GELİŞTİRİLMİŞ YAPAY SİNİR AĞI ALGORİTMALARI VE UYGULAMALARI

Engin TAŞ

Doktora Tezi

İstatistik Anabilim Dalı

Haziran - 2008

JÜRİ VE ENSTİTÜ ONAYI

Engin Taş'ın "**Geliştirilmiş Yapay Sinir Ağı Algoritmaları ve Uygulamaları**" başlıklı **İstatistik Anabilim Dalındaki**, Doktora tezi.....tarihinde, aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

	Adı-Soyadı	**İmza**
Üye (Tez Danışmanı)	**: Prof. Dr. Memmedağa MEMMEDLİ**	**……**
Üye	**: Prof. Dr. Embiya AĞAOĞLU**	**……**
Üye	**: Prof. Dr. Vakıf CAFER**	**……**
Üye	**: Prof. Dr. Ayşen APAYDIN**	**……**
Üye	**: Doç. Dr. Yusuf OYSAL**	**……**

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …… tarih ve …… sayılı kararıyla onaylanmıştır.

Enstitü Müdürü

ÖZET

Doktora Tezi

GELİŞTİRİLMİŞ
YAPAY SİNİR AĞI ALGORİTMALARI
VE UYGULAMALARI

Engin TAŞ

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
İstatistik Anabilim Dalı

Danışman: Prof. Dr. Memmedağa MEMMEDLİ
2008, 103 sayfa

Bu tez çalışmasında, ilk önce deterministik kuadratik performans fonksiyonu durumunda momentumlu gradyan düşümü algoritmasının kararlılığı ve yakınsama hızı incelenmiştir. Teorik incelemeler sonucunda, hızlı yakınsamayı sağlayan etkin öğrenme oranı ve momentum faktörü formülleri, Hessian'ın en büyük ve en küçük özdeğerlerini kullanarak belirlenmiştir. Bu yaklaşım rassal olarak oluşturulan kuadratik test problemleri üzerinde denenmiş ve etkin öğrenme parametreleriyle çalışan algoritmanın diğer geleneksel momentumlu gradyan düşümü algoritmalarından daha üstün performans gösterdiği gözlenmiştir. Kuadratik performans fonksiyonu için elde edilen etkin öğrenme parametreleri, hatanın, ağ ağırlıklarının herhangi bir doğrusal olmayan fonksiyonu olduğu genel duruma uyarlanmıştır. Genel durumda etkin parametrelerle çalışacak momentumlu gradyan düşümü algoritmasının dört farklı versiyonu önerilmiştir. Geliştirilen algoritmalar gerçek veri kümesine sahip güncel test problemleri üzerinde diğer geleneksel gradyan düşümü algoritmaları ile karşılaştırılmıştır. Etkin öğrenme parametreli algoritmaların, diğer geleneksel gradyan düşümü algoritmalara göre genelde daha iyi yakınsama performansı gösterdiği gözlenmiştir.

Anahtar Kelimeler: Geri-Yayılım, Momentum, Gradyan Düşümü , Kararlılık, Yakınsama Hızı, Hessian

ABSTRACT

PhD Dissertation

IMPROVED ARTIFICIAL NEURAL NETWORK ALGORITHMS AND APPLICATIONS

Engin TAŞ

Anadolu University
Graduate School of Sciences
Statistics Program

Supervisor: Prof. Dr. Memmedağa MEMMEDLİ
2008, 103 pages

In this thesis, at first, stability and convergence speed of gradient descent with momentum algorithm is analyzed in the case of a deterministic quadratic performance function. As a consequence of theoretical analyzes, effective learning rate and momentum factor formulas which improve convergence speed are determined from the largest and smallest eigenvalue of the Hessian. This approach is tested on randomly generated test problems and results indicate that the algorithm with effective learning parameters outperforms other conventional gradient descent with momentum algorithms. Effective learning parameters obtained for the quadratic performance function are adapted to the general case where the performance is any nonlinear function of the network weights. Four different versions of gradient descent with momentum algorithm which works compatible with the effective learning parameters are proposed in the general case. Developed algorithms are compared with other conventional gradient descent algorithms on the up-to-date test problems. It is observed that algorithms with the effective learning parameters show better convergence performance than the other conventional gradient descent algorithms in general.

Keywords: Backpropagation, Momentum, Gradient Descent, Stability, Convergence Speed, Hessian

TEŞEKKÜR

Kendisinden her konuda çok şey öğrendiğim değerli ve sevgili danışman hocam sayın Prof. Dr. Memmedağa MEMMEDLİ'ye çok teşekkür ederim.

Çalışmam süresince desteklerini esirgemeyen Prof. Dr. Embiya AĞAOĞLU'na, İstatistik Bölümündeki değerli Hocalarıma ve Araştırma Görevlisi arkadaşlarıma çok teşekkür ederim.

Hayatım boyunca her zaman yanımda olan, desteklerini esirgemeyen sevgili ağabeylerime, ablama ve beni yetiştiren anneme ve babama derin sevgi ve saygılarımla…

Engin TAŞ

Haziran 2008

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

1. GİRİŞ

Geri-yayılım (Backpropagation-BP) algoritması yapay sinir ağlarında oldukça sık kullanılan popüler bir öğrenme algoritmasıdır, çünkü kavramsal olarak basittir, hesapsal olarak verimlidir ve genelde başarılı bir şekilde çalışır. Bununla birlikte, bir sinir ağını tasarlarken ve geri-yayılım algoritmasıyla eğitirken gizli katman sayısı, gizli katmandaki nöron sayısı, transfer fonksiyonu, eğitim ve test veri kümesi, öğrenme oranı ve momentum faktörü gibi keyfi seçimlerin yapılması gerekir. Bu seçimler kritik olabilir. Bu seçimlerin yapılabilmesi için kesin bir yöntem veya teori yoktur.

Bu nedenle yapay sinir ağları üzerindeki en yoğun çalışmalar bu ağların eğitilmesi konusunda olmaktadır. Zira etkin bir algoritma ile eğitilebilen, yani öğrenebilen ağlar, yeni şekilleri tanıyabilmekte (şekil tanıma) veya verilen bir girdinin hangi sınıfa ait olduğuna karar verebilmektedir (sınıflandırma).

İleri beslemeli yapay sinir ağlarının eğitimi için geri-yayılım (BP) algoritması zor problemlerde bile güçlü olduğunu göstermiştir. Ama, BP' nin yüksek performansına, ağ parametrelerini düzenlemek için gerekli olan uzun eğitim zamanı maliyetinde erişilir ki bu gerçek uygulamalarda caydırıcı olabilir. Oldukça basit problemlerde bile, standart BP sıkça eğitim örneklerinin tamamının yüzlerce veya binlerce kez işlendiği uzun eğitim süreci gerektirir. Bu süreci hızlandırmak için literatürde birçok çalışma yapılmıştır.

Daha hızlı algoritmalar üzerindeki çalışmalar kaba olarak iki kategoriye ayrılır. Birinci kategori sezgisel tekniklerin geliştirilmesini içerir. Sezgisel teknikler öğrenme oranının değişimi, momentum kullanımı ve değişkenlerin yeniden ölçeklendirilmesi gibi tekniklerdir (Vogl ve ark. 1988; Jacobs 1988; Tollenaere 1990; Rigler ve ark. 1990, Brent 1973). İkinci kategori ise standart sayısal optimizasyon teknikleri üzerinde yoğunlaşmıştır (Shanno 1990; Barnard 1992; Battiti 1992; Charalambous 1992).

Geri-yayılım, sinir ağının ağırlıklarının fonksiyonu olan hata kareler toplamının minimumunun bulunmasına yönelik gradyan düşümü optimizasyon metodudur. Momentumlu geri-yayılım algoritmasının (BPM) kararlılığı ve yakınsama hızı gibi önemli özellikleri, öğrenme ve momentum katsayılarının

seçimine bağlı olarak değişim göstermektedir. Momentum katsayısının gradyan düşüm algoritmasına dahil edilmesi, algoritmanın yakınsama özelliklerini geliştirir, ama bunun yanında seçilmesi gereken yeni bir parametre ortaya çıkarır. Bu parametrelerin seçimi problemi de kendi başına bir optimizasyon problemidir. Bu konuda literatürde değişik yaklaşımlar bulunmaktadır.

Jacobs (1988) öğrenme oranı ve momentum faktörünün güncellenmesi için basit bir metot olan *delta-bar-delta* kuralını önermiştir. Bu çalışmada, hata sinyallerinin gözlemlerine dayanarak öğrenme oranı ve momentum faktörünün dinamik olarak arttırılması veya azaltılması önerilmiştir. Jacobs'un metodunun varyasyonları Vogel ve ark. (1988) (*bold-driver* tekniği) ve Battiti (1989) tarafından rapor edilmiş, Allred ve Kelly (1990) tarafından ayrıntılı olarak karşılaştırılmıştır.

Başka bir sezgisel yaklaşım Fahlman (1988) tarafından *quickprop* adıyla verilmiştir. Buradaki temel fikir hata yüzeyini her bir ağırlığın bir fonksiyonu olarak kuadratik bir polinomla (örneğin Parabol ile) yaklaşık olarak tahmin etmek ve daha sonra hata fonksiyonunun iki ardışık değerini ve bunların gradyanlarını değerlendirerek polinomun katsayılarını belirlemektir. İterasyonun sonraki adımında ağırlık parametresi parabolün minimumuna atanır.

Qian (1999), sürekli zaman limitinde momentum parametresinin, tutucu bir kuvvet alanı içinde yapışkan bir ortamda hareket eden Newton partikülleri kümesine benzer olduğunu göstermiştir . Yerel minimum civarında sistemin davranışının, bir eşli ve sönümlü harmonik dalgalanıcılara denk olduğu belirtilmiştir.

Torii ve Hagan (2002), kuadratik performans fonksiyonları için dik iniş eğitiminde momentum faktörünün etkisini analiz etmiştir. Momentum katsayısının değerinin algoritmanın yakınsama özelliklerini nasıl değiştirdiğini göstermiştir.

Bhaya ve Kaszkurewicz (2004), kuadratik hata fonksiyonu için öğrenme oranı ve momentum parametrelerinin özel seçimiyle, momentumlu gradyan düşümü algoritmasının eşlenik gradyan algoritması (Hestenes ve Stiefel 1952, Greenbaum 1997) gibi çalışabileceğini göstermiştir.

Günümüzde incelenen problemlerin boyutları günden güne artmakta ve veri setleri daha da büyük genişliklere ulaşmaktadır. Bu nedenle gradyana dayalı metotların önemi günden güne daha iyi anlaşılmaktadır. Çünkü ikinci düzey bilgiyi de kullanan klasik optimizasyon metotları bu tür büyük ölçekli problemlerde hesaplama ve zaman maliyeti açısından elverişsiz duruma gelir. Momentumlu gradyan düşümü algoritmasının güvenli ve güçlü global yakınsama özelliklerinden yararlanmak için öğrenme parametrelerinin iyi ayarlanmış olması gerekir. Zira bu algoritmanın yakınsama performansı üzerinde kritik bir öneme sahiptir. Bu nedenle bu doktora tez çalışmasında momentumlu gradyan düşümü algoritması için etkin öğrenme parametreleri yaklaşımı sunulmuştur.

İkinci bölümde geri-yayılım algoritması kısaca anlatılmış ve geri-yayılım algoritmasının aslında temelinde bir gradyan düşümü metodu olduğu ifade edilmiş ve geri-yayılım algoritmasını hızlandırmak için önerilen sezgisel yaklaşımlara değinilmiştir.

Üçüncü bölümde *kuadratik hata fonksiyonu* durumunda momentumlu gradyan düşümünün *kararlılığı* önceki çalışmalardan farklı bir yaklaşımla incelenmiş ve sonuç olarak kuadratik hata fonksiyonu için yakınsamayı hızlandıran *etkin öğrenme oranı ve momentum faktörü* belirlenmiştir. Etkin parametreler rassal olarak oluşturulan kuadratik problemlerde denenmiştir.

Dördüncü bölümde kuadratik hata fonksiyonu durumunda elde edilen etkin öğrenme parametreleri hata fonksiyonunun ağın ağırlıklarının herhangi bir fonksiyonu olduğu genel durumda çalışacak şekilde uyarlanmıştır. Etkin öğrenme parametrelerini temelde iki farklı yöntemle belirleyen momentumlu gradyan düşümü algoritmasının *dört farklı versiyonu geliştirilmiştir*.

Beşinci bölümde, geliştirilen algoritmalar ve literatürde popüler olan geleneksel momentumlu gradyan düşümü algoritmaları yapay sinir ağlarında sıkça kullanılan test problemleri üzerinde *karşılaştırılmıştır*.

2. GERİ-YAYILIM VE GRADYAN DÜŞÜMÜ

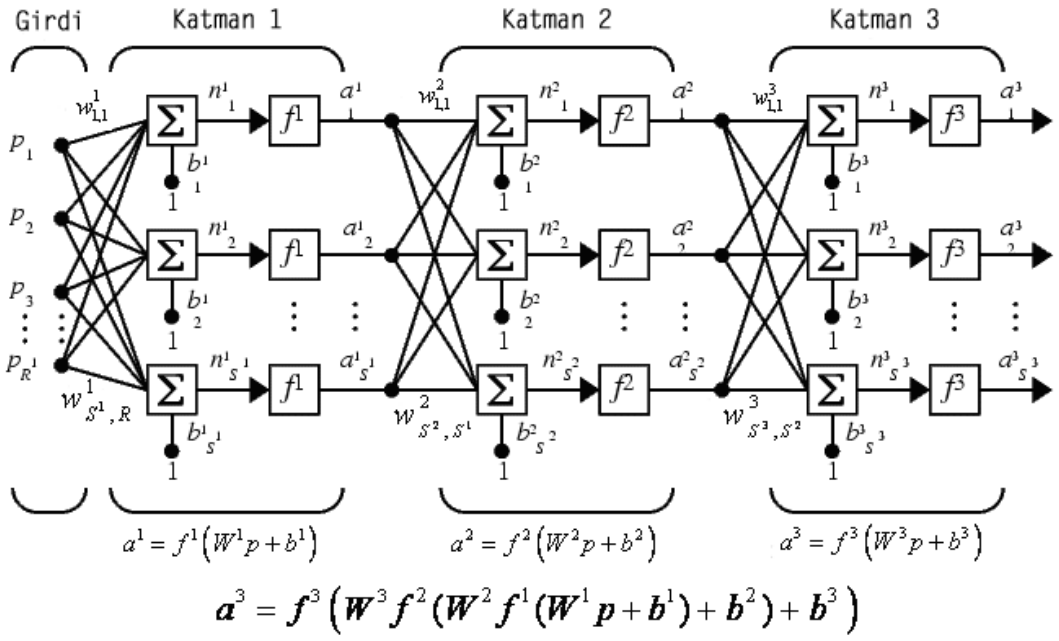
Frank Rosenblatt'ın (1958) Algılayıcı (Perceptron) öğrenme kuralı ve Bernard Widrow ve Marcian Hoff'un (1960) en küçük hata kareler ortalaması (EHKO) algoritması tek-katmanlı algılayıcı-benzeri ağlarını eğitmek için tasarlanmıştı. Bu tek katmanlı ağlar sadece doğrusal olarak ayrılabilen sınıflandırma problemlerini çözebilen kısıtlı ağlardı. Bunun üstesinden gelebilmek için Rosenblatt ve Widrow doğrusal olmayan problemleri de çözebilecek çok katmanlı algılayıcıları önerdiler. Ama algılayıcı için geliştirmiş oldukları eğitim algoritmasını bu daha güçlü ağları eğitmek için genelleştiremediler.

Çok katmanlı ağları eğitebilen bir algoritmanın ilk açıklaması Paul Werbos'un 1974'deki tezinde bulunmaktadır (Werbos 1974). Bu tezde sunulan algoritma genel ağlar içindi. Sinir ağları ise bunun özel bir haliydi ve bu nedenle sinir ağları topluluğunda yayılmadı. 1980'lerin ortalarına kadar geriye yayılım algoritması (backpropagation) bağımsız olarak David Rumelhart, Geoffrey Hinton ve Ronald Williams (Rumelhart ve ark. 1986), David Parker (1985) ve Yann Le Cun (1985) tarafından bulunmuştur. Geri-yayılım algoritması "Parallel Distributed Processing" adlı kitabta yer aldıktan sonra popüler olarak sıkça kullanılmaya başlanmıştır (Rumelhart ve Mc Clelland 1986). Bu kitabın yayınından sonra sinir ağları alanında bir araştırma patlaması yaşanmıştır. Geri-yayılım algoritmasıyla eğitilen çok katmanlı algılayıcı şu an en çok kullanılan yapay sinir ağıdır.

2.1 Çok Katmanlı Algılayıcı

Çok katmanlı bir algılayıcı Şekil 2.1'deki gibi gösterilebilir. Her katman kendi W ağırlık matrisine, kendi b sapma vektörüne, bir net girdi vektörü n'ye ve bir çıktı vektörü a'ya sahiptir. Katmanları belirlemek için üst indis kullanılmıştır. Yukarıda tanımlanan herhangi bir değişkenin hangi katmana ait olduğu üst indisinde katman numarası ile gösterilmiştir. Böylece, birinci katmanın ağırlık matrisi W^1 olarak, ikinci katmanın ağırlık matrisi W^2 olarak ve son

katmanın ağırlık matrisi W^3 olarak gösterilmiştir. Bu ağ üç katmanlı bir ağdır. Son katman çıktı katmanı olarak adlandırılır. Diğer katmanlar ise gizli katmanlardır. Bu notasyon üç katmanlı bir ağ için Şekil 2.1'de kullanılmıştır.



Şekil 2.1 Üç katmanlı ağ.

Şekil 2.1'den görüleceği üzere birinci katmanda R girdi, S^1 nöron, ikinci katmanda S^2 nöron bulunur vs. Farklı katmanlar farklı sayıda nöron içerebilir. Birinci ve ikinci katmanların çıktısı, ikinci ve üçüncü katmanların girdisi olur. Bu nedenle, ikinci katman $R = S^1$ girdili, $S = S^2$ nöronlu ve $S^1 \times S^2$ boyutlu bir ağırlık matrisine sahip tek katmanlı bir ağ olarak görülebilir. İkinci katmanın girdisi a^1, çıktısı ise a^2'dir. Çıktısı ağın çıktısı olan katman çıktı katmanı olarak adlandırılır. Diğer katmanlar gizli katmanlar olarak adlandırılır. Şekil 2.1'de gösterilen ağ bir çıktı katmanına (katman 3) ve iki gizli katmana sahiptir (katman 1 ve katman 2) (Hagan ve ark. 1996).

2.2 Geri-Yayılım Algoritması

Geri-yayılım algoritmasını geliştirirken Şekil 2.1'in kompakt bir halini kullanmak daha kolay olacaktır (Şekil 2.2). Çok katmanlı ağlarda, bir katmanın çıktısı takip eden katmanın girdisi haline gelir. Bu işlemi ifade eden denklemler

$$a^{m+1} = f^{m+1}\left(W^{m+1}a^m + b^{m+1}\right), \qquad m = 0,1,...,M-1 \tag{2.1}$$

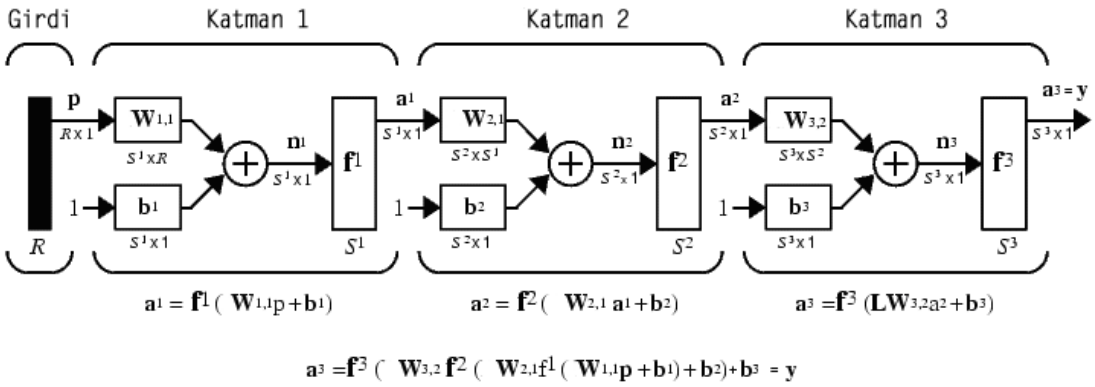
biçimindedir. Burada M ağdaki katmanların sayısıdır. İlk katmandaki nöronlar dış girdi sinyallerini alırlar:

$$a^0 = p \tag{2.2}$$

eşitliği (2.1)'in başlangıç noktasını verir. Son katmandaki nöronların çıktıları

$$a = a^M \tag{2.3}$$

biçiminde ağın çıktıları olur.



Şekil 2.2 Kompakt şekilde üç katmanlı bir ağ (Hagan ve ark. 1996).

2.2.1 Performans İndeksi

Çok katmanlı ağlardaki geri-yayılım algoritması performans indeksi olarak hata kareler ortalamasını kullanır. Algoritmaya

$$\{p_1,t_1\},\{p_2,t_2\},...,\{p_Q,t_Q\}, \tag{2.4}$$

örnek çiftlerinden oluşan bir veri kümesi sunulur. Burada p_q, $q=1,...,Q$ ağa verilen bir girdi, t_q ise karşılık gelen hedeftir. Her girdi ağa uygulandığında, ağın çıktısı hedef ile karşılaştırılır. Algoritma ağın parametrelerini, hata kareler ortalaması

$$F(x)=E[e^2]=E[(t-a)^2] \tag{2.4}$$

minimize edecek şekilde düzenlenmelidir, burada $E[\cdot]$ beklenen değeri gösterir ve tüm girdi/hedef kümeleri üzerinden alınır, x ağın ağırlıkları ve sapmalarının vektörüdür. Eğer ağ çoklu çıktıya sahipse (2.4)

$$F(x)=E[e^Te]=E[(t-a)^T(t-a)] \tag{2.5}$$

ifadesine genelleşir. En küçük hata kareler ortalamasında olduğu gibi hata kareler ortalaması

$$\hat{F}(x)=\big(t(k)-a(k)\big)^T\big(t(k)-a(k)\big)=e^T(k)e(k) \tag{2.6}$$

şeklinde yaklaşık olarak hesaplanır, burada hata karenin beklenen değeri, k iterasyonundaki hata kareyle değiştirilmiştir. Yaklaşık hata kareler ortalaması için gradyan düşümü algoritması

$$w_{i,j}^{m}(k+1) = w_{i,j}^{m}(k) - \eta \frac{\partial \hat{F}}{\partial w_{i,j}^{m}},$$
$$b_{i}^{m}(k+1) = b_{i}^{m}(k) - \eta \frac{\partial \hat{F}}{\partial b_{i}^{m}} \tag{2.7}$$

şeklinde verilebilir. Burada η öğrenme oranıdır (Hagan ve ark. 1996).

2.2.2 Zincir Kuralı

Tek katmanlı bir ağda (2.7) kısmi türevlerini hesaplamak oldukça kolaydır. Çok katmanlı bir ağda ise hata gizli katmanlardaki ağırlıkların açık bir fonksiyonu değildir, bu nedenle bu türevler kolay bir şekilde hesaplanamaz. Hata, gizli katmanlardaki ağırlıkların karmaşık bir fonksiyonu olduğundan, türevleri hesaplamak için analizdeki zincir kuralı kullanılır:

$$\frac{df(n(w))}{dw} = \frac{df(n)}{dn} \times \frac{dn(w)}{dw}. \tag{2.8}$$

Denklem (2.7)'deki türevleri bulmak için zincir kuralı kullanılırsa

$$\frac{\partial \hat{F}}{\partial w_{i,j}^{m}} = \frac{\partial \hat{F}}{\partial n_{i}^{m}} \times \frac{\partial n_{i}^{m}}{\partial w_{i,j}^{m}}, \tag{2.9}$$

$$\frac{\partial \hat{F}}{\partial b_{i}^{m}} = \frac{\partial \hat{F}}{\partial n_{i}^{m}} \times \frac{\partial n_{i}^{m}}{\partial b_{i}^{m}} \tag{2.10}$$

elde edilir. Bu denklemlerdeki ikinci terim kolay bir şekilde hesaplanabilir, çünkü m katmanına gelen net girdi, o katmandaki ağırlıkların ve sapmanın açık bir fonksiyonudur:

$$n_i^m = \sum_{j=1}^{s^{m-1}} w_{i,j}^m a_j^{m-1} + b_i^m. \tag{2.11}$$

Buradan

$$\frac{\partial n_i^m}{\partial w_{i,j}^m} = a_j^{m-1}, \qquad \frac{\partial n_i^m}{\partial b_i^m} = 1 \tag{2.12}$$

elde edilir.

$$s_i^m \equiv \frac{\partial \hat{F}}{\partial n_i^m} \tag{2.13}$$

tanımlaması yapılırsa ($\hat{F}$'nın m katmanındaki net girdinin i. elemanındaki değişime *duyarlılığı*), (2.9) ve (2.10)

$$\frac{\partial \hat{F}}{\partial w_{i,j}^m} = s_i^m a_j^{m-1}, \tag{2.14}$$

$$\frac{\partial \hat{F}}{\partial b_i^m} = s_i^m \tag{2.15}$$

ifadelerine sadeleşir. Şimdi yaklaşık gradyan düşümü algoritması

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \eta s_i^m a_j^{m-1}, \tag{2.16}$$

$$b_i^m(k+1) = b_i^m(k) - \eta s_i^m \tag{2.17}$$

Biçiminde verilebilir. Matris formunda:

$$W^m(k+1) = W^m(k) - \eta s^m (a^{m-1})^T , \tag{2.18}$$

$$b^m(k+1) = b^m(k) - \eta s^m \tag{2.19}$$

olarak tanımlanır. Burada

$$s^m \equiv \frac{\partial \hat{F}}{\partial n^m} = \begin{bmatrix} \frac{\partial \hat{F}}{\partial n_1^m} \\ \frac{\partial \hat{F}}{\partial n_2^m} \\ . \\ . \\ . \\ \frac{\partial \hat{F}}{\partial n_{s^m}^m} \end{bmatrix}. \tag{2.20}$$

2.2.3 Duyarlılıkların Geri-Yayılımı

s^m duyarlılıklarını hesaplamak, zincir kuralının bir başka uygulamasını gerektirir. Algoritmaya geri-yayılım ismini veren bu süreçtir, çünkü m katmanındaki duyarlılığın $m+1$ katmanındaki duyarlılıktan hesaplanmasını sağlayan bir yineleme ilişkisini ifade eder. Duyarlılıklar için yineleme ilişkisini türetmek için Jakobiyen matrisi kullanılır:

$$\frac{\partial n^{m+1}}{\partial n^m} \equiv \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdot & \cdot & \cdot & \frac{\partial n_1^{m+1}}{\partial n_{s^m}^m} \\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdot & \cdot & \cdot & \frac{\partial n_2^{m+1}}{\partial n_{s^m}^m} \\ \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & & & & \cdot \\ \cdot & \cdot & & & & \cdot \\ \frac{\partial n_{s^{m+1}}^{m+1}}{\partial n_1^m} & \frac{\partial n_{s^{m+1}}^{m+1}}{\partial n_2^m} & \cdot & \cdot & \cdot & \frac{\partial n_{s^{m+1}}^{m+1}}{\partial n_{s^m}^m} \end{bmatrix}. \tag{2.21}$$

Bu matrisin i, j elemanı ele alınsın:

$$\begin{aligned} \frac{\partial n_i^{m+1}}{\partial n_j^m} &= \frac{\partial \left(\sum_{l=1}^{m} w_{i,l}^{m+1} a_l^m + b_i^{m+1} \right)}{\partial n_j^m} = w_{i,j}^{m+1} \frac{\partial a_j^m}{\partial n_j^m} \\ &= w_{i,j}^{m+1} \frac{\partial f^m(n_j^m)}{\partial n_j^m} = w_{i,j}^{m+1} \dot{f}^m(n_j^m), \end{aligned} \tag{2.22}$$

burada

$$\dot{f}^m(n_j^m) = \frac{\partial f^m(n_j^m)}{\partial n_j^m}. \tag{2.23}$$

Bu nedenle Jakobiyen matrisi

$$\frac{\partial n^{m+1}}{\partial n^m} = W^{m+1} \dot{F}^m(n^m), \tag{2.24}$$

biçiminde yazılabilir.

$$\dot{F}^m(n^m) = \begin{bmatrix} \dot{f}^m(n_1^m) & 0 & \cdots & 0 \\ 0 & \dot{f}^m(n_2^m) & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \dot{f}^m(n_{s^m}^m) \end{bmatrix} \tag{2.25}$$

Duyarlılık için yineleme ilişkisi zincir kuralını kullanarak matris formunda

$$\begin{aligned} s^m &= \frac{\partial \hat{F}}{\partial n^m} = \left(\frac{\partial n^{m+1}}{\partial n^m}\right)\frac{\partial \hat{F}}{\partial n^{m+1}} = \dot{F}^m(n^m)(W^{m+1})^T \frac{\partial \hat{F}}{\partial n^{m+1}} \\ &= \dot{F}^m(n^m)(W^{m+1})^T s^{m+1}. \end{aligned} \tag{2.26}$$

biçiminde yazılabilir. Duyarlılıklar son katmandan ilk katmana doğru geriye yayımlanırlar, geri-yayılım algoritması da ismini buradan alır:

$$s^M \to s^{M-1} \to \ldots \to s^2 \to s^1. \tag{2.27}$$

Aslında geri-yayılım algoritması temelini sayısal optimizasyondaki gradyan düşümü metodundan alır. *Sadece bir yapay sinir ağında gradyanı hesaplamak için öncelikle duyarlılıkları geri yaymanız gerekir.* Geri-yayılımın güzelliği, transfer fonksiyonlarının özelliklerine dayanarak, gradyanın hesaplanması için zincir kuralının çok verimli bir uygulamasına sahip olmasıdır. Son olarak, geri-yayılım algoritmasının tamamlanması için bir adım daha gerekir. Yineleme ilişkisi için başlangıç noktası s^M bulunmalıdır. Bu da son katmanda elde edilir:

$$s_i^M = \frac{\partial \hat{F}}{\partial n_i^M} = \frac{\partial (t-a)^T(t-a)}{\partial n_i^M} = \frac{\partial \sum_{j=1}^{s^M}(t_j - a_j)^2}{\partial n_i^M} = -2(t_i - a_i)\frac{\partial a_i}{\partial n_i^M}. \tag{2.28}$$

$$\frac{\partial a_i}{\partial n_i^M} = \frac{\partial a_i^M}{\partial n_i^M} = \frac{\partial f^M(n_i^M)}{\partial n_i^M} = f^M(n_i^M) \tag{2.29}$$

eşitliğinden (2.28) eşitliği

$$s_i^M = -2(t_i - a_i) f^M(n_i^M). \tag{2.30}$$

şeklinde yazılabilir. Matris formunda ifade edilirse

$$s^M = -\dot{F}^M(n^M)(t-a) \tag{2.31}$$

olur.

EKHO algoritması, öğrenme oranı çok büyük olmadığı sürece hata kareler ortalamasını minimize eden çözüme yakınsar. Çünkü tek katmanlı doğrusal bir ağda hata kareler ortalaması kuadratik bir fonksiyondur. Kuadratik fonksiyon tek bir durağan noktaya sahiptir. Ek olarak, kuadratik fonksiyonun Hessian matrisi sabittir ve bu nedenle fonksiyonun eğriselliği verilen bir yönde değişiklik göstermez ve fonksiyonun seviye eğrileri eliptiktir.

Basit geri-yayılım EKHO algoritmasını genelleştirilmiş şeklidir. EKHO gibi, aynı zamanda hata kareler ortalamasını minimize etmek için yaklaşık gradyan düşümü algoritmasıdır. Aslında basit geri-yayılım, tek katmanlı doğrusal bir ağda kullanıldığında EKHO algoritmasına denktir. Ama çok katmanlı ağlara uygulandığında, basit geri-yayılımın özellikleri oldukça farklıdır. Tek katmanlı bir doğrusal ağda performans yüzeyi tek minimum noktasına ve sabit eğriselliğe sahipken, çok katmanlı bir ağda birçok yerel minimum noktasına sahiptir ve eğrisellik parametre uzayının farklı bölgelerinde büyük ölçüde değişkendir. Örneğin hata fonksiyonu büyük bir olasılıkla kuadratik değildir. Eğrisellik çok değişken olduğu için gradyan düşümü algoritması için uygun bir öğrenme oranı seçmek zordur. Bazı bölgelerde, yüzey düz olduğu durumda büyük bir öğrenme oranına ihtiyaç duyulurken, eğrisellik yüksek olduğu diğer bölgeler ise küçük bir öğrenme oranını gerektirir. Ayrıca ağda sigmoid transfer fonksiyonları

kullanıldığı için performans yüzeyinde düz bölgeler mutlaka oluşur. Çünkü sigmoid fonksiyon büyük girdiler için çok düzdür (Hagan ve ark. 1996).

2.3 Geri-Yayılım Algoritmasında Sezgisel Düzenlemeler

Geri-yayılım algoritmasının (gradyan düşümü) bazı dezavantajları daha önce ifade edildi. Geri-yayılım algoritmasını geliştirmek için literatürde iki popüler sezgisel metot önerilmiştir.

Momentum

Gradyan düşümü algoritması ıraksamaya başladığı anda dar bir vadide ileri ve geri salınımlar yapmaya başlar. Eğer algoritmanın performans yüzeyinde izlediği yolu parametrelerde gerçekleştirilen güncellemelerin ortalaması alınarak bir filtreden geçirilirse, salınımlar yumuşatılır ve daha kararlı bir yol elde edilir. Bu gradyan düşümü algoritmasına momentum terimi eklenerek gerçekleştirilebilir. Geri-yayılım algoritmasına momentum terimi eklendiğinde aşağıdaki denklemler elde edilir:

$$\Delta W^m(k) = \mu \Delta W^m(k-1) - (1-\mu)\eta s^m (a^{m-1})^T , \tag{2.18}$$

$$\Delta b^m(k) = \mu \Delta b^m(k-1) - (1-\mu)\eta s^m , \tag{2.19}$$

Literatürde çoğu kaynakta $(1-\mu)\eta$ ifadesinin tümü öğrenme oranı olarak kabul edilmiştir. Buradaki açık gösterim ise daha sağlıklıdır (Bu konu Bölüm 3'de daha ayrıntılı olarak açıklanmıştır). Momentum faktörü ile algoritmanın kararlılığı korunurken daha büyük bir öğrenme oranı kullanılmasına olanak sağlanır. Momentum faktörünün başka bir özelliği, performans yüzeyinde algoritmanın izlediği yol tutarlı bir yön ise yakınsamayı hızlandırmaya çalışır. Diğer bir deyişle, momentum faktörü algoritmaya eylemsizlik katarak salınımları yumuşatır ve algoritmanın belirli bir yönde kararlı bir şekilde ilerlemesini sağlar (Hagan ve ark. 1996).

Değişken Öğrenme Oranı

Hata yüzeyinin düz olduğu bölgelerde öğrenme oranı arttırılarak ve eğimin arttığı bölgelerde ise öğrenme oranını azaltarak yakınsama hızlandırılabilir. Çok katmanlı ağlarda hata yüzeyi kuadratik bir fonksiyon değildir. Yüzeyin şekli parametre uzayının farklı bölgelerinde çok farklı olabilir. Bu nedenle yakınsama hızı, eğitim sırasında öğrenme oranını ayarlayarak arttırılabilir. Burada dikkat edilmesi gereken nokta, öğrenme oranının ne zaman değiştirileceğini ve ne kadar değiştirileceğini belirlemektir.

Öğrenme oranının değişimi için birçok yaklaşım bulunmaktadır. Toplu güncelleme ile çalışan genel bir yaklaşım öğrenme oranının algoritmanın performansına göre değiştirilmesidir (Vogl ve ark. 1988). Değişken öğrenme oranlı geri-yayılım algoritmasının kuralları aşağıdaki gibi verilebilir:

1. Eğer hata kareler (tüm veri kümesi üzerinden) bir ağırlık güncellemesinden sonra belirli bir yüzdeden (tipik olarak yüzde bir ile yüzde beş arasında) daha fazla artıyorsa, o zaman ağırlık güncellemesi iptal edilir, öğrenme oranı belirli bir $0 < \rho < 1$ faktörüyle çarpılır ve momentum katsayısı μ sıfıra eşitlenir.
2. Eğer hata kareler bir ağırlık güncellemesinden sonra azalıyorsa, ağırlık güncellemesi kabul edilir ve öğrenme oranı belirli bir $\tau > 1$ faktörüyle çarpılır. Eğer μ momentum katsayısı daha önce sıfıra eşitlenmişse, kendi orijinal değerine geri döndürülür.
3. Eğer hata kareler belirli bir eşik değeri ζ 'den daha az artarsa, o zaman ağırlık güncellemesi kabul edilir ama öğrenme oranı değiştirilmez. Eğer μ daha önce sıfıra eşitlenmişse, kendi orijinal değerine geri döndürülür.

Değişken öğrenme oranlı algoritma üzerinde gerçekleştirilmiş daha başka birçok yaklaşım bulunmaktadır. Jacobs (1988) ağdaki her parametrenin (ağırlık veya sapma) kendi öğrenme oranına sahip olduğu *delta-bar-delta* öğrenme kuralını önermiştir. Bu yaklaşımda algoritma, herhangi bir ağ parametresinin değişimi belirli iterasyonlar için aynı yönde olursa, o parametrenin öğrenme oranını arttırır. Parametre değişiminin yönü değişken ise o parametrenin öğrenme oranı azaltılır.

Tollenaere'nin (1990) *SuperSAB* algoritması delta-bar-delta kuralına benzerdir, fakat öğrenme oranlarını düzenlemek için daha karmaşık kurallara sahiptir.

Gradyan düşümü geri-yayılım algoritmasında başka bir sezgisel düzenleme Fahlman'ın (1988) *quickprop* algoritmasıdır. Hata yüzeyinin parabolik ve minimum noktası civarında yukarı konkav olduğunu varsayar ve her ağırlığın etkisi bağımsız olarak dikkate alınır.

Gradyan düşümü geri-yayılım algoritmasında gerçekleştirilen sezgisel düzenlemeler bazı problemler için çok daha hızlı yakınsama sağlayabilir. Buna rağmen, bu metotların iki temel sorunu vardır. *Birincisi,* gradyan düşümü geri-yayılım algoritması için gereken tek parametre sadece öğrenme oranı iken bu düzenlemeler çeşitli parametrelerin seçimini gerektirir (örn. ζ, ρ ve μ). Bazı karmaşık sezgisel düzenlemelerde beş veya altı parametrenin seçilmesi gerekir. Genellikle algoritmanın performansı bu parametrelerdeki değişikliklere duyarlıdır. Ayrıca, parametrelerin seçimi problem bağımlıdır. Bu düzenlemelerdeki *ikinci* sorun, bazı problemlerde standart algoritma sonuç olarak bir çözüm bulup yakınsamasına rağmen düzeltilmiş algoritmalar bazı durumlarda yakınsamamaktadır. Bu tür sorunlar, daha karmaşık algoritmalar kullanıldığında daha çok gerçekleşme eğilimindedir.

2.4 Doğru Arama Yöntemleri

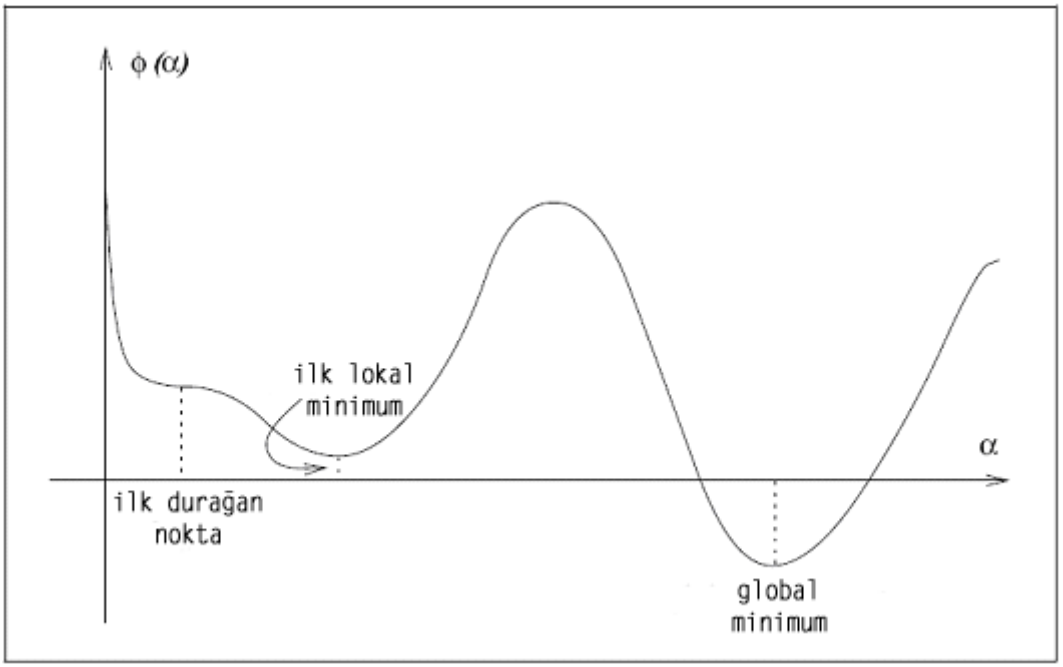
Bu kesimden itibaren ağın ağırlıkları (w) ve sapmaları (b), toplu olarak x ile gösterilecektir. Öğrenme oranını belirlemek için başka bir yaklaşımda doğru arama yöntemlerini kullanmaktır. Doğru arama stratejisinde, algoritma bir p_k yönü seçer ve şu anki x_k noktasından daha düşük fonksiyon değerine sahip yeni x_{k+1} adımına geçmek için bu yön boyunca arama yapar. p_k yönü boyunca ne kadar uzağa gidileceği yani bir α_k adım uzunluğu bir-boyutlu minimizasyon problemi

$$\min_{\alpha_k>0} f(x_k + \alpha_k p_k) \tag{2.20}$$

yaklaşık olarak çözülerek bulunabilir. Denklem (2.20) kesin olarak çözülerek, p_k yönünden maksimum fayda elde edilebilir, ama kesin bir minimizasyon hem pahalı hem de gereksizdir. Bunun yerine, doğru arama algoritması (2.20)'nin minimumunu çok da hassasiyet göstermeden yaklaşık buluncaya kadar belirli sayıda deneysel adım uzunlukları üretir. Yeni iterasyon noktasında yeni bir arama yönü ve adım uzunluğu hesaplanır ve süreç bu şekilde tekrarlanır. Doğru arama iterasyonu

$$x_{k+1} = x_k + \alpha_k p_k , \tag{2.21}$$

şeklinde verilir, pozitif skaler α_k adım uzunluğu olarak tanımlanır. Doğru arama yöntemlerinin başarısı p_k yönü ve adım uzunluğunun birlikte etkin seçimlerine dayanır.



Şekil 2.3 Global minimumdaki ideal adım uzunluğu.

Çoğu doğru arama algoritması p_k'nın azalan yön olmasını gerektirir; $p_k^T \nabla f_k < 0$ olması, çünkü bu özellik f fonksiyonunun bu yönde azalacağını garanti eder. Ayrıca, arama yönü

$$p_k = -B_k^{-1}\nabla f_k\,, \tag{2.22}$$

şeklinde verilebilir, B_k simetrik ve tekil olmayan (yani tersi olan) matristir. Dik iniş (Steepest descent) metodunda B_k sadece birim matris I, Newton metodunda ise kesin Hessian $\nabla^2 f(x_k)$ ve quasi-Newton metotlarında ise B_k düşük-ranklı bir formül yardımıyla her iterasyonda güncellenen Hessian matrisinin bir tahminidir. p_k (2.22) ile tanımlandığında ve B_k mutlak pozitif ise, $p_k^T \nabla f_k = -\nabla f_k^T B_k^{-1} \nabla f_k < 0$ olur ve böylece p_k azalan bir yöndür (Nocedal ve Wright 1999).

2.4.1 Adım Uzunluğu

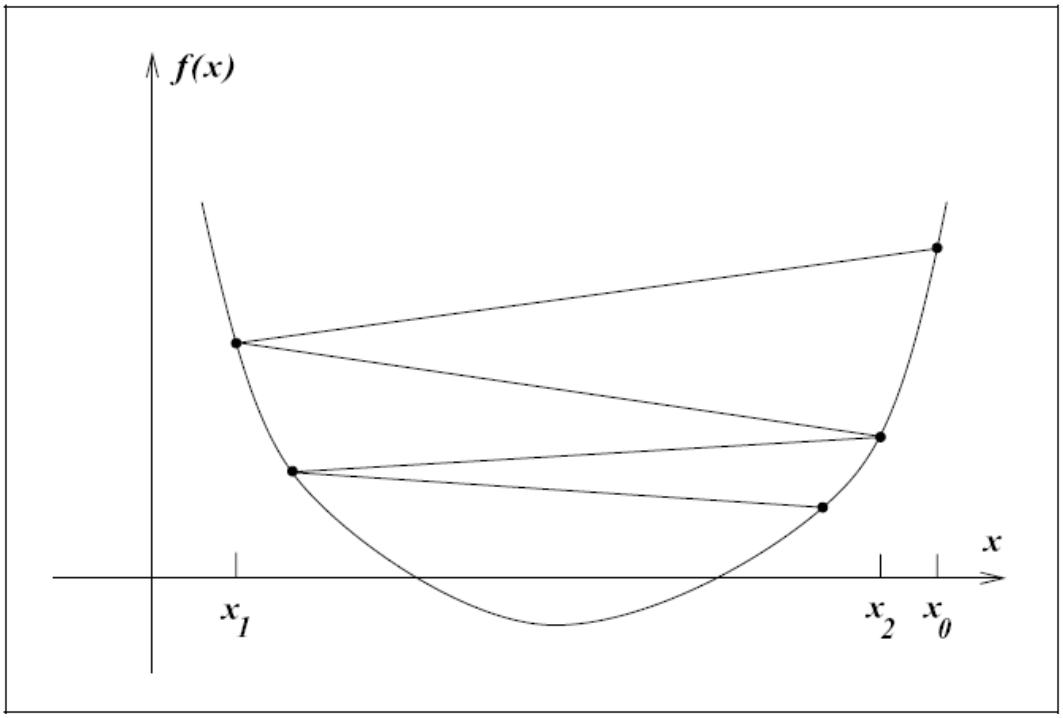
Adım uzunluğu α_k hesaplanırken, bir seçimle karşılaşılır. α_k seçimi, f'ye belirli bir azalma verecek şekilde ama aynı zamanda çok fazla zaman harcamadan yapılmak istenir. İdeal seçim aşağıda tanımlanan tek değişkenli $\phi(.)$ fonksiyonunun global minimumudur.

$$\phi(\alpha) = f(x_k + \alpha p_k), \qquad \alpha > 0 \tag{2.23}$$

Ama genellikle bu değeri bulmak çok maliyetlidir (Şekil 2.3). ϕ'nin yerel minimumunu bulmak bile genellikle amaç fonksiyonunun ve gradyan ∇f'nin çok fazla hesaplanmasını gerektirir. Daha pratik stratejiler, f'de yeterli azalmaları gerçekleştirecek adım uzunluğunu bulmaya yönelik kesin olmayan doğru aramayı minimum çabayla gerçekleştirirler.

Sıradan doğru arama algoritmaları, belirli koşulları sağlayan değeri kabul edinceye kadar α için aday değerler serisini denerler. *Doğru arama iki aşamada yapılır.* Bir *sınırlandırma aşaması* arzu edilen adım uzunluğu değerlerini içeren bir aralık bulur ve bir *ikiye bölme veya interpolasyon* aşaması bu aralıkta iyi bir adım uzunluğu hesaplar.

α_k üzerine koyulabilecek basit bir koşul f'de bir azalmanın sağlanmasıdır, örneğin $f(x_k + \alpha_k p_k) < f(x_k)$. Ama tek başına bu koşulun uygun olmadığı Şekil 2.4'de gösterilmiştir, minimum $f^* = -1$ iken fonksiyon değerlerinden oluşan süreç $\{5/k\}$, $k = 0,1,...,$ sonuç olarak 0'a yakınsar (Nocedal ve Wright 1999). Burada karşılaşılan sorun f'de yeterli azalmanın garanti altına alınmamasında kaynaklanır (Nocedal ve Wright 1999).



Şekil 2.4 f'de yetersiz azalma.

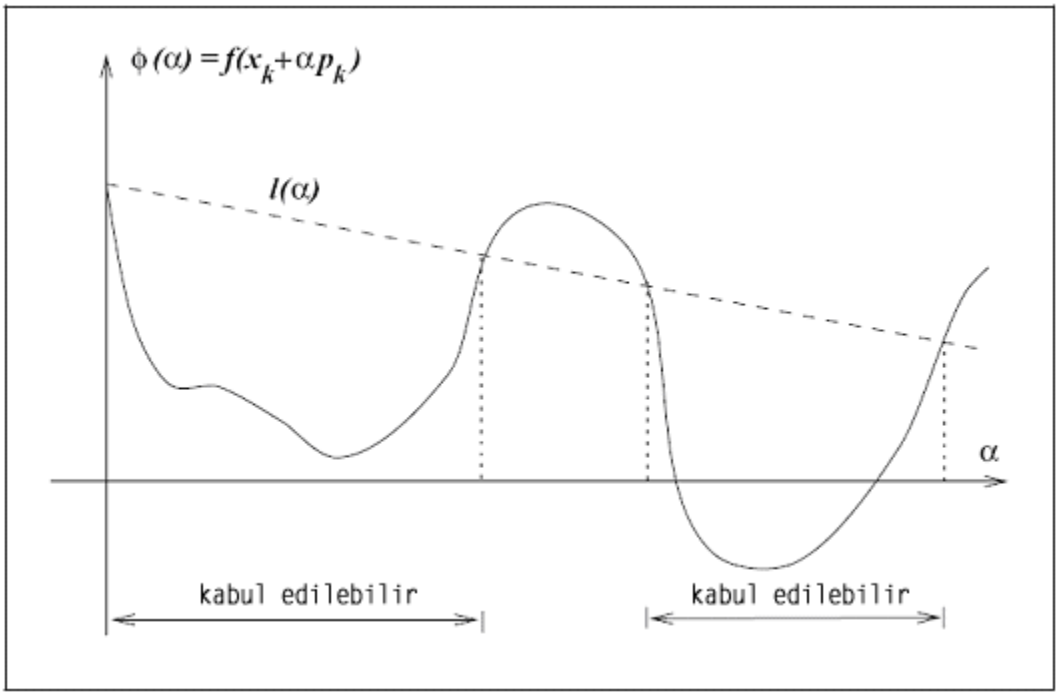
2.4.2 Wolfe Koşulları

Popüler bir kesin olmayan doğru arama koşulu, α_k'nın her şeyden önce amaç fonksiyonu f'de

$$f(x_k + \alpha p_k) \le f(x_k) + c_1 \alpha \nabla f_k^T p_k, \qquad c_1 \in (0,1) \tag{2.24}$$

eşitsizliğiyle ölçülen şekilde yeterli bir azalma vermesini şart koşar:

Başka bir deyişle f'deki azalma hem adım uzunluğu α_k hem de yönsel türev $\nabla f_k^T p_k$ ile oranlı olmalıdır. (2.24) eşitsizliği bazen *Armijo koşulu* olarak da tanımlanır. Yeterli azalma koşulu Şekil 2.5'de gösterilmiştir. (2.24)'ün sağ tarafı, doğrusal bir fonksiyon $l(\alpha)$ ile ifade edilebilir. $l(.)$ fonksiyonu negatif eğime sahiptir. $c_1 \nabla f_k^T p_k$, $c_1 \in (0,1)$ olduğundan α'nın küçük pozitif değerleri için ϕ'nin grafiğinin üzerinde yer alır. Yeterli azalma koşulu α'nın sadece $\phi(\alpha) \le l(\alpha)$ koşulu sağlanırsa kabul edilebileceğini belirtir. Bu koşulun sağlandığı aralıklar Şekil 2.5'de gösterilmiştir. Pratikte, c_1 oldukça küçük seçilir, örneğin $c_1 = 10^{-4}$ (Nocedal ve Wright 1999).

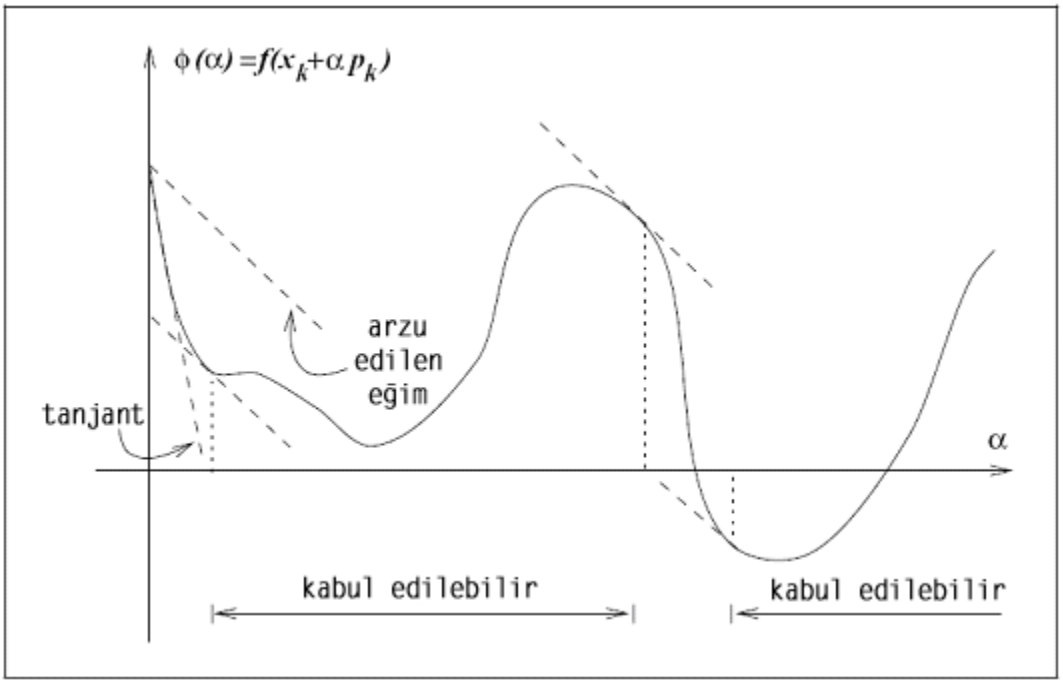


Şekil 2.5 Yeterli azalma koşulu.

Yeterli azalma koşulu tek başına algoritmanın kabul edilebilir bir ilerleme kaydetmesini sağlayamaz, çünkü Şekil 2.5'den görüldüğü gibi, α'nın yeterince küçük tüm değerleri için sağlanır. Kabul edilemeyecek kadar küçük adımları elemek için, *eğrisellik koşulu* adı altında ikinci bir gereksinim eklenir bu da α_k'nın

$$\nabla f(x_k + \alpha_k p_k)^T p_k \geq c_2 \nabla f_k^T p_k, \tag{2.25}$$

koşulunu belirli bir $c_2 \in (c_1, 1)$ sabiti için sağlamasını gerektirir. c_1 ise (2.24)'ten gelen sabittir. Sol taraf sadece $\phi'(\alpha_k)$ türevidir, böylece eğrisellik koşulu $\phi(\alpha_k)$'nın eğiminin gradyan $\phi'(0)$'dan c_2 kat daha büyük olmasını sağlar. Bu mantıklıdır çünkü $\phi'(\alpha)$ eğimi çok negatif ise, seçilen yön boyunca ilerleyerek f'nin önemli derecede azalacağı bilgisine sahip olunur. Diğer yandan, eğer eğim biraz negatifse veya hatta pozitifse, bu yönde f'de daha fazla bir azalma beklenemeyeceğinin bir göstergesidir ve bu nedenle doğru aramayı sona erdirmek mantıklı olur. Eğrisellik koşulu Şekil 2.6'da gösterilmiştir. c_2'nin tipik değerleri p_k arama yönü Newton veya quasi-Newton metoduyla seçildiğinde 0.9 ve doğrusal olmayan eş gradyan metoduyla seçildiğinde 0.1'dir (Nocedal ve Wright 1999).

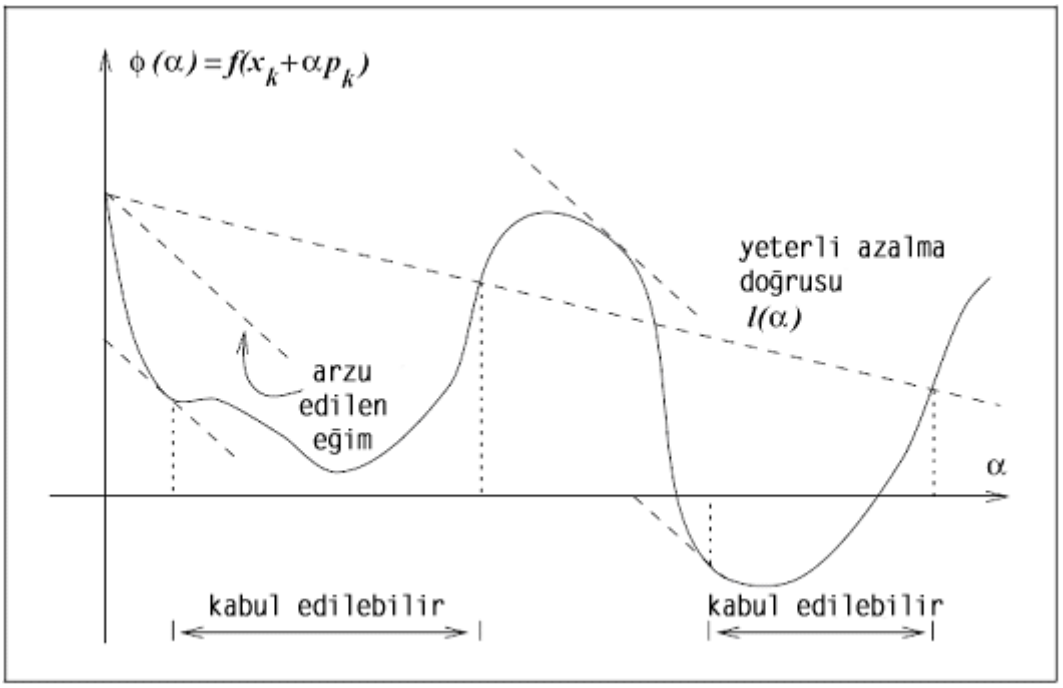


Şekil 2.6 Eğrisellik koşulu.

Yeterli azalma ve eğrisellik koşulu beraberce

$$f(x_k + \alpha_k p_k) \le f(x_k) + c_1 \alpha_k \nabla f_k^T p_k ,$$
$$\nabla f(x_k + \alpha_k p_k)^T p_k \ge c_2 \nabla f_k^T p_k , \qquad 0 < c_1 < c_2 < 1 \tag{2.26}$$

şeklinde ifade edilir ve Wolfe koşulları olarak bilinir. Şekil 2.7'de görüldüğü gibi Wolfe koşullarını sağlayan bir adım uzunluğu özellikle ϕ'nin minimumuna yakın düşmeyebilir. Buna rağmen eğrisellik koşulu, α_k'yı en azından ϕ'nin bir yerel minimumuna veya durağan noktasının yakın komşuluğuna düşecek şekilde düzenlenebilir.



Şekil 2.7 Wolfe koşullarını sağlayan adım uzunlukları.

Ciddi Wolfe koşulları α_k'nın

$$f(x_k + \alpha_k p_k) \le f(x_k) + c_1 \alpha_k \nabla f_k^T p_k ,$$
$$\left|\nabla f(x_k + \alpha_k p_k)^T p_k\right| \le c_2 \left|\nabla f_k^T p_k\right| , \quad 0 < c_1 < c_2 < 1$$

koşullarını sağlamasını gerektirir. Wolfe koşullarından tek farkı $\phi'(\alpha_k)$ türevinin büyük pozitif olmasına izin verilmez. Bu durumda, ϕ'nin durağan noktalarına uzak noktalar çıkartılmış olur (Nocedal ve Wright 1999).

3. KUADRATİK FONKSİYON İÇİN KARARLILIK VE YAKINSAMA HIZI

Momentumlu gradyan düşümü algoritmasında öğrenme parametrelerinin, yani öğrenme oranı ve momentum faktörünün seçimi algoritmanın yakınsama tavrı üzerinde önemli bir etkiye sahiptir. Eğer bu parametreler dikkatli seçilmezse algoritmanın yakınsama süresi oldukça uzayabilir ve hatta belirli bir sınırın dışında seçildiğinde algoritma yakınsamayabilir. Bu nedenle momentumlu gradyan düşümü algoritmasının kararlılığı ve yakınsama hızı üzerine literatürde çok sayıda çalışma yapılmıştır ve halen devam etmektedir (Plaut ve ark. 1986, Williams 1991, Yu ve Chen 1997).

Yapay sinir ağlarında da performans fonksiyonu olarak kullanılan hata kareler fonksiyonu yerel minimum civarında yaklaşık olarak kuadratik olduğu için son zamanlardaki çalışmalar kuadratik hata fonksiyonlarının analizi üzerinde yoğunlaşmıştır. Bu bölümde, ağın hata fonksiyonu kuadratik olduğunda momentumlu gradyan düşümü algoritmasının kararlılığı farklı bir yaklaşımla incelenmiştir. Algoritmayı kuadratik hata fonksiyonu durumunda kararlı kılan ve yakınsama hızını çok iyi derecede iyileştiren etkin öğrenme parametrelerinin hesaplanması için gerekli formüller elde edilmiştir. Daha sonra hata fonksiyonunun kuadratik olduğu küçük boyutlu rassal problemlerde hesaplanan etkin parametrelerin gerçekteki optimum öğrenme parametreleriyle ne kadar örtüştüğü incelenmiştir. Orta boyutlu kuadratik hata fonksiyonlu problemlerde ise etkin parametrelerle çalışan momentumlu gradyan düşümü algoritmasının performansı diğer geleneksel momentumlu gradyan düşümü algoritmalarıyla karşılaştırılmıştır.

3.1 Momentumlu Gradyan Düşümü Algoritmasının Fiziki Yorumu

Qian (1999) momentumlu gradyan düşümü algoritması ile tutucu bir kuvvet alanı etkisi altında v sürtünme katsayısı ile yapışkan bir ortamda hareket eden bir

nokta kütlenin Newton denklemi arasındaki benzerliği göstermiştir. Ayrıca momentum ile Newton partiküllerinin kütlesi arasındaki ilişkiyi de ifade etmiştir.

$E(x)$ potansiyel enerjisine sahip bir tutucu kuvvet alanı altında v sürtünme katsayısı ile yapışkan bir ortamda hareket eden bir m nokta kütle için Newton denklemi

$$m\frac{d^2x}{dt^2}+v\frac{dx}{dt}=-\nabla_x E(x) \tag{3.1}$$

biçiminde verilebilir, burada m hareket eden nokta kütle, x partikülün koordinat vektörü ve $\nabla_x E(x)$, E potansiyel enerji fonksiyonunun x' e göre gradyan vektörüdür. $m=0$ olduğunda (3.1)

$$\frac{dx}{dt}=-\eta\nabla_x E(x) \tag{3.2}$$

momentumsuz temel gradyan düşümü algoritması haline gelir, η öğrenme oranının (3.1)'deki $\frac{1}{v}$' ye eşit olduğuna dikkat etmek gerekir. (3.1) kesikli şekilde

$$m\frac{x(t+\Delta t)-x(t-\Delta t)}{\Delta t^2}+v\frac{x(t+\Delta t)-x(t)}{\Delta t}=-\nabla_x E(x) \tag{3.3}$$

gibi yazılabilir. (3.3), $x_{t+\Delta t}$, x_t, $x_{t-\Delta t}$ terimlerine göre yazılırsa,

$$x_{t+\Delta t}-x_t=-\frac{(\Delta t)^2}{(m+v\Delta t)}\nabla_x E(x)+\frac{m}{(m+v\Delta t)}(x_t-x_{t-\Delta t})$$

veya $\Delta t=1$ alarak

$$x_{t+1} - x_t = -\frac{1}{m+v}\nabla_x E(x) + \frac{m}{m+v}(x_t - x_{t-1}). \tag{3.4}$$

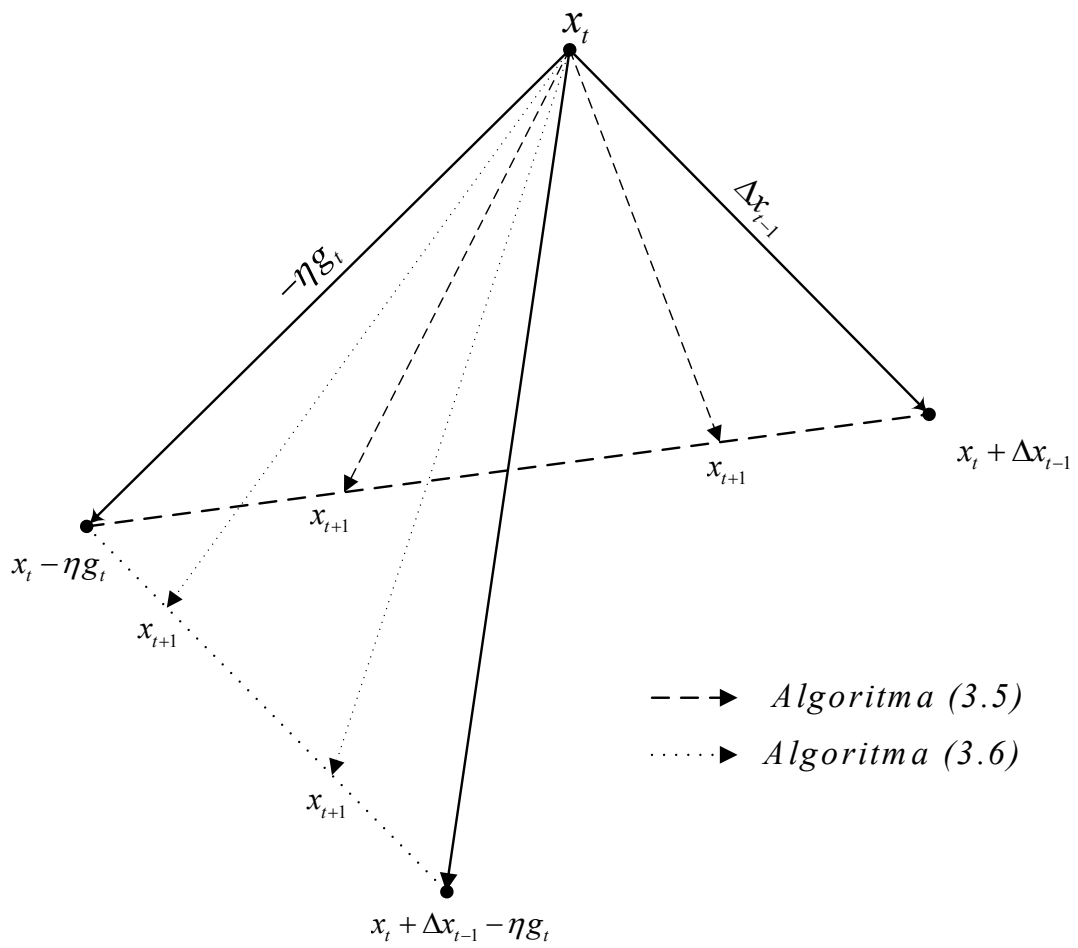
elde edilir. $\mu = \frac{m}{m+v}$ terimi momentum faktörü ve $\eta = \frac{1}{v}$ öğrenme oranı olarak kabul edilirse, (3.4)

$$x_{t+1} - x_t = -(1-\mu)\eta\nabla_x E(x) + \mu(x_t - x_{t-1}). \tag{3.5}$$

şeklinde yazılabilir. (3.5) denklemi, Torii ve Hagan (2002) tarafından kullanılan momentumlu gradyan düşümü iterasyon sürecidir. Qian (1999), çalışmasında $(1-\mu)\eta = \frac{1}{m+v}$ 'yi ε ile göstererek, tüm ifadeyi öğrenme oranı olarak algılamıştır (momentum faktörü ilgili çalışmada p ile gösterilmiştir):

$$x_{t+1} - x_t = -\varepsilon \cdot \nabla E(x) + \mu(x_t - x_{t-1}) \tag{3.6}$$

Yapay sinir ağları hakkında çoğu kitap ve makalede momentumlu gradyan düşümü algoritması (3.6) ile verilmiştir (Kamarthi ve Pittner 1999; Haykin 1999; Bishop 1995). (3.6)'da, ε (η) öğrenme oranının, momentum faktörü μ den bağımsız olarak seçilebileceği görülür. Ama (3.5)'de, μ'nün değişiminin ε'ı etkileyeceği açıktır. Belirli bir η için, momentum faktörü $0 \le \mu \le 1$ aralığında değiştiğinde (3.5) iterasyonu uygulanırsa, $(t+1)$ zamanında sistemin durumu x_{t+1}, $(x_t + \Delta x_{t-1})$ ve $(x_t - \eta \cdot g_t)$ noktalarının konveks kombinasyonu ile belirlenir, (3.6) uygulandığında ise sistemin durumu x_{t+1}, $(x_t + \Delta x_{t-1} - \eta \cdot g_t)$ ve $(x_t - \eta \cdot g_t)$ noktalarının konveks kombinasyonu ile belirlenir (g_t, t adımındaki hata fonksiyonunun gradyanıdır) (şekil 3.1). (Qian 1999)'daki fiziki yorum momentumlu gradyan düşümü algoritmasının (3.5)'deki şeklinin daha uygun olduğu fikrini ortaya koyar. Şekil 3.1'den görüleceği üzere, geometrik açıdan da (3.5) algoritmasının yeni x_{t+1} durumu için daha geniş bir oluşum imkanı sağladığı açıktır.



Şekil 3.1 Momentumlu gradyan düşümü algoritmasının iki farklı parametrik tipinin geometrik gösterimi.

Denklem (3.5)'de, momentum faktörü kütle rolü oynar, eğer $m = 0$ ise $\mu = 0$ olur ve aynı şekilde bu ifadenin tersi de geçerlidir. (3.5)'deki potansiyel enerji fonksiyonu $E(x)$ hata fonksiyonu gibi düşünülebilir, yani koordinat vektörü x'in seçimi, $E(x)$ fonksiyonunun minimizasyonu problemi ile ilişkilidir. Kararlı durumda, ağırlık vektörü x'in (veya denk olarak, partikülün hızının) değişimi sıfıra ulaşır.

x_0'ın, $E(x)$ fonksiyonunun yerel minimumu olduğu varsayılsın ($\nabla_x E(x_0) = 0$), o zaman x_0'ın komşuluğunda,

$$\nabla E(x) \approx \nabla E(x_0) + H(x - x_0) = H(x - x_0), \tag{3.7}$$

burada

$$H = (h_{ij}) \ , \ h_{ij} = \left. \frac{\partial^2 E(x)}{\partial x_i \partial x_j} \right|_{x_0} ,$$

simetrik pozitif tanımlı Hessian matrisidir. (3.5)'de (3.7)'yi düşünerek, yerel minimum x_0 civarında, (3.5)

$$x_{t+1} - x_t = -(1-\mu)\eta H x_t + \mu(x_t - x_{t-1}) + (1-\mu)\eta b \tag{3.8}$$

biçiminde yeniden yazılabilir veya

$$x_{t+1} = [(1+\mu)I - (1-\mu)\eta H]x_t - \mu x_{t-1} + (1-\mu)\eta b \tag{3.9}$$

şeklinde ifade edilebilir, Hx_0 ifadesi (3.8) ve (3.9)'da b ile gösterilmiştir ($Hx_0 = b$). Momentum parametresi $-1 < \mu < 1$ aralığında olduğunda, momentumlu gradyan düşümü algoritması kararlı olur. μ'nün negatif değerleri yakınsamayı yavaşlattığı için (Phalsankar ve Sastry 1994) momentumun $0 < \mu < 1$ aralığından seçilmesi daha uygundur (Torii ve Hagan 2002).

3.2 Kuadratik Hata Fonksiyonları için Momentumlu Gradyan Düşümü Algoritmasının Kararlılığı

(3.5)'deki gradyan düşümü algoritması, $\nabla_x E(x)$ fonksiyonunun x_0 yerel minimum civarındaki (3.7) yaklaşımı kullanılarak (3.8) veya (3.9) şeklinde yazılabilir. (3.8),

$$F(x) = \frac{1}{2} x^T H x - b^T x + c \tag{3.10}$$

kuadratik hata fonksiyonunun minimize edilmesi için momentumlu gradyan düşümü algoritmasıdır. H, $n \times n$ simetrik pozitif tanımlı bir matris; b, n-boyutlu bir vektör ve c ise bir sabittir. F kuadratik fonksiyonunun x noktasındaki gradyanı $\nabla F(x) = Hx - b$'dir. $b - Hx = r$, $Hx = b$ doğrusal denkleminin çözümü açısından artık gibi düşünülebilir ve $\nabla F(x) = -r$ eşitliği alınabilir. (3.10)'un minimumunu veren ağırlık vektörü x^*,

$$Hx = b\,. \tag{3.11}$$

doğrusal denkleminin çözüm vektörüdür. Bu nedenle (3.10)'un minimumunu bulan herhangi bir iteratif süreç, (3.11)'de verilen denklemin çözümü için de bir iteratif süreç olarak kabul edilebilir. H simetrik ve pozitif tanımlı olduğu için, uygun bir Q ortogonal matrisi ile köşegenleştirilebilir (Q, H'nin ortonormal özvektörleri ile oluşturulan bir matristir):

$$H = QKQ^T \quad , \quad QQ^T = I\,,$$

K, H'nin k_i, $i = \overline{1,n}$ özdeğerlerinden oluşan bir köşegen matristir. $x' = Q^T x$ dönüşümü uygulanırsa, (3.9) aşağıdaki gibi yazılabilir,

$$x'_{t+1} = [(1+\mu)I - (1-\mu)\eta K]x'_t - \mu x'_{t-1} + (1-\mu)\eta b'\,, \tag{3.12}$$

burada $b' = Q^T b$. (3.12),

$$x'_{i,t+1} = [1 + \mu - (1-\mu)\eta\kappa_i]x'_{i,t} - \mu x'_{i,t-1} + (1-\mu)\eta b'_i \;,\; i = \overline{1,n}\,. \tag{3.13}$$

biçiminde koordinatlara göre yazılabilir. Böylece x vektörünün koordinatları, x' vektörünün koordinatlarının doğrusal kombinasyonu olarak elde edilir. Yapay denklem $x'_{i,t} = x'_{i,t}$ eklenerek (3.13) matris formunda yazılabilir:

$$\tilde{x}'_{i,t+1} = P_i \tilde{x}'_{i,t} + d_i \ , \ \tilde{x}'_{i,t} = \begin{pmatrix} x'_{i,t-1} \\ x'_{i,t} \end{pmatrix}, \ i = 1,2,...,n \,, \tag{3.14}$$

burada $P_i = \begin{pmatrix} 0 & 1 \\ -\mu & 1+\mu-(1-\mu)\eta\kappa_i \end{pmatrix}$, 2×2 bir matris ve $d_i = \begin{bmatrix} 0 \\ (1-\mu)\eta b_i \end{bmatrix}$, iki boyutlu bir vektördür ($i = 1,2,...,n$).

(3.13) veya (3.14) ile verilen doğrusal dinamik sistem, P_i matrisinin özdeğerlerinin büyüklüğü (modu) bir'den küçük olduğunda kararlı olur (Brogan 1991). Bu sayede, (3.9) momentumlu gradyan düşümü algoritmasının kararlılığı ile P_i matrisini özdeğerlerinin büyüklüğü arasında bir ilişki kurulmuş olur.

P_i ($i = 1,2,...,n$) matrisinin özdeğerlerini bulmak için karşılık gelen karakteristik denklem yazılırsa:

$$\left| P_i - \lambda I \right| = \begin{vmatrix} -\lambda & 1 \\ -\mu & 1+\mu-(1-\mu)\eta\kappa_i - \lambda \end{vmatrix} = 0, \ \ i = 1,2,...,n \,.$$

Buradan, P_i matrisinin λ özdeğerlerinin, aşağıdaki kuadratik denklemlerin kökleri olduğu bulunur (Qian 1999):

$$\lambda^2 - [(1+\mu)-(1-\mu)\eta\kappa_i]\lambda + \mu = 0, \ \ i = 1,2,...,n \,, \tag{3.15}$$

κ_i ($i = 1,2,...,n$), $n \times n$ H simetrik pozitif tanımlı matrisin özdeğerleridir. Her bir κ_i'ye ($i = 1,2,...,n$) reel veya kompleks iki kök karşılık gelir.

Doğrusal iteratif süreç (3.14)'ün kararlılığı için, (3.15)'in her bir kökünün büyüklüğünün (modunun) 1'den küçük olması gerekir. Bu nedenle, (3.8) veya (3.9) gradyan düşümü algoritmasının kararlılığı problemi (3.15) denkleminin incelenmesi haline gelir. (3.15)'in H matrisinin herhangi κ özdeğerine karşılık gelen kökleri

$$\lambda = \frac{[(1+\mu)-(1-\mu)\eta\kappa] \pm \sqrt{[(1+\mu)-(1-\mu)\eta\kappa]^2 - 4\mu}}{2}. \qquad (3.16)$$

şeklinde hesaplanır, (3.15) in sol tarafındaki kuadratik fonksiyon ele alınırsa (λ ya göre)

$$\varphi(\lambda) = \lambda^2 - [(1+\mu)-(1-\mu)\eta\kappa]\lambda + \mu\,. \qquad (3.17)$$

Bu kuadratik formun diskriminantı

$$D = [(1+\mu)-(1-\mu)\eta\kappa]^2 - 4\mu\,,$$

veya momentum faktörü μ ' ye göre yazılırsa

$$D(\mu) = (1+\eta\kappa)^2\mu^2 - 2(1+\eta^2\kappa^2)\mu + (1-\eta\kappa)^2 \qquad (3.18)$$

şeklinde elde edilir. $D<0$ olduğunda (3.15)'in kökleri, büyüklüğü $|\lambda| = \sqrt{\mu}$ 'ye eşit olan eşlenik kompleks sayılardır. (3.18) kuadratik formu [0,1] aralığında iki farklı köke sahiptir:

$$\mu_1 = \frac{(1-\eta\kappa)^2}{(1+\eta\kappa)^2}, \quad \mu_2 = 1,$$

ve bu nedenle $D(\mu)$ 'nün işareti aşağıdaki gibi belirlenir:

$$D(\mu)\begin{cases} <0 & , \quad S(\eta\kappa) < \mu < 1 \\ =0 & , \quad \mu = 1 \quad veya \quad \mu = S(\eta\kappa)\,. \\ >0 & , \quad \mu > 1 \quad veya \quad \mu < S(\eta\kappa) \end{cases} \qquad (3.19)$$

(3.19)'da $S(\eta\kappa) = \frac{(1-\eta\kappa)^2}{(1+\eta\kappa)^2}$, η öğrenme oranı ve κ, H matrisinin herhangi bir

özdeğeridir. $S(\eta\kappa)$, $\eta\kappa$ ' nın bir fonksiyonu olarak aşağıdaki özelliklere sahiptir:

$S(\eta\kappa)$, $0 \le \eta\kappa \le 1$ aralığında 1'den 0'a azalandır ve $\eta\kappa = 1$'de 0 minimum değerini alır; $\eta\kappa > 1$ olduğunda ise 0'dan $+\infty$'a artandır. $S(\eta\kappa)$, $0 \le \eta\kappa \le 2$ aralığında konveks, $(2, +\infty)$ aralığında ise konkavdır. $\eta\kappa = 2$ dönme noktasıdır (Şekil 3.3) (Taş 2003).

Aşağıda (Torii ve Hagan 2002)'deki kararlılık sonuçlarına benzer bir teorem verilmiştir ve bu tez çalışmasında ispat farklı bir yoldan gerçekleştirilir.

Teorem 1 (Kararlılık). η'nın öğrenme oranı ve κ_i'nin $(i = 1, 2, ..., n)$ simetrik pozitif tanımlı H matrisinin özdeğerleri olduğunu varsayılsın. Eğer $0 < \eta\kappa_i \le 2$, $(i = 1, 2, ..., n)$ ise, o zaman (3.8) momentumlu gradyan düşümü algoritması, (0,1) aralığındaki herhangi bir μ momentum faktörü için kararlı olur. $\max_i \eta\kappa_i > 2$ olduğunda ise, (3.8) algoritması $\max_i \frac{\eta\kappa_i - 2}{\eta\kappa_i + 2} < \mu < 1$ aralığındaki herhangi bir μ momentum faktörü için kararlı olur.

İspat. Algoritma (3.8)'in kararlılığını ispatlayabilmek için, P matrisinin özdeğerlerinin büyüklüğünün 1'den küçük olduğunu göstermek gerekir. (3.15)'in herhangi bir kompleks kökünün büyüklüğü $|\lambda| = \sqrt{\mu}$'dür ve $0 < \mu < 1$ koşulu sağlandığında 1'den küçüktür. O zaman geriye (3.15)'in veya (3.17) kuadratik fonksiyonunun herhangi bir reel kökünün mutlak değerinin 1'den küçük olduğunu göstermek kalır.

(3.17) ile tanımlanan $\varphi(\lambda)$ kuadratik formu için

$$\varphi(0) = \mu > 0 . \tag{3.20}$$

$\varphi(\lambda)$ fonksiyonunun minumumu,

$$\lambda_{\min} = \frac{(1+\mu) - (1-\mu)\eta\kappa}{2} \tag{3.21}$$

ve minimum değeri

$$\varphi(\lambda_{\min}) = -\frac{D(\mu)}{4}. \tag{3.22}$$

Eşitlik (3.22)'ye dayanarak, $D(\mu) \geq 0$ olduğu durumda, $\varphi(\lambda_{\min}) \leq 0$ olduğu görülür. λ özdeğerleri $\eta\kappa$ 'ya göre incelenirse;

a) eğer $0 < \mu < 1$ için $\eta\kappa = 1$ ise, o zaman $D(\mu) < 0$ 'dır, λ özdeğerleri kompleks sayılardır ve bu durumda $|\lambda| = \sqrt{\mu} < 1$ 'dir.

b) eğer $0 < \eta\kappa \leq 2$ ve $\eta\kappa \neq 1$ ise, o zaman $0 < \mu < 1$ aralığındaki μ momentum faktörü seçimine bağlı olarak, $D(\mu)$ sıfırdan küçük, sıfırdan büyük veya sıfıra eşit olabilir. Bu durumda, (3.21) ile tanımlanan $\lambda_{\min}$ aşağıdaki gibi değerlendirilebilir

$$\frac{3\mu - 1}{2} < \lambda_{\min} = \frac{(1+\mu) - (1-\mu)\eta\kappa}{2} < \frac{1+\mu}{2}. \tag{3.23}$$

μ, (0,1) aralığında değiştiği için, $\lambda_{\min}$ 'in üst ve alt sınırı (3.23) kullanılarak bulunabilir:

$$\frac{-1}{2} < \lambda_{\min} < 1. \tag{3.24}$$

Diğer taraftan,

$$\varphi(1) = (1-\mu)\eta\kappa > 0, \quad (\text{çünkü } \mu < 1) \tag{3.25}$$

ve

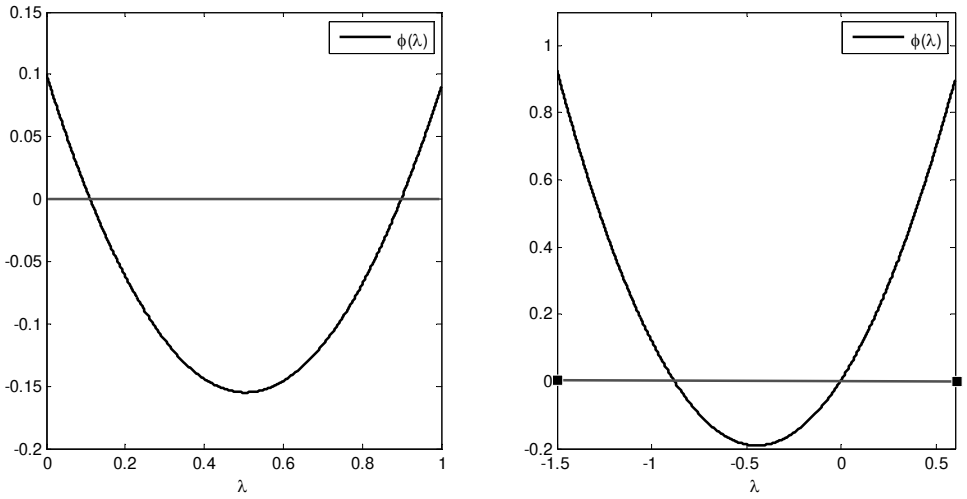
$$\varphi(-1) = 1 + (1+\mu) - (1-\mu)\eta\kappa + \mu = 2(1+\mu) - (1-\mu)\eta\kappa \tag{3.26}$$

Denklem (3.26)'ya göre $\varphi(-1)$, $0 < \eta\kappa < 2$ aralığında $(\eta\kappa)$'ya göre azalandır ve bu nedenle,

$$\varphi(-1) \geq 2(1+\mu) - (1-\mu)2 = 4\mu > 0 \tag{3.27}$$

olur. Böylece, $0 < \eta\kappa \leq 2$ koşulu sağlandığında, $0 < \mu < 1$ aralığındaki herhangi bir momentum faktörü için $\varphi(-1) > 0$ olur.

Şimdi, $D(\mu) > 0$ olduğu durumda (3.20), (3.25) ve (3.27) ye göre, $\varphi(0)$, $\varphi(1)$, $\varphi(-1) > 0$ ve (3.24)'den $\frac{-1}{2} < \lambda_{\min} < 1$ olduğu görülür. Bu nedenle, $\varphi(\lambda)$'nın grafiği şekil 3.2'deki şematik grafiklerden birine benzer olacaktır ve her iki durumda da $|\lambda| < 1$ olduğu görülür.



Şekil 3.2 $\varphi(\lambda)$ ' nın şematik grafikleri.

c) Şimdi $\eta\kappa > 2$ koşulunun sağlandığı varsayılsın. O zaman $\frac{\eta\kappa - 2}{\eta\kappa + 2} < \mu < 1$ aralığında olan momentum faktörleri için (3.17) kuadratik formunun reel köklerinin mutlak değerinin 1'den küçük olacağı gösterilebilir. Bunun için, $\lambda_{\min} \in (-1, 0)$ ve $\varphi(-1) > 0$ koşullarının sağlanması yeterli olacaktır. (3.26)' ya göre,

$$\varphi(-1) = (\eta\kappa + 2)\mu - (\eta\kappa - 2).$$

elde edilir, bu sayede, $\eta\kappa > 2$ durumunda, $\varphi(-1) > 0$ olabilmesi için

$$\mu > \frac{\eta\kappa - 2}{\eta\kappa + 2} \tag{3.28}$$

eşitsizliğinin sağlanması gerekir. Diğer taraftan, $0 < \mu < 1$ aralığında $D(\mu)$ için (3.19) ile tanımlanmış $D(\mu) > 0$ koşulu

$$\mu < \frac{(\eta\kappa - 1)^2}{(\eta\kappa + 1)^2} \tag{3.29}$$

olduğunda sağlanır. $\eta\kappa > 1$ olduğu durumda (3.21) ve (3.29) dan

$$\lambda_{\min} = \frac{1}{2}\left[(\eta\kappa + 1)\mu - (\eta\kappa - 1)\right] = \frac{1 - \eta\kappa}{1 + \eta\kappa} < 0 \tag{3.30}$$

olduğu bulunur. Buradan (3.21) ve (3.19)'a göre

$$0 < (\eta\kappa + 2)\mu - (\eta\kappa - 2) = [(\eta\kappa + 1)\mu - (\eta\kappa - 1)] + \mu + 1 = 2\lambda_{\min} + (\mu + 1)$$

eşitliği elde edilir. μ, 0 ile 1 arasından değerler aldığı için, $\lambda_{\min} > -\frac{\mu + 1}{2} > -1$ olduğu bulunur. Bu nedenle, (3.15)' in kökleri $\lambda_{\min} \in (-1, 0)$ olduğu sürece (-1,0) aralığında yerleşir ve μ momentum faktörü $\frac{\eta\kappa - 2}{\eta\kappa + 2} < \mu < 1$ aralığında değişir. Sonuç olarak, (3.8) iteratif süreci $\max_i \frac{\eta\kappa_i - 2}{\eta\kappa_i + 2} < \mu < 1$ koşulu sağlandığında kararlıdır. ■

Not 1. Teorem 3.1'in ispatına dikkat edilirse, aşağıdaki ifadelerin doğru olduğu görülür:

Eğer $0 < \eta\kappa < 1$ ve $D(\mu) > 0$ ise (3.15)' in uygun kökleri (0,1) aralığında yerleşir;

Eğer $1 < \eta\kappa \leq 2$ ve $D(\mu) > 0$ ise (3.15)' in uygun kökleri (-1,0) aralığında yerleşir.

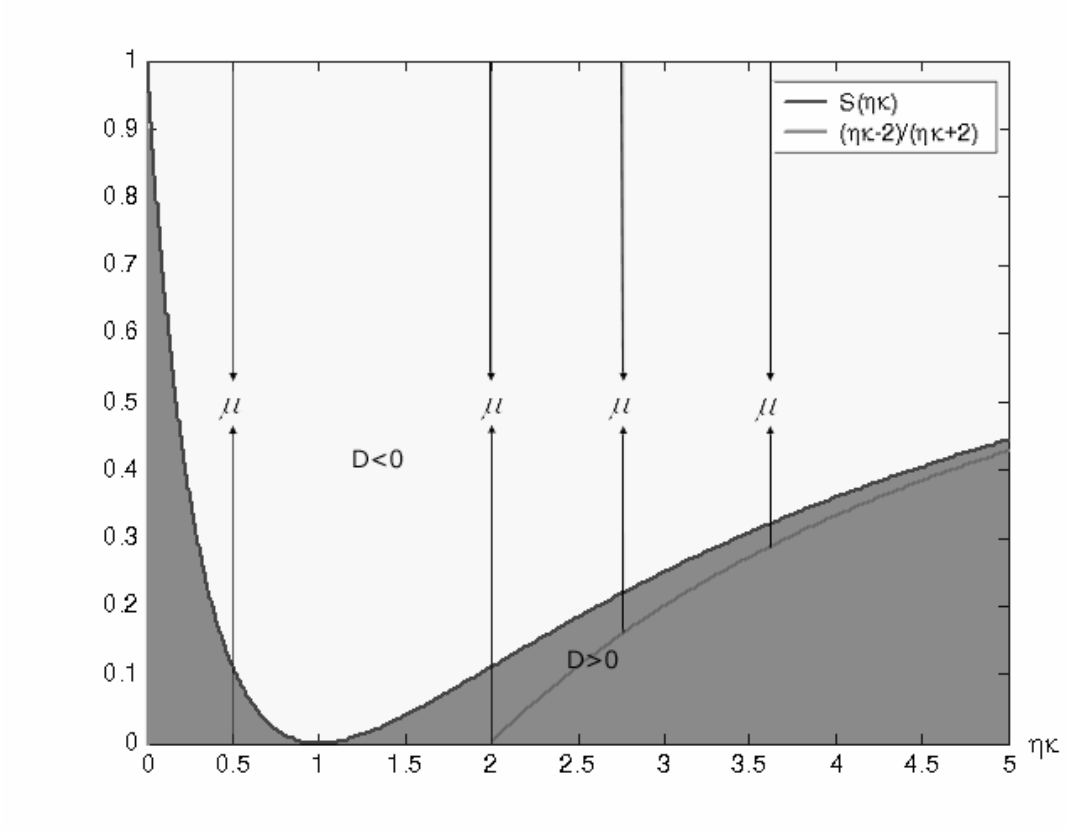
Not 2. Teorem 3.1 kısa şekilde aşağıdaki gibi ifade edilebilir:

η öğrenme oranı ve κ_i, $i = 1,2,...,n$ simetrik pozitif tanımlı H matrisinin özdeğerleri olsun, o zaman (3.8) momentumlu gradyan düşümü algoritması aşağıda verilen aralıktaki momentum faktörleri için kararlıdır.

$$\max\{0\ ,\ \max_i \frac{\eta\kappa_i - 2}{\eta\kappa_i + 2}\} < \mu < 1$$

Not 3. Teorem 1'in ispatından şu açıktır: μ momentum faktörü $-1 < \mu < 1$ aralığında değiştiğinde, $\max_i \frac{\eta\kappa_i - 2}{\eta\kappa_i + 2} < \mu < 1$ koşulu (3.8) algoritmasının kararlı olması için gerekli ve yeterli koşuldur.

Şekil 3.3'de, (3.8) iteratif sürecinin kararlılığı için μ momentum katsayısının $\eta\kappa$ 'ya göre değişim aralıkları geometrik olarak gösterilmiştir.



Şekil 3.3 $S(\eta\kappa)$, $\dfrac{\eta\kappa-2}{\eta\kappa+2}$ fonksiyonlarının ve geçerli μ aralıklarının grafiği

3.3 Önceki Çalışmalarla Karşılaştırma

(Qian 1999) ve (Torii ve Hagan 2002) çalışmalarında elde edilen sonuçlar arasında benzerlikler bulunmaktadır. Bu tez çalışmasında ise (3.15)'in kökleri önceki çalışmalardan farklı bir yaklaşımla incelenmiştir. Bunu açık olarak görebilmek için öncelikle (Qian 1999)'da elde edilen sonuçları yorumlamaya çalışalım.

Daha önce açıklandığı üzere, (3.8)'in kararlılığı için (3.16)'da verilen $\lambda_{i,1}, \lambda_{i,2}$, $i=\overline{1,n}$ köklerinin büyüklüğünün 1'den küçük olması gerekir. (Qian 1999)'da sayfa 148'deki sonuç 3'e göre

$Max\left(\left|\lambda_{i,1}\right|,\left|\lambda_{i,2}\right|\right)<1$ *ve bu nedenle (14) ile ifade edilen sistem ancak ve ancak*

$$-1 < p < 1 \;\; ve \;\; 0 < \varepsilon\kappa_i < 2 + 2p \tag{3.31}$$

koşulları sağlandığında yakınsar.

Denklem (3.31)'de p ve ε, bu çalışmada sırasıyla μ ve $(1-\mu)\eta$'ya karşılık gelir. Eğer (3.31)'i bu çalışmada kullanılan terimlere dönüştürülürse, (3.31)'deki ikinci eşitsizlik aşağıdaki gibi yazılabilir;

$$0 < (1-\mu)\eta\kappa_i < 2 + 2\mu\,. \tag{3.32}$$

Buradan $0 < \eta\kappa_i < \dfrac{2+2\mu}{1-\mu}$ eşitsizliğini veya

$$\mu > \frac{\eta\kappa_i - 2}{\eta\kappa_i + 2} \;, \quad i = \overline{1,n} \tag{3.33}$$

elde edilir. (3.31) bu çalışmada Teorem 1'de verilen, (3.8) sisteminin kararlılığı için momentum faktörünün sağlaması gereken koşula denktir. (3.33)'de $\eta\kappa_i \le 2$, $i = \overline{1,n}$ olduğunda μ parametresi için alt sınır sıfır olacaktır ($0 < \mu < 1$ olduğu düşünülürse) aksi halde eğer $\eta\kappa_i > 2$ ise o zaman μ parametresinin alt sınırı $\max\limits_i \dfrac{\eta\kappa_i - 2}{\eta\kappa_i + 2}$ olacaktır. Bu durumda, (3.8) sisteminin yakınsak olması için $\dfrac{\eta\kappa_i - 2}{\eta\kappa_i + 2} < \mu < 1$ koşulunun sağlanması gerekir. Bu nedenle (Qian 1999)'daki sonuç 3 şu şekilde yazılabilir:

Açıklama 1: (3.14) ile tanımlanmış sistem ancak ve ancak

$$\frac{\eta\kappa_i - 2}{\eta\kappa_i + 2} < \mu < 1, \quad i = 1,2,...,n$$

koşulu sağlandığında kararlıdır.

İyi bir yakınsama sağlayan durum (Qian 1999)'daki (35) formülü ile reel köklerin birbirine eşit olduğu $\lambda_{i,1} = \lambda_{i,2}$, yani (3.15) kuadratik denkleminin diskriminantının sıfıra eşit olduğu durum olarak ifade edilmiştir.
(Qian 1999)'daki (35) formülü

$$p = (1 - \sqrt{\varepsilon\kappa_i})^2 \tag{3.34}$$

şeklinde verilmiştir.
$\mu = p,\ \ \varepsilon = (1-\mu)\eta$ eşitlikleri dikkate alınırsa, (3.34) aşağıdaki şekilde yazılabilir

$$\mu = \left(1 - \sqrt{(1-\mu)\eta\kappa_i}\right)^2. \tag{3.35}$$

(3.35)'den

$$\mu = \frac{(1-\eta\kappa_i)^2}{(1-\eta\kappa_i)^2} \tag{3.36}$$

bulunur. (3.36)'daki $\frac{(1-\eta\kappa_i)^2}{(1+\eta\kappa_i)^2}$ ifadesi (3.19)'da tanımlanmış $S(\eta\kappa_i)$'dir. (3.19)'a göre, (3.36) sağlandığında $D(\mu) = 0$ olur, diğer bir deyişle (3.15) kuadratik denklemin veya (3.17) kuadratik formun diskriminantı sıfır olur ve $\mu = S(\eta\kappa)$ değerini alır. Buna göre de aşağıdaki açıklama yapılabilir.

Açıklama 2. Eğer μ momentum faktörü verilen bir η öğrenme oranı ve κ_i özdeğerleri için (3.36)'yı sağlarsa, o zaman (3.13) ün x_i' bileşeni için iyi bir yakınsama hızı elde edilmiş olur.

Bu sonuca dayanarak, her x_i', $i = 1,...,n$ bileşeni için uygun bir μ_i seçilmesi fikri akla gelebilir. Ancak, buradaki amaç (3.8) sisteminin hızlı yakınsamasını gerçekleştirmektir. Bu durumda, $x = Qx'$ dönüşümü, x_i bileşeninin

x_j''lerin ($j = 1, 2, ..., n$) bir doğrusal kombinasyonu olduğunu gösterir ve bu nedenle (3.36) ya uygun μ seçimi zorlaşır.

Açıklama 2'ye benzer öneriler (Torii ve Hagan 2002)'de de yer alır. Torii ve Hagan (2002) verilen bir öğrenme oranı için, λ özdeğerlerinin kompleks kalmasını sağlayan en küçük momentum faktörünün iyi bir yakınsama sağlayabileceğini açıklamıştır. Öyle ki, Torii ve Hagan (2002) momentum faktörünü genel olarak aşağıdaki gibi seçmeyi önerir

$$\mu = \max_i \frac{(1-\eta\kappa_i)^2}{(1+\eta\kappa_i)^2} = \max_i S(\eta k_i).$$

Bu aslında Qian'ın (1999) vermiş olduğu öneriye benzerdir.

3.4 Yakınsama Hızı

(Torii ve Hagan 2002)'de açıklandığı gibi, momentumlu gradyan düşümü algoritmasının yakınsama hızı, λ özdeğerlerinin büyüklüğüne bağlıdır; λ *özdeğerleri kompleks olduğunda, büyüklükleri ne kadar küçük olursa yakınsama o kadar hızlı olur.* Bu verilen bir öğrenme oranı için $\mu = \max_i S(\eta\kappa_i) = \max_i \frac{(1-\eta\kappa_i)^2}{(1+\eta\kappa_i)^2}$ seçiminin genelde daha iyi bir yakınsama sağlayacağını gösterir. Torii ve Hagan (2002), η öğrenme oranının uygun seçimi üzerine herhangi bir inceleme yapmamıştır. Aslında, η öğrenme oranının daha iyi bir seçimi kompleks λ özdeğerlerin büyüklüklerini daha da küçültecektir. Bu tez çalışmasında uygun $\eta = \eta^0$'ın aşağıdaki minimax probleminden belirlenmesi önerilir:

$$\max_i \frac{(1-\eta^0\kappa_i)^2}{(1+\eta^0\kappa_i)^2} = \min_{\eta>0} \max_i \frac{(1-\eta\kappa_i)^2}{(1+\eta\kappa_i)^2}. \qquad (3.37)$$

Bu durumda, momentum faktörü $\mu = \mu^0 = \max_i \frac{(1-\eta^0\kappa_i)^2}{(1+\eta^0\kappa_i)^2}$ olarak seçildiğinde iyi

bir yakınsama hızı elde edilmiş olur.

Simetrik pozitif tanımlı H matrisinin özdeğerlerinin κ_n en küçük ve κ_1 en büyük olmak üzere şu şekilde sıralandığı varsayılsın: $0 < \kappa_n \le \kappa_{n-1} \le ... \le \kappa_2 \le \kappa_1$. Bu durumda, $S_i(\eta) = S(\eta\kappa_i) = \dfrac{(1-\eta\kappa_i)^2}{(1+\eta\kappa_i)^2}$, $i = 1,2,...,n$ foknsiyonlarının η'ya göre grafikleri şekil 3.4 de gösterilmiştir. $S_1(\eta)$ ve $S_n(\eta)$ fonksiyonlarının sıfırdan farklı olan keşisim noktası $\eta_{1,n}$ ile işaretlensin. $S_1(\eta) = S_n(\eta)$ eşitliğinden, diğer bir ifadeyle $\dfrac{(1-\eta\kappa_1)^2}{(1+\eta\kappa_1)^2} = \dfrac{(1-\eta\kappa_n)^2}{(1+\eta\kappa_n)^2}$'den $\eta_{1,n} = \dfrac{1}{\sqrt{\kappa_1\kappa_n}}$ olduğu bulunur.

Lemma 1. i) Eğer $0 < \eta \le 1/\sqrt{\kappa_i\kappa_n}$ ise, o zaman $S_n(\eta) \ge S_i(\eta),\ i \in \{1,2,...,n-1\}$.

ii) Eğer $\eta \ge 1/\sqrt{\kappa_1\kappa_i}$ ise, o zaman $S_1(\eta) \ge S_i(\eta),\ i \in \{2,...,n\}$.

İspat. i) $S_n(\eta) \ge S_i(\eta),\ i \ne n$, başka bir ifadeyle

$$\frac{(1-\eta\kappa_n)^2}{(1+\eta\kappa_n)^2} \ge \frac{(1-\eta\kappa_i)^2}{(1+\eta\kappa_i)^2}, \qquad \eta > 0 \tag{3.38}$$

eşitsizliğini ele alalım. (3.38), $\kappa_n = \kappa_i$ olduğunda herhangi bir $\eta > 0$ için sağlanır. Bu nedenle, $\kappa_n < \kappa_i$ olduğu durum incelendiğinde (3.38)'den

$$2\eta(\kappa_i - \kappa_n)(2 - 2\eta^2\kappa_i\kappa_n) \ge 0$$

elde edilir. $\eta(\kappa_i - \kappa_n) > 0$ olduğu için, $2 - 2\eta^2\kappa_i\kappa_n \ge 0$'dır. Buradan (3.38)'in $0 < \eta \le 1/\sqrt{\kappa_i\kappa_n}$ aralığındaki herhangi bir η için sağlandığı elde edilir.

ii) Şimdi $S_1(\eta) \ge S_i(\eta)$, $i \ne 1$ veya

$$\frac{(1-\eta\kappa_1)^2}{(1+\eta\kappa_1)^2} \ge \frac{(1-\eta\kappa_i)^2}{(1+\eta\kappa_i)^2}, \qquad \eta > 0$$

eşitsizliği düşünülürse, bu eşitsizlik

$$2\eta(\kappa_i - \kappa_1)(2 - 2\eta^2\kappa_i\kappa_1) \geq 0 .$$

şeklinde yeniden yazılabilir. $\kappa_1 > \kappa_i$ için $\eta(\kappa_i - \kappa_1) < 0$ olduğundan $2 - 2\eta^2\kappa_i\kappa_1) \leq 0$ 'dır ve buradan $\eta \geq 1/\sqrt{\kappa_1\kappa_i}$ için $S_1(\eta) \geq S_i(\eta)$ eşitsizliğinin sağlandığı sonucuna varılır. ■

Teorem 2. $\eta^0 = 1/\sqrt{\kappa_1\kappa_n}$, minimax problemi (3.37)'nin tek çözüm noktasıdır:

$$\min_{0<\eta} \max_i S_i(\eta) = \max_i S_i(\eta^0) .$$

İspat. Öncelikle $\tilde{S}(\eta) = \max_i S_i(\eta) = \begin{cases} S_n(\eta) & , \quad 0 < \eta \leq \eta^0 \\ S_1(\eta) & , \quad \eta \geq \eta^0 \end{cases}$ olduğunu gösterelim.

$0 < \kappa_n \leq \kappa_{n-1} \leq ... \leq \kappa_{i+1} \leq \kappa_i \leq ... \leq \kappa_2 \leq \kappa_1$ ilişkisinden yararlanarak

$$\frac{1}{\sqrt{\kappa_1\kappa_n}} \leq \frac{1}{\sqrt{\kappa_2\kappa_n}} \leq ... \leq \frac{1}{\sqrt{\kappa_i\kappa_n}} \leq ... \leq \frac{1}{\sqrt{\kappa_{n-1}\kappa_n}}$$

yazılabilir. Bu nedenle, $0 < \eta \leq 1/\sqrt{\kappa_1\kappa_n}$ aralığında, Lemma 1' e göre

$$S_n(\eta) \geq S_i(\eta), \quad i = 1, 2, ..., n-1 .$$

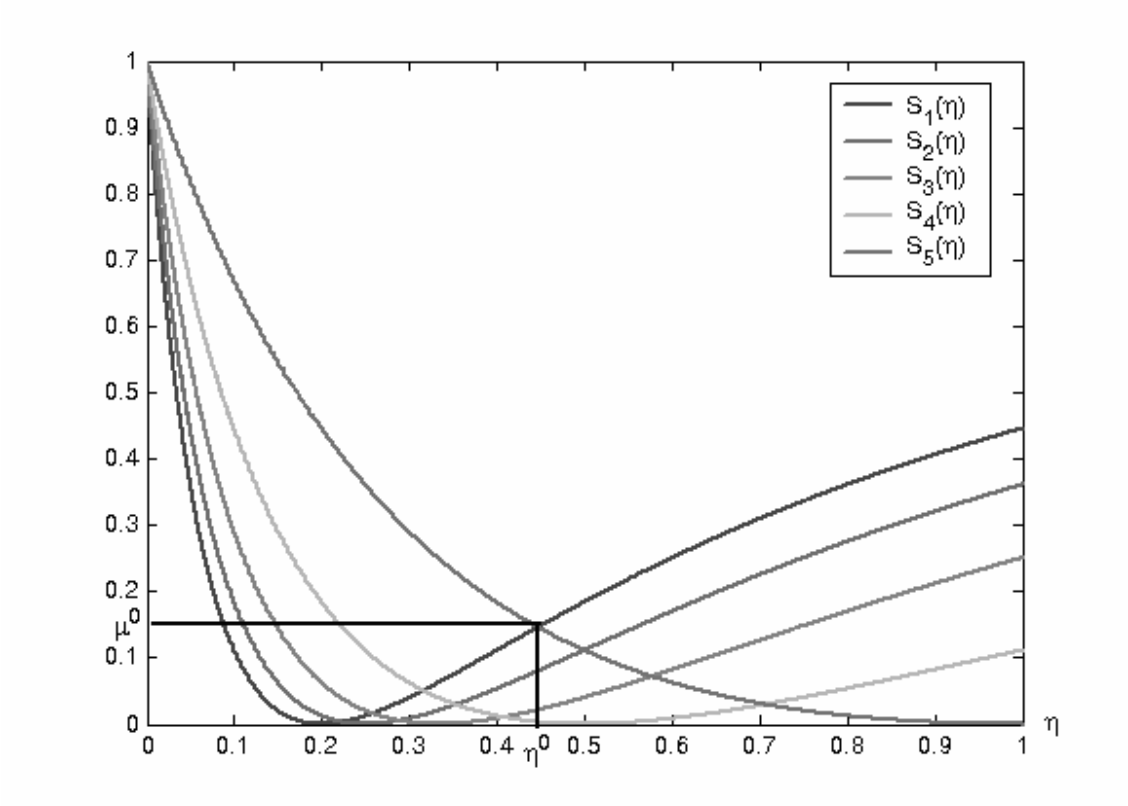
Diğer taraftan,

$$\frac{1}{\sqrt{\kappa_1\kappa_2}} \leq \frac{1}{\sqrt{\kappa_1\kappa_3}} \leq ... \leq \frac{1}{\sqrt{\kappa_1\kappa_i}} \leq ... \leq \frac{1}{\sqrt{\kappa_1\kappa_n}}$$

olduğu için yine Lemma 1' e göre $\eta \geq 1/\sqrt{\kappa_1\kappa_n}$ aralığında $S_1(\eta) \geq S_j(\eta), \quad j = 2, ..., n$ eşitsizlikleri sağlanır.

$\eta = \eta^0$ da, $\tilde{S}(\eta^0) = S_1(\eta^0) = S_n(\eta^0) = \dfrac{\left(\sqrt{k_1} - \sqrt{k_n}\right)^2}{\left(\sqrt{k_1} + \sqrt{k_n}\right)^2}$. Şimdi η^0'ın $\tilde{S}(\eta)$'nın

minimumu olduğu gösterilebilir. Kesim 3.2'de verilmiş olan $S(\eta\kappa)$'nın özellikleri ele alındığında, $S_n(\eta) = S(\eta\kappa_n)$, $0 < \eta\kappa_n \le 1$ aralığında azalandır. $\eta^0\kappa_n = \dfrac{\kappa_n}{\sqrt{\kappa_1\kappa_n}} \le 1$ olduğu için, $S_n(\eta)$, $0 < \eta \le \eta^0$ aralığında minimumunu $\eta = \eta^0 = \dfrac{1}{\sqrt{\kappa_1\kappa_n}}$ noktasında alır. Aynı şekilde, $\eta = \eta^0$'ın $\eta^0 \le \eta < +\infty$ aralığında $S_1(\eta)$'nın da minimumu olduğu açıktır. Bu nedenle $\eta = \eta^0$, $\tilde{S}(\eta)$ fonksiyonunun minimumudur ve η^0'a karşılık gelen momentum faktörü $\mu^0 = \tilde{S}(\eta^0) = \dfrac{\left(\sqrt{\kappa_1} - \sqrt{\kappa_n}\right)^2}{\left(\sqrt{\kappa_1} + \sqrt{\kappa_n}\right)^2}$ olur.■



Şekil 3.4 $\tilde{S}(\eta) = \max_i S_i(\eta)$ grafiği ve etkin η^0, μ^0 parametreleri.

Sonuç olarak, etkin öğrenme oranı ve momentum faktörü

$$\eta^0 = \frac{1}{\sqrt{\kappa_1 \kappa_n}}, \qquad \mu^0 = \frac{\left(\sqrt{\frac{\kappa_1}{\kappa_n}} - 1\right)^2}{\left(\sqrt{\frac{\kappa_1}{\kappa_n}} + 1\right)^2}, \; k_n > 0. \tag{3.39}$$

Formülleri ile verilir. (3.39) kullanılarak aşağıdaki ilişkiler elde edilebilir:

$$(1-\mu^0)\eta^0 = \frac{4}{\left(\sqrt{\kappa_1} + \sqrt{\kappa_n}\right)^2} \tag{3.40}$$

$$\text{Eğer } \frac{\kappa_1}{\kappa_n} \to 1 \text{ ise, } \mu^0 \to 0 \tag{3.41}$$

$$\text{Eğer } \frac{\kappa_1}{\kappa_n} \to \infty \text{ ise, } \mu^0 \to 1 \tag{3.42}$$

$\eta = \eta^0$ ve $\mu = \mu^0$ seçimleri kullanıldığında, (3.40)'dan (3.8) algoritmasındaki gradyan terimi üzerinde Hessian matrisinin sadece en büyük ve en küçük özdeğerlerinin etkisi olduğu anlaşılır. Gradyana bağlı olan adım büyüklüğü Hessian matrisinin ilgili özdeğerleri büyüdüğünde küçülür.

Denklem (3.41), Hessian matrisinin özdeğerleri birbirine yakın olduğunda momentum faktörünün etkisinin azaldığını gösterir. Hessian matrisinin özdeğerlerinin aralığı geniş olduğunda ise momentum faktörünün etkisi artar.

3.5 Deneyler

Bu kesimdeki deneylerin ilk kısmında, hata fonksiyonu kuadratik olduğunda, küçük boyutlu problemlerde kesim 3.4'de önerilen etkin öğrenme oranı ve momentum faktörünün gradyan düşümü algoritmasının yakınsama tavrını nasıl değiştirdiği ve ne kadar hızlandırdığı test edilmiş ve bu etkin değerlerin belirlenen parametre aralığındaki optimum değerlerle ne kadar örtüştüğü gözlemlenmiştir. Bunun için farklı özdeğer dağılımlarına sahip Hessian matrisli problemler oluşturarak öncelikle geçerli parametre aralığındaki tüm

kombinasyonlar denenmiş ve en iyi yakınsama performansına sahip parametre çifti tespit edilmiştir. Daha açık bir ifadeyle, önce η öğrenme oranı [0.01, 0.99] aralığından seçilmiş ve her bir öğrenme oranı için momentum faktörünün [0.01, 0.99] aralığındaki tüm değerleri momentumlu gradyan düşümü algoritmasında denenmiştir. Bu işlem sonucunda [0.01, 0.99] aralığındaki öğrenme oranı ve momentum faktörü olası çifti için tüm kombinasyonlar denenmiş olur. Bu deneylerin sonucunda en iyi performansa sahip öğrenme oranı ve momentum faktörü çifti belirlenir. Farklı özdeğer dağılımlarına sahip küçük boyutlu problemler için çizelge 3.1'in ikinci sütununda bu değerler verilmiştir. Daha sonra önerilen etkin öğrenme oranı ve momentum faktörü (3.39) formülü kullanılarak hesaplanmış ve (3.8) algoritması bu etkin parametre değerleriyle çalıştırılmıştır. Çizelge 3.1'in üçüncü sütununda hesaplanan etkin parametre değerleri verilmiş ve genel olarak sonuçlar özetlenmiştir. Dikkat edilmelidir ki, küçük boyutlu problemlerde bile en iyi öğrenme oranı ve momentum faktörünü denemelerle bulmak ciddi bir zaman alır ve bu değerler problemden probleme değişiklik gösterir. Etkin öğrenme oranı ve momentum faktörünün (3.39) formülü ile hesaplanması ise oldukça kolaydır.

Çizelge 3.1 5 boyutlu rassal problemlerde gerçekleştirilen deneylerin sonuçları

Özdeğerler	**Deneme ile bulunan öğrenme parametreleri**			**(3.39)'la hesaplanan etkin öğrenme parametreleri**		
	η	μ	İterasyon sayısı	η^0	μ^0	İterasyon sayısı
1, 2, 3, 4, 5	0.45	0.15	18	0.45	0.15	21
75.83, 37.95, 40.49, 56.21, 55.31	0.02	0.05	15	0.02	0.03	14
2.29, 7.19, 17.67, 19.46, 18.68	0.15	0.25	28	0.15	0.24	30
1.60, 138.03, 99.63, 51.02, 62.76	0.06	0.67	98	0.07	0.65	104

Çizelge 3.1'deki sonuçlar incelendiğinde (3.39) kullanılarak hesaplanan etkin öğrenme parametre değerlerinin, deneme yolu ile bulunan en iyi öğrenme parametre değerlerine genel olarak çok yakın olduğu görülür ve deneme yolu ile bulunan en iyi öğrenme parametrelerinin gerçekleştirdiği iyi yakınsama performansını da aynı şekilde gerçekleştirir. Diğer bir deyişle (3.39) formülü, kuadratik hata fonksiyonu durumunda momentumlu gradyan düşümü algoritması için optimale yakın öğrenme parametre değerlerinin elde edilmesine olanak sağlar.

Deneylerin ikinci kısmında, etkin öğrenme oranı ve momentum faktörü momentumlu gradyan düşümü (GDM) algoritmasına uyarlanmıştır ve bu algoritma GDME olarak adlandırılmıştır. GDME algoritmasının performansı kuadratik hata fonksiyonu durumunda geleneksel momentumlu gradyan düşümü GDM ve momentumlu ve adaptif öğrenme oranlı gradyan düşümü (GDX) algoritmalarıyla karşılaştırılmıştır. Algoritmalar karşılaştırılırken rassal olarak oluşturulmuş küçük ve orta ölçekli kuadratik hata fonksiyonlu test problemleri kullanılmıştır. Sonuçlar çizelge 3.2'de özetlenmiştir.

Problemler, problem boyutu n (ağırlıkların sayısı) ve (3.10) amaç fonksiyonundaki Hessian matrisinin (H) koşul sayısı kontrol edilebilecek şekilde oluşturulmuştur. Daha sade olması için, kuadratik fonksiyonun durağan noktasının orijinde olduğu ve orada sıfır değerini aldığı varsayılmıştır. Bu (3.10) daki b ve c terimlerinin her ikisinin de sıfır olmasını gerektirir; b sıfır olmadığı zaman fonksiyonun değeri her noktada b kadar artar, seviye eğrilerinin şekli değişmez, c'nin sıfır olmadığı ve H'nin tersinir olduğu durumda, seviye eğrileri değişmez ama fonksiyonun durağan noktası $x^* = -H^{-1}b$ noktasına kayar. Bu nedenle, (3.10) amaç fonksiyonu sadece H matrisiyle belirlenir. H şu şekilde oluşturulabilir: $H = QKQ^T$, burada Q rassal olarak oluşturulmuş ortogonal matris ve K köşegen elemanları pozitif sayılar olan bir köşegen matristir. Q ortogonal matrisini oluşturmak için, her bir elemanı standart normal dağılımdan seçilen belirli bir kare matrisin QR dekompozisyonu kullanılır. H'nin koşul sayısı K'nın köşegen elemanları tarafından belirlenir ve K'nın köşegen elemanları

$$K_{11} = 1/t'$$

$$K_{ii} = (t')^{u_i},\ i = 2,...,n-1;$$

$$K_{nn} = t';$$

şeklinde seçilir, t' kullanıcı tarafından arzu edilen koşul sayısı t'nin karekökü ve her bir u_i (-1,+1) aralığında uniform değişkendir (Lenard ve Minkoff 1984).

Çizelge 3.2 Farklı boyuta ve koşul sayısına sahip problemlerde deney sonuçları

n=10 (problem boyutu)	**Koşul Sayısı**							
Algoritmalar	10		100		1000		10000	
	Devir	*Zaman (sn)*	*Devir*	*Zaman*	*Devir*	*Zaman*	*Devir*	*Zaman*
GDME	**38**	**0.01**	**157**	**0.04**	**557**	**0.03**	**1967**	**0.06**
GDX	643	0.05	1139	0.09	6411	0.781	60919	7.461
GDM	6370	0.22	19854	1.112	59487	1.933	1.68E+05	5.507
n=100	**Koşul Sayısı**							
Algoritmalar	10		100		1000		10000	
	Devir	*Zaman*	*Devir*	*Zaman*	*Devir*	*Zaman*	*Devir*	*Zaman*
GDME	**40**	**0.06**	**158**	**0.05**	**544**	**0.06**	**1963**	**0.14**
GDX	802	0.13	1564	0.261	6817	1.152	69838	12.147
GDM	6524	0.39	19395	1.151	58003	3.545	1.73E+05	10.646
n=1000	**Koşul Sayısı**							
Algoritmalar	10		100		1000		10000	
	Devir	*Zaman*	*Devir*	*Zaman*	*Devir*	*Zaman*	*Devir*	*Zaman*
GDME	**42**	**9.474**	**164**	**7.03**	**567**	**12.658**	**1950**	**23.965**
GDX	881	16.213	1546	29.512	7631	144.22	74674	1384.3
GDM	6985	67.677	20912	204.15	63781	623.49	1.82E+05	1819.3

Kuadratik bir hata fonksiyonu durumunda algoritmalar karşılaştırıldığında, etkin öğrenme parametereleriyle çalışan momentumlu gradyan düşümü algoritması GDME'nin, hem devir sayılarına hem de zamana (sn.) göre diğer algoritmalardan daha iyi bir yakınsama performansına sahip olduğu görülür. GDME hem devir sayılarına hem de zamana göre karşılaştırıldığında tüm problemlerde en iyi performansı göstermiştir. Özellikle, problem boyutunun ve koşul sayısının büyük olduğu durumlarda etkin parametrelerin başarısı daha iyi görülür.

Hessian'ın özdeğer dağılımının etkisini değerlendirmek için yapılan deneyler, algoritmaların yakınsama performansını esas olarak problemin koşul sayısının etkilediğini gösterir. En büyük ve en küçük özdeğerin dışındaki özdeğerler algoritmaların yakınsama performansı üzerinde bir etkiye sahiptir ama bu etki en büyük κ_1 ve en küçük κ_n özdeğerinin beraber yaptığı etkiyle karşılaştırıldığında oldukça küçüktür. Bu deki deneylerden elde edilen sonuçlar (Çizelge 3.3) daha önce ifade edilen (3.40), (3.41) ve (3.42) ilişkilerinden çıkarılan düşünceleri destekler.

Çizelge 3.3 Farklı özdeğer dağılımına sahip problemlerde deney sonuçları

Durum	**Özdeğerler**	**Öğrenme Oranı η^0**	**Momentum Faktörü μ^0**	**Gradyan Katsayısı**	**İterasyon sayısı**
$\frac{\kappa_1}{\kappa_n} \to 1$	0.001,0.0009, 0.0008, 0.0007, 0.0006	1291	0.016133	1270.2	12
	0.1, 0.09, 0.08, 0.07, 0.06	12.91	0.016133	12.702	12
	100000, 99999, 99998, 99997, 99996	1.00E-05	1.00E-10	1.00E-05	4
	1000, 999.9, 999.8, 999.7, 999.6	0.001	1.00E-08	0.0010002	4
$\frac{\kappa_1}{\kappa_n} \to \infty$	1, 0.9, 0.8, 0.7, 1e-5	316.23	0.98743	3.9748	4987
	1, 0.9, 0.8, 0.7, 1e-6	1000	0.99601	3.992	16601
	1, 0.9, 0.8, 0.7, 1e-7	3162.3	0.99874	3.9975	56253
	1, 0.9, 0.8, 0.7, 1e-10	1.00E+05	0.99996	3.9999	>100000
	10000, 0.9, 0.8, 0.7, 1	0.01	0.96079	3.92E-04	2384
	100000, 0.9, 0.8, 0.7, 1	0.0031623	0.98743	3.97E-05	8425
	1000000, 0.9, 0.8, 0.7, 1	0.001	0.99601	3.99E-06	29293
	10000, 9999, 9998, 9997, 1	0.01	0.96079	3.92E-04	1947
	100000, 99999, 99998, 99997, 1	0.0031623	0.98743	3.97E-05	6537
	1000000, 999999, 999998, 999997, 1	0.001	0.99601	3.99E-06	22090

4. ETKİN ÖĞRENME PARAMETRELERİ İLE GERİ YAYILIM ALGORİTMASININ HIZLANDIRILMASI

Üçüncü bölümde kuadratik hata fonksiyonu durumunda momentumlu gradyan düşümü algoritmasının kararlılığı incelenmiş ve yakınsama hızını oldukça iyileştiren etkin öğrenme oranı ve momentum faktörü belirlenmişti. Bu bölümde ise kuadratik hata fonksiyonu durumunda elde edilen etkin öğrenme parametrelerinin, hatanın ağ ağırlıklarının herhangi bir doğrusal olmayan fonksiyonu olduğu genel duruma uyarlanabilmesi amaçlanmaktadır. Etkin öğrenme parametrelerini hesaplamak için gerekli olan bilgi ağın herhangi bir ağırlık noktasındaki Hessian matrisinin en büyük ve en küçük özdeğeridir. İhtiyacımız olan bilgi küçük olmasına karşın, bu bilgiye ulaşmak oldukça maliyetlidir. Diğer yandan etkin parametreler eğitim sürecinin her adımında hesaplanacağı için hesaplama ve zaman maliyeti, problem boyutuna, veri kümesine ve ağırlıkların güncellenme yöntemine bağlı olarak katlanarak artar. Her adımda Hessian'ın hesaplanması oldukça zor ve hatta bazı durumlarda imkânsız hale gelebilir.

Üçüncü bölümde elde edilen etkin parametrelerin genel durumda momentumlu gradyan düşümü algoritmasına uyarlanması, ağın Hessian'ının veya Hessian'ın en büyük ve en küçük özdeğerlerinin hesaplanması gibi yeni problemler oluşturur. Momentumlu gradyan düşümü algoritmasının pratikte hızlandırılması ancak bu problemlerin etkin ve makul bir maliyetle çözülmesine bağlıdır. Alt kesimlerde bu problemlerin çözümüne ait yaklaşımlar ve bu yaklaşımlar çerçevesinde geliştirilmiş algoritmalar incelenecektir.

Literatürde genel olarak çevrimiçi (online) ve toplu (batch) olmak üzere iki farklı ağırlık güncelleme yöntemi kullanılır (Bishop 1995; Haykin 1999). Çevrimiçinde öncelikle bir örneklem genişliği belirlenir ve ağırlık güncellemesi bu örneklemin ağa sunulmasının ardından gerçekleşir. Eğer belirli bir genişlikte örneklem yerine her defasında sadece bir örnek (eğitim çifti) ağa sunulursa buna tam çevrimiçi güncelleme denir. Topluda ise ağırlık güncellemesi tüm eğitim örnekleri ağa sunulduktan sonra gerçekleşir. (3.10) Deterministik hata fonksiyonu durumunda momentumlu gradyan düşümü algoritmasının yakınsamasını oldukça

hızlandıran etkin parametreler ancak ağın gerçek Hessian matrisi H kullanılarak belirlenir. Genel durumda gerçek Hessian'a ve dolayısıyla Hessian'ın en büyük ve en küçük özdeğerlerinin yaklaşık tahminleri ancak tüm veri kümesi kullanılarak elde edilebilir. Tüm veri kümesi kullanılmazsa, gerçekleştirilen hesaplamaların sonucunda elde edilen Hessian, gerçek Hessian'ın gürültülü tahminleri olur ve dolayısıyla özdeğerler de gerçek değerlerin gürültülü tahminleri olur. Bu durum, bu çalışmada incelenmeyen farklı bir problemdir. Daha sonra anlatılan yaklaşımlar temel olarak toplu ağırlık güncellemesine göre tasarlanmıştır ve tüm veri kümesini kullanır.

Şimdi, bir veya daha çok gizli katmana sahip çok katmanlı bir ileri beslemeli sinir ağının eğitiminde kullanılan momentumlu geri-yayılım algoritmasını ele alınsın. $x \in R^n$ ağırlık vektörü, $E(x)$ tüm girdi örnekleri kümesi için hata kareler toplamını temsil eden hata fonksiyonu olsun. Gizli katmanlarda sigmoid biçimli aktivasyon fonksiyonları kullanıldığı için $E(x)$ hata fonksiyonu genellikle doğrusal olmayan bir fonksiyondur. *Geri-yayılım* $E(x)$ hata fonksiyonunun minimumunun bulunması için bir *gradyan düşümü* algoritmasıdır ve bu nedenle

$$x_{t+1} - x_t = -(1-\mu)\eta \nabla_x E(x_t) + \mu(x_t - x_{t-1}), \tag{4.1}$$

şeklinde yazılabilir, burada $\nabla_x E(x_t) = \left(\frac{\partial E(x)}{\partial x_1}, ..., \frac{\partial E(x)}{\partial x_n} \right)_{x=x_t}$, $E(x)$ fonksiyonunun $x = x_t$ noktasındaki uygun gradyan vektörü, η öğrenme oranı ve μ momentum faktörüdür. (4.1), ağırlık vektörünün t adımındaki toplu güncelleme formülüdür. Algoritmaya bağlı olarak η ve μ katsayıları her adımda sabit veya değişken olabilir.

Üçüncü bölümde, (3.10) kuadratik fonksiyonunun minimizasyonunda kullanılan momentumlu gradyan düşümü algoritmasının yakınsama hızı (3.39) etkin öğrenme parametreleri kullanılarak oldukça geliştirilmiştir. Etkin öğrenme parametrelerinin, hatanın ağ ağırlıklarının herhangi bir doğrusal olmayan fonksiyonu olduğu genel durumdaki (4.1) algoritmasına uyarlanabilmesi için

$E(x)$ hata fonksiyonunun her adımda (3.10) kuadratik formuna indirgenmesi gerekir ve bu şekilde etkin parametreler kullanılabilir. Bunun için öncelikle her adımda yerel kuadratik yaklaşım kullanılır.

4.1 Momentumlu Geri-Yayılım Algoritmasında Hata Fonksiyonunun Yerel Kuadratik Yaklaşımı

Belirli bir $\hat{x}$ ağırlığı civarında ikinci dereceden Taylor açılımını kullanarak hata fonksiyonu $E(x)$ 'in yerel kuadratik yaklaşımı yapılabilir:

$$E(x) \approx E(\hat{x}) + (x-\hat{x})^T \nabla_x E(\hat{x}) + \frac{1}{2}(x-\hat{x})^T H(x-\hat{x}) . \tag{4.2}$$

Burada H Hessian matrisidir:

$$H = (h_{ij}) \ , \ h_{ij} = \left. \frac{\partial^2 E(x)}{\partial x_i \partial x_j} \right|_{\hat{x}} . \tag{4.3}$$

$E(x)$ hata fonksiyonunun, yerel minimum olan özel bir x^* noktası civarındaki yerel kuadratik yaklaşımında ise, $\nabla E(x^*) = 0$ olacağı için, (4.2)' de doğrusal terim yer almaz ve bu durumda (4.2)

$$E(x) = E(x^*) + \frac{1}{2}(x-x^*)^T H(x-x^*)$$

şekline gelir.

Genel duruma aykırı olmadığı için, (4.2)'de $\hat{x} = 0$ alınabilir ve bu durumda (4.2)

$$F(x) = \frac{1}{2} x^T Hx - b^T x + c \tag{4.4}$$

kuadratik fonksiyonu şeklinde yazılabilir, burada $b = -\nabla_x E(\hat{x})$ ve $c = E(\hat{x})$. Bu şekilde, her t adımında (4.1) BPM algoritması

$$x_{t+1} - x_t = -(1-\mu_t)\eta_t H x_t + \mu_t (x_t - x_{t-1}) + (1-\mu_t)\eta_t b \tag{4.5}$$

veya

$$x_{t+1} = [(1+\mu_t)I - (1-\mu_t)\eta_t H]x_t - \mu_t x_{t-1} + (1-\mu_t)\eta_t b\,, \tag{4.6}$$

şekline dönüşür, burada $b = Hx_0$'dır. (4.5) veya (4.6) uygun olarak (3.8) veya (3.9) algoritmalarının aynısıdır, buradaki tek farklılık η ve μ parametrelerinin t adımına bağlı olarak değişmesidir.

Bölüm 3'de, H Hessian matrisinin pozitif tanımlı olduğu durumda, (4.4) kuadratik hata fonksiyonu için bu algoritmaların hızlı yakınsamasını sağlayan etkin öğrenme oranı ve momentum faktörünün (3.39) formülleri ile belirlendiği gösterildi. Genel hata fonksiyonu durumuna uygun (4.1) algoritması için, her bir t adımında (4.2) hata fonksiyonunun yerel kuadratik yaklaşımını gerçekleştirerek, (3.39) formülleri ile (4.5) algoritmasının hızlı yakınsamasını sağlayan etkin öğrenme oranı η ve momentum faktörü μ'nün belirlenmesi önemli rol oynayabilir. (3.39)'a göre her t adımında etkin η_t^0 ve μ_t^0 katsayıları x_t noktasına uygun simetrik pozitif tanımlı (4.3) Hessian matrisin en büyük ve en küçük özdeğerleri kullanılarak hesaplanır. Bu yaklaşımla (4.1) algoritmasında etkin η_t

ve μ_t parametrelerinin bir dinamik seçimi gerçekleştirilir.

4.2 Geliştirilmiş Momentumlu Geri-Yayılım Algoritmaları için Genel Yapı

Her adımda etkin öğrenme parametreleriyle çalışacak algoritmaların ana yapısı aşağıdaki gibi verilebilir. Bu ana yapı toplu ağırlık güncellemesini kullanır (Bishop 1995; Haykin 1999). Daha sonra momentumlu gradyan düşümü algoritması bu ana yapı çerçevesinde farklı yaklaşımlarla çalışacak şekilde düzenlenecektir.

Algoritma (Ana Yapı)

- x_0 başlangıç noktası ve yakınsama toleransı $\varepsilon > 0$ verilsin,

1. *Başlangıç Hessian matrisinin tanımlanması*

2. $k \leftarrow 0$;

3. Durma koşulu **sağlanmadığı** sürece devam et: (ana döngü)

(a) H_k *(Hessian)'nın en büyük* κ_1 *ve en küçük* κ_n *özdeğerini hesapla;*

(b) η_k ve μ_k 'yı (3.39) formülü ile hesapla;

(c) Arama yönünü belirle, $p_k = -(1-\mu_k)\eta_k \nabla E(x_k) + \mu_k p_{k-1}$;

(d) Ağırlığı güncelle, $x_{k+1} = x_k + \alpha_k p_k$, α_k ' yı doğru arama ile belirle;

(e) $k \leftarrow k+1$;

4. Dur (ana döngü)**.**

(4.1) iterasyonunun her adımında hata fonksiyonunun yerel kuadratik yaklaşımı yapıldıktan sonra etkin öğrenme parametrelerini belirleyebilmek için herhangi bir ağırlık noktasındaki uygun Hessian'ın en büyük ve en küçük özdeğerlerinin hesaplanması gerekir. Bu hesaplamanın etkin bir şekilde ve makul bir maliyetle gerçekleştirilebilmesi için iki farklı yaklaşım önerilmiştir:

1. Hessian matrisinin her adımda iteratif tahminini kullanarak özdeğerlerin yaklaşık olarak hesaplanması (Nocedal ve Wright 1999; Gill, Murray ve Wright 1981).

2. Hessian matrisini hesaplamadan kesin Hessian-vektör çarpımını (Pearlmutter 1994; Moller 1993), Güç iterasyonuyla (Golub ve Van Loan 1996) beraber kullanarak en büyük ve en küçük özdeğerin yaklaşık olarak hesaplanması.

4.3 Hessian Matrisinin İteratif Yaklaşık Hesaplanması

1950'lerin ortalarında, Argonne Ulusal Laboratuarında çalışan W.C. Davidon, uzun bir optimizasyon hesabını gerçekleştirmek için koordinat düşümü metodunu kullanıyordu. O zamanlarda bilgisayarlar çok kararlı olmadığı için, Davidon'un karşılaştığı sorun daha doğrusu yaşadığı hayal kırıklığı, hesaplama bitmeden bilgisayar sisteminin çökmesiydi. Bu nedenle Davidon iterasyonu hızlandırmak için bir yol bulmaya karar verdi. Geliştirdiği algoritma; ilk quasi-Newton algoritması, doğrusal olmayan optimizasyondaki en devrimci fikirlerden biri olmuştur. Kısa bir süre içinde yeni algoritmanın var olan metotlardan çok daha hızlı ve daha güvenilir olduğu Fletcher ve Powell tarafından ispatlanmıştır. Bu olağanüstü ilerleme doğrusal olmayan optimizasyonu bir anda değiştirmiştir. Takip eden yirmi sene boyunca, algoritmanın birçok çeşidi önerilmiş ve yüzlerce makale onların çalışmasına adanmıştır. İlginç olan tarihsel bir ironi, Davidon'un makalesi (1959) yayına kabul edilmemiştir; 1991'de SIAM Journal on Optimization'ın ilk sayısında yayınlanana kadar 30 seneden fazla bir süre teknik rapor olarak kalmıştır.

Quasi-Newton metotları, örneğin dik iniş gibi, her iterasyonda amaç fonksiyonunun sadece gradyanına ihtiyaç duyar. Gradyanlardaki değişim ölçülerek, amaç fonksiyonun bir modeli oluşturulur ve ikinci türevlere ihtiyaç duyulmadığından, quasi-Newton metotları bazen Newton'un metodundan daha etkilidir.

Hessian'ın iteratif olarak yaklaşık hesaplanmasına quasi-Newton yöntemlerinde geniş yer verilmiştir. En popüler quasi-Newton algoritmalarından olan BFGS, Broyden, Fletcher, Goldfarb ve Shanno tarafından geliştirilmiştir (Nocedal ve Wright 1999; Gill, Murray ve Wright 1981, Fletcher 1987). Bu

kesimde, BFGS algoritmasının türetilmesi (ve BFGS ile yakından ilişkili DFP algoritması), gelişim süreci ve bazı önemli teorik özellikleri verilecektir.

x_k noktasında genel hata fonksiyonu $E(x)$'e uygun kuadratik modeli dikkate alınırsa:

$$m_k(p) = E(x_k) + \nabla E(x_k)^T p + \frac{1}{2} p^T H_k p \,. \tag{4.7}$$

Burada H_k, her iterasyonda düzeltilecek veya güncellenecek $n \times n$ boyutlu simetrik pozitif tanımlı bir Hessian matrisidir. (4.7) fonksiyonunun $p = 0$'daki değeri ve gradyanı sırasıyla $E(x_k)$ ve $\nabla E(x_k)$'dır. Bu konveks kuadratik modeli minimize eden p_k yönü

$$p_k = -H_k^{-1} \nabla f_k \,, \tag{4.8}$$

biçiminde verilebilir, p_k quasi-Newton algoritmasında arama yönü olarak kullanılır. Yeni iterasyon

$$x_{k+1} = x_k + \alpha_k p_k \,, \tag{4.9}$$

adım uzunluğu α_k (2.26) Wolfe koşullarını sağlayacak şekilde seçilir. Bu iterasyon doğru aramalı Newton metoduna oldukça benzerdir; temel fark ise gerçek Hessian'ın yerine yaklaşık Hessian H_k kullanılır.

Davidon (1959), H_k'yı her iterasyonda yeniden hesaplamak yerine, en yakın adımlarda ölçülen eğriselliği yansıtacak basit bir güncelleme önermiştir. Yeni bir x_{k+1} iterasyonu oluşturulsun ve

$$m_{k+1}(p) = f_{k+1} + \nabla f_{k+1}^T p + \frac{1}{2} p^T H_{k+1} p \tag{4.10}$$

şeklinde yeni bir kuadratik model oluşturulsun. Bu durumda en son adımda kazanılan bilgilere dayanarak H_{k+1} üzerinde hangi koşulların konulması gerekir? Mantıklı bir koşul; m_{k+1}'in gradyanının amaç fonksiyonu f'nin gradyanıyla en son iki iterasyon x_k ve x_{k+1}'de eşleşmesidir. $\nabla m_{k+1}(0)$ kesin olarak ∇f_{k+1} olduğu için, bu koşulların ikincisi otomatik olarak sağlanır. İlk koşul matematiksel olarak

$$\nabla m_{k+1}(-\alpha_k p_k) = \nabla f_{k+1} - \alpha_k H_{k+1} p_k = \nabla f_k \,. \tag{4.11}$$

biçiminde yazılabilir, tekrar düzenlenirse

$$H_{k+1}\alpha_k p_k = \nabla f_{k+1} - \nabla f_k \tag{4.12}$$

elde edilir. Notasyonu kolaylaştırmak için

$$s_k = x_{k+1} - x_k, \qquad y_k = \nabla f_{k+1} - \nabla f_k \,, \tag{4.13}$$

olarak tanımlansın. Bu durumda (4.12)

$$H_{k+1} s_k = y_k \,. \tag{4.14}$$

biçiminde yazılabilir. Bu formüle *secant (kesen)* denklemi denir.

Yer değiştirme s_k ve *gradyan değişimi* y_k verildiğinde secant denklemi, s_k'yı y_k'ya dönüştüren H_{k+1} matrisinin simetrik pozitif tanımlı olmasını gerektirir. Bu sadece s_k ve y_k için aşağıdaki *eğrisellik koşulu* sağlandığında gerçekleşebilir;

$$s_k^T y_k > 0 \,, \tag{4.15}$$

f ciddi konveks olduğunda, (4.15) eşitsizliği herhangi iki x_k ve x_{k+1} noktaları için sağlanıyor. Ama bu koşul konveks olmayan fonksiyonlar için her zaman

geçerli değildir. Bu durumda α'yı seçen doğru arama yordamlarına kısıtlamalar koyarak (4.15)'in sağlanmasının açıkça zorlanması gerekir. Aslında, doğru aramaya Wolfe veya ciddi Wolfe koşulları konulursa (4.15)'in sağlanması garanti edilir. Bunu gösterebilmek için, (4.13) ve (2.26)'dan $\nabla f_{k+1}^T s_k \geq c_2 \nabla f_k^T s_k$ olur ve bu nedenle

$$y_k^T s_k \geq (c_2 - 1)\alpha_k \nabla f_k^T p_k . \tag{4.16}$$

$c_2 < 1$ ve p_k azalan bir yön olduğundan, sağ taraftaki terim pozitif olacaktır ve eğrisellik koşulu (4.14) sağlanacaktır.

Eğrisellik koşulu sağlandığında, sekant denklemi (4.14)'ün her zaman bir H_{k+1} çözümü vardır. Aslında, (4.14) sonsuz sayıda çözüm kabul eder, çünkü simetrik bir matriste $n(n+1)/2$ serbestlik derecesi vardır ve sekant denklemi sadece n koşulu temsil eder. Pozitif tanımlılık gerekliliği, n tane ek eşitsizlik oluşturur, tüm temel minörler pozitif olmalıdır, ama bu koşullar kalan serbestlik derecesini yok etmez.

H_{k+1}'i tek olarak belirleyebilmek için, şu ek koşulu koyarız; *sekant denklemini sağlayan tüm simetrik matrisler arasından, H_{k+1}, bir anlamda, şu anki matris H_k'ya en yakın olanıdır.* Diğer bir deyişle, aşağıdaki problem çözülmeye çalışılır

$$\begin{aligned} &\min_H \|H - H_k\|, \\ &H = H^T, \qquad Hs_k = y_k , \end{aligned} \tag{4.17}$$

burada s_k ve y_k (4.15)'i sağlar ve H_k simetrik ve pozitif tanımlıdır. Birçok matris normu (4.17)'de kullanılabilir ve her bir norm farklı bir quasi-Newton metoduna yol açar. (4.17) minimizasyon probleminin kolay çözümüne olanak sağlayan ve ölçek-değişmez optimizasyon metoduna yol açan bir norm,

$$\|A\|_w \equiv \left\|W^{1/2} A W^{1/2}\right\|_F, \tag{4.18}$$

biçiminde tanımlanan ağırlıklandırılmış Frobenius normudur (Golub ve Van Loan 1996).

$\|\bullet\|_F$, şu şekilde tanımlanır, $\|C\|_F^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^2$. W ağırlığı, $Wy_k = s_k$ ilişkisini sağlayan *herhangi* bir matris olarak seçilebilir. Somut olması açısından, $W = \bar{G}_k^{-1}$ şeklinde seçilebilir, burada $\bar{G}_k$,

$$\bar{G}_k = \left[\int_0^1 \nabla^2 f(x_k + \tau\alpha_k p_k) d\tau\right] \tag{4.19}$$

biçiminde tanımlanan ortalama Hessian'dır.

$$y_k = \bar{G}_k \alpha_k p_k = \bar{G}_k s_k \tag{4.20}$$

özelliği Taylor teoreminden görülebilir. Bu ağırlıklandırma matrisi ve Frobenius norm ile, (4.17)'nin tek çözümü

$$\text{(DFP)} \qquad H_{k+1} = (I - \gamma_k y_k s_k^T) H_k (I - \gamma_k y_k s_k^T) + \gamma_k y_k y_k^T . \tag{4.21}$$

$$\gamma_k = \frac{1}{y_k^T s_k}$$

biçiminde verilir. (4.21), Davidon tarafından orijinal olarak 1959 yılında önerilmiştir ve sonradan Fletcher ve Powell tarafından çalışılmış, uygulanmış ve popüler hale gelmiştir. Bu nedenle DFP güncelleme formülü olarak bilinir.

H_k'nın tersi olarak tanımlanan $B_k = H_k^{-1}$ metodun uygulanmasında yararlıdır, çünkü arama yönü (4.8)'in matris-vektör çarpımı anlamında hesaplanmasına olanak verir. Sherman-Morrison-Woodbury formülünü kullanarak

(Ek-1), (4.21)'deki H_k'nın DFP güncellemesine karşılık gelen ters Hessian yaklaşığı B_k'nın güncellenmesi için

$$\text{(DFP)} \qquad B_{k+1} = B_k - \frac{B_k y_k y_k^T B_k}{y_k B_k y_k} + \frac{s_k s_k^T}{y_k^T s_k} \tag{4.22}$$

ifadesi türetilebilir. (4.22)'nin sağ tarafındaki son iki terimin rank-bir matrisler olduğuna dikkat edilirse, B_k bir rank-iki düzenlemeye girer. (4.21)'inde H_k'nın rank-iki düzenlemesi olduğunu görmek kolaydır. Bu quasi-Newton güncellemesinin temel fikridir: İterasyon matrislerini baştan itibaren her iterasyonda yeniden hesaplamak yerine, amaç fonksiyonu hakkında en yakın zamanda gözlemlenen bilgi ile şu anki Hessian yaklaşığında gömülü halde bulunan bilgiyi birleştiren basit bir düzenleme uygulanır.

DFP güncelleme formülü oldukça etkilidir ama kısa zaman sonra şu an için tüm quasi-Newton güncelleme formülleri arasında en etkin olarak kabul edilen BFGS formülü onun yerini almıştır. BFGS güncellemesi (4.21)'e yol açan ifadede basit bir değişiklik yapılarak türetilebilir. Hessian yaklaşıkları H_k üzerine koşullar koymak yerine, onların tersi B_k'lar üzerine benzer koşullar konulur. Güncellenen yaklaşık B_{k+1} simetrik ve pozitif tanımlı olmalıdır ve bu durumda sekant denklemi (4.14)

$$B_{k+1} y_k = s_k .$$

gibi yazılır. B_k'ya yakın olma koşulu şimdi (4.17)'nin aşağıdaki benzeri ile tanımlanır:

$$\begin{aligned} &\min_B \left\| B - B_k \right\|. \\ &B = B^T, \qquad B y_k = s_k . \end{aligned} \tag{4.23}$$

Norm yine ağırlıklandırılmış Frobenius normudur, ağırlık matrisi W ise şimdi $Ws_k = y_k$ koşulunu sağlar (somut olması için, yine W (4.19)'da tanımlanan ortalama Hessian $\bar{G}_k$ ile verilir). (4.17)'nin tek çözümü B_{k+1}

$$\text{(BFGS)} \qquad B_{k+1} = (I - \rho_k s_k y_k^T) B_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T , \qquad (4.24)$$

$$\rho_k = \frac{1}{y_k^T s_k}$$

biçiminde verilir. BFGS güncelleme formülüyle sonuçlanan (4.17) minimizasyon problemine dikkat edilirse, güncellenen Hessian yaklaşıklarının açık olarak pozitif tanımlı olması gerekli değildir. Fakat H_k pozitif tanımlı olduğunda H_{k+1}'inde pozitif tanımlı olacağı aşağıdaki ifadeden kolayca görülebilir. Öncelikle, (4.16)'dan $y_k^T s_k$ pozitifdir, bu nedenle (4.24) güncelleme formülü iyi tanımlanmıştır. Herhangi bir sıfır olmayan vektör için

$$z^T H_{k+1} z = w^T H_k w + \rho_k (z^T s_k)^2 \geq 0$$

olur, burada $w = z - \rho_k y_k (s_k^T z)$ olarak tanımlanmıştı. Sağ taraf sadece $s_k^T z = 0$ olursa sıfır olur, ama bu durumda $w = z \neq 0$ olur, bu da ilk terimin sıfırdan büyük olduğunu ima eder. Bu nedenle, H_{k+1} pozitif tanımlıdır.

Broyden Ailesi

Literatürde BFGS ve DFP gibi daha birçok quasi-Newton güncelleme formülü bulunmaktadır. Bunların içinde özel öneme sahip olan

$$H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} + \phi_k (s_k^T H_k s_k) v_k v_k^T , \qquad (4.25)$$

genel formülüyle ifade edilen bir güncellemeler ailesidir. ϕ_k bir skaler parametredir ve

$$v_k = \left[\frac{y_k}{y_k^T s_k} - \frac{H_k s_k}{s_k^T H_k s_k} \right] \tag{4.26}$$

biçiminde verilir. BFGS ve DFP metotları Broyden sınıfının üyeleridir; $\phi_k = 0$ olduğunda BFGS ve ϕ_k=1 olduğunda ise DFP metotları elde edilir. Bu nedenle (4.25) bu iki metodun "doğrusal kombinasyonu" olarak

$$H_{k+1} = (1-\phi_k) H_{k+1}^{BFGS} + \phi_k H_{k+1}^{DFP} .$$

biçiminde yazılabilir. Bu ilişki Broyden ailesinin tüm üyelerinin sekant denklemini sağladığını gösterir, zaten BFGS ve DFP matrisleri de sekant denklemini sağlar. Ayrıca, $s_k^T y_k > 0$ olduğunda BFGS ve DFP güncellemesi Hessian yaklaşıklarının pozitif tanımlılığını koruduğu için, bu ilişki aynı özelliğin $\phi_k \geq 0$ olduğunda Broyden ailesi için de geçerli olacağını ima eder (Nocedal ve Wright 1999, Gill ve ark. 1981).

Momentumlu gradyan düşümü algoritmasının yakınsama hızını geliştirmek için gerekli olan bilgi Hessian matrisinin en büyük ve en küçük özdeğeridir. Bu bilgiyi elde etmek için quasi-Newton metotlarında kullanılan Hessian matrisinin yaklaşık hesaplanması metotları kendi geliştireceğimiz algoritmalara adapte edilebilir (Tas ve Memmedli 2007, Memmedli ve Tas 2006, Memmedli ve Tas 2007a, Mammadov ve ark. 2007). Bu yaklaşımla kazanılacak önemli bir avantaj ise her adımda Broyden Ailesine üye BFGS ve DFP metotlarıyla iteratif olarak yaklaşık hesaplanan Hessian matrislerinin her zaman pozitif tanımlı olmasıdır. Bu bize pratikte önemli bir avantaj sağlar. Çünkü, genel hata fonksiyonunun yerel kuadratik yaklaşımı ile her adımda elde edeceğimiz Hessian matrisleri gerçekte pozitif tanımlı olmayabilir. Ama BFGS ve DFP metotları bizi bu sorundan kurtarır. Aksi halde, uğraşmamız gereken başka bir problemimiz daha olacaktı.

Daha önce değinildiği üzere bu alt kesimde anlatılan Hessian'ın yaklaşık hesaplanması metotları toplu ağırlık güncellemesi ile çalışır ve sürecin her adımında tüm veri seti kullanılır.

4.4 İteratif Hessian Yaklaşıklarını Kullanarak Geliştirilen Algoritmalar

BFGS ve DFP güncellemelerinde temel fikir, Hessian matrislerini baştan itibaren her iterasyonda yeniden hesaplamak yerine, amaç fonksiyonu hakkında en yakın zamanda gözlemlenen bilgi ile şu anki Hessian yaklaşığında gömülü halde bulunan bilgiyi birleştiren basit bir düzenleme uygulamaktır.

Şimdi kesim 4.2'de verilmiş olan ana algoritma yapısı tekrar hatırlanırsa; 3 (c) adımındaki arama yönünü belirleyebilmek için 3 (b) adımında etkin öğrenme parametreleri η ve μ'nün hesaplanması gerekir. Bunun içinde 3 (a) adımında Hessian matrisinin en büyük ve en küçük özdeğerlerine ihtiyaç duyulur. Bu hesaplamalar oluşan her ağırlık noktasında, diğer bir ifadeyle iterasyonun her adımında gerçekleştirileceği için her adımda Hessian'ı kesin olarak hesaplamak oldukça maliyetlidir. Bunun yerine yukarıda anlatılan quasi-Newton metotlarında kullanılan iteratif Hessian yaklaşıkları kullanılabilir. Burada dikkat edilmesi gereken nokta, BFGS ve DFP güncellemelerinin sağlıklı çalışabilmesi için algoritmanın 3 (c) adımında etkin öğrenme parametreleriyle hesaplanan yönün azalan bir yön olması gerekliliğidir ve bu nedenle 3 (d) adımında Wolfe koşullarını veya ciddi Wolfe koşullarını sağlayan bir adım uzunluğunun belirlenmesi gerekir. Ayrıca BFGS ve DFP güncellemeleri daha önce belirtildiği üzere Hessian matrisinin pozitif tanımlılığını korur ve bu özelliğiyle de ciddi bir avantaj oluşturur.

BFGS Hessian güncellemesi ile çalışacak etkin öğrenme parametreli momentumlu gradyan düşümü algoritması aşağıdaki gibi verilebilir:

Algoritma A1

- x_0 başlangıç noktası ve yakınsama toleransı $\varepsilon > 0$ verilsin,

1. $k \leftarrow 0$;

2. Eğer ilk iterasyon ise;

Başlangıç Hessian'ını birim matris olarak tanımla, $H_0 = I$,

Başlangıç yönünü gradyanın ters yönü olarak seç, $p_0 = -\nabla E(x_0)$

3. Eğer ilk iterasyon değilse

Yakınsama koşulu **sağlanana** kadar devam et: (ana döngü)

(a) Ağırlık değişimini hesapla, $s_k = x_{k+1} - x_k$;

(b) Gradyan değişimini hesapla, $y_k = \nabla E(x_{k+1}) - \nabla E(x_k)$;

(c) Hessian'ı hesapla, $H_{k+1} = H_k - \dfrac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \dfrac{y_k y_k^T}{y_k^T s_k}$;

(d) Hessian'ın en büyük κ_1 ve en küçük özdeğeri κ_n'i yaklaşık olarak hesapla;

(e) η_k ve μ_k'yı (3.39) formülü ile hesapla;

(f) Arama yönünü hesapla, $p_k = -(1-\mu_k)\eta_k \nabla E(x_k) + \mu_k p_{k-1}$;

(g) Ağırlığı güncelle, $x_{k+1} = x_k + \alpha_k p_k$, α_k'yı Wolfe koşullarını sağlayan doğru arama ile belirle;

(h) $k \leftarrow k+1$;

4. Dur (ana döngü)**.**

DFP Hessian güncellemesi ile çalışacak etkin öğrenme parametreli momentumlu gradyan düşümü algoritması ise aşağıdaki gibi verilebilir:

Algoritma A2

- x_0 başlangıç noktası ve yakınsama toleransı $\varepsilon > 0$ verilsin,

1. $k \leftarrow 0$;

2. Eğer ilk iterasyon ise;

Başlangıç Hessian'ını birim matris olarak tanımla, $H_0 = I$,

Başlangıç yönünü gradyanın ters yönü olarak seç, $p_0 = -\nabla E(x_0)$

3. Eğer ilk iterasyon değilse

Yakınsama koşulu **sağlanana** kadar devam et: (ana döngü)

(a) Ağırlık değişimini hesapla, $s_k = x_{k+1} - x_k$;

(b) Gradyan değişimini hesapla, $y_k = \nabla E(x_{k+1}) - \nabla E(x_k)$;

(c) v_k 'yı hesapla, $v_k = \left[\frac{y_k}{y_k^T s_k} - \frac{H_k s_k}{s_k^T H_k s_k} \right]$;

(d) Hessian'ı hesapla, $H_{k+1} = H_k - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} + (s_k^T H_k s_k) v_k v_k^T$;

(e) Hessian'ın en büyük κ_1 ve en küçük özdeğeri κ_n 'i yaklaşık olarak hesapla;

(f) η_k ve μ_k 'yı (3.39) formülü ile hesapla;

(g) Arama yönünü hesapla, $p_k = -(1-\mu_k)\eta_k \nabla E(x_k) + \mu_k p_{k-1}$;

(h) Ağırlığı güncelle, $x_{k+1} = x_k + \alpha_k p_k$, α_k 'yı Wolfe koşullarını sağlayan doğru arama ile belirle;

(i) $k \leftarrow k+1$;

4. Dur (ana döngü)**.**

A1 ve A2 algoritmaları arasındaki tek fark kullandıkları Hessian güncellemeleridir. Aslında BFGS ve DFP güncellemeleri quasi-Newton metotlarında Hessian'ın tersini yaklaşık olarak hesaplar. Burada ise Broyden ailesi yardımıyla Hessian'ın kendisi yaklaşık olarak hesaplanır.

4.5 Hessian Matrisini Hesaplamadan En Büyük ve En Küçük Özdeğerlerinin Hesaplanması

Geriye yayılım ağları gibi büyük öğrenme makineleri binlerce özgün parametreye sahip olabilir. Tüm Hessian matrisini hesaplamak ve daha sonra bu matrisin özdeğerlerini bulmak kaçınılması gereken maliyetli bir işlemdir. Özellikle büyük boyutlu ve çok fazla sayıda örnek içeren geniş çaplı veri kümesine sahip olan problemlerde tüm Hessian'ı iterasyonun her adımında hesaplamak, her adımda kaydetmek ve özdeğerlerini hesaplamak oldukça maliyetli, zor ve bazen de imkânsız olabilir. Bu nedenle, Hessian matrisini hesaplayıp daha sonra özdeğerleri hesaplamak yerine, Hessian'ın en büyük ve en küçük özdeğerinin makul bir zamanda hesaplamak için farklı bir yol bulmak oldukça umut verici gözükür. Bu yaklaşımı gerçekleştirebilmek için güzel ve

pratik bir yol aşağıda verilen iki temel fikre dayanır: 1-Taylor açılımı, 2-Güç iterasyonu. Bu metot genel olarak herhangi bir diferansiyellenebilen amaç fonksiyonuna uygulanabilir.

Taylor açılımı: Gradyanının ağırlık uzayında bir nokta civarında Taylor açılımı

$$\nabla_x(x+\Delta x) = \nabla_x(x) + H\Delta x + O(\|\Delta x\|^2)$$

biçiminde verilir, burada x ağırlık uzayında bir nokta, Δx x'deki değişim (perturbation), ∇_x gradyan, $\partial E / \partial x_i$ kısmi türevlerinin vektörü ve H ise Hessian, x'in elemanlarının her bir çiftine göre ikinci mertebeden türevlerin matrisidir. Burada $\Delta x = rv$ olarak seçilirse; v, bir vektör ve r küçük bir sayıdır. Hv çarpımı

$$H(rv) = rHv = \nabla_x(x+rv) - \nabla_x(x) + O(r^2)$$

biçiminde hesaplanabilir, veya r'ye bölünürse

$$Hv = \frac{\nabla_x(x+rv) - \nabla_x(x)}{r} + O(r). \qquad (4.26)$$

olur. Bu denklem, gradyanın verimli olarak hesaplanabildiği herhangi bir sistemde Hv çarpımını bulmak için basit bir yaklaşık hesap algoritması oluşturur. Bu çarpımı hesaplamak için gereken zaman da gradyanı hesaplamak için gereken zamana eşittir (x noktasındaki gradyanın hesaplandığı varsayılırsa). Doğrudan çarpım n ağırlık olduğu durumda $O(n^2)$ düzeyinde bir işlem olmasına rağmen (4.26) bir $O(n)$ işlemdir. Alışılagelmiş yapay sinir ağları bağlamında, bu işlem iki ileri yayılım ve iki geri-yayılım yapılarak gerçekleştirilebilir.

Bu formülün bir dezavantajı nümerik ve yuvarlama problemlerine karşı duyarlı olmasıdır. $O(r)$ teriminin önemsiz olması için r sabitinin yeterince küçük olması gerekir. Ama r küçüldükçe $x+rv$ toplamında büyük sayılar küçük sayılara eklenir, bu da v'de kesinlik kaybına neden olur. Benzer bir kesinlik kaybı da orijinal gradyan değiştirilmiş gradyandan çıkarıldığında gerçekleşir, çünkü

neredeyse iki eş vektör aralarındaki küçük farkı elde etmek için birbirinden çıkartılır (Pearlmutter 1994).

4.5.1 $\mathrm{R}\{\cdot\}$ Tekniği

Hv çarpımını sadece yaklaşık olarak hesaplayan (4.26) yerine bu çarpımı kesin olarak hesaplayan bir algoritma Pearlmutter (1994) tarafından geliştirilmiştir. Bu teknik aynı zamanda kesim 4.5'de değinilen nümerik problemlerden kurtarır. Bunu yapabilmek için, öncelikle (4.26) denkleminin $r \to 0$ 'a giderken limitini alırız. Sol taraf Hv olarak kalırken sağ taraf bir türevin tanımına uyar ve böylece

$$Hv = \lim_{r \to 0} \frac{\nabla_x(x + rv) - \nabla_x(x)}{r} = \frac{\partial}{\partial r} \nabla_x(x + rv)\bigg|_{r=0} \tag{4.27}$$

biçiminde verilebilir. Herhangi bir sistemin gradyanını hesaplayan bir algoritma, bu yeni niceliği hesaplayacak şekilde kolaylıkla dönüştürülebilir. Bu dönüşümü gerçekleştirmek için anahtar nokta

$$R_v\{f(x)\} \equiv \frac{\partial}{\partial r} f(x + rv)\bigg|_{r=0}, \tag{4.28}$$

gibi bir operatör tanımlamaktır (Pearlmutter 1994). Bu durumda $Hv = R_v\{\nabla_x(x)\}$ olur (Karışıklık olmaması için $R_v\{\cdot\}$, bundan sonra $R\{\cdot\}$ ile gösterilecektir). Daha sonra gradyanın zaten hesaplandığı geri-yayılım algoritmasında $R\{\cdot\}$ operatörü her bir denkleme uygulanabilir. Çünkü $R\{\cdot\}$ bir diferansiyel operatördür, diferansiyel operatörler için geçerli olan kurallara uyar, örneğin:

$$\begin{aligned} R\{cf(x)\} &= cR\{f(x)\} \\ R\{cf(x)\} &= cR\{f(x)\} \end{aligned} \tag{4.29}$$

$$R\{f(x)+g(x)\} = R\{f(x)\} + R\{g(x)\}$$
$$R\{f(x)g(x)\} = R\{f(x)\}g(x) + f(x)R\{g(x)\}$$
$$R\{f(g(x))\} = f'(g(x))R\{g(x)\}$$
$$R\left\{\frac{df(x)}{dt}\right\} = \frac{dR\{f(x)\}}{dt}$$

Dikkat edilirse aynı zamanda

$$R\{x\} = v \tag{4.30}$$

dir. Bu kurallar, normal olarak gradyanı hesaplamada kullanılan denklemlerden yeni bir denklemler kümesi türetmek ve buradan yeni R-değişkenleri kümesi oluşturmak için yeterlidir. Bu yeni denklemler orijinal gradyan hesabındaki değişkenlerin kendi sağ tarafında kullanılmasına olanak sağlar. Bu gradyan hesabına eklenmiş bir sistem olarak düşünülebilir, geriye yayılımdaki gradyan hesabının, hata ölçümünün ileri beslemeli hesaplanmasına eklenmiş olması gibi. Bu eklenmiş sistem, istenilen Hv vektörü ile kesin olarak aynı olan $R\{\nabla_x\}$ vektörünü hesaplar.

$\mathrm{R}\{\cdot\}$ Tekniğinin Geri-Yayılım Ağına Uygulanması

Geriye yayılım ağında $R\{\cdot\}$ tekniği, gradyanı hesaplayan denklemleri Hv'yi, Hessian H'ın bir v vektörüyle çarpımını, hesaplayan denklemler şekline dönüştürmek için kullanılabilir. Genellikle E birçok örnek için hataların toplamıdır, $E = \sum_p E_p$. Bu nedenle $\nabla_x E$ ve H tüm örnekler üzerinden alınan toplamlardır, $H = \sum_p H_p$, ve $Hv = \sum_p H_p v$. Açık olması için, örnekler üzerindeki bu dış toplam gerekli olmadıkça burada gösterilmeyecektir, gradyan ve Hv yordamları sadece tek bir örnek için gösterilmiştir.

Bir geriye yayılım ağında Hv çarpımını etkin bir şekilde hesaplamak için yukarıdaki yordam uygulanarak bir denklemler kümesinden oluşan $R\{backprop\}$

algoritması türetilebilir. Aslında $R\{backprop\}$ algoritması birbirinden bağımsız olarak Werbos (1988) tarafından bulunmuştur. Werbos (1988) $Hv = \nabla_x (v \cdot \nabla_x E)$ hesaplamak için bir geriye yayılım süreci olarak türetmiştir. İlginç olarak bu türetim burada verilenin dualidir çünkü denklemleri yönü ters çevrilmiştir, $\nabla_x E$ algoritmasının geri geçişi Hv algoritmasında ileri geçiş haline gelir, burada ise denklemlerin yönü değiştirilmez. Aynı algoritma aynı zamanda Møller (1993) tarafından da başka bir türetim ile bulunmuştur.

Uyumluluk için x ağırlıkları kaynak ve hedef birimlerini gösterecek şekilde indekslenecektir, i. birimden j. birime olan ağırlık x_{ij} ile gösterilir. v, x ile aynı boyuta sahip olduğu için v'nin elemanları da aynı şekilde indekslenir.

Ağın ileri geçişinin hesaplaması

$$\begin{aligned} z_i &= \sum_j x_{ji} y_j \,, \\ y_i &= \sigma_i(z_i) + I_i \end{aligned} \tag{4.31}$$

biçiminde verilebilir, burada $\sigma_i(\cdot)$ i. birimin doğrusal olmayan transfer fonksiyonu, z_i i. birime gelen toplam girdi, y_i i. birimin çıktısı ve I_i i. birime dışarıdan (ağın dışından) gelen girdidir.

Hata ölçümü $E = E(y)$ olsun ve y_i'ye göre basit doğrudan türevi $e_i = dE / dy_i$ olsun. e_i'nin sadece y_i'ye bağlı olduğunu ve herhangi bir $j \neq i$ için y_j'ye bağlı olmadığı varsayılmıştır. Bu zaten hata kareler veya çapraz entropi gibi çoğu genel hata ölçümünde geçerlidir (Hinton 1989). Böylece $e_i(y_i)$ basit bir fonksiyon olarak yazılabilir. O zaman geri geçiş

$$\begin{aligned} \frac{\partial E}{\partial y_i} &= e_i(y_i) + \sum_j x_{ij} \frac{\partial E}{\partial z_j}, \\ \frac{\partial E}{\partial z_i} &= \sigma_i'(z_i) \frac{\partial E}{\partial y_i}, \end{aligned} \tag{4.32}$$

$$\frac{\partial E}{\partial x_{ij}} = y_i \frac{\partial E}{\partial z_j}$$

biçiminde verilebilir. $R\{\cdot\}$ tekniği (4.31) ve (4.32) denklemlerine uygulanırsa, ileri geçiş için

$$\begin{aligned} R\{z_i\} &= \sum_j (x_{ji} R\{y_j\} + v_{ji} y_j), \\ R\{y_i\} &= R\{z_i\} \sigma_i'(z_i) \end{aligned} \tag{4.33}$$

ve geri geçiş için

$$\begin{aligned} R\left\{\frac{\partial E}{\partial y_i}\right\} &= e_i'(y_i) R\{y_i\} + \sum_j \left(x_{ij} R\left\{\frac{\partial E}{\partial z_j}\right\} + v_{ij} \frac{\partial E}{\partial z_j} \right), \\ R\left\{\frac{\partial E}{\partial z_i}\right\} &= \sigma_i'(z_i) R\left\{\frac{\partial E}{\partial y_i}\right\} + R\{z_i\} \sigma_i''(z_i) \frac{\partial E}{\partial y_i}, \\ R\left\{\frac{\partial E}{\partial x_{ij}}\right\} &= y_i R\left\{\frac{\partial E}{\partial z_j}\right\} + R\{y_i\} \frac{\partial E}{\partial z_j} \end{aligned} \tag{4.34}$$

eşitlikleri elde edilir. Elemanları $R\left\{\frac{\partial E}{\partial x_{ij}}\right\}$ olan vektör aslında $R\{\nabla_x\} = Hv$'dir ve asıl hesaplanmak istenilen niceliktir.

Hata kareler toplamı için $e_i'(y_i) = y_i - d_i$ olur burada d_i i. çıktı birimi için istenilen hedef değeridir, bu nedenle $e_i'(y_i) = 1$ olur. Bu (4.34)'ü basit çıktı birimleri için sadeleştirir ve $R\left\{\frac{\partial E}{\partial y_i}\right\} = R\{y_i\}$ olur.

4.5.2 Güç iterasyonu

$H \in \mathbb{R}^{n \times n}$ simetrik matrisi ve $\|v_0\| = 1$ olan rassal bir v_0 vektörü için Güç iterasyonu

$k = 1, 2, \ldots$ için

- $w_k = Hv_{k-1}$
- $v_k = w_k / \|w_k\|$
- $\lambda_k = v_k^T H v_k$

biçiminde verilebilir. Güç iterasyonu sonucunda λ, H'nın en büyük özdeğeri λ_1'e, v ise bu en büyük özdeğere karşılık gelen özvektör v_1'e yakınsar. Bu iterasyonda durdurma koşullarına dikkat edilmemiştir, sadece döngü "$k = 1, 2, \ldots$ için" ifadesi ile belirtilmiştir. Tabii ki, pratikte durdurma koşulları çok önemlidir ve probleme göre karar verilmesi gerekir. Sonucun oldukça kesin olması istendiğinde iterasyon sayısı büyür ve zaman olarak maliyetlidir. Buna rağmen, en büyük özdeğerin iyi bir tahmini az sayıda iterasyonla elde edilebilir (tipik olarak 10 iterasyonda).

Bu yöntem gradyan düşümü algoritmasında öğrenme oranı parametresini çevrimiçi (online) tahmin etmek için geliştirilen tekniğin bir bölümü olarak Le Cun ve ark. (1993) tarafından kullanılmıştır.

İlginç olarak, bu metot ufak bir değişiklikle en küçük özdeğer ve karşılık gelen özvektörü de bulacak şekilde düzenlenebilir (Le Cun ve ark. 1993). Öncelikle, en büyük özdeğer λ_1 hesaplanmalıdır (veya üstten sınırlandırılmalı). Daha sonra, $\|z\| = 1$ olan rassal bir z vektörü için

$$z \leftarrow \frac{1}{r}\left(\nabla_x E\left(x + rz\right) - \nabla_x E\left(x\right)\right) - \lambda_1 z \tag{4.28}$$

iterasyonunu gerçekleştirerek $(H-\lambda_1 I)$'nın en küçük özdeğeri ve karşılık gelen özvektör hesaplanabilir, bu da H matrisinin en küçük özdeğeri ile aynıdır.

Sonuç olarak, Güç iterasyonu Hessian matrisinin kendisini hesaplamadan onun en büyük ve en küçük özdeğerlerinin yaklaşık olarak hesaplanmasında kullanılabilir. Güç iterasyonunun 2 (a) adımındaki Hv_{k-1} çarpımlarını yaklaşık olarak hesaplamak için Taylor açılımını kullanan (4.26), kesin olarak hesaplamak içinse $R\{\cdot\}$ tekniği verimli bir şekilde kullanılabilir.

Hessian matrisinin kendisini hesaplamadan onun en büyük ve en küçük özdeğerini yaklaşık olarak hesaplamak için (4.26)'yı kullanan algoritma 4.1 oluşturulmuştur.

Algoritma 4.1

1. n boyutlu v ve z vektörlerini rassal olarak seç. Küçük bir sabit sayı r'yi seç, örneğin r=0.01.

2. x ağırlık noktasındaki gradyanı tüm veri seti için hesapla, $G=\nabla E(x)$.

3. v ve z vektörlerini normalleştir. $N(v)=v/\|v\|$, $N(z)=z/\|z\|$. (ana döngü)

4. x ağırlık vektörüne $rN(v)$'yi ekle ve x_1 olarak kaydet.

5. x ağırlık vektörüne $rN(z)$'yi ekle ve x_2 olarak kaydet.

6. x_1 ağırlık noktasındaki gradyanı hesapla, $G_1=\nabla E\left(x+rN\left(v\right)\right)$.

7. x_2 ağırlık noktasındaki gradyanı hesapla, $G_2=\nabla E\left(x+rN\left(z\right)\right)$.

8. v vektörünü güncelle, $v\leftarrow\frac{1}{r}\left(G_1-G\right)$.

9. En büyük özdeğer tahminini hesapla, $\lambda_1=\frac{v^T Hv}{v^T v}$.

10. z vektörünü güncelle, $z\leftarrow\frac{1}{r}\left(G_2-G\right)-\lambda_1 N(z)$.

11. 3-10 adımlarını k defa tekrar et. (örneğin $k=10$) (ana döngü sonu).

12. En küçük özdeğer tahminini hesapla, $\lambda_n=\frac{z^T Hz}{z^T z}$.

Le Cun ve ark. (1993), gradyan düşümü algoritmasının tam çevrimiçi eğitiminde kullanılacak ve öğrenme süresince sabit olan öğrenme oranını belirlemek için Hessian'ın sadece en büyük özdeğerini Taylor açılımı ve hareketli ortalama kullanarak çevrimiçi (online) tahmin etmiştir. Burada kullanılan veri seti büyük ve tekrarlı veriden oluştuğu için en büyük özdeğer tahmin edilirken veri setinin tümü kullanılmamıştır. Bunun yerine veri setinin küçük bir bölümü kullanılarak Hessian'ın en büyük özdeğeri tahmin edilmiş ve eğitim süreci boyunca sabit olan öğrenme oranı bu özdeğerden yararlanılarak hesaplanmıştır. Bu çalışmada ise etkin parametrelerle uyumlu çalışacak algoritmalarda ağırlık güncellemesi toplu (batch) olarak yapılır ve tüm veri seti kullanılır.

Algoritma 4.1 Hv_{k-1} çarpımlarını Taylor açılımını kullanarak yaklaşık olarak hesaplar. Hv_{k-1} çarpımlarını kesin olarak hesaplayan $\mathrm{R}\{\cdot\}$ tekniğini kullanan algoritma ise aşağıdaki gibi verilebilir.

Algoritma 4.2

1. n boyutlu v ve z vektörlerini rassal olarak seç.

2. v ve z vektörlerini normalleştir. $N(v) = v/\|v\|$, $N(z) = z/\|z\|$ (ana döngü).

3. $HN(v)$ çarpımını $\mathrm{R}\{\cdot\}$ tekniğini kullanarak kesin olarak hesapla.

4. v vektörünü güncelle, $v \leftarrow HN(v)$.

5. En büyük özdeğer tahminini hesapla, $\lambda_1 = \dfrac{v^T Hv}{v^T v}$.

6. $HN(z)$ çarpımını $\mathrm{R}\{\cdot\}$ tekniğini kullanarak kesin olarak hesapla.

7. z vektörünü güncelle, $z \leftarrow HN(z) - \lambda_1 N(z)$.

8. 2-7 adımlarını k defa tekrar et. (örneğin $k = 10$) (ana döngü sonu).

9. En küçük özdeğer tahminini hesapla, $\lambda_n = \dfrac{z^T Hz}{z^T z}$.

4.6. Hessian-Vektör Çarpımı ve Güç İterasyonu ile Geliştirilen Algoritmalar

Hessian'ın en büyük ve en küçük özdeğerini yaklaşık olarak hesaplamak için Hessian-vektör çarpımı ve Güç iterasyonunu kullanan etkin öğrenme parametreli momentumlu gradyan düşümü algoritması A3 adı altında verilebilir (Memmedli ve Taş 2007b).

Algoritma A3

- x_0 başlangıç noktası ve yakınsama toleransı $\varepsilon > 0$ verilsin,

1. $k \leftarrow 0$;

2. Eğer ilk iterasyon ise;

Başlangıç Hessian'ını birim matris olarak tanımla, $H_0 = I$,

Başlangıç yönünü gradyanın ters yönü olarak seç, $p_0 = -\nabla E(x_0)$

3. Eğer ilk iterasyon değilse

Yakınsama koşulu **sağlanana** kadar devam et: (ana döngü)

(a) Hessian'ın en büyük κ_1 ve en küçük κ_n özdeğerini Algoritma 4.1 ile yaklaşık olarak hesapla;

(b) Eğer en küçük özdeğer negatifse, özdeğerlere yapay bir düzeltme uygula.

(c) η_k ve μ_k'yı (3.39) formülü ile hesapla;

(d) Arama yönünü hesapla, $p_k = -(1-\mu_k)\eta_k \nabla E(x_k) + \mu_k p_{k-1}$;

(e) Ağırlığı güncelle, $x_{k+1} = x_k + \alpha_k p_k$, α_k'yı doğru arama ile belirle;

(f) $k \leftarrow k+1$;

4. Dur (ana döngü)**.**

Algoritma A3'ün çalıştırılmasında karşılaşılan temel problem, belirli bir ağırlık noktasında hesaplanan özdeğerlerin negatif olmasıdır (özellikle en küçük özdeğerin). Çünkü Hessian matrisi her ağırlık noktasında pozitif tanımlı olmayabilir. Bu probleme üçüncü bölümde de değinilmişti ve orada sorun kendiliğinde çözülmüştü çünkü gerçek Hessian'ın yerine kullanılan BFGS ve DFP metotlarıyla elde edilen Hessian yaklaşıkları her zaman pozitif tanımlılığını

koruyordu. Burada ise elde edilmeye çalışılan Hessian matrisini gerçek özdeğerleridir ve negatif değerler alabilir. Bu problemin basit bir çözümü özdeğerlere pozitif bir sayı ekleyerek özdeğerleri pozitif hale getirmektir ve dolayısıyla Hessian matrisini pozitif tanımlı hale getirmektir. Bu yaklaşım ikinci düzey metotlardan olan Levenberg-Marquardt da kullanılır. Doğru arama, arama yönünün azalan bir yön olmasını sağlar ve bu şekilde uygun Hessian matrisinin pozitif tanımlılığını mümkün olduğunca korumaya çalışır. Doğru arama ile bulunan nokta eğer yerel minimum civarında bir nokta olursa, Hessian bu bölgede pozitif tanımlı olacak ve etkin öğrenme parametreleriyle uyumlu olarak çalışan algoritmalar daha iyi yakınsama performansı gösterecektir.

Sonuç olarak kesim 4.2'de verilmiş olan ana algoritma yapısı altında temelde iki farklı yaklaşım kullanan pratikte dört farklı algoritma düzenlenmiştir. Hessian yaklaşık hesabını temel alan A1 ve A2 algoritmaları sırasıyla, Broyden Ailesinin üyesi olan DFP ve BFGS Hessian güncellemelerini kullanır. BFGS ve DFP Hessian güncellemelerinin önemli bir avantajı Hessian matrisinin pozitif tanımlılığını korumalarıdır. Bu da A1 ve A2 algoritmalarının sahip olduğu önemli bir özelliktir. A1 ve A2 algoritmalarında bu güncelleme yöntemleriyle hesaplanan özdeğerler yaklaşıktır çünkü hesaplanan Hessian matrisleri yaklaşıktır. Diğer taraftan A3 algoritması Hessian matrisini hesaplamadan Hessian'ın bir vektörle çarpımını Güç iterasyonuyla beraber kullanarak sadece en büyük ve en küçük özdeğeri yaklaşık olarak bulmaya çalışır. Bu yaklaşımda özellikle büyük boyutlu problemlerde hem saklama maliyeti hem de hesaplama maliyeti açısından önemli bir avantaj sağlar. Bu yaklaşımda da hesaplanan özdeğerler yaklaşıktır.

Etkin öğrenme parametrelerinin iterasyon süreci boyunca sağlıklı bir şekilde hesaplanabilmesi için her adımda amaç fonksiyonunun değerinin ve gradyanının hesaplanması gerekir. Amaç fonksiyonunun ve gradyanının her defasında hesaplanması ağın hata toplamının tüm veri seti için hesaplanmasını gerektirir. Özellikle A3 algoritmasında, en büyük ve en küçük özdeğerlerin hesaplanmasında kullanılan Güç iterasyonu çok sayıda Hessian-vektör çarpımını gerektirir (her adım için en az 20 kez). Veri kümeleri ise günden güne daha büyük boyutlu hale gelmektedir. Bu nedenle örnek sayısının çok olduğu büyük çaplı veri kümesine sahip problemlerde etkin öğrenme parametrelerinin hesaplanması ciddi bir

hesaplama ve zaman maliyeti oluşturur. Bu sorun geliştirilen algoritmalar için gerçekleştirilen karşılaştırmalarda açıkça ortaya çıkmıştır. Bu nedenle, etkin öğrenme parametre güncellemelerini iterasyon sürecinin her adımında yapmamak, belirli adımlarda yapmak ilk bakışta daha mantıklı görünür. Ne yazık ki, A1 ve A2 algoritmalarında bu yaklaşım gerçekleştirilemez çünkü bu algoritmalar her adımda Hessian matrisini iteratif olarak hesaplar ve sürecin herhangi bir adımındaki Hessian güncellemesi bir öncekine dayanır. Ama A3 algoritmasında bu yaklaşım uygulanabilir.

İlk olarak, etkin öğrenme parametrelerini sadece belirli adımlarda hesaplayacak şekilde düzenlenen A4 algoritmasında özdeğerleri yaklaşık olarak hesaplamak yerine kesin Hessian-vektör çarpımlarını (Pearlmutter 1994) kullanarak kesin olarak hesaplamak düşünülmüştür. Ama bu durumda öğrenme parametrelerini belirli adımlarda hesaplayarak kazanılan zaman, özdeğerlerin kesin hesabında harcanmış olur. Diğer bir deyişle, özdeğerleri kesin olarak hesaplamak ciddi bir avantaj sağlamamış ve ekstra bir zaman maliyeti oluşturmuştur. Bu durum gerçekleştirilen deneylerde gözlemlenmiştir. Bu nedenle A4 algoritmasında da algoritma 4.2 kullanılarak özdeğerler yaklaşık olarak hesaplanmıştır ve bu sayede etkin öğrenme parametrelerini her adım yerine belirli adımlarda yaklaşık olarak hesaplayarak, büyük çaplı ve tekrarlı veriye sahip problemlerde bu yaklaşımla ciddi bir zaman kazancı sağlanmıştır.

A4 algoritmasında hata fonksiyonunun değişimi n (problem boyutu) iterasyon boyunca belirli bir eşik değerden (0.005) daha az oranda azalmışsa öğrenme parametreleri güncellenir (A4 algoritması -3 (b)). Eğer hata fonksiyonu azalmaya devam ediyorsa herhangi bir parametre güncellemesi gerçekleştirilmez. Çünkü hata fonksiyonundaki azalma az olmasına rağmen iterasyonlarda öğrenme parametreleri her defasında yeniden hesaplanmadığı için ciddi bir zaman tasarrufu sağlanmış olur. A4'ün pratikte uygulanmasında temel sorun A3'de olduğu gibi oluşan herhangi bir ağırlık noktasında kesin olarak hesaplanan özdeğerlerin negatif olması olasılığıdır (özellikle en küçük özdeğer). Bu durumda A4 algoritmasında da A3 algoritmasındaki aynı yapay düzeltme uygulanarak özdeğerler pozitif hale getirilir.

Algoritma A4

- x_0 başlangıç noktası ve yakınsama toleransı $\varepsilon > 0$ verilsin,

1. $k \leftarrow 0$;

2. Eğer ilk iterasyon ise;

Başlangıç Hessian'ını birim matris olarak tanımla, $H_0 = I$;

Başlangıç yönünü gradyanın ters yönü olarak seç, $p_0 = -\nabla E(x_0)$.

3. Eğer ilk iterasyon değilse

Yakınsama koşulu **sağlanana** kadar devam et: (ana döngü)

(a) Oran = $E(x_k) / E(x_{k-1})$;

(b) Eğer (mod(iterasyon sayısı, n)=0 ve Oran > 0.995) ise

- Hessian'ın en büyük κ_1 ve en küçük κ_n özdeğerini Algoritma 4.2 ile yaklaşık olarak hesapla;
- Eğer en küçük özdeğer negatifse, özdeğerlere yapay bir düzeltme uygula;
- η_k ve μ_k'yı (3.39) formülü ile hesapla.

(c) Arama yönünü hesapla, $p_k = -(1-\mu_k)\eta_k \nabla E(x_k) + \mu_k p_{k-1}$;

(d) Ağırlığı güncelle, $x_{k+1} = x_k + \alpha_k p_k$, α_k'yı doğru arama ile belirle;

(e) $k \leftarrow k+1$;

4. Dur (ana döngü)**.**

5. ETKİN PARAMETRELİ VE GELENEKSEL MOMENTUMLU GERİ-YAYILIM ALGORİTMALARININ KARŞILAŞTIRILMASI

Bu bölümde, etkin öğrenme parametreleriyle uyumlu olarak çalışacak algoritmalar, önceki bölümde ifade edilen yaklaşımları kullanacak şekilde düzenlenmiştir. Düzenlenen algoritmalar, doğru aramanın kullanıldığı klasik momentumlu gradyan düşümü (GDM) ve adaptif öğrenme oranlı ve momentumlu gradyan düşümü (GDX) ile karşılaştırılmıştır. Etkin parametrelerle çalışacak olan algoritmalarda (A1, A2, A3 ve A4) adım büyüklüğünün belirlenmesi için doğru arama kullanır. Bu nedenle, karşılaştırmaların sağlıklı olması açısından GDX ve GDM algoritmaları da doğru arama ile çalışacak şekilde düzenlenmiştir. Doğru aramalı GDX ve GDM sırasıyla OGDX ve OGDM olarak adlandırılmıştır. Doğru arama ile çalışan gradyan düşümü algoritmaları literatürde optimum gradyan düşümü olarak da bilinir. Bu sayede, GDM ve GDX algoritmaları en iyi performansı verecek şekilde düzenlenmiş olur. Karşılaştırmalar iki grup veri seti üzerinde gerçekleştirilmiştir. Birinci grupta, ufak boyutlu problemler ele alınmıştır (XOR, Parite3, Parite4, Parite5 problemleri). İkinci grupta ise UCI veri ambarından daha büyük boyutlu ve daha büyük veri kümesine sahip olan gerçek problemler ele alınmıştır (Güneş patlaması, Çiçek, Cam, İyonosfer verileri) (Asuncion ve Newman 2007). Parite7 problemi de bu problem grubuna dahil edilmiştir.

XOR problemi küçük boyutlu sınıflandırma problemleri içinde belki de en popüler olanıdır. Bu problem ikilik (binary) sistemde iki girdiden oluşur. Yapay sinir ağı, dört örnek çiftini karşılık gelen çıktıya eşlemeye çalışır.

Parite problemleri ise XOR probleminin daha büyük boyutlu şekli olarak düşünülebilir; örneğin Parite3 ikilik sistemde üç adet ikili girdiden oluşur ve çıktı yine tek boyutludur.

İyonosfer verisi Labrador Goose koyunda bulunan bir radar sistemi tarafından toplanmıştır. Bu sistem 16 yüksek-frekanslı aşamalı bir antenler topluluğundan oluşur. “İyi” radar dönütleri, iyonosferde belirli bir yapının kanıtını gösterir. “Kötü” dönütler ise böyle bir yapının olmadığını gösterir; bu sinyaller iyonosfer tabakasından geçer. Bu veri setinde 34 tane sürekli girdi değişkeni ve

“iyi” ve “kötü” olarak 1 tane ikili çıktı değişkeni bulunmaktadır. Toplam 351 tane gözlem içermektedir ve bu problem bir sınıflandırma problemidir.

Iris (çiçek) veritabanı belki de örnek tanıma literatüründe bulunan en iyi bilinen veritabanıdır. Fisher’in makalesi (1936) bu alanda yayınlanmış klasik bir makaledir ve bugüne kadar sıkça referans edilmiştir (Duda ve Hart 1973). Veri seti 3 sınıftan oluşur ve her bir sınıf belirli bir iris tabakasını simgeler ve her sınıftan 50 tane örnek bulunur. Bu sınıflardan bir tanesi diğer ikisinden doğrusal olarak ayrılabilir; diğerleri doğrusal olarak ayrılabilen sınıflar değildir. Bu problemde 4 tane girdi ve bir çıktı değişkeni bulunur. Veri seti 150 örnekten oluşmaktadır. Amaç ise örneğin iris tabakasının hangi sınıfa ait olduğunu tahmin etmektir.

Cam veritabanı 214 veriden oluşmaktadır. 10 tane gerçel girdi değişkeninden oluşur. Problem camın hangi sınıf cam olduğuna karar vermektir. Kriminolojik araştırmalarda suç mahalinde bulunan camları sınıflandırmak için bu problem ortaya konmuştur.

Güneş patlaması veritabanı 24 saatlik bir periyotta oluşmuş her biri belirli bir güneş patlaması çeşidine karşı gelecek şekilde 3 potansiyel sınıf içermektedir. Her örnek güneş üzerinde 1 aktif bölge için yakalanmış karakteristikleri temsil eder. Kısaca her sınıf 24 saatlik bir periyotta oluşan kendi sınıfına ait güneş patlamalarının sayar. Toplam 1389 örnek ve 10 girdi değişkeninden oluşmaktadır. Problemler küçük ve orta ölçekli olmak üzere iki gruba ayrılmıştır ve problemlerde kullanılan ağ mimarileri aşağıdaki gibidir:

Problem Kümesi I	XOR	Parite3	Parite4	Parite5
Girdi	2	3	4	5
Gizli Nöron Sayısı	3	4	7	8
Çıktı	1	1	1	1

Problem Kümesi II	Güneş (Sun Flare)	Çiçek (Iris)	Cam (Glass)	Parite7 (Parity7)	İyonosfer (Ionosphere)
Girdi	24	4	9	7	34
Gizli Nöron Sayısı	4	8	6	14	8
Çıktı	3	1	1	1	1

Girdiler ortalaması sıfır ve standart sapması 1 olacak şekilde, hedefler ise [-1,1] aralığına yerleşecek şekilde dönüştürülmüştür. Gizli katmanlar için tanjant sigmoid, çıktı katmanı için basit doğrusal transfer fonksiyonu kullanılmıştır. Eğitim süreci hata fonksiyonunun değeri küçük bir eşik değerinden (0.02) küçük oluncaya kadar devam ettirilmiştir. Karşılaştırma gerçekleştirilirken aşağıdaki adımlar izlenmiştir:

Karşılaştırma Yordamı

1. Girdileri ortalaması sıfır ve standart sapması 1 olacak şekilde dönüştür.
2. Hedefleri [-1,1] aralığına yerleşecek şekilde dönüştür.
3. Girdi ve çıktı kümesini rassal olarak yeniden oluştur.
4. Karşılaştırılacak eğitim algoritmalarını belirle, hata eşik değerini belirle
5. Ağı daha önce belirlenen mimaride oluştur.
6. $k = 1$ den 10 a kadar (ana döngü)
a) $k.$ deneyin başlangıç ağırlıklarını belirle
7. $i = 1$ den karşılaştırılacak algoritma sayısına kadar
a) Ağın ağrılıklarını $k.$ deneyin başlangıç ağırlıkları olarak ata.
b) Maksimum iterasyon sayısını ve hedef hata değerini belirle
c) $i.$ eğitim algoritması ile ağı hedef hata değerine ulaşılana kadar eğit.
d) Eğitimin sonucunda gerçekleşen devir sayısını ve geçen zamanı sakla.
8. Dön adım 7. (yeni i)
9. Dön adım 6. (ana döngü)

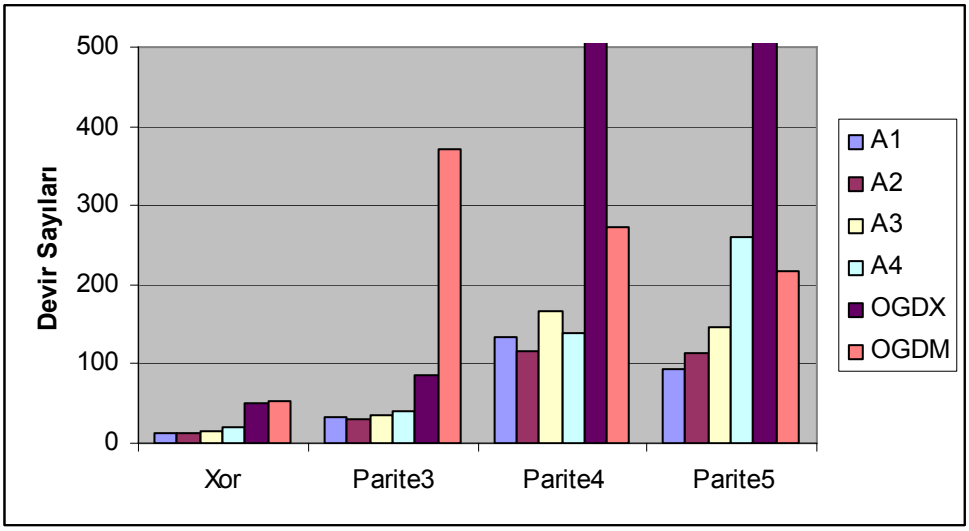
Eğitim süreci 10 farklı başlangıç ağırlığında başlatılmış ve her eğitim algoritması için elde edilen devir sayıları ve çalışma zamanlarının ortalaması çizelge 5.1 de verilmiştir.

Çizelge 5.1 I. Problem kümesi için karşılaştırılan algoritmaların devir sayıları ve çalışma zamanları (sn).

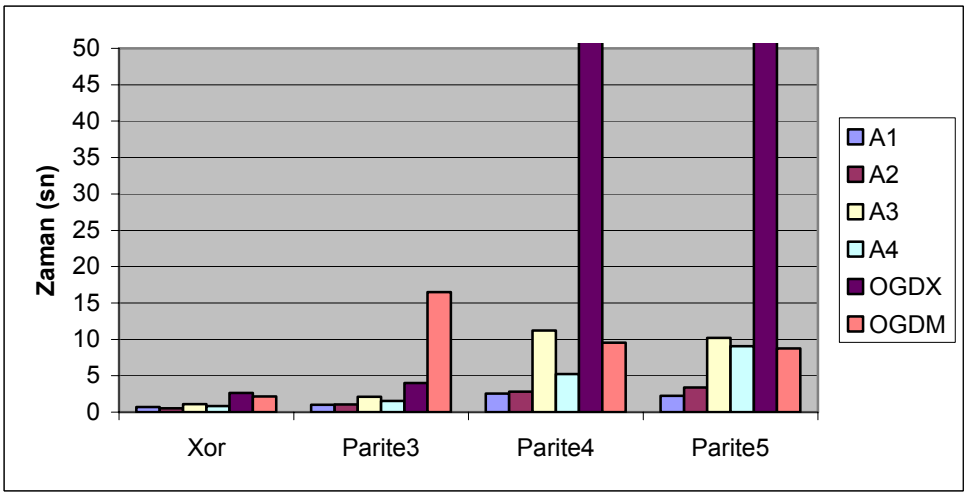
Algoritma	**Kriter**	**XOR**	**Parite 3**	**Parite 4**	**Parite 5**
A1	*Devir*	12.7	32	134.6	93.50
	Zaman(sn)	0.69	1.00	2.56	2.23
A2	*Devir*	11.8	30.4	115.60	113.40
	Zaman(sn)	0.53	1.04	2.80	3.38
A3	*Devir*	14.6	34.1	165.70	147.30
	Zaman(sn)	1.09	2.10	11.21	10.21
A4	*Devir*	19	40.60	139.40	261.30
	Zaman(sn)	0.82	1.53	5.24	9.04
OGDX	*Devir*	50.6	85.90	1176.5	1017.7
	Zaman(sn)	2.62	4.01	58.53	70.58
OGDM	*Devir*	52.6	370.4	271.80	217.4
	Zaman(sn)	2.17	16.48	9.53	8.74

Birinci problem grubunda etkin parametrelerle çalışan A1, A2, A3, A4 algoritmalarının tümü OGDM ve OGDX algoritmalarından daha az adımda yakınsamıştır. Ayrıca yakınsama zamanları da OGDX ve OGDM algoritmalarından daha azdır (Şekil 5.1 ve Şekil 5.2). Hessian veya ters Hessian yaklaşımlarını kullanan A1 ve A2 algoritmaları, Hessian-vektör çarpımını ve Güç iterasyonunu beraber kullanan A3 ve A4 algoritmalarına göre daha az zamanda yakınsamıştır. Bu grupta en iyi performansı Hessian'ın tersini DFP güncellemesi ile yaklaşık olarak hesaplayan A1 algoritması göstermiştir.

Şekil 5.1 I. Problem kümesi için karşılaştırılan algoritmaların devir sayıları grafiği.



Şekil 5.2 I. Problem kümesi için karşılaştırılan algoritmaların çalışma zamanı (sn) grafiği.



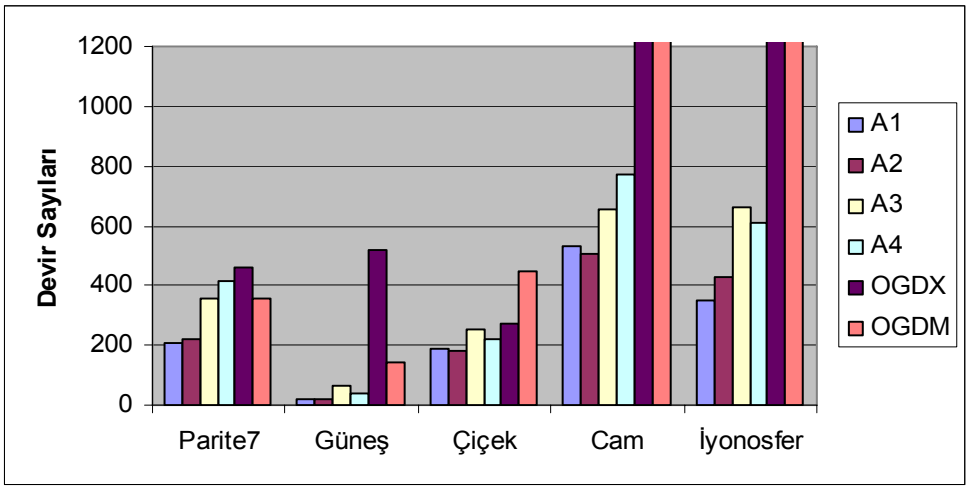
Çizelge 5.2 II. Problem kümesi için karşılaştırılan algoritmaların devir sayıları ve çalışma zamanları (sn).

Algoritma	Kriter	Parite7	Güneş	Çiçek	Cam	İyonosfer
A1	*Devir*	206.6	20.1	187.9	532.2	348.4
	Zaman(sn)	8.52	1.23	4.66	36.16	33.99
A2	*Devir*	222.5	17	179.1	507.4	426.3
	Zaman(sn)	14.94	0.96	4.79	85.79	141.48
A3	*Devir*	355.2	63.3	251.1	656.9	663.1
	Zaman(sn)	43.97	13.81	23.82	108.91	123.17
A4	*Devir*	414.7	40.9	221.1	773.1	612.6
	Zaman(sn)	22.93	4.08	10.12	52.82	70.52
OGDX	*Devir*	460.9	521.9	269.6	5807.9	3717.1
	Zaman(sn)	32.27	434.95	15.03	529.42	326.61
OGDM	*Devir*	359.9	140.5	448.5	7314.5	5508.1
	Zaman(sn)	18.39	13.56	19.46	570.34	372.93

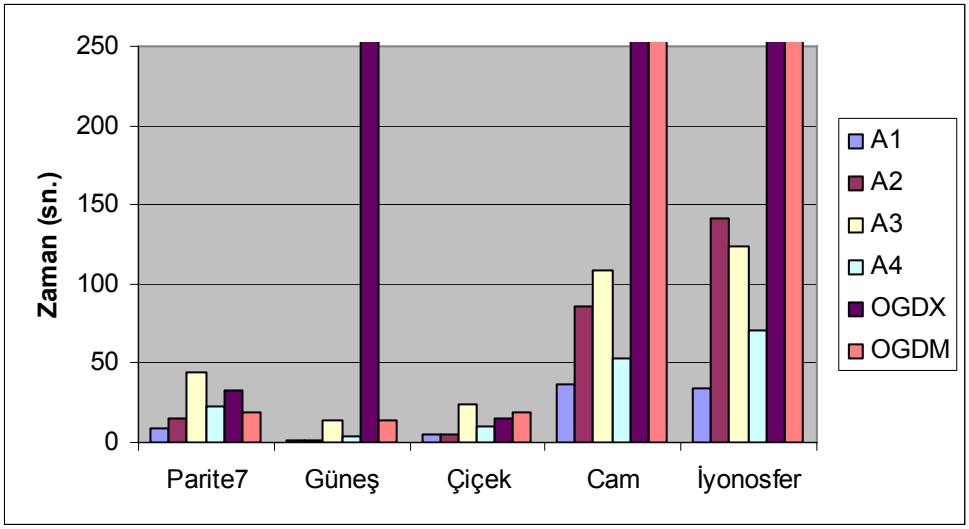
İkinci problem grubunda, iterasyon (devir) sayılarına bakıldığında etkin öğrenme parametreleriyle çalışan tüm algoritmalar doğru aramalı momentumlu gradyan düşümü algoritmaları OGDX ve OGDM'den daha az iterasyonda çözüme yakınsamıştır. İterasyon sayılarına göre en iyi performansı Broyden ailesine üye BFGS ve DFP güncellemelerinin kullanıldığı A1 ve A2 algoritmaları göstermiştir. Bu algoritmaları Hessian-vektör çarpımını ve Güç iterasyonunu beraber kullanan A3 ve A4 takip eder. Yakınsama zamanları göz önüne alındığında ise en iyi performansı A1 göstermiştir. A2 ikinci en iyi performansa sahiptir (Şekil 5.3 ve Şekil 5.4). A1 ve A2 algoritmalarının yakınsama zamanı iterasyon sayısına göre karşılaştırıldığında, A2 iterasyon başına A1'e oranla daha fazla zaman alır. Bunun nedeni A2'nin kullandığı DFP güncellemesinin A1'in kullandığı BFGS güncellemesine göre daha fazla hesaplama maliyetine sahip olmasıdır (DFP güncellemesi (4.25)'de $\phi_k = 1$'dir). A3 ve A4 algoritmalarında iterasyon sayısı-zaman karşılaştırması yapıldığında ise iterasyon sayılarının birbirine yakın olduğu

halde A3'ün yakınsaması A4'e göre oldukça fazla zaman alır, çünkü A3 algoritması etkin öğrenme parametrelerini her adımda hesaplar ve bu hesaplama yoğun Hessian-vektör çarpımlarını içerdiği için zaman olarak oldukça maliyetlidir. A4 ise etkin öğrenme parametrelerini problem boyutuna ve hata fonksiyonundaki değişime göre belirli adımlarda gerçekleştirir. Bunun sonucunda A4'ün yakınsaması A3'e göre daha fazla iterasyonda olmasına rağmen A4 daha az zaman alır. A4'de kullanılan bu yaklaşım özellikle büyük veri kümesine sahip büyük boyutlu problemlerde algoritmanın zaman maliyetini oldukça azaltır.

Şekil 5.3 II. Problem kümesi için karşılaştırılan algoritmaların devir sayıları grafiği.



Şekil 5.4 II. Problem kümesi için karşılaştırılan algoritmaların çalışma zamanı (sn) grafiği.



Bu kesimde, iterasyon sayısına ve zamana göre verilen karşılaştırma sonuçları her problem için 10 farklı ağırlık noktasından başlatılan deneylerin ortalamasıdır. Elde edilen bu değerlerin standart sapmalarına bakıldığında, OGDX ve OGDM algoritmaları gerçekleştirilen tüm deneylerde yüksek standart sapma değerlerine sahipken etkin öğrenme parametreli A1, A2, A3 ve A4 algoritmaları bu algoritmalara kıyasla çok küçük standart sapma değerlerine sahiptir. Gerçekleştirilen deneyler her problem için en az 3 olmak üzere farklı ağ mimarilerini kapsayacak biçimde gerçekleştirilmiştir. Bunun sonucunda, etkin öğrenme parametreleri ile çalışan A1, A2, A3 ve A4 algoritmalarının değişen ağ yapısına OGDX ve OGDM den daha az duyarlı olduğu ve her zaman aynı performansı kararlı bir şekilde gerçekleştirdiği gözlemlenmiştir. Diğer yandan, OGDX ve OGDM için bunu söylemek oldukça zordur, bu algoritmalar değişen ağ yapısına oldukça duyarlıdır, örneğin; A1, A2, A3 ve A4 algoritmalarının önemli bir avantajı büyük boyutlu ağ mimarilerine gereksinim duymamasıdır, küçük boyutlu ağ mimarilerinde de aynı performansı OGDX ve OGDM'den çok daha iyi bir şekilde gösterir. OGDX ve OGDM'nin ise özellikle küçük ağ mimarilerinde

istenilen eşik hata değerine ulaşamadığı, yerel minimum civarında çok fazla oyalandığı ve yakınsama sorunu yaşadığı gözlenmiştir.

OGDX ve OGDM den gözlemlenen bu sorunların temelinde öğrenme parametrelerinin seçimi yatmaktadır. OGDM algoritmasında öğrenme oranı ve momentum katsayısı iterasyon süreci boyunca sabit götürülür, bu ciddi bir dezavantaj oluşturur çünkü bu sabit seçimin iterasyon süresince oluşan farklı ağırlık noktalarında iyi performans göstermez ve algoritmanın yakınsaması sağlansa da bu çok verimli ve sağlıklı olmaz. OGDX algoritması, süreç boyunca adaptif seçimli öğrenme oranını kullanarak bu sorunun üstesinden gelmeye çalışır ama bu yaklaşımda eğitilecek verinin önemli unsurlarını dikkate almaz ve gösterdiği performans da OGDM den farklı değildir hatta bazı problemlerde OGDM daha iyi performans göstermiştir. Bu da problemin veri yapısının dikkate alınmamasının bir sonucudur.

6. SONUÇ VE ÖNERİLER

Bu doktora tezi çalışmasında, öncelikle hata fonksiyonunun kuadratik olduğu durumda momentumlu gradyan düşümü algoritması için etkin öğrenme oranı ve momentum faktörünün eş-zamanlı olarak Hessian'ın en büyük ve en küçük özdeğerleri ile belirlenmesi teorik olarak önerilmiştir. Oluşturulan rassal kuadratik test problemleri üzerinde gerçekleştirilen deneylerde önerilen etkin öğrenme parametrelerinin momentumlu gradyan düşümü algoritmasının yakınsama tavrını geliştirdiği ve yakınsama hızını diğer geleneksel gradyan düşümü algoritmalara göre daha çok hızlandırdığı gözlenmiştir.

Daha sonra etkin öğrenme parametreleri, yapay sinir ağında hatanın ağırlıkların doğrusal olmayan bir fonksiyonu olduğu genel durum için yerel kuadratik yaklaşım yardımıyla BPM algoritmasına uyarlanmıştır. Bunun için momentumlu gradyan düşümü algoritmasının her adımında oluşan ağırlık noktasında etkin öğrenme parametreleri, uygun Hessian'ın en büyük ve en küçük özdeğerlerini kullanarak, önerilen formüllerle hesaplanmıştır.

Hessian'ın en büyük ve en küçük özdeğerini yaklaşık olarak hesaplamak için önerilen *birinci yaklaşımda*, quasi-Newton metotlarındaki popüler BFGS ve DFP güncellemeleri ile her adımda Hessian yaklaşık olarak hesaplanır ve buradan da en büyük ve en küçük özdeğer elde edilir (A1 ve A2 algoritmaları). Birinci yaklaşım $O(N^2)$ maliyete sahiptir. *İkinci yaklaşımda* ise Hessian-vektör çarpımını ve Güç iterasyonunu beraber kullanarak Hessian'ın kendisini hesaplamadan, sadece en büyük ve en küçük özdeğer yaklaşık olarak hesaplanır (A3 ve A4 algoritmaları). Bu yaklaşım gradyan hesabına eşdeğer $O(N)$ maliyetlidir ve büyük boyutlu problemlerde de sağlıklı bir şekilde çalışır.

Literatürde klasik gradyan düşümü algoritmalarında, yakınsamayı hızlandıran öğrenme parametrelerinin seçimi için bazı sezgisel yöntemler önerilmesine rağmen bu yaklaşımlar farklı ağ mimarilerinde ve farklı özelliklere sahip veri kümelerinde her zaman sağlıklı çalışmaz. Bu tez çalışmasında geliştirilen A1, A2, A3, A4 algoritmalarında ise her adımda ortaya çıkan ağırlık noktasında veri kümesinden elde edilen eğrisellik bilgisini gradyan bilgisiyle

birleştirerek etkin öğrenme oranı ve momentum faktörü (3.39) formülleri kullanılarak eş-zamanlı ve otomatik olarak belirlenir. Bu yaklaşım öğrenme parametrelerini veri kümesine, ağ mimarisine ve oluşan ağırlık noktasına göre adapte ederek gradyan düşümü algoritmasının yakınsama tavrını iyileştirir ve yakınsamasını hızını arttırır. Bu sonuç gerçekleştirilen deneylerden de açıkça görülmektedir.

Büyük genişlikte veri kümesine sahip problemlerde gerek A1 ve A2 algoritmalarında her adımda gerçekleştirilen Hessian yaklaşımları, gerekse A3 algoritmasında çok sayıda gerçekleştirilen Hessian-vektör çarpımları bazı durumlarda karşılanamayacak şekilde maliyetli olur. A4 algoritması etkin parametreleri her adım yerine belirli adımlarda hesaplayarak, bu tip problemlerde önemli bir avantaj sağlar. Bu algoritmada etkin parametrelerin güncellenmesi önceden belirlenmiş kriterler sağlandığında gerçekleşir. A4 algoritmasının verimliliği parametre güncelleme kriterlerinin daha sağlıklı bir şekilde geliştirilmesi ile daha da artırılabilir.

Günümüzde karşılaşılan problemlerin boyutu ve veri kümelerinin genişliği giderek artmaktadır. Toplu ağırlık güncellemesi ile çalışan klasik optimizasyon teknikleri küçük ve orta boyutlu problemlerde her ne kadar çok iyi yakınsama özelliklerine sahip olsa da, büyük genişlikte veri kümesine sahip büyük boyutlu problemlerde çok maliyetli olduklarından, pratikte uygulanması çok elverişli değildir. Bu nedenle son yıllarda toplu ağırlık güncellemesi ile çalışan metotlar yerine stokastik yaklaşım kullanan gradyan temelli metotlar günden güne daha da önemli bir konuma gelmektedir.

Stokastik (çevrimiçi) gradyan düşümü metotları tüm veri setinden hesaplanan gradyanlar yerine veri kümesinden küçük örneklemler kullanarak elde edilen gradyan tahminleri ile çalışır. Bu yaklaşım da hesaplama maliyetini çok büyük oranda azaltır: tekrarlı veriye sahip büyük boyutlu veri kümelerinde basit stokastik gradyan düşümü birçok karmaşık ikinci düzey toplu metotlardan çok daha üstün performans gösterir.

Bu doktora tez çalışmasında momentumlu gradyan düşümü için önerilen etkin öğrenme parametrelerinin ileride stokastik yaklaşımla çalışan gradyan düşümü metotlarına da uyarlanabilmesi amaçlanmaktadır.

KAYNAKLAR

Allred, L. G. ve Kelly, G. E. (1990), "Supervised learning techniques for backpropagation networks," *Proceedings of the International Joint Conference on Neural Networks I*, San Diego 702-709.

Asuncion, A & Newman, D.J. (2007), *UCI Machine Learning Repository* [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

Barnard E. (1992), "Optimization for training neural nets," *IEEE Transactions on Neural Networks*, **3** (2), 232-240

Battiti R. (1992), "First and second order methods for learning: Between steepest descent and Newton's method," *Neural Computation*, **4** (2), 141-166

Battiti, R. (1989), "Accelerated backpropagation learning: two optimization methods," *Complex Systems*, **3**, 331-342.

Bhaya, A. ve Kaszkurewicz E. (2004), "Steepest descent with momentum for quadratic functions is a version of the conjugate gradient method," *Neural Networks* **17**, 65-71

Bishop, C. M. (1995), *Neural networks for pattern recognition*. Oxford Univ. Press

Brent, R. P. (1973), *Algorithms for Minimization without Derivatives*. Englewood Cliffs, NJ: Prentice-Hall

Brogan, W.L. (1991), *Modern Control Theory*. Englewood Cliffs, NJ: Prentice-Hall

Charalambous, C. (1992), "Conjugate gradient algorithm for efficient training of artificial neural networks," *IEEE Proceedings*, **139** (3), 301-310

Davidon W. C. (1959), *Variable metric method for minimization*, Technical ReportANL–5990 (revised), Argonne National Laboratory, Argonne, II

Duda,R.O., & Hart,P.E. (1973) Pattern Classification and Scene Analysis. (Q327.D83) John Wiley & Sons. ISBN 0-471-22361-1. See page 218.

Fahlman, S. E. (1988), "Faster-learning variations on back-propagation: an empirical study," *Proceedings of the 1988 Connectionist Models Summer School*, 38-51. Pittsburgh, PA: Morgan Kaufmann.

Fisher, R. A. (1936), "The use of multiple measurements in taxonomic problems," *Annual Eugenics*, **7**, Part II, 179-188.

Fletcher, R. (1987), *Practical Methods of Optimization (Second ed.)* New York: John Wiley

Gill, P. E., W. Murray ve M. H. Wright. (1981), *Practical Optimization*, New York: Academic Press.

Golub G. ve Van Loan C. (1996), *Matrix Computations*. John Hopkins University Press, Baltimore, MD, 3rd edition.

Greenbaum, A. (1997), *Iterative methods for solving linear systems*. Philadelphia: SIAM

Hagan, M.T., Demuth, H.B. ve Beale, M.H. (1996), *Neural Network Design.* Boston, MA: PWS

Hagiwara M. ve Sato A. (1995). "Analysis of momentum term in backpropagation," *IEICE Trans. Inform. Syst*., **E78-D** (8). Aug.

Haykin, S. (1999), *Neural Networks (2nd ed.)*. Upper Saddle River. NJ:Prentice-Hall

Hestenes, M. R. ve Stiefel E. (1952), "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards* **49** (6), 409-436.

Hinton, G.E. (1989), "Connectionist learning procedures," *Artificial Intelligence*, **40**, 185-234.

Jacobs, R. A. (1988), "Increased rates of convergence through learning rate adaptation," *Neural Networks*, **1** (4), 295-308

Kamarthi, S. V. ve Pittner, S. (1999), "Accelerating neural network training using weight extrapolations," *Neural Networks*, **12** (9), 1285-1299

Le Cun, Y. (1985), "Une procedure d'apprentissage pour reseau a seuil assymetrique," *Cognitiva*, **85**, 599-604

Le Cun, Y., Simard, P. Y. ve Pearlmutter, B. A. (1993), "Automatic learning rate maximization by on-line estimation of the Hessian's eigenvectors," *In (Hanson et al., 1993)*, 156-163.

Lenard M. L., Minkoff M. (1984), "Randomly Generated Test Problems for Positive Definite Quadratic Programming," *ACM Transactions on Mathematical Software (TOMS)*, **10** (1), 86-96.

Mammadov, M., Tas E ve Omay R.E. (2007), "Accelerating Backpropagation using Effective Parameters at each step and an Experimental Evaluation," *Journal of Statistical Computation and Simulation* (basımda).

Memmedli M., Tas E. (2006), "An Improvement of Backpropagation Algorithm with Effective Dynamic Learning Rate and Momentum," *WSEAS Transactions on Mathematics*, **7** (5), July.

M. Memmedli, E.Taş. (2007a), "Her Adımda Etkin Öğrenme Parametrelerinin Seçimi ile Geriye Yayılım Algoritmasını Geliştirilmesi," *5. İstatistik Kongresi*, 20-24 Mayıs, Antalya

Memmedli M., Tas E. (2007b), "BPM Algorithms with Near-Optimal Learning Parameters," *ICSTE-2007*, 19-21 November, Baku.

Møller, M. (1993), "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks* **6** (4), 525-533

Nocedal J. ve Wright S. (1999), *Numerical Optimization*. Springer

Parker, D. B. (1985), *Learning-logic: Casting the cortex of the human brain in silicon*. Technical Report TR-47, Center for Computational Research in Economics and Management Science, MIT, Cambridge, MA

Pearlmutter, B. A. (1994), "Fast exact multiplication by the Hessian," *Neural Computation* **6** (1), 147-160.

Phansalkar, V.V. ve Sastry, P.S. (1994), "Analysis of the backpropagation algorithm with momentum," *IEEE Trans. Neural Networks*, **5**. May**.**

Plaut, D.; Nowlan, S. ve Hinton, G.E. (1986), *Experiments on learning by backpropagation*. Technical Report CMU-CS-86-126, Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA

Qian, N. (1999), "On the momentum term in gradient descent learning algorithms," *Neural Networks*, **12** (1), 145-151

Rigler, A. K., Irwine, J. M. ve Vogl, T. P. (1990), "Rescaling of variables in back propagation learning," *Neural Networks*, **3** (5), 561-573

Rosenblatt, F. (1958), "The Perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, **65**, 386-408

Rumelhart, D. E., Hinton, G. E., ve Williams R. J. (1986), "Learning representations by back-propagating errors," *Nature*, **323**, 533-536

Rumelhart, D. E. ve Mc Clelland, J. L. eds, (1986), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. **1**, Cambridge, MA:MIT Press

Shanno, D. F. (1990), "Recent advances in numerical techniques for large-scale optimization," *Neural Networks for Control*, Miller, Sutton and Werbos, eds., Cambridge, MA:MIT Press

Taş, E. (2003), *Yapay Sinir Ağlarında Dik İniş ve Eşlenik Gradyan Eğitim Algoritmalarının Karşılaştırılması*, Yüksek Lisans Tezi, Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Eskişehir.

Tas E. ve Memmedli M. (2007), "Modified Gradient Descent Algorithms based on Effective Parameter Selection," *22nd European Conference on Operational Research EURO XXII*, 8-11 July, Prag.

Tollenaere, T. (1990), "SuperSAB: Fast adaptive back propagation with good scaling properties," *Neural Networks*, **3** (5), 561-573

Torii, M. ve Hagan, M. T. (2002), "Stability of steepest descent with momentum for quadratic functions," *IEEE Transactions on Neural Networks*, **13** (3), 752-756

Vogl, T. P., Mangis, J. K., Zigler, A. K., Zink W. T. ve Alkon, D. L. (1988), "Accelerating the convergence of the backpropagation method," *Biological Cybernetics*, **59**, 256-264

Werbos, P. J. (1974), *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. PhD Thesis, Harward University, Cambridge, MA

Werbos, P. J. (1988), "Backpropagation: Past and future," *IEEE International Conference on Neural Networks*, **1**, 343–353, San Diego, CA.

Widrow, B. ve Hoff, M.E. (1960), *Adaptive switching circuits*, IRE WESCON Convention Record, 96-104

Williams, P. M. (1991), *A Marquardt algorithm for choosing the step-size in backpropagation learning with conjugate gradients*. Technical Report CSRP **299**, University of Sussex, Brighton, UK

Yu, X.-H. ve Chen, G.-A. (1997), "Efficient backpropagation learning using optimal learning rate and momentum," *Neural Networks*, **10** (3), 517-527

Ek-1 Sherman-Morrison-Woodbury Formülü

Tekil (singular) olmayan A kare matrisi rank-bir güncellemeye girerse aşağıdaki şekle gelir:

$$\overline{A} = A + ab^T ,$$

burada $a, b \in R^n$, o zaman eğer $\overline{A}$ tekil olmayan (tersi olan) matris ise

$$\overline{A}^{-1} = A^{-1} - \frac{A^{-1}ab^T A^{-1}}{1 + b^T A^{-1}a}$$

eşitliğini elde edilir.