T.C.

ONDOKUZ MAYIS ÜNİVERSİTESİ LİSANSÜSTÜ

EĞİTİM ENSTİTÜSÜ BİLGİSAYAR

MÜHENDİSLİĞİ

OPTİMİZASYON-BM615

Altın Oran Arama ve Bracketing Yöntemi ile Fonksiyon Optimizasyonu Raporu

FIRAT KAAN BİTMEZ -

23281855

SAMSUN, 2024-2025 Eğitim Öğretim Yılı Güz Yarıyılı

1. Giriş

Bu rapor, referans kitap "**Engineering Optimization**"ta sayfa 298'de yer alan 5.8 numaralı örnekte anlatılan Altın Oran Arama algoritmasını kullanarak Bracketing Yöntemi'nin uygulanmasını bir pratik çalışmayla değerlendirmeyi amaçlamaktadır. Proje, bir fonksiyonun belirli bir aralıkta minimumunu bulmaya odaklanır ve Altın Oran kullanılarak yapılan iterasyonlarla arama aralığı daraltılır. Bu yöntem, arama bölgesini sistematik olarak daraltarak istenen tolerans seviyesine ulaşılmasını sağlar ve minimum değerin yaklaşık çözümünü sunar. Aynı zamanda, bu yöntem hem hesaplama maliyetini azaltarak hem de daha hızlı bir yakınsama sağlayarak optimizasyon problemlerinde kullanışllı bir seçenek sunar.

Altın Oran Arama yöntemi, özellikle tek tepe noktalarına sahip fonksiyonlarda kullanılmak üzere tasarlanmış bir tekniktir ve bu yöntemle elde edilen sonuçlar, hem teorik olarak hem de pratik uygulamalarda tatmin edici bir doğruluk sunar. Bu raporda, bahsi geçen yöntemin matematiksel altyapısı detaylandırılacak ve kodlama aşamasında izlenen adımlar ayrıntılarıyla analiz edilecektir. Ayrıca, algoritmanın güçlü ve zayıf yönleri de ortaya konularak gelecekteki olası iyileştirmelere ve çeşitli uygulama alanlarına da değinilecektir.

2. Matematiksel Arka Plan (Braketing ve Altın Oran Arama Yöntemi)

Bracketing Yöntemi, bir fonksiyonun belirli bir aralık içinde minimum veya maksimum değerini bulmaya yönelik temel bir optimizasyon tekniğidir. Bu yöntem, özellikle yalnızca bir minimum veya maksimum noktası (unimodal) olan fonksiyonlar için uygundur. Arama alanını sistematik olarak daraltarak belirlenen bir tolerans seviyesine ulaşmayı hedefler. Bracketing yöntemi, fonksiyonun türev bilgisine ihtiyaç duymadan yalnızca fonksiyon değerlerini kullanarak işlem yapar, bu da türevi alınamayan veya karmaşık yapıya sahip fonksiyonlar için önemli bir avantaj sağlar.

Altın Oran Arama Yöntemi ise, Bracketing yönteminin daha verimli bir alt türü olarak karşımıza çıkar. Bu yöntemde aralık, Altın Oran kullanılarak dengeli bir şekilde daraltılır ve bu sayede gereksiz hesaplamalar en aza indirilir. Altın Oran Arama Yöntemi, hem teorik hem de pratik açıdan optimizasyon problemlerinde sıkça tercih edilir.

2.1 Bracketing Yöntemi

Bracketing yöntemi, belirlenen bir başlangıç aralığı içinde bir fonksiyonun minimumunu bulmak için kullanılır. Yöntemin temel prensibi, her iterasyonda aralığı daraltarak minimum değerin yer aldığına inanılan alt aralığı seçmek ve bu işlem tolerans seviyesine ulaşılana kadar devam etmektir

Bracketing Yöntemin önemli özellikleri:

- **Türev Gerektirmez**: Bracketing yöntemi, fonksiyonun türev bilgisine ihtiyaç duymaz ve yalnızca fonksiyon değerleriyle çalışır.
- **Unimodal Fonksiyonlara Uygundur**: Bu yöntem, yalnızca bir minimum veya maksimum noktası bulunan fonksiyonlar için idealdir.
- **Dengeli Daraltma**: Aralık sistematik olarak küçültüldüğü için çözüm süreci düzenlidir ve hızlı yakınsama sağlanır.

Bu yöntem nasıl çalışır:

- **Başlangıç aralığını** ([a,b]) belirleyerek çalışmaya başlar.
- Aralıkta seçilen iki noktada (x1,x2) fonksiyon değerleri karşılaştırılarak daraltılacak alt aralık belirlenir.
- **Unimodal (tek tepe veya çukur) fonksiyonlar** için uygundur.

Matematiksel olarak:

1. **İki noktanın tanımı**: $x1=b-\phi\cdot(b-a)$, $x2=a+\phi\cdot(b-a)$

   Burada $\phi= /5-12\ /2 \approx 0.618$ Altın Oran katsayısıdır.
2. **Fonksiyon değerlerinin karşılaştırılması**: Eger $f(x1)>f(x2)$ ise $a=x1$,aksi halde $b=x2$

   Bu işlem, her iterasyonda aralığı daraltarak tolerans seviyesine kadar devam eder. Tolerans ε sağlandığında:

   $$|b-a|<\varepsilon$$

   yakınsama gerçekleşmiş olur.

2.2 Altın Oran Arama Yöntemi

Altın Oran Arama Yöntemi, Bracketing yönteminin etkinliğini artırmak için kullanılan bir tekniktir. Bu yöntemde, Altın Oran ($\phi=/5-12\ /2\approx 0.618$) kullanılarak aralık, dengeli bir şekilde daraltılır. Altın Oran, aralığı iki alt aralığa bölerken ideal dengeyi sağlayarak optimizasyon sürecinin hızlanmasına olanak tanır.

Altın Oran Arama Yönteminin Uygulama Adımları:

1. **Başlangıç Aralığı**: İlk olarak [a,b] aralığı belirlenir.
2. **İki Noktanın Tanımı**:
   - $x1=b-\phi\cdot(b-a)$
   - $x2=a+\phi\cdot(b-a)$

Burada ϕ, Altın Oran katsayısıdır.

3. **Fonksiyon Değerlerinin Karşılaştırılması**:
    - Eğer f(x1)>f(x2)f(x_1) > f(x_2)f(x1)>f(x2) ise a=x1a = x_1a=x1,
    - Aksi halde b=x2.
4. **Aralığın Güncellenmesi**: Her iterasyonda, minimuma daha yakın olduğu düşünülen alt aralık seçilir ve işlem, tolerans (|b−a|<ε) sağlanana kadar devam eder.

Altın Oran Arama Yönteminin Avantajları:

- **Dengeli Daraltma**: Altın Oran, her iterasyonda aralığı sistematik olarak daraltarak gereksiz hesaplamaları önler.
- **Türev Gerektirmez**: Karmaşık veya türev alınamaz fonksiyonlar için uygundur.
- **Yakınsama Garantisi**: Her adımda aralık daraltıldığı için unimodal fonksiyonlarda yakınsama garantilidir.
- **Hızlı ve Verimli**: Dengeli daraltma sayesinde algoritma daha az iterasyonla sonuca ulaşır.

3. Kullanılan Algoritma ve Kodlama

3.1. Python Algoritmasının Yapısı

Altın Oran Arama Yöntemi, Python dilinde kodlanarak optimize edilmiştir. Bu algoritmanın amacı, verilen bir unimodal fonksiyonun belirli bir aralık içinde minimum noktasını bulmaktır. Kodlama sürecinde iterasyonlar sistematik bir şekilde gerçekleştirilir ve sonuçlar matematiksel doğrulukla elde edilir.

Amaç Fonksiyonu: Optimizasyon yapılan fonksiyon, türev alınması zor olabilecek bir yapıya sahiptir ve şu şekilde tanımlanmıştır:

f(x)=0.65−1+x20.75−0.65·arctan(x1)

Bu fonksiyonun özellikleri:

- **Unimodal:** Fonksiyonun sadece bir minimum noktası vardır.
- **Sürekli:** Optimizasyon algoritması için uygun bir yapı sağlar.
- **Türev Gerekmez:** Fonksiyon değerlerine dayalı olarak çalışır.

3.2. Algoritmanın Detaylı Adımları

Altın Oran Arama algoritması aşağıdaki adımlarla uygulanmıştır:

1. **Başlangıç Aralığının Belirlenmesi:**
   Optimizasyonun başlangıç aralığı [a,b]olarak tanımlanır. Bu çalışmada başlangıç aralığı olarak [−2,2]seçilmiştir.
2. **Altın Oran Sabitinin Hesaplanması:**
   Altın Oran (ϕ), aralıkları dengeli bir şekilde daraltmak için kullanılır. Matematiksel olarak:

$$\phi = /5-1\ /2 \approx 0.618$$

3. **İlk Noktaların Hesaplanması:**
   Aralığın uçlarındaki iki nokta (x1,x2x_1, x_2x1,x2) şu şekilde hesaplanır:

$$x1=b-\phi\cdot(b-a), x2=a+\phi\cdot(b-a)$$

4. **Fonksiyon Değerlerinin Karşılaştırılması:**
   Hesaplanan noktalardaki fonksiyon değerleri (f(x1),f(x2) karşılaştırılır:

- Eğer f(x1)>f(x2), minimum nokta [x1,b] aralığında bulunur, bu durumda a=x1 olarak güncellenir.
- Aksi durumda, minimum nokta [a,x2] aralığında bulunur, bu durumda b=x2 olarak güncellenir.

5. **İterasyon Döngüsü:**
   Yukarıdaki işlemler, aralık uzunluğu (∣b−a∣) belirlenen tolerans (ϵ=10−6) değerinden küçük olana kadar tekrar edilir.
6. **Sonuçların Hesaplanması:**
   Algoritma, minimum nokta tahmini olarak aralığın ortasını döndürür:

$$\text{Min Nokta: } x=2a+b, \text{Fonksiyon Degeri: } f(x)$$

Python Kodunun Çalışma Prensipleri

Kod, yukarıdaki adımları takip ederek yapılandırılmıştır. İşte Python uygulaması:

```
import numpy as np

def f(x):
    if abs(x) < 1e-8:
        x = np.sign(x) * 1e-8  # Bölme hatasını önlemek için epsilon ekleme
    return 0.65 - (0.75 / (1 + x**2)) - 0.65 * np.arctan(1 / x)

def bracketing_method(f, a, b, tol=1e-6):
    phi = (np.sqrt(5) - 1) / 2
    x1, x2 = b - phi * (b - a), a + phi * (b - a)
    f1, f2 = f(x1), f(x2)

    while abs(b - a) > tol:
        if f1 > f2:
```

```
            a, x1, f1 = x1, x2, f2
            x2 = a + phi * (b - a)
            f2 = f(x2)
        else:
            b, x2, f2 = x2, x1, f1
            x1 = b - phi * (b - a)
            f1 = f(x1)

    return (a + b) / 2, f((a + b) / 2)

# Çalıştırma:
min_point, min_value = bracketing_method(f, -2, 2, tol=1e-6)
print(f"Minimum Nokta: x = {min_point}, f(x) = {min_value}")
```

Parametreler

Giriş Parametreleri:

- a,b: Başlangıç aralığı ([−2,2]).
- tolerans: Yakınsama toleransı (10^{-6}).
- f(x): Optimizasyonu yapılacak fonksiyon.

Çıktılar:

- Minimum nokta ve bu noktadaki fonksiyon değeri.

4. Sonuçların Detaylı Analizi

4.1. İterasyon Detayları

Altın Oran Arama algoritması, başlangıç aralığından başlayarak her iterasyonda aralığı daraltır. İşte iterasyonların detaylı bir analizi:

1. **Başlangıç Durumu:**
    - **Aralık:** [−2,2]
    - **İlk Noktalar:** x1=−0.4721,x2=0.4721
    - **Fonksiyon Değerleri:** f(x1)=0.771,f(x2)=−0.698
2. **1. İterasyon:**
    - f(x1)>f(x2) bu nedenle aralık [−0.4721,2] olarak daraltılır.
    - Yeni noktalar: x1=0.4721,x2=1.0557
    - Fonksiyon Değerleri: f(x1)=−0.698,f(x2)=−0.198
3. **2. İterasyon:**

        - $f(x1) \le f(x2)$, aralık $[-0.4721, 1.0557]$ olur.
        - Güncellenen noktalar ve fonksiyon değerleri hesaplanır.

4. **Son İterasyon:**
    - Aralık tolerans seviyesine (10^{-6}) ulaştığında durur.
    - **Son Aralık:** $[0.000000, 0.000001]$
    - **Minimum Nokta:** $x \approx 0.000$
    - **Minimum Değer:** $f(x) \approx -1.121$

4.2. Çıktıların Yorumu

- **Minimum Nokta:** Algoritma, minimum nokta olarak $x \approx 0.000$ sonucunu döndürmüştür.
- **Minimum Değer:** Bu noktadaki fonksiyon değeri $f(x) \approx -1.121$olarak hesaplanmıştır.
- **Yakınsama Süreci:** Aralık her iterasyonda sistematik olarak daraltılmış ve yakınsama başarılı bir şekilde sağlanmıştır.

Sonuçların Doğrulanması

- **Matematiksel Tutarlılık:** Sonuçlar, Altın Oran Arama yönteminin teorik temelleriyle uyumlu çıkmıştır.
- **Hızlı Yakınsama:** Her iterasyonda aralığın $\phi \approx 0.618$ oranında daraltılması, algoritmanın hızlı yakınsamasını sağlamıştır.

5. Yöntemin Değerlendirilmesi

5.1 Algoritmanın Avantajları

1. **Hızlı Yakınsama**: Altın Oran, her iterasyonda dengeli bir daraltma sağlar.
2. **Türev Bilgisine İhtiyaç Duymaması**: Karmaşık fonksiyonlarda kullanılabilir.
3. **Kapsamlı Uygulama Alanı**: Matematik, mühendislik ve ekonomi alanlarında yaygın olarak kullanılabilir.

5.2 Eksiklikler

- **Yalnızca unimodal fonksiyonlar için uygundur**. Multimodal fonksiyonlarda farklı yöntemlerle birleştirilmesi gerekebilir.
- **Hassasiyet ve hız arasında bir denge sağlanmalıdır**. Tolerans değeri küçüldükçe işlem süresi uzar.

6. SONUÇ

Altın Oran Arama yöntemi, özellikle türev alınamayan veya karmaşık fonksiyonların optimizasyonunda etkili bir tekniktir. Bu çalışma, yöntemin hem matematiksel temellerini hem de pratik uygulamasını başarılı bir şekilde göstermiştir. Kodun çıktıları, teorik olarak beklenen sonuçlarla uyumlu olup yöntemin doğruluğunu kanıtlamaktadır.

Bu yöntem, geniş bir uygulama alanına sahip olup büyük ölçekli optimizasyon problemlerinde güvenilir bir araçtır. **Gelecek çalışmalar**, yöntemin multimodal fonksiyonlara uygulanabilirliğini artırmaya odaklanabilir.

7.KAYNAKLAR

- Rao, S. S. (1996). ***Engineering Optimization: Theory and Practice*** (3rd ed.). Wiley-Interscience. (Page: 310-340)
- Livio, M. (2003)**. *The Golden Ratio: The Story of Phi, the World's Most Astonishing*** *Number*. Broadway Books.
- Bitmez, F. K. Optimization Repository. ***GitHub***. https://github.com/firatkaanbitmez/Optimization