benzer şekilde, Şekil-7'deki F9'un yakınsama eğrisi göz önüne alınarak benzer yorumlama yapılabilir. Diğer bütün kıyaslama test fonksiyonları için KOA ve KKO algoritmaları arasında 0.05 anlamlılık düzeyinde önemli bir fark bulunmaktadır. O halde önerilen KKO algoritmasının klasik KOA'dan farklı bir algoritma olduğu söylenebilir.

### 5.2. Önerilen KKO algoritmasının literatürdeki yöntemlerle karşılaştırılması

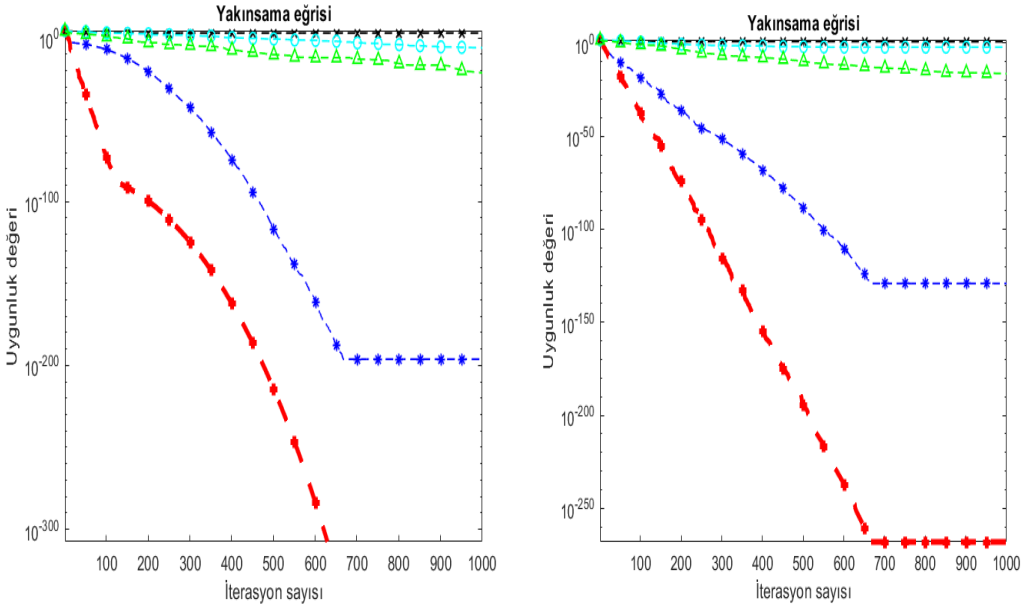
Bu bölümde KKO ve KOA algoritmaları, literatürdeki diğer iyi bilinen; diferansiyel evrim, yapay arı kolonisi ve parçacık sürü optimizasyonu algoritmalarıyla karşılaştırılmıştır. Bütün çalışmalarda, aday çözüm sayısı 20 olarak alınmış ve 1000 iterasyonda, 50 tekrar ile çalıştırılmıştır. Karşılaştırma yapılan algoritmaların kendilerine has parametreleri Tablo-4'de verilmiştir. Tablo-4, genellikle çalışmalarda kullanılan parametreler esas alınarak oluşturulmuştur [53].

**Tablo 4.** Karşılaştırılan algoritmalar için parametre ayarları

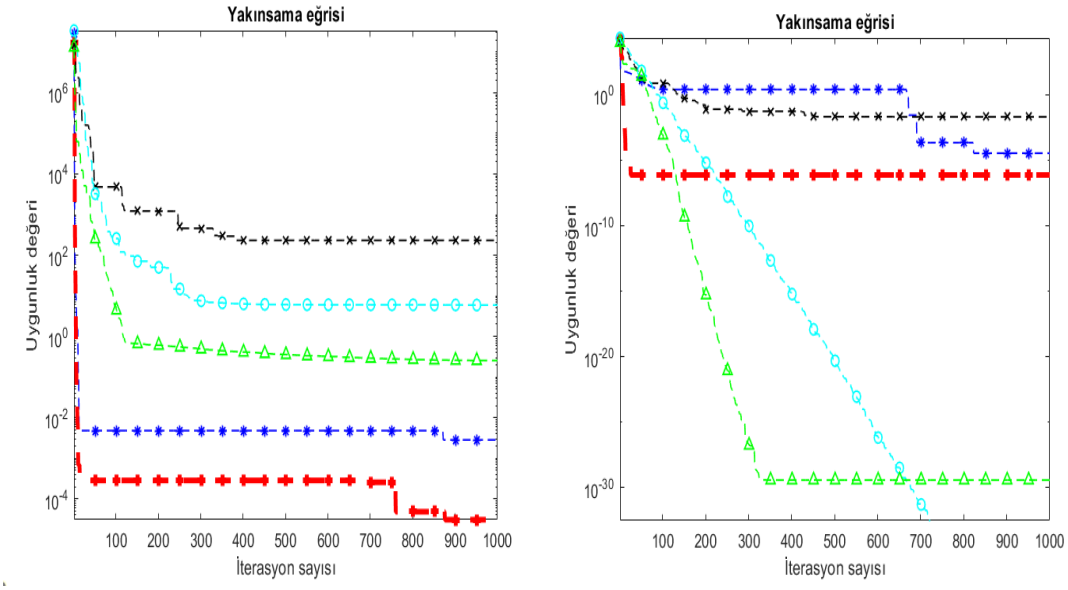
| Algoritma | Parametre ayarları |
|---|---|
| Yapay Arı Kolonisi | $\alpha = 1$ |
| Diferansiyel Evrim | Ölçekleme Faktörü= 0.5; Çaprazlama olasılığı= 0.5 |
| Parçacık Sürü Optimizasyonu | $c_1 = 1.5;\ c_2 = 2;\ v_{max} = 6$ |



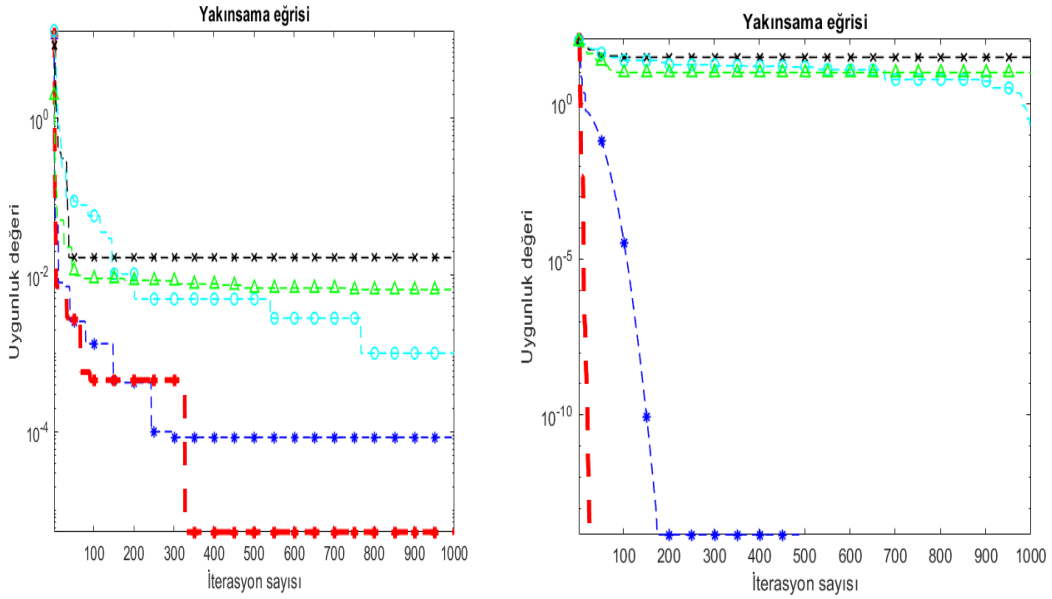

**Şekil 7.** F1-F2 kalite test fonksiyonları için karşılaştırma analizleri



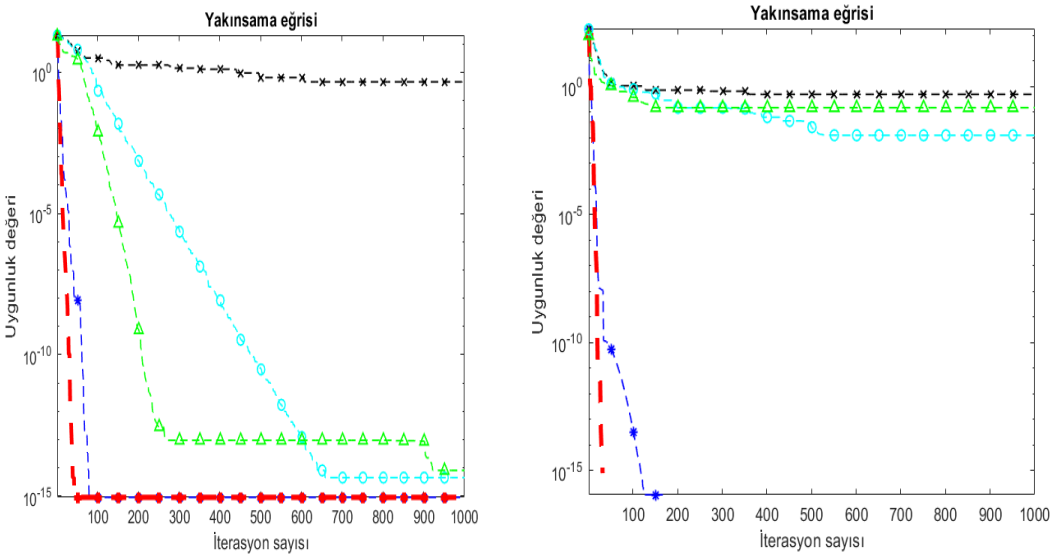

**Şekil 8**. F3-F4 kalite test fonksiyonları için karşılaştırma analizleri



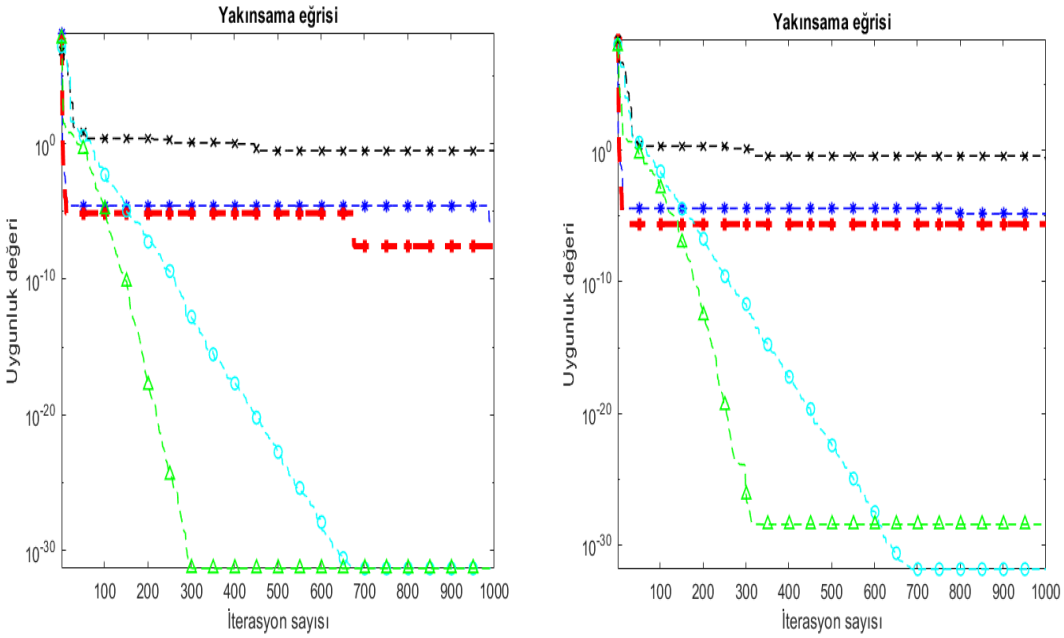

**Şekil 9**. F5-F6 kalite test fonksiyonları için karşılaştırma analizleri



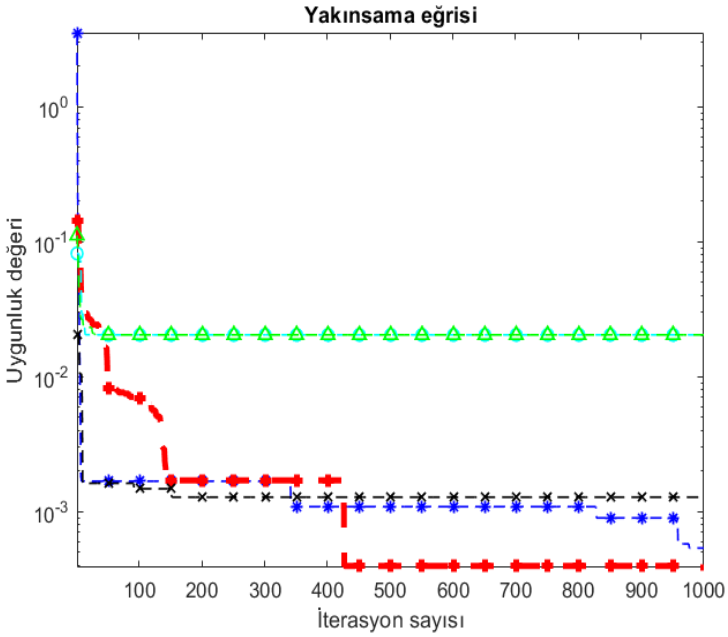

**Şekil 10**. F7-F8 kalite test fonksiyonları için karşılaştırma analizleri




**Şekil 11**. F9-F10 kalite test fonksiyonları için karşılaştırma analizleri




**Şekil 12**. F11-F12 kalite test fonksiyonları için karşılaştırma analizleri




**Şekil 13**. F13 kalite test fonksiyonu için karşılaştırma analizleri

Şekil 7-13 arasındaki figürler, KKO algoritmasının literatürdeki diğer bazı temel optimizasyon yöntemleri ile karşılaştırılması sonucundaki yakınsama davranışlarını sırasıyla göstermektedir. Şekil 7'de F1 ve F2, Şekil 8'de F3, F4, Şekil 9'da F5, Şekil 10'da F7, Şekil 13'de, F13 fonksiyonları için KKO'nın küresel minimuma diğer yöntemlerden daha iyi yakınsadığı ve daha düşük değerlere ulaştığı görülebilir. Diğer yandan KKO algoritması Şekil 10'da F8 ve Şekil 11'de F10 fonksiyonları için düşük iterasyonlarda direkt olarak küresel minimum değere yakınsamıştır. Son olarak, Şekil 9'da F6 ve Şekil 12'de F11 ve F12 fonksiyonlarında DE ve PSO daha iyi yakınsama göstermektedirler.

**Tablo 5.** KKO algoritmasının diğer yöntemler ile arasındaki Wilcoxon işaretli-sıralı test sonuçları

| Algoritmalar | YAK | | DE | | PSO | |
|---|---|---|---|---|---|---|
| Kalite test fonksiyonları | p değeri | Önem | p değeri | Önem | p değeri | Önem |
| F1 | 7.5569E-10 | + | 7.5569E-10 | + | 7.5569E-10 | + |
| F2 | 7.5569E-10 | + | 7.5569E-10 | + | 7.5569E-10 | + |
| F3 | 7.5569E-10 | + | 7.5569E-10 | + | 7.5569E-10 | + |
| F4 | 7.5569E-10 | + | 7.5569E-10 | + | 7.5569E-10 | + |
| F5 | 7.5569E-10 | + | 7.5569E-10 | + | 7.5569E-10 | + |
| F6 | 7.5569E-10 | + | 7.5569E-10 | + | 7.5569E-10 | + |
| F7 | 7.5569E-10 | + | 7.5569E-10 | + | 7.5569E-10 | + |
| F8 | 7.5569E-10 | + | 2.0195E-04 | + | 7.5382E-10 | + |
| F9 | 7.5569E-10 | + | 2.5396E-10 | + | 9.4640E-10 | + |
| F10 | 7.5569E-10 | + | 3.5862E-04 | + | 7.5569E-10 | + |
| F11 | 7.5569E-10 | + | 7.5569E-10 | + | 1.4172E-08 | + |
| F12 | 7.5569E-10 | + | 7.5569E-10 | + | 7.832E-01 | - |
| F13 | 5.3067E-07 | + | 6.3945E-06 | + | 1.9410E-05 | + |
| Toplam | 13/0 | | 13/0 | | 12/1 | |

Tablo-5 göz önüne alındığında, önerilen KKO ile YAK, DE ve PSO arasında Wilcoxon işaretli-sıralı test sonuçlarına göre %5 anlamlılık düzeyinde belirgin bir fark vardır. Dolayısıyla hem Tablo 3 hem de Tablo 5 dikkate

alınırsa, önerilen KKO algoritmasının hem orijinal KOA'dan hem de literatürdeki diğer üç algoritmadan (YAK, DE, PSO) farklı olduğu söylenebilir.

## 6. Sonuçlar

Bu çalışmada son zamanlarda önerilen KOA yönteminin rastgele değerleri kaotik haritalar ile yer değiştirilmiştir. Önerilen KKO algoritması, klasik KOA ile kalite test fonksiyonları üzerinde karşılaştırılmıştır. Kaotik harita olarak Gauss haritası kullanılmış, sonuçlar tablo ve şekiller ile desteklenmiştir. Kaotik haritaların tekrar edilememezlik ve ergodiklik özellikleri sayesinde KKO algoritması çözüm uzayını daha geniş taramaktadır. Dolayısıyla yerel minimum noktada takılmadan küresel minimum noktaya yakınsama ihtimalini arttırmaktadır. Tablo-2'de, F9 ve F13 hariç diğer bütün kalite test fonksiyonlarında önerilen KKO algoritması daha iyi sonuç vermektedir. Bütün bulgular göz önüne alındığında, KKO'nın, KOA'dan çok daha iyi yakınsama kabiliyetine sahip olduğu görülebilir. Özellikle Şekil-3 ve 4'de, F1, F2, F3, F4, F10 kalite test fonksiyonlarında KOA yerel minimum noktasında takılırken, KKO'nın küresel minimum noktaya daha yakın bir yakınsama sergilediği görülebilir. Diğer yandan Tablo-2'den, F1, F3 ve F10 fonksiyonlarında, direkt küresel minimum olan 0 noktasına yakınsadığı görülebilir. Diğer yandan, KKO algoritması literatürde sıkça kullanılan YAK, DE ve PSO algoritmaları ile karşılaştırılmıştır. Şekil 7-13'de, KKO'nun diğer algoritmalar ile karşılaştırılmasındaki yakınsama analizlerine yer verilmiştir. F1, F2, F3, F4, F5, F7, F8, F9, F10 ve F13 kalite test fonksiyonlarında KKO'nun diğer algoritmalardan daha iyi yakınsama kabiliyetinde sahip olduğu görülebilir. Ayrıca KKO ile YAK, DE ve PSO arasında yapılan Wilcoxon işaretli-sıralı teste göre, KKO'nun diğer 3 yöntemden istatistiksel olarak farklı olduğu söylenebilir. Bütün deneysel ve istatistiksel sonuçlar göz önüne alınırsa KKO algoritmasının yeni bir algoritma olduğu ve düşük iterasyonlarda küresel minimum değere yakınsama açısından başarılı sonuçlar ürettiği söylenebilir. KKO, başlangıç parametrelerinin öğrenilmesi için Sinir ağları, Aşırı öğrenme, ANFIS gibi yöntemlerde de uygulanabilir.

## Kaynaklar


[1] Osman, I.H. and J.P. Kelly, Meta-heuristics theory and applications. Journal of the Operational Research Society, 1997. 48(6): p. 657-657.

[2] Laporte, G., et al., Classical and modern heuristics for the vehicle routing problem. International transactions in operational research, 2000. 7(4-5): p. 285-300.

[3] Kumar, A., et al., A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. Swarm and Evolutionary Computation, 2020. 56: p. 100693.

[4] Andrei, N., An unconstrained optimization test functions collection. Adv. Model. Optim, 2008. 10(1): p. 147-161.

[5] Angira, R. and B. Babu, Optimization of process synthesis and design problems: A modified differential evolution approach. Chemical Engineering Science, 2006. 61(14): p. 4707-4721.

[6] He, X. and Y. Zhou, Enhancing the performance of differential evolution with covariance matrix self-adaptation. Applied Soft Computing, 2018. 64: p. 227-243.

[7] Kockanat, S. and N. Karaboga, Medical image denoising using metaheuristics, in Metaheuristics for medicine and biology. 2017, Springer. p. 155-169.

[8] Kumar, A., S. Singh, and A. Kumar. Grey wolf optimizer and other metaheuristic optimization techniques with image processing as their applications: a review. in IOP Conference Series: Materials Science and Engineering. 2021. IOP Publishing.

[9] Nakib, A. and E.-G. Talbi, Metaheuristics for medicine and biology. Vol. 704. 2017: Springer.

[10] 1Črepinšek, M., S.-H. Liu, and M. Mernik, Exploration and exploitation in evolutionary algorithms: A survey. ACM computing surveys (CSUR), 2013. 45(3): p. 1-33.

[11] 1Ab Wahab, M.N., S. Nefti-Meziani, and A. Atyabi, A comprehensive review of swarm optimization algorithms. PloS one, 2015. 10(5): p. e0122827.

[12] 1Kennedy, J. and R. Eberhart. Particle swarm optimization. in Proceedings of ICNN'95-international conference on neural networks. 1995. IEEE.

[13] 1Hussien, A.G., et al., Crow search algorithm: theory, recent advances, and applications. IEEE Access, 2020. 8: p. 173548-173565.

[14] Mirjalili, S., Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Computing and Applications, 2016. 27(4): p. 1053-1073.

[15] Karaboga, D. and B. Basturk, On the performance of artificial bee colony (ABC) algorithm. Applied soft computing, 2008. 8(1): p. 687-697.

[16] Yang, X.-S. and S. Deb, Engineering optimisation by cuckoo search. International Journal of Mathematical Modelling and Numerical Optimisation, 2010. 1(4): p. 330-343.

[17] Mirjalili, S. and A. Lewis, The whale optimization algorithm. Advances in engineering software, 2016. 95: p. 51-67.

[18] Mirjalili, S., S.M. Mirjalili, and A. Lewis, Grey wolf optimizer. Advances in engineering software, 2014. 69: p. 46-61.
[19] Heidari, A.A., et al., Harris hawks optimization: Algorithm and applications. Future generation computer systems, 2019. 97: p. 849-872.
[20] Abualigah, L., et al., Aquila Optimizer: A novel meta-heuristic optimization Algorithm. Computers & Industrial Engineering, 2021: p. 107250.
[21] Mirjalili, S., S.M. Mirjalili, and A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Computing and Applications, 2016. 27(2): p. 495-513.
[22] Kaveh, A. and S. Talatahari, A novel heuristic optimization method: charged system search. Acta Mechanica, 2010. 213(3): p. 267-289.
[23] Rashedi, E., H. Nezamabadi-Pour, and S. Saryazdi, GSA: a gravitational search algorithm. Information sciences, 2009. 179(13): p. 2232-2248.
[24] Mirjalili, S., SCA: a sine cosine algorithm for solving optimization problems. Knowledge-based systems, 2016. 96: p. 120-133.
[25] Abualigah, L., et al., The arithmetic optimization algorithm. Computer methods in applied mechanics and engineering, 2021. 376: p. 113609.
[26] Booker, L.B., D.E. Goldberg, and J.H. Holland, Classifier systems and genetic algorithms. Artificial intelligence, 1989. 40(1-3): p. 235-282.
[27] Hansen, N., S.D. Müller, and P. Koumoutsakos, Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evolutionary computation, 2003. 11(1): p. 1-18.
[28] Storn, R. and K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 1997. 11(4): p. 341-359.
[29] Atashpaz-Gargari, E. and C. Lucas. Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. in 2007 IEEE congress on evolutionary computation. 2007. Ieee.
[30] Rao, R.V., V.J. Savsani, and D. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Computer-Aided Design, 2011. 43(3): p. 303-315.
[31] Askari, Q., I. Younas, and M. Saeed, Political Optimizer: A novel socio-inspired meta-heuristic for global optimization. Knowledge-Based Systems, 2020. 195: p. 105709.
[32] Wolpert, D.H. and W.G. Macready, No free lunch theorems for optimization. IEEE transactions on evolutionary computation, 1997. 1(1): p. 67-82.
[33] Jenkinson, O., Ergodic optimization in dynamical systems. Ergodic Theory and Dynamical Systems, 2019. 39(10): p. 2593-2618.
[34] Kaur, G. and S. Arora, Chaotic whale optimization algorithm. Journal of Computational Design and Engineering, 2018. 5(3): p. 275-284.
[35] Kiefer, J., Sequential minimax search for a maximum. Proceedings of the American mathematical society, 1953. 4(3): p. 502-506.
[36] Koupaei, J.A., S.M.M. Hosseini, and F.M. Ghaini, A new optimization algorithm based on chaotic maps and golden section search method. Engineering Applications of Artificial Intelligence, 2016. 50: p. 201-214.
[37] Kohli, M. and S. Arora, Chaotic grey wolf optimization algorithm for constrained optimization problems. Journal of computational design and engineering, 2018. 5(4): p. 458-472.
[38] 38. Dunia, S. and S. Ramzy, Chaotic Sine-Cosine Algorithms. International Journal of Soft Computing, 2018. 13(3): p. 108-122.
[39] Altay, E.V. and B. Alatas, Bird swarm algorithms with chaotic mapping. Artificial Intelligence Review, 2020. 53(2): p. 1373-1414.
[40] Steenhof, K., M.N. Kochert, and T.L. Mcdonald, Interactive effects of prey and weather on golden eagle reproduction. Journal of Animal Ecology, 1997: p. 350-362.
[41] Carnie, S.K., Food habits of nesting golden eagles in the coast ranges of California. The Condor, 1954. 56(1): p. 3-12.
[42] Meinertzhagen, R., How do larger raptorial birds hunt their prey. Ibis, 1940. 4: p. 530-535.
[43] Dekker, D., HUNTING BEHAVIOR OF GOLDEN EAGLES, AQUILA-CHRYSAETOS, MIGRATING IN SOUTHWESTERN ALBERTA. 1985, OTTAWA FIELD-NATURALISTS CLUB PO BOX 35069, WESTGATE PO, OTTAWA ON K1Z 1A2 …. p. 383-385.
[44] Watson, J., The golden eagle. 2010: Bloomsbury Publishing.
[45] Eubank, S. and D. Farmer, An introduction to chaos and randomness, in 1989 lectures in complex systems. Proceedings: Lectures, Volume 2. 1990.
[46] dos Santos Coelho, L. and V.C. Mariani, Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization. Expert Systems with Applications, 2008. 34(3): p. 1905-1913.
[47] Ollagnier, J.M., Ergodic theory and statistical mechanics. Vol. 1115. 2007: Springer.
[48] Derrac, J., et al., A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation, 2011. 1(1): p. 3-18.
[49] Güvenc, U. and F. Katırcıoğlu, Escape velocity: a new operator for gravitational search algorithm. Neural Computing and Applications, 2019. 31(1): p. 27-42.
[50] KATIRCIOĞLU, F. and U. GÜVENÇ, Sequentially Modified Gravitational Search Algorithm for Image Enhancement. Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 2020. 8(4): p. 2266-2288.

[51] García, S., et al., A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. Journal of Heuristics, 2009. 15(6): p. 617-644.
[52] Wilcoxon, F., Individual comparisons by ranking methods, in Breakthroughs in statistics. 1992, Springer. p. 196-202.
[53] Yang, Y., et al., Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. Expert Systems with Applications, 2021. 177: p. 114864.