Erciyes Üniversitesi
Fen Bilimleri Enstitüsü Dergisi
Cilt 38, Sayı 1, 2022

Erciyes University
Journal of Institue Of Science and Technology
Volume 38, Issue 1, 2022

Yönsel Türev Tabanlı Yakınsama Yaklaşımlarının Karşılaştırmalı Analizi

Ersan YAZAN*[1], Muhammed Fatih TALU[2],

*[1] Adıyaman Üniversitesi Besni MYO Bilgisayar Teknolojileri, ADIYAMAN
[2] İnönü Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği, MALATYA

(Alınış / Received: 18.03.2022, Kabul / Accepted: 14.04.2022, Online Yayınlanma / Published Online: 30.04.2022)

Anahtar Kelimeler
Yönsel türev,
Momentum, Adadelta,
Adagrad, Adam, Newton

Öz: Yönsel türev tabanlı yakınsama yaklaşımları, doğrusal olmayan bir maliyet fonksiyonunu (J(θ)) minimize veya maksimize etmek için θ parametre vektörünü iteratif olarak güncelleme mantığına dayanmaktadır. Yapay öğrenme alanında sık kullanılan bu yaklaşımlarla güncelleme yapılırken maliyet fonksiyonunun ilgili parametreye göre yönsel türev bilgisi kullanılır. Bu çalışmada, literatürde kabul görmüş türev tabanlı yaklaşımların (Gradient Descent, Momentum, Adadelta, Adagrad, Rms, Adam, Newton ve BFGS) doğrusal olmayan fonksiyonlar üzerindeki yakınsama performansları incelenmiştir. Yaklaşım performanslarını kıyaslayabilmek için optimal noktaya yakınsama hızı ve toplam hesaplama maliyeti kıstasları değerlendirilmiştir.

Comparative Analysis of Gradient Based Convergence Approaches

Keywords
Gradient Descent,
Momentum, Adadelta,
Adagrad, Adam, Newton

Abstract: Directional derivative-based convergence approaches are based on the logic of iteratively updating the parameter vector θ to minimize or maximize a nonlinear cost function (J(θ)). While updating with these approaches, which are frequently used in the field of artificial learning, directional derivative information of the cost function is used according to the relevant parameter. In this study, the convergence performances of derivative-based approaches (Gradient Descent, Momentum, Adadelta, Adagrad, Rms, Adam, Newton and BFGS) accepted in the literature on nonlinear functions were investigated. In order to compare the approach performances, the speed of convergence to the optimal point and the total computational cost criteria were evaluated.

*İlgili Yazar, email: eyazan@adiyaman.edu.tr

1. Giriş

Mühendislik uygulamalarında çok önemli bir yere sahip olan optimizasyon, en basit tanımıyla, belirlenen koşullar altında en iyi sonucun alınmasıdır. Bir sistemin gerçekleştirilme sürecinde bazı kararların alınması gerekir. Bu kararların amacı, sistemin gerçekleştirilmesi için gereken çabayı en aza indirgemek veya sistemden beklenen kazanımı olabildiğince arttırmaktır. İstenen çaba veya kazanım, belirli karar değişkenlerinin bir fonksiyonu olarak ifade edilebildiğinden optimizasyon, bir fonksiyonun belirlenen amaç doğrultusunda maksimum veya minimum değerini veren koşulları bulma süreci olarak tanımlanabilir.

Optimizasyon yöntemlerinin temeli matematik bilimine dayanır. Bu nedenle matematik bilimindeki birçok gelişme optimizasyon yöntemlerinin geliştirilmesine temel oluşturmuştur. Newton ve Leibniz'in matematiğe katkısı optimizasyonun diferansiyel yöntemler geliştirilmesini mümkün kılmış, Bernoulli, Euler, Lagrange ve Weirstrass, fonksiyonların minimizasyonuyla alanın temellerini atmışlar, Cauchy ise kısıtlamasız minimizasyon problemlerini çözmek için en dik iniş metodunun ilk uygulamasını gerçekleştirmiştir [1]. Bunların dışında birçok gelişme optimizasyon alanında

ilerleme kat edilmesine yardımcı olmuştur. Bütün bu gelişmelere rağmen, sayısal bilgisayar teknolojisinin ortaya çıkışına kadar bu alandaki çalışmalarda kaydedilen ilerleme çok az denilebilecek kadar olabilmiştir. Bilgisayar teknolojisindeki gelişmeyle birlikte optimizasyon tekniklerinin uygulanması mümkün olmuş ve yeni metotlar üzerine daha fazla araştırma yapılmıştır.

Optimizasyon alanında geliştirilen yöntemler hemen her mühendislik alanında kullanıldığı gibi bilgisayar bilimlerinde de oldukça kullanım alanına sahiptir. Yapay öğrenme ve görüntü işleme bu alanların başında gelmektedir. Yapay öğrenme alanında problem fark etmeksizin, yapay sinir ağlarındaki hücreler arası ağırlık değerlerinin güncellenmesi için kullanılan bu yöntemler [2]-[4], görüntü işleme alanında şekil bölütleme, parmak izi tanıma, plaka tanıma gibi birçok alanda kullanılmaktadırlar [5]-[7]. Ayrıca bu iki alan temelinde medikal [8]-[10] ve sinyal işleme [11] gibi diğer farklı alanlardaki problemlerin çözümünde de kullanılmaktadırlar.

Genel bir optimizasyon probleminin matematiksel ifadesi eşitlik 1'deki gibi gösterilebilir:

$$\begin{aligned} &\text{Minimize } f(\theta), \theta = (\theta_1, \theta_2, \ldots . \theta_n)^T \\ &\text{Kısıtlamalar;} \\ &g_j(\theta) < \theta, j = 1, 2, \ldots m \\ &h_k(\theta) = \theta, k = 1, 2, \ldots n \end{aligned} \tag{1}$$

Burada θ tasarım değişkeni, f(θ) tasarım değişkenine bağlı minimize edilmek istenen amaç (hedef) fonksiyon, g(θ) ve h(θ) ise sırayla eşitsizlik ve eşitlik kısıtlayıcıları olarak adlandırılır. Bu şekilde tanımlanan optimizasyon problemleri; kısıtlayıcılı optimizasyon problemi olarak adlandırılır ve çözüm kısıtlayıcılar ile çevrelenmiş bölgenin içerisinde aranır. Bazı problemler ise kısıtlayıcı içermez, bu tarz problemler ise kısıtlayıcısız optimizasyon problemleri olarak adlandırılır. Bu iki problem türü haricinde optimizasyon problemlerinin farklı şekillerde sınıflandırılması yapılmıştır. Bu sınıflandırmaları genel olarak şu şekilde tanımlayabiliriz; amaç fonksiyonun ve varsa kısıtlama fonksiyonlarının doğrusal olup olmamasına bağlı olarak; doğrusal (lineer) programlama veya doğrusal olmayan (nonlineer) programlama, tasarım değişkenleri negatif olmayan tamsayı değerler alan lineer problemler tamsayı programlama, aksi durumda tamsayı olmayan programlama olarak isimlendirilirler. Bir diğer sınıflandırma da tasarım değişkenlerinin sürekli değerler olduğu sürekli optimizasyon problemleri veya sınırlı değerlerden oluştuğu ayrık optimizasyon problemleri şeklindedir.

Problemlerin sınıflandırıldığı gibi, bu problemlerin çözümü için geliştirilen yöntemler de çözüm yaklaşımına göre sınıflandırılmaktadır. Klasik optimizasyon yöntemleri genel olarak Doğrudan (direct) yöntemler (araştırma yöntemleri) ve Dolaylı (indirect) yöntemler (optimallik kriterleri) olmak üzere ikiye ayrılır [12]. Dolaylı yöntemlerde bir noktanın optimum nokta olabilmesi için gerek ve yeter şartları sağlaması gerekir. Problem çözümünde öncelikle bu şartları sağlayan aday noktalar bulunur, sonrasında bu aday noktalara göre çözüm uygulanarak optimum nokta elde edilmeye çalışılır. Doğrudan yöntemlerde ise, tahmini bir başlangıç noktası belirlenerek optimum nokta olarak kabul edilir ve bu noktaya göre çözüm elde edilir. Bu başlangıç değerine göre elde edilen çözüm genellikle optimallik şartlarını sağlamaz. Bu nedenle gerçek optimum nokta bulunana kadar tekrarlı (iteratif) olarak işlemler yinelenir.

Bu çalışmada doğrudan yöntemler sınıfına giren ve Kısıtlamasız Doğrusal Olmayan (Nonlineer) problemlerin çözümünde kullanılan Türev Tabanlı Yöntemler kıyaslamalı bir şekilde incelenmiştir. İkinci bölümde, kullanılan türev tabanlı yöntemler hakkında detaylı bilgi verilmektedir. Üçüncü bölümde yöntemlerin belirli test fonksiyonları üzerinde uygulanarak performansları gözlemlenmiş ve sonuçları gösterilmiştir. Son bölümde ise çalışmanın genel sonuçları üzerinde değerlendirmeler yapılmıştır.

2. Materyal ve Metot

Türev tabanlı yöntemler, optimum noktayı bulmak için türev bilgisini kullanırlar. Bu nedenle, bu yöntemlerin kullanıldığı problemlerde amaç fonksiyonunun, uygulanabilir aralıkta sürekli ve türevlenebilir olması gerekmektedir. Bu algoritmalar, tasarım değişkenleri için gelişi güzel seçilen değerlerle işleme başlar ve amaç fonksiyonunun o noktadaki türevini alır. Bulunulan noktadan türev yönünde giderek optimum noktaya ulaşmaya çalışır. Bu yöntemler lokal bilgileri kullandıkları için varılacak optimum nokta lokal optimum noktadır. Bu yöntemlerin genel olarak işleyişi Eşitlik. 2'de vektörel formda gösterilmiştir.

$$\theta_{t+1} = \theta_t + \Delta\theta_t; \quad t = 0, 1, 2, \ldots \tag{2}$$

Parametrelerin güncelleme işlemi optimal noktaya varıncaya kadar tekrar edilir. Optimum noktada türev değeri 0'a eşittir. Burada; θ, tasarım değişkeni, t iterasyon numarası, Δθ ise mevcut noktadaki değişim miktarını ifade etmektedir. Değişim miktarının hesaplanması genel olarak Eşitlik.3 'te gösterildiği gibi ifade edilebilir.

$$\Delta\theta_t = \alpha d_t \tag{3}$$

Burada d, ilerleme yönü; α, adım büyüklüğü olarak ifade edilmektedir. Eşitlik.3'te gösterilen değişim miktarının hesaplanması bu iki değerin belirlenmesi problemini barındırmaktadır. Yönsel türev tabanlı yaklaşımlar, ilerleme yönünde atılacak adım büyüklüğünün hesaplanması noktasında farklılık gösterirler.

2.1. Türevsel iniş (Gradient descent)

1847'de Cauchy tarafından ortaya atılan yöntem [13] "Dereceli Alçalma Yöntemi" olarak da adlandırılır ve kısıtlamasız optimizasyon problemlerinin çözümünde kullanılan türev tabanlı yöntemlerin ilkidir. Bu yöntemde ilerleme yönünü hesaplarken amaç fonksiyonunun birinci türevi alındığı için "birinci dereceden yöntem" olarak nitelendirilir.

Bu yöntem, θ parametrelerine bağlı maliyet fonksiyonunu (J(θ)) minimize etmek için kullanılır. Başlangıçta rastgele belirlenen θ parametrelerinin her adımda türevin tersi yönünde güncellenerek optimum değere ulaşılıncaya kadar tekrar edilmesi ile gerçekleştirilir:

$$\theta_t = \theta_{t-1} - \alpha.\nabla_\theta J(\theta) \tag{4}$$

Eşitlik.4'te yer alan θ güncellenecek parametre vektörü, α adım büyüklüğü ve $\nabla_\theta J(\theta)$ maliyet fonksiyonunun gradientini göstermektedir. Gradient descent (türevsel iniş) yönteminin, "Batch", "Mini-Batch" ve "Online (SGD)" olmak üzere üç farklı kullanış şekli bulunmaktadır [14]. Bu yöntemler maliyet hesabında kullanılan veri miktarına göre birbirinden ayrılırlar. Batch'de, maliyet fonksiyonu hesaplanırken tüm eğitim kümesi kullanılırken, mini-batch'de belirli sayıdaki örüntüler kullanır, online'da her bir örüntü geldiğinde parametreler güncellenir. Dolayısıyla Batch kullanımda yavaş çalışır ancak yakınsama tutarlılığı en doğru olandır. Buna karşılık online kullanım her bir eğitim seti için güncelleme işlemi gerçekleştirdiği için daha hızlı çalışır ve yerel minimuma daha erken sürede ulaşabilir, ancak yakınsama hatası yüksektir.

$$\theta_t = \theta_{t-1} - \alpha.\nabla_\theta J(\theta; x^i, y^i) \tag{5}$$

Mini-Batch kullanım her iki yöntemin avantajlarını barındıracak şekilde düşünülerek geliştirilmiştir. Güncelleme işlemi, her bir n adet örüntü için gerçekleştirilir.

$$\theta_t = \theta_{t-1} - \alpha.\nabla_\theta J(\theta; x^{i+n}, y^{i+n}) \tag{6}$$

Türevsel İniş yönteminin bazı dezavantajları vardır. Bu dezavantajlar şunlardır:

- Yöntem lokal minimuma yakınsamayı garanti eder, fakat bunun için çok fazla sayıda iterasyon gerekebilir
- Her iterasyon diğerlerinden bağımsız bir şekilde başlatılır ve bu da verimsiz olabilir. Önceki iterasyon bilgileri kullanılmaz.

Her iterasyonda ilerleme yönünü belirlemek için sadece birinci dereceden bilgiler kullanılır. Bu da yöntemin yakınsama kabiliyetini düşürür. Pratikte, ilk birkaç iterasyon sonucunda amaç fonksiyonunda önemli ölçüde bir azalma olduğu gözlemlenebilir, fakat sonraki iterasyonlarda bu azalma yavaşlamaktadır.

2.2. Momentum

Momentum yöntemi yönsel türevin sadece anlık değil, geçmiş değerlerinide oransal bir şekilde hesaba katar ve daha etkili bir yakınsama özelliği gösterir [15].

$$v_t = \rho.v_{t-1} + \alpha.\nabla_\theta J(\theta; x^i, y^i) \tag{7}$$

$$\theta_t = \theta_{t-1} - v_t \tag{8}$$

Eşitlik.7'de görülen ρ parametresi 0 ile 1 arasında olup, bir sonraki θ' nın hesaplanmasında önceki türevlerin hangi oranda hesaba alınacağını belirlemek için kullanılır.

2.3. Adagrad

Adagrad yöntemi [16] sabit adım büyüklüğünden kaynaklanan problemi ortadan kaldıran bir yöntemdir. Bu yönteme göre, başlangıçta bir adım büyüklüğü belirlenerek ilk iterasyon gerçekleştirilir. Sonraki iterasyonlarda bu adım büyüklüğü güncellenerek işlemler gerçekleştirilir.

$$G_t = G_{t-1} + \nabla J(\theta_{t-1})^2 \tag{9}$$

$$\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{G_{t-1} + \epsilon}} . \nabla J(\theta_{t-1}) \tag{10}$$

G_t, geçmiş gradient bilgisini tutan, parametre sayısı ile aynı boyutta bir matristir ve adım büyüklüğünün ölçeklenmesinde kullanılır. Bu matrisin her bir elemanı ilgili parametrenin önceki gradientlerinin karelerinin toplamına eşittir. Eşitlik.10' da yer alan ϵ terimi, sıfıra bölünmeyi önlemek için kullanılır ve çok küçük bir değerdir.

2.4. Adadelta

Adagrad yöntemi her ne kadar adım büyüklüğü parametresi problemine odaklanmış bir yöntem olsa da ilk iterasyonda bir adımbüyüklüğü değerine ihtiyaç duyar. Ayrıca dikkatle incelendiğinde iterasyon sayısı arttıkça G parametresi büyüyeceği için adım büyüklüğünün sıfıra yaklaştığı görülmektedir. Bu da belirli bir iterasyondan sonra yakınsama hızının azalmasına sebepiyet verir. İşte bu iki durum Adadelta [17] yönteminin geliştirilmesinde önemli bir etken olmuştur. Adadelta yöntemi, Adagrad yönteminin bu iki dezavantajını ortadan kaldırmıştır.

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1-\rho) g_t^2 \tag{11}$$

$$\Delta\theta_t = -\frac{RMS[\Delta\theta]_{t-1}}{RMS[g]_t} . g_t \tag{12}$$

$$RMS[g]_t = \sqrt{E[g^2]_t + \epsilon} \tag{13}$$

$$E[\theta^2]_t = \rho E[\theta^2]_{t-1} + \Delta\theta_t^2 \tag{14}$$

$$G_t = G_{t-1} + \nabla J(\theta_{t-1})^2 \tag{15}$$

$$\theta_{t+1} = \theta_t - \Delta\theta_t \tag{16}$$

Eşitlik. 11'de görüldüğü üzere, Adagrad yönteminde olduğu gibi önceki türevler (gradientler) karelerinin tamamı olacak şekilde değil, belirli bir oranda alınmaktadır. Bunu da momentum yönteminde olduğu gibi ρ sabit terimi ile ayarlamaktadır.

2.5. RMSProp

Geoff Hinton tarafından önerilmiş ve onun ders notlarında yer alan RMSProp yöntemi [18], Adadelta ile yakın zamanda birbirinden bağımsız olarak ortaya atılmıştır [14]. Adadelta yöntemine benzer fakat öğrenme katsayısı parametresini kullanır.

$$E[g^2]_t = \rho E[g^2]_{t-1} + (1-\rho) g_t^2 \tag{17}$$

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{E[g^2]_t + \epsilon}} . g_t \tag{18}$$

Notlarında ρ = 0.9 ve α = 0.001 olarak alınmasını öneren Hinton, bu değerlele yöntemin iyi sonuçlar vereceğini belirtmiştir.

2.6. Adam

Adaptive Moment Estimation (Adam) yöntemi [19] de sabit adım büyüklüğü yerine, her iterasyonda güncellenen bir adım büyüklüğü kullanan yöntemlerdendir. Bu yöntemde önceki türev değerinin karesel toplamları da hesaba katılmaktadır.

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{19}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{20}$$

$$\acute{m}_t = \frac{m_t}{1-\beta_1^t} \tag{21}$$

$$\tilde{v}_t = \frac{v_t}{1-\beta_2^t} \tag{22}$$

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\tilde{v}_t + \epsilon}} . \acute{m}_t \tag{23}$$

Ayrıca çalışmada, ß$_1$ = 0 ve çok küçük bir ß$_2$ seçildiğinde Adagrad yöntemine karşılık geldiği belirtilmiştir.

2.7. Newton

Newton tarafından önerilen ve kendi adıyla anılan yöntem, parametre güncellemesinde, sadece birinci kısmi türevi değil ikinci türev değerini de kullanır [20]. Bu nedenle "ikinci dereceden yöntem" olarak nitelendirilir. Birinci dereceden yöntemlere göre daha etkili olan yöntem, özellikle optimum noktaya belirli bir mesafeden sonra çok hızlı bir yakınsama özelliği gösterir. Temel fikri Taylor serisi genişlemesine dayanan yöntem Eşitlik.24 'te gösterildiği gibidir.

$$\theta_{t+1} = \theta_t + \frac{f'(\theta)}{f''(\theta)} \tag{24}$$

Eşitlik.24 'te de görüldüğü gibi adım büyüklüğü (α) parametresi kullanılmamaktadır. Bu nedenle yöntem her zaman için yerel optimum değere yakınsamayı garanti etmez. Bu problem, adım büyüklüğünü de eşitliğe ekleyerek giderilebilir.

Newton yöntemi kuadratik iç bükey fonksiyonlarda çok başarılı olurken içbükey olmayan fonksiyonlarda aynı başarıyı gösterememektedir [18]. Başka bir deyişle içbükeylik azaldıkça Newton yönteminin performansı da azalmaktadır.

Newton yönteminin en büyük dezavantajı her iterasyonda ikinci dereceden türevlerin (Hessian matrisi) hesaplanmasıdır. Bu işlem genellikle oldukça zaman alır, hatta bazı uygulamalarda bu türevi hesaplamak mümkün olmayabilir. Bir diğer dezavantajı da, Hessian matrisinin tekil olması gerekliliğidir. Fakat her iterasyonda Hessian matrisi tekil olmayabilir. Bundan dolayı ilerleme yönü bizi optimum noktaya götürmeyebilir. Optimum noktaya gidebilmek için Hessian matrisinin pozitif tanımlı ve tekil olması gerekmektedir.

2.7. Bfgs

Newton yöntemindeki, Hessian matrisinin oluşturulması ile ilgili dezavantajları ortadan kaldırmak için bazı yöntemler önerilmiştir [21 - 24]. Önerilen bu yöntemler, her iterasyonda Hessian matrisi veya Hessian matrisinin tersine yaklaşık bir matris üreterek, Hessian matrisi yerine kullanmışlardır. BFGS (Broyden-Fletcher-Goldfarb-Shanno) yöntemi [25] Hessian matrisine yaklaşık matris üreten yöntemlerden en popüler olanıdır. Yöntemin algoritması şu şekildedir:

Adım 1: Başlangıç değişken atamalarını yap.

Parametreler: θ_t=0.1rand()

Simetrik pozitif tanımlı yaklaşım matrisi: $H_{t=0}$=I

Yakınsama parametresi: $\varepsilon=10^{-4}$

İterasyon sayısı: t=0

Yakınsama büyüklüğü: $c_{t=0} = \nabla f(\theta_{t=0})$

Adım 2: $\|c_t\| < \varepsilon$ ise algoritmayı sonlandır.
Adım 3: İlerleme yönünü: $d_t = -c_t {H_t}^{-1}$
Adım 4: Parametreleri güncelle: $\theta_{t+1} = \theta_t + \alpha d_t$
Adım 5: Hessian matrisini güncelle: $H_{t+1} = H_t + D_t + E_t$
D ve E, doğrulama matrisleri olup Eşitlik.25'de gösterildiği şekilde hesaplanırlar

$$D_t = \frac{y_t {y_t}^T}{(s_t . y_t)}, \quad E_t = \frac{-c_t {c_t}^T}{(c_t . d_t)} \tag{25}$$

Adım 6: Değişimleri hesapla

$$y_t = c_{t+1} - c_t \tag{26}$$

$$s_t = \alpha d_t \tag{27}$$

Adım 7: $t = t + 1$yap ve Adım 2'ye git.

Algoritmadan da anlaşıldığı üzere, ilk adımda Hessian yaklaşım matrisi birim matris (I) olduğundan dolayı, ilk iterasyon Türev Azaltma Yöntemi ile aynıdır. Sonraki iterasyonlarda bu matrisin değeri değişeceğinden dolayı daha hızlı bir yakınsama davranışı gösterir.

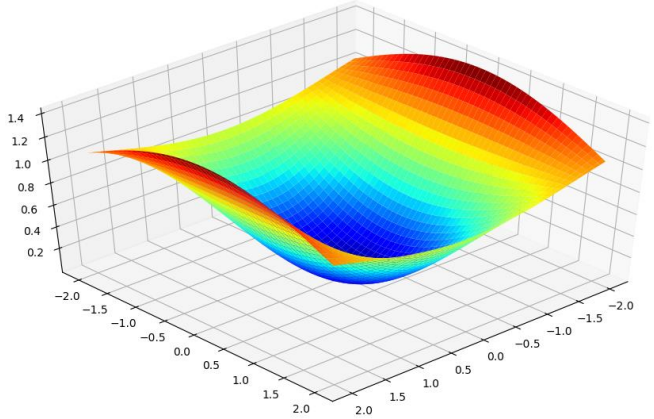
3. Bulgular

Yaklaşımların optimum noktaya yakınsama karakteristiklerini gözlemlemek amacıyla iki ayrı test fonksiyonu (Griewank ve Branin [26]) kullanılmıştır. Uygulama Python dili ile kodlanmış olup, her bir yöntem ile ayrı ayrı güncellenen parametreler, fonksiyon yüzey grafiği üzerinde gösterilerek bir animasyon oluşturulmuş, oluşturulan animasyon sayesinde yöntemlerin farklı başlangıç noktalarından optimum noktaya ilerleme hızları görsel olarak da gözlemlenmiştir.

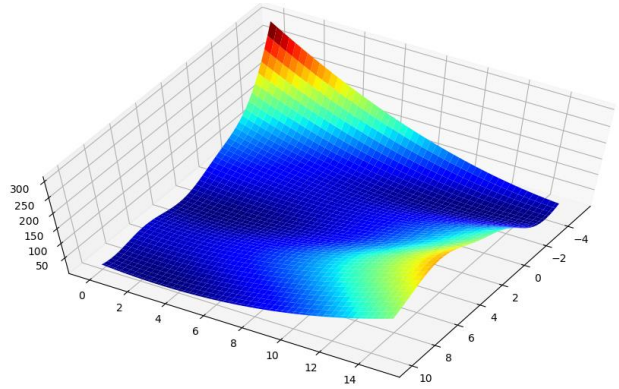
Eşitlik.28'de matematiksel denklemi verilen Griewank fonksiyonu; iki parametre (d=2) için $x_{opt} = (0,0)$ noktasında $f(x_{opt}) = 0$ olacak şekilde tek bir global minimum değere sahiptir.

$$f(x) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{28}$$

Fonksiyonun [-2, 2] aralığında çizilen yüzey grafiği Şekil 1'de gösterilmiştir.



Şekil 1. Griewank fonksiyonu yüzey grafiği



Şekil 2. Branin fonksiyonu yüzey grafiği

$$f(x) = a(x_2 - bx_1^2 + cx_1 - r)^2 + s(1-t)\cos(x_1) + s \tag{29}$$

Eşitlik.29'da yer alan Branin fonksiyonunda a,b,c,r,s ve t sabitler olup değerleri şu şekildedir: a= 1, b=5.1/(4π^2), c = 5/π, r =6, s = 10 ve t = 1/(8 π). Branin fonksiyonu iki parametre için $f(x_{opt}) = 0.397887$ global minimum değerine sahiptir. Bu glonal minimum değere üç farklı noktada erişilir. Bu noktalardaki parametrelerin değerleri $x_{opt} = (-\pi, 12.275), (\pi, 2.275), (9.4278, 2.475)$ şeklindedir. Branin fonksiyonunun $x_1 \in [-5,10]$, $x_2 \in [0,15]$ aralıklarında elde edilen yüzey grafiği Şekil 2'de görülmektedir. Yöntemler ilk olarak Griewank fonksiyonu üzerinde farklı başlangıç noktaları için ayrı ayrı test edilmiş ve belirli bir yakınsama değerine göre (ε) elde edilen sonuçlar Tablo 1 ve Tablo 2'de gösterilmiştir.

Tablo 1'den de görüldüğü üzere Adagrad, Adadelta, RMS ve Adam yaklaşımları genel olarak en başarılı sonuçları veren yöntemler olmuşlardır. Momentum yöntemi [1.8, 1.8] başlangıç noktasında diğer başlangıç noktalarına göre daha iyi bir performans sergilemiştir. BFGS yaklaşımı SGD'den daha hızlı yakınsamış olsa da yakın değerler elde edilmiştir. Newton yöntemi ise [0.5, 0.5] başlangıç noktasında hem hız hem de optimum değere yakınlık konusunda çok başarılı bir performans göstermiş fakat optimum noktadan uzaklaştıkça yakınsama özelliği gösterememiş bu nedenle bir müddet sonra iterasyon durdurulmuştur.

Aynı uygulama yakınsama değeri $\varepsilon = 10^{-6}$ olacak şekilde küçültülerek yeniden çalıştırılmış ve optimum noktaya belirli yakınlıkta yöntemlerin davranışları gözlemlenmek istenmiştir. Bu uygulama sonucunda [0.5, 0.5] başlangıç noktasında Newton yöntemi yine en hızlı yöntem olmuş Adadelta ve RMS yöntemleri sırayla ikinci ve üçüncü olmuşlardır. Uygulama [0.8, 0.8] ve [1, 1] başlangıç noktalarından çalıştırıldığında Adagrad ve Adadelta sıralamada yer değiştirmiş Adadelta yöntemi daha hızlı yakınsama özelliği göstermiştir. Bunun haricinde genel olarak yöntemlerin ilk yakınsama değerine göre aynı hız sıralamasında yakınsadıkları gözlemlenmiştir.

Tablo 1.Griewank fonksiyonu için ε = 0.001 yakınsama değerine göre yöntemlerin yakınsadığı optimum değerler.

	x_1	x_2			x_1	x_2			x_1	x_2			x_1	x_2	
Başlangıç Noktası	0.5	0.5			0.8	0.8			1	1			1.8	1.8	
	Optimum Değer				**Optimum Değer**				**Optimum Değer**				**Optimum Değer**		
Yaklaşım	**x1opt**	**x2opt**	**İtr.**		**x1opt**	**x2opt**	**İtr.**		**x1**	**x2**	**İtr.**		**x1opt**	**x2opt**	**İtr.**
SGD	0.060	0.178	21		0.039	0.185	31		0.031	0.185	37		0.017	0.190	76
Momentum	0.076	0.144	23		-0.067	0.136	34		-0.056	0.168	35		0.033	0.083	45
Adadelta	0.033	0.164	2		0.011	0.102	4		0.034	0.172	4		2.79e-4	0.122	10
Adagrad	-0.099	-0.109	3		0.101	0.093	2		0.011	0.072	1		0.034	0.130	6
Adam	-0.069	-0.149	14		0.115	-0.012	24		0.120	-0.021	25		0.115	-0.062	37
RMS	0.060	0.176	2		0.055	0.152	4		0.068	0.168	5		0.006	0.161	14
Newton	0.004	0.006	2		-1.572	-2.222	80		4.717	6.667	80		1.572	2.222	80
BFGS	0.059	0.178	20		0.040	0.190	28		0.031	0.189	33		0.017	0.191	70

Tablo 2.Griewank fonksiyonu için yaklaşımların yakınsadığı optimum noktalardaki fonksiyon değeri (ε = 0.001)

	x_1	x_2			x_1	x_2			x_1	x_2			x_1	x_2	
Başlangıç Noktası	0.5	0.5			0.8	0.8			1	1			1.8	1.8	
	Optimum Değer				**Optimum Değer**				**Optimum Değer**				**Optimum Değer**		
Yaklaşım	F_{opt}		**İtr.**		F_{opt}		**İtr.**		F_{opt}		**İtr.**		F_{opt}		**İtr.**
SGD	0.0097		21		0.0093		31		0.0090		37		0.0092		76
Momentum	0.0080		23		0.0069		34		0.0086		35		0.0023		45
Adadelta	0.0072		2		0.0027		4		0.0080		4		0.0038		10
Adagrad	0.0078		3		0.0073		2		0.0013		1		0.0048		6
Adam	0.0079		14		0.0067		24		0073		25		0.0076		37
RMS	0.0095		2		0.0073		4		0.0094		5		0.0065		14
Newton	1.7013e-05		2		1.0019		80		1.0167		80		1.0019		80
BFGS	0.0096		20		0.0098		28		0.0094		33		0.0093		70

Yaklaşımlar, Branin fonksiyonu için $\varepsilon = 10^{-6}$ yakınsama değeri ile çalıştırılmış ve elde edilen sonuçlar Tablo 3 ve Tablo 4'te gösterilmiştir. Tablo 3 incelendiğinde yöntemlerin aynı noktalardan başlamasına rağmen farklı optimum noktalara yakınsadığı görülmektedir.

Tablo 3.Branin fonksiyonu için ε = 0.000001 yakınsama değerine göre yöntemlerin yakınsadığı optimum değerler.

	x_1	x_2			x_1	x_2			x_1	x_2			x_1	x_2	
Başlangıç Noktası	-3	3			-4	1			6	6			-4	12	
	Optimum Değer				**Optimum Değer**				**Optimum Değer**				**Optimum Değer**		
Yaklaşım	**x1opt**	**x2opt**	**İtr.**		**x1opt**	**x2opt**	**İtr.**		**x1opt**	**x2opt**	**İtr.**		**x1opt**	**x2opt**	**İtr.**
SGD	3.141	2.276	45		3.141	2.276	40		3.141	2.276	42		-1.899	12.061	250
Momentum	nan	nan	250		nan	nan	250		9.420	2.470	250		15.714	12.885	250
Adadelta	-3.141	12.274	131		-3.141	12.274	158		3.141	2.276	59		-3.142	12.276	43
Adagrad	-3.139	12.262	250		-3.084	12.037	250		3.141	2.276	65		-3.142	12.276	42
Adam	3.142	2.276	105		3.141	2.276	142		3.141	2.275	117		-3.142	12.275	120
RMS	-3.183	12.234	250		-3.116	12.298	250		3.141	2.276	82		-3.142	12.276	49
Newton	-3.142	12.275	2		-3.141	12.275	3		6.283	1.1	250		-3.142	12.275	3
BFGS	3.141	2.276	56		3.141	2.276	48		3.141	2.276	56		-1.888	12.047	250

Tablo 4.Branin fonksiyonu için yaklaşımların yakınsadığı optimum noktalardaki fonksiyon değeri (ε=0.000001)

	x_1	x_2			x_1	x_2			x_1	x_2			x_1	x_2	
Başlangıç Noktası	-3	3			-4	1			6	6			-4	12	
	Optimum Değer				**Optimum Değer**				**Optimum Değer**				**Optimum Değer**		
Yaklaşım	**Fopt**		**İtr.**		**Fopt**		**İtr.**		**Fopt**		**İtr.**		**Fopt**		**İtr.**
SGD	0.397889		45		0.397889		40		0.397889		42		13.524034		250
Momentum	nan		250		nan		250		0.397997		250		0.398086		250
Adadelta	0.397889		131		0.397889		158		0.397889		59		0.397888		43
Adagrad	0.397965		250		0.423815		250		0.397889		65		0.397888		42
Adam	0.397889		105		0.397889		142		0.397889		117		0.397889		120
RMS	0.425923		250		0.408158		250		0.397889		82		0.397888		49
Newton	0.397889		2		0.397891		3		19.60211		250		0.397889		3
BFGS	0.397889		56		0.397889		48		0.397889		56		13.670019		250

Newton yaklaşımı uygun noktalardan başlatıldığında optimum noktaya çok hızlı bir şekilde yakınsamıştır. [6, 6] noktasından başlatıldığında ise optimum noktaya yakınsama özelliği gösterememiştir. Adagrad ve RMS yöntemleri [-3, 3] ve [-4, 1] noktalarından başlatıldığında iyi bir yakınsama gösterememiş, Momentum yöntemi ise sadece [6, 6] noktasından başlatıldığında optimum noktaya yakınsayabilmiştir. Adadelta yöntemi genel olarak tüm başlangıç noktalarından optimum noktaya yakınsamıştır. BFGS yöntemi [-4, 12] haricindeki diğer noktalardan başlatıldığında optimum noktaya yakınsamıştır.

Deneysel çalışmalarda dikkat çeken bir nokta şudur ki; Branin kullanıldığında online kullanımda (SGD) Newton haricinde diğer yöntemlere göre beklenenin aksine daha hızlı bir yakınsama gözlemlenmiştir.

4. Tartışma ve Sonuç

Bu çalışmada türev tabanlı sekiz farklı optimal noktaya yakınsama yaklaşımının davranışı belirli fonksiyonlar kullanılarak incelenmiştir. Yaklaşımların farklı geometrik yüzeylerdeki yakınsama kabiliyetlerinin daha iyi anlaşılabilmesi için fonksiyonların yüzey grafikleri çizilmiş ve yaklaşım noktaları anlık grafiğe eklenerek bir animasyon oluşturulmuştur. Böylece yakınsama kabiliyetleri açık bir şekilde incelenebilmiştir. Yapılan deneysel çalışmalar neticesinde yakınsama performanslarında başlangıç noktasının önemli bir etken olduğu ve daha fazla iterasyonda yakınsama gösteren bir yöntemin az sayıda iterasyon sonucunda yakınsayan yöntemlere göre optimum noktaya daha iyi yakınsayabildiği gözlemlenmiştir. Ayrıca birden fazla optimum noktası olan Branin fonksiyonunda yöntemlerin aynı noktadan başlamalarına rağmen farklı optimum noktalara yakınsadığı gözlemlenmiştir.

Kaynakça

[1] Rao, S. S. 2009. Engineering optimization: theory and practice, John Wiley & Sons, 2009

[2] Buber, E., Şahingoz, O. K. 2017. Makine Öğrenmesi sistemi ile görüntü İşleme ve en uygun parametrelerin ayarlanması, Artificial Intelligence and Data Processing Symposium (IDAP), 16-17 Eylül, Malatya,16-17.

[3] Williams, R. J., Peng, J. 1990. An efficient gradient-based algorithm for on-line training of recurrent network trajectories, Neural computation 2(4), 490-501.

[4] Møller, M. F. 1993. A scaled conjugate gradient algorithm for fast supervised learning. Neural networks 6(4), 525-533.

[5] Çapar, A,, Gökmen, M. 2011. Gradyan temelli şekil bölütleme ve tanıma, ITU Journal Series D: Engineering, 10(3), 15-26.

[6] ALTUN, A. A. 2007. Esnek hesaplama yöntemleri ile otomatik parmakizi tanıma. Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Konya

[7] Çevik, K. K. 2010. Yapay zeka yöntemleri ile araç plaka tanıma sistemi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, Konya.

[8] Klein, S., Staring, M., Pluim, J.P.W. 2007. Evaluation of optimization methods for nonrigid medical image registration using mutual information and B-splines, IEEE transactions on image processing 16(12), 2879-2890

[9] Staib, L. H., Duncan, J. S. 1996. Model-based deformable surface finding for medical images, IEEE transactions on medical imaging 15(5), 720-731

[10] Chakraborty, A., Staib, L. H., Duncan, J. S. 1996. Deformable boundary finding in medical images by integrating gradient and region information, IEEE Transactions on Medical Imaging, 15(6), 859-870

[11] Akyılmaz,E., Demirkesen, C., Nar, F., Okman, E., Çetin, M. 2013. Interactive ship segmentation in SAR images, Signal Processing and Communications Applications Conference (SIU), 24-26 Nisan, Kıbrıs, 1-4

[12] Karaboğa, D. 2014. Yapay Zeka Optimizasyon Algoritmaları, Nobel Akademi Yayıncılık, 225s.

[13] Cauchy, A. 1847. Méthode générale pour la résolution des systemes d'équations simultanée, Comp. Rend. Sci. 25, 536-538

[14] Yazan, E., Talu, M. F. 2017. Comparison of the stochastic gradient descent based optimization techniques. Artificial Intelligence and Data Processing Symposium (IDAP), 16-17 Eylül, Malatya

[15] Qian, N. 1999. On the momentum term in gradient descent learning algorithms. Neural Networks, The Official Journal of the International Neural Network Society, 12(1), 145–151

[16] Duchi, J., Hazan, E., Singer, Y. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research, 12, 2121–2159

[17] Zeiler, M.D. 2012. ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701

[18] Tijmen, T., Hinton, G. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 4(2).

[19] Kingma, D. P., Ba, J. L. 2015. Adam: a Method for Stochastic Optimization. International Conference on Learning Representations,7-9 Mayıs, San Diego, 1-13

[20] Greenstadt, J. 1967. On the relative efficiencies of gradient methods, Mathematics of Computation, 21(99), 360-367

[21] Broyden, C. G. 1965. A class of methods for solving nonlinear simultaneous equations, Mathematics of computation 19(92), 577-593

[22] Broyden, C. G. Quasi-Newton methods and their application to function minimisation, Mathematics of computation, 21, 368-381

[23] Barnes, J. G. P. 1965. An algorithm for solving non-linear equations based on the secant method, The Computer Journal 8(1), 66-72

[24] D. Goldfarb, 1970. A family of variable-metric methods derived by variational means, Mathematics of computation, 24(109), 23-26.

[25] Nocedal , J., Wright, S. J. 2006. Sequential quadratic programming. Springer New York

[26] Optimization Test Functions. https://www.sfu.ca/~ssurjano/optimization.html (Erişim Tarihi: 14.01.2022)