Optimizasyon Teknikleri

Ders Notu – 4

KISITSIZ OPTIMIZASYON

Prof. Dr. Bilal ALATAŞ

İÇERİK

- **KISITSIZ OPTIMIZASYON PROBLEMİ**
- **ARAŞTIRMA YÖNÜ (SEARCH DIRECTION) KONSEPTİ**
- **KISITSIZ OPTİMİZASYON METOTLARI**
  - **STEEPEST DESCENT METODU**
  - **ESLENIK GRADYENT METODU**
  - **NEWTON METODU**
  - **QUASI-NEWTON METODU**

KISITSIZ OPTIMIZASYON PROBLEMI

Kisitsiz optimizasyon problemi tanimi:

✓ ***Minimize***

$$f(x) = f(x_1, x_2, \ldots\ldots, x_n)$$

- Lineer problemlerden farkli olarak, nonlineer problemler için global optimumu bulan matematiksel bir yöntem yoktur (konveks problemler hariç).

- Kisitsiz optimizasyon problemlerinin çogunda *sezgisel* (*heuristic*) metotlar kullanilir.

- Kisitsiz optimizasyon yöntemlerinde çogunlukla search direction (arastirma yönü) metotlari kullanilir.

En iyi çözümü araştırılacak model,

enb(veya enk)$f(x_1, x_2, \ldots\ldots, x_n)$ $(x_1, x_2, \ldots\ldots, x_n) \in R^n$ .....(1)

olsun.

Bu modelde $x_1, x_2, \ldots\ldots, x_n$; n sayıdaki değişkenler ve $f(x_1, x_2, \ldots\ldots, x_n)$ bunların bir fonksiyonudur. Bu fonksiyonun x^* gibi bir noktada en iyi çözüme sahip olabilmesi için $f(x_1, x_2, \ldots\ldots, x_n)$ 'in bütün noktalarda birinci ve ikinci dereceden kısmi türevleri bulunabilmeli ve bunlar sürekli olmalıdır.

$f(x_1, x_2, \ldots\ldots, x_n)$ fonksiyonunun x_i 'ye göre kısmi türevinin x^*'daki değeri $\frac{\partial f(x^*)}{\partial f(xi)}$ olsun.

$x^* = (x_1^*, x_2^*, \ldots.. x_n^*)$ in (1)'deki amaç fonksiyonunun yerel uç noktası olabilmesi için gerekli koşul aşağıdaki teoremle açıklanmıştır.

Teorem-1: x^*,(1)'de verilen problemin bir yerel uç noktası ise $\frac{\partial f(x^*)}{\partial f(xi)}$ =0 olur.

Tanım: i = 1, 2, ….., n için $\frac{\partial f(x^*)}{\partial f(xi)}$ = 0 eşitliğini gerçekleyen x^* 'a çok değişkenli $f(x_1, x_2, \ldots\ldots, x_n)$ fonksiyonunun duraksama noktası denir.

Bir duraksama noktasının yerel en büyük, yerel en küçük veya ne yerel en büyük ne de yerel en küçük olması koşulları aşağıda açıklanmıştır.

Teorem-2: k=1, 2, ….., n için $H_k(x^*) > 0$ ise, duraksama noktası (x^*) (1)'in yerel en küçüğüne karşılık gelir.

Teorem-2': k=1, 2, ….., n için $H_k(x^*)(-1)^k$ ile aynı işarete sahip ve sıfır değilse, duraksama noktası (x^*) (1)'in yerel en büyük olduğu noktadır.

Teorem-2'': $H_n(x^*)\neq 0$ ise ayrıca **2** ve **2'** teoremler sağlanmıyorsa, x^* yerel uç nokta olamaz.

Yerel uç nokta olmayan bir duraksama noktasına "dönüm noktası" denir. Duraksama noktası x^*' a göre $H_n(x^*)=0$ ise, x^* yerel en küçük, yerel en büyük veya dönüm noktası olabilir.

$f(x_1, x_2, \ldots\ldots, x_n)$ konkav ve (1) ile açıklanan problem en büyükleme amaçlı ise en iyi çözüm $f(x_1, x_2, \ldots\ldots, x_n)$'in duraksama noktasında gerçekleşir. Benzer şekilde $f(x_1, x_2, \ldots\ldots, x_n)$ konveks, problem en küçükleme amaçlı ise en iyi çözüm fonksiyonun duraksama noktasında ortaya çıkar.

Örnek-1: $f(x, y)=x^3+3xy^2-6x^2-y^2+1$ fonksiyonunun yerel en büyük, yerel en küçük ve dönüm noktalarını araştırınız.

$\frac{\partial f}{\partial x}=3x^2+3y^2-12x$

$\frac{\partial f}{\partial y}=6xy-2y$

Fonksiyonun duraksama noktalarının bulunması amacıyla gerçekleştirilen işlemler aşağıda gösterilmiştir.

$\frac{\partial f}{\partial x}=3x^2+3y^2-12x=0$ ……(I)

$\frac{\partial f}{\partial y}=6xy-2y=0$……(II)

Önce (II) yi inceleyelim. Bu eşitlik i)y=0 veya ii) 6x-2=0 yani $x=\frac{1}{3}$ olması durumunda sağlanır. y=0 değeri (I) de yerine koyulunca x şöyle hesaplanır:

$3x^2-12x=0$ veya $3x(x-4)=0$ buradan da iii) x=0 veya iv) x=4 ii) den elde edilen x=1/3 değeri (1) nolu eşitlikte x yerine yazılırsa, y aşağıdaki gibi hesaplanır.

$3.(1/9)+3.y^2-12.(1/3)=0$

$3y^2=4-(1/3)=11/3$

$Y^2=11/9$ ise $y=\pm\sqrt{11}/3$

Böylece (0,0),(4,0),(1/3, $\sqrt{11}/3$),(1/3,- $\sqrt{11}/3$) noktalarından herbirini Hessian matrisine yerleştirelim. Önce (0,0) noktasını alalım. Bu noktada H(0,0) aşağıdaki gibi olur;

$$H(0,0)=\begin{bmatrix} -12 & 0 \\ 0 & -2 \end{bmatrix}$$

$H_1(0,0)=-12<0$ ve $H_2(0,0)=24>0$ olur, Teorem-2' ye göre (0,0) yerel en büyüktür.

(4,0) noktasını inceleyelim. Bu noktada H(4,0) aşağıdaki gibi elde edilir;

$$H(4,0)=\begin{bmatrix} 12 & 0 \\ 0 & 22 \end{bmatrix}$$

$H_1=(4,0)=12>0$ ve $H_2(4,0)=264$ olur. Teorem-2'ye göre (4,0) yerel en küçüktür.

(1/3, $\sqrt{11}$/3) noktasi için belirlenen Hessian matrisi ve diğer işlemler aşağıda gösterilmiştir;

$$H(1/3, \sqrt{11}/3)=\begin{bmatrix} -10 & 2\sqrt{11} \\ 2\sqrt{11} & 0 \end{bmatrix}$$

$H_1(1/3, \sqrt{11}/3)=-10<0$

$H_2(1/3, \sqrt{11}/3)=-44<0$ elde edilir. Teorem 2 ve 2' sağlanmadığından ve $H_2 \neq 0$ olduğundan (1/3, $\sqrt{11}$/3) dönüm noktasıdır.

Son olarak (1/3,$-\sqrt{11}$/3) noktasını inceleyelim. Bu noktadaki Hessian matrisi aşağıda gösterilmiştir.

H(1/3,$-\sqrt{11}$/3)=$\begin{bmatrix} -10 & -2\sqrt{11} \\ -2\sqrt{11} & 0 \end{bmatrix}$ Buna göre;

H_1 (1/3,$-\sqrt{11}$/3)=-10 ve H_2 (1/3,$-\sqrt{11}$/3)=-44 olarak belirlenir. 2 ve 2' nolu Teoremler saglanmadığından ve ayrıca $H_2 \neq 0$ olduğundan, (1/3,$-\sqrt{11}$/3) dönüm noktasıdır

ARASTIRMA YÖNÜ (SEACH DIRECTION) KONSEPTI

- Optimizasyon problemi çözümünde çogunlukla *search direction* (arastirma yönü) yöntemi kullanilir.
- *Search direction* optimizasyon yöntemlerinde yeni nokta eski noktanin biraz modifikasyonu (düzeltilmesiyle) elde edilir.

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)}$$

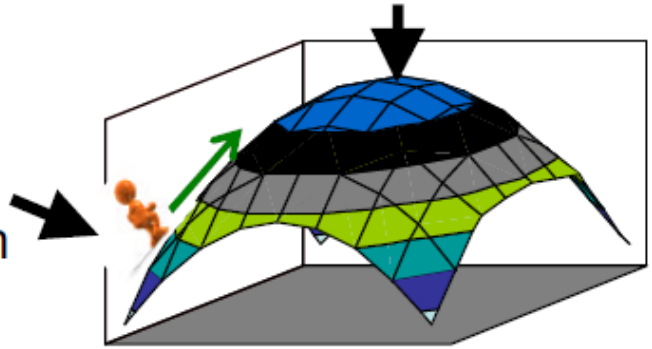
Modifikasyon miktari = ***alfa(k)*d(k)***

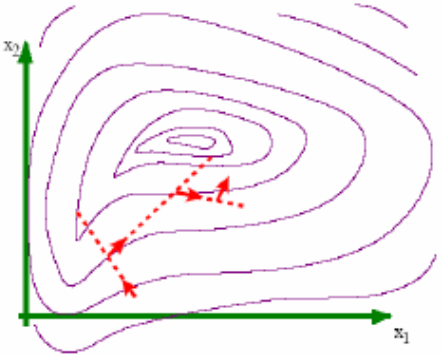
Mdifikasyon miktari kolaylik için iki parçaya ayrilmistir: **alfa(k)**, **d(k)**

d(k) : bulundugumuz noktadan bizi daha iyi bir noktaya götürecek yön (search direction).

alfa: adim uzunlugu (***d(k)*** yönü üzerinde fonksiyonun degerini düsüren adim uzunlugu.

<u>**Not**</u>: Metotlar birbirinden ***d(k)***' nin hesaplanmasinda ayrilirlar





İTERATİF ARAŞTIRMA İŞLEMİ

Adım 1. Bir başlangıç çözümü $x^{(0)}$ al. İterasyon sayacını sıfırla

Adım 2. Araştırma uzayında bu nokta için bir araştırma yönü ($d^{(t)}$) belirle.

Sınırlamasız optimizasyonda maliyet fonksiyonu ile gradyentin hesabı

Sınırlamalı optimizasyonda sınırlama fonksiyonlarına ihtiyaç duyar.

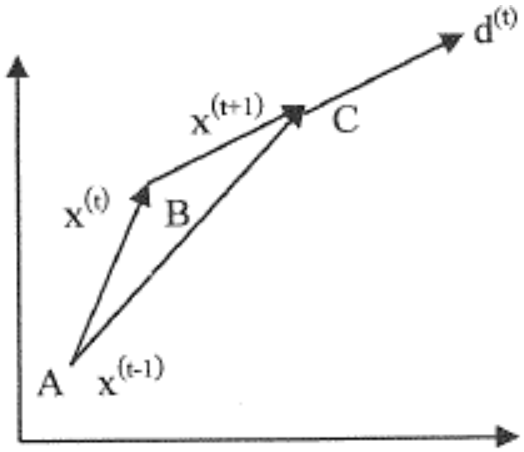
Adım 3. Araştırmanın yakınsamasını kontrol et. Eğer yakınsamış ise bitir, yoksa devam et.

Adım 4. Pozitif bir adım büyüklüğü ($\alpha^{(t)}$) belirle.

Adım 5. Yeni çözümü aşağıdaki formül ile hesapla.

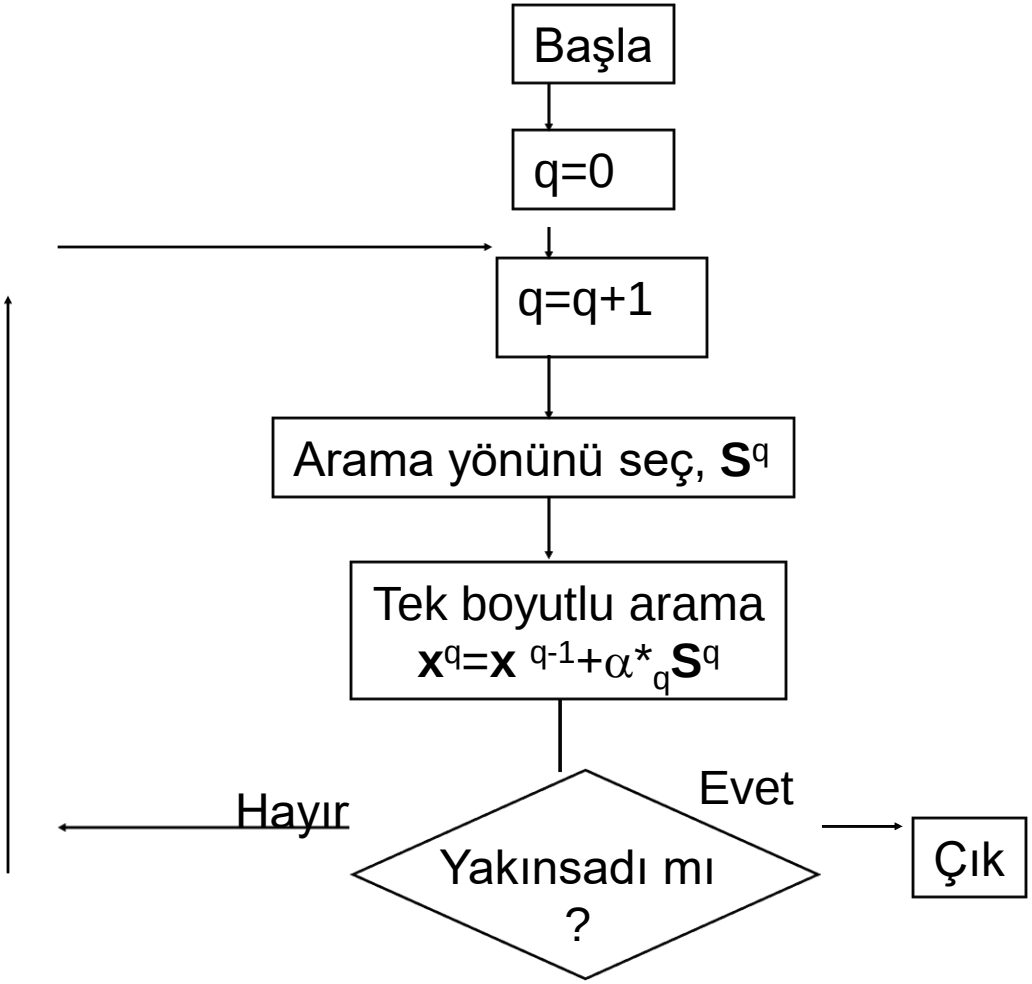
$$x^{(t+1)}=x^{(t)}+ \alpha^{(t)}d^{(t)}$$

Adım 6. İterasyon sayacını bir arttır (t=t+1) ve Adım 2'ye git.



Şekil 1.3. Araştırma işlemi: $x^{(t-1)}$ önceki çözümü $x^{(t)}$ mevcut çözümü, $x^{(t+1)}$ erişilecek bir sonraki çözümü ve $d^{(t)}$ de araştırma yönünü temsil etmektedir.

Genel optimizasyon stratejisi



KISITSIZ OPTIMIZASYON METOTLARI

Search direction metodunu direk veya dolayli olarak kullanan kisitsiz optimizasyon metotlari:

- **Steepest descent (en dik inis) metodu**
- **Eslenik gradyent (Conjugate gradient) metodu**
  - ✓ **Fletcher – Reeves Metodu**
  - ✓ **Polak - Robiee Metodu**
- **Newton metodu**
- **Quasi-Newton metotlari**
  - ✓ **DFP metodu**
  - ✓ **BFGS Metodu**

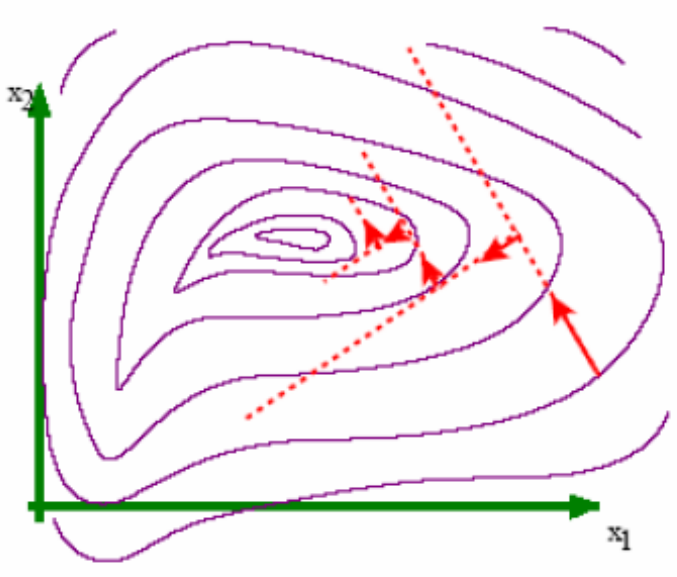
STEEPEST DESCENT METODU

- Steepest descent metodu çok degiskenli problemler için, türev (gradyent) esasli optimizasyon metotlarinin en basitidir.

- Search direction (arastirma yönü) için fonksiyonun gradyent yönünün tersi kullanilir:

$$d^{(k)} = -\frac{\nabla f}{|\nabla f|}$$

- . Bu yön kisa vadede fonksiyonun degerini en hizli düsüren yöndür.



enb $f(x_1,x_2,\ldots\ldots,x_n)$, $(x_1,x_2,\ldots\ldots,x_n) \in R^n$ .....(2)

Tanım: $x=(x_1,x_2,\ldots\ldots,x_n) \in R^n$ vektörünün uzunluğu $\|x\|$ ile gösterilir ve $\|x\|=(x_1^2+x_2^2+\ldots+x_n^2)^{1/2}$ bağıntısından hesaplanır.

N boyutlu her vektör R^n içinde bir yön tanimlar. Herhangi bir yön sonsuz sayıda vektörle gösterilebilir. Örneğin; (1,1), (2,2), (3,3) vektörlerinin hepsi iki boyutlu uzayda pozitif hareketle 45°lik açıdaki yönü gösterir. Herhangi bir x vektörü için $x/\|x\|$ vektörünün uzunluğu 1 birimdir ve bu vektör x ile aynı yöndedir. Buna göre R^n deki herhangi bir yön uzunluğu 1 birim olan bir vektör (birim vektör) ile açıklanabilir. x=(6,8) vektörünü birim vektör olarak açıklayalım. $\|x\|=(6^2+8^2)^{1/2}=10$ olduğundan x=(6,8) vektörünün yönü birim vektör (6/10,8/10) ile ilişkilendirilir. Herhangi bir x vektörü için tanımlanan $x/\|x\|$ birim vektörüne x'e karşılık gelen "normalleştirilmiş vektör" denir. R^n deki herhangi bir yön, o yönü tanımlayan normalleştirilmiş vektörlerle açıklanabilir.

Konuya açıklık kazandırma için bütün noktalarda kısmi türevlere sahip $f(x_1,x_2,\ldots\ldots,x_n)$ fonksiyonunu ele alalım.

Tanım: $f(x_1,x_2,\ldots\ldots,x_n)$'in birinci mertebeden kısmi türevleriyle oluşturulan n elemanlı vektöre $f(x_1,x_2,\ldots\ldots,x_n)$'in gradyant vektörü denir ve aşağıdaki gibi gösterilir.

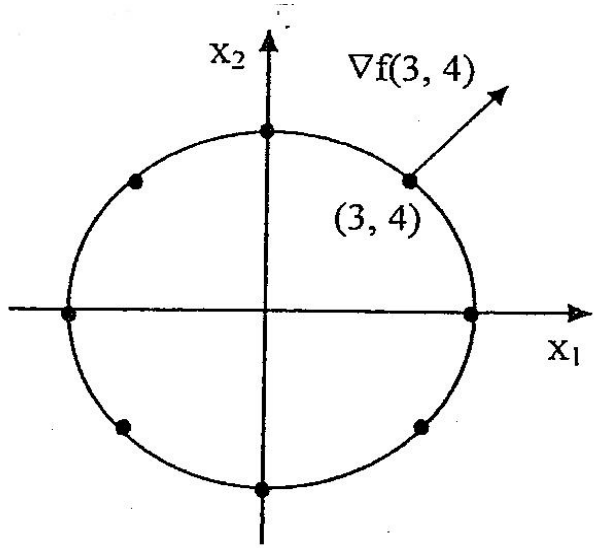
$$\nabla f(x)=\left[\frac{\partial f(x)}{\partial(x1)}, \frac{\partial f(x)}{\partial(x2)},\ldots\ldots, \frac{\partial f(x)}{\partial(xn)}\right]$$

$\nabla f(x)$, $\frac{\nabla f(x)}{II\nabla f(x)II}$ ile aynı yöndedir.

Mesela $f(x,y)=x_1^2+x_2^2$ ise $\nabla f(x)=(2x_1,2x_2)$ olur. $x_1=3$ ve $x_2=4$ olarak seçildiğinde $\nabla f(3,4)= (6,8)$ olarak elde edilir. $\|\nabla f(3,4)\|=10$ olduğundan, $\nabla f(3,4)$'un yönü $(6/10,8/10)=(0.6,0.8)$ olarak belirlenir.

$f(x_1,x_2,\ldots\ldots,x_n)=f(x^*)$ eğrisi üzerindeki bir x^* noktasında tanımlanan $\frac{\nabla f(x*)}{\|\nabla f(x*)\|}$ birim vektörü $f(x^*)$ eğrisine diktir.

Mesela, birim vektör, $\frac{\nabla f(x3,4)}{\|\nabla f(3,4)\|}$ $=(0.6,0.8)$, $x_1^2+x_2^2=25$ eğrisine (3,4) noktasında diktir.



$\frac{\partial f(x)}{\partial(xi)}$'in tanımıyla açıklandığı gibi x_i'nin değeri,δ küçük bir değer olmak üzere, δ kadar arttırılırsa f(x)'in değeri yaklaşık $\delta\frac{\partial f(x)}{\partial(xi)}$ kadar artar. Bir x noktasından δ uzunluğunda ve normalleştirilmiş sütun vektörü d yönünde hareket ettiğimizi varsayalım. Bu durumda f(x), $\frac{\nabla f(x)}{II\nabla f(x)II}$ ile d'nin skaler çarpımının δ kati kadar artar. Yani,f(x)'deki artış $\delta\frac{\nabla f(x)}{II\nabla f(x)II}$ d'ye eşit olur. Buna göre $\delta\frac{\nabla f(x)}{II\nabla f(x)II}$ d>0 ise x'den d yönünde uzaklaşılması f(x)'in değerinin artmasına; $\delta\frac{\nabla f(x)}{II\nabla f(x)II}$ d <0 ise, x'den d yönünde uzaklaşılması f(x)'in değerinin azalmasına yol açacaktır.

$f(x_1,x_2)=x_1^2+x_2^2$ fonksiyonunu dikkate alarak (3,4) noktasından 45°'lik açıyla δ kadar uzaklaştığımızı düşünelim. Bu durumda $f(x_1,x_2)$ fonksiyonunun değerindeki değişiklik şu şekilde hesaplanacaktır. 45°'lik açı $(1/\sqrt{2},1/\sqrt{2})$ vektörüyle gösterildiğinden $d=(1/\sqrt{2},1/\sqrt{2})$ olur. $\frac{\nabla f(3,4)}{II\nabla f(3,4)II}\cong(0.6,0.8)$ olduğundan $f(x_1,x_2)$'in değeri yaklaşık,

$\delta[0.6\ \ 0.8][\frac{1/\sqrt{2},}{1/\sqrt{2},}]=0.99\delta$ kadar artacaktır.

Bir önceki kesimde açıklandığı gibi v^* (2) ile açıklanan problemin en iyi çözümü ise $\nabla f(v^*)=0$ olur.v_0 gibi bir noktada bulunduğumuzu ve (2)'yi çözen v^* noktasını bulmak istediğimizi varsayalım. v^*'I bulmak için $f(x_1,x_2,.....,x_n)$'I artan oranda arttıran yönde hareket edilmesi gerekir. v'den çok az uzaklaşarak f(x)'in en hızlı bicimde artmasını sağlayabilmek için yapılacak hareketin en büyük artma doğrultusunu verir.

En dik artış yöntemi uygulaması için yaklaşık bir v_0 başlangıç noktası seçilir, fonksiyonu en büyükleyen noktaya ulaşıncaya kadar $v_1, v_2, ..$ noktalar dizisi türetilir. Başlangıç noktası v_0 olsun. v_0'dan ∇v_0 yönünde uzaklaşarak t'nin bazı negatif olmayan değerleri icin $v_1 = v_0 + t_0 \nabla f(v_0)$ bağıntısıyla açıklanan noktadır. v_0'dan $v_1 = v_0 + t_0 \nabla f(v_0)$'a hareket edildiğinde t_0, aşağıdaki tek boyutlu en iyileme problem için çözümdür.

$$\text{enb} f(v_0 + t_0 \nabla f(v_0)) \qquad t_0 \geq 0 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

ulaşılan yeni noktaya göre tanımlanan $\|\nabla f(v_1)\|$ uzaklığı çok küçük (0.01'den küçükse) ise, çözüm bulunmuş olur. $\|\nabla f(v_1)\|$ yeterince küçük değilse bu kez, v_1'den $\|\nabla f(v_1)\|$ yönünde t_1 mesafesinde uzaklaşılır. t_0'in bulunmasında olduğu gibi t_1, aşağıdaki fonksiyonun çözüm sonucuyla belirlenir.

$$\text{enb} f(v_1 + t_1 \nabla f(v_1)) \qquad t_1 \geq 0$$

Bu çözümle, $v_2 = v_1 + t_1 \nabla f(v_1)$ noktasına ulaşılır. $\|\nabla f(v_2)\|$ yeterince küçükse, v_2 problemin en iyi çözümü olur. $\|\nabla f(v_2)\|$ yeterince küçük değilse işlemler en küçük $\|\nabla f(v_n)\|$ uzaklığını veren v_n noktasına ulaşıncaya kadar tekrarlanır. Sonuçta v_n, $f(x_1, x_2, \ldots\ldots, x_n)$'in duraksama noktasına en yakin nokta olarak seçilir.

Kısaca En Dik İniş

- *f*'de en büyük düşüşü veren yönde hareket et

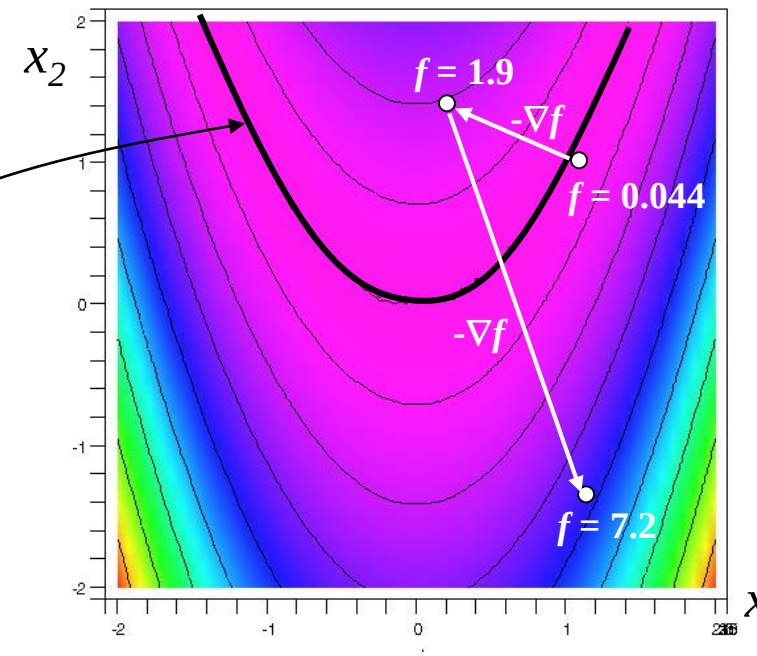
Taylor:
$$f(\mathbf{x}+h\mathbf{s}) = f(\mathbf{x}) + (\nabla f)^T \mathbf{s}h + o(h^2)$$

$$\Rightarrow df = f(\mathbf{x}+h\mathbf{s}) - f(\mathbf{x}) \approx (\nabla f)^T \mathbf{s}h$$

En iyi yön:
$$\mathbf{s} = -\nabla f$$

- Örnek:
$$f = x_1^4 - 2x_1^2 x_2 + x_2^2$$

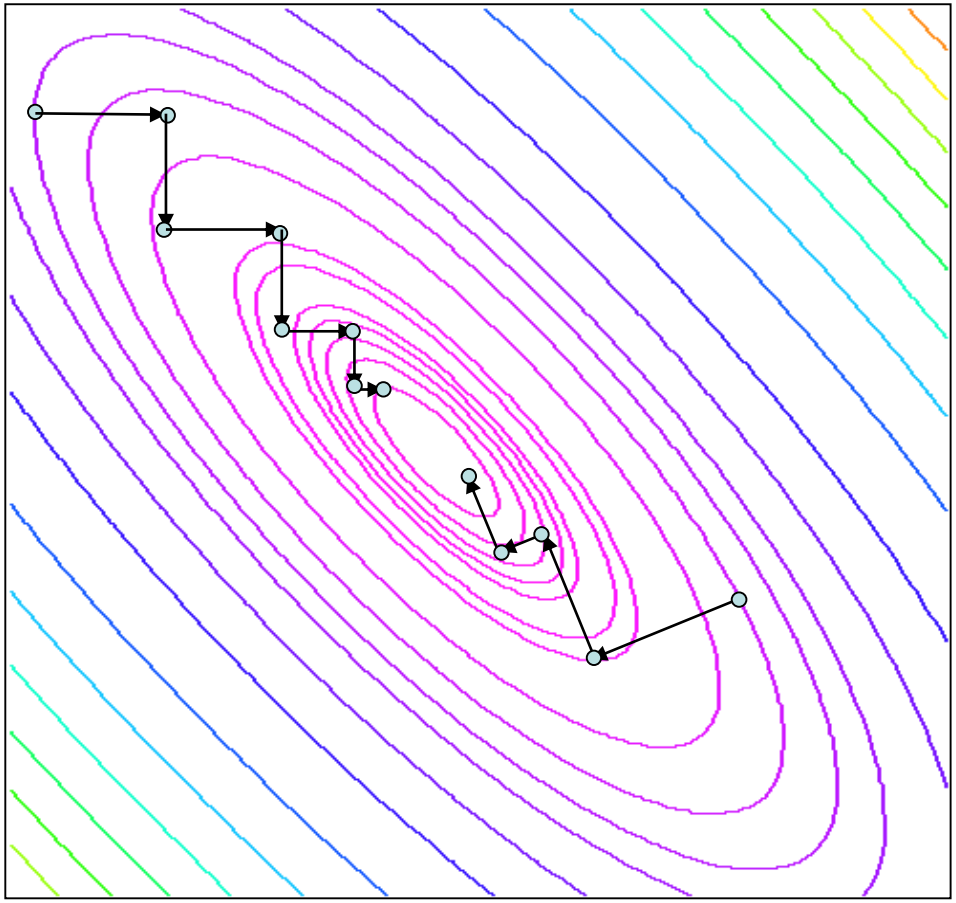
$$\nabla f = \begin{pmatrix} 4x_1^3 - 4x_1x_2 \\ -2x_1^2 + 2x_2 \end{pmatrix} = 2(x_1^2 - x_2)\begin{pmatrix} 2x_1 \\ -1 \end{pmatrix}$$



Uzaklaşma olur! Çare: *çizgi arama*

En Dik İniş Yakınsama

- Zik-zak yakınsama davranışı:



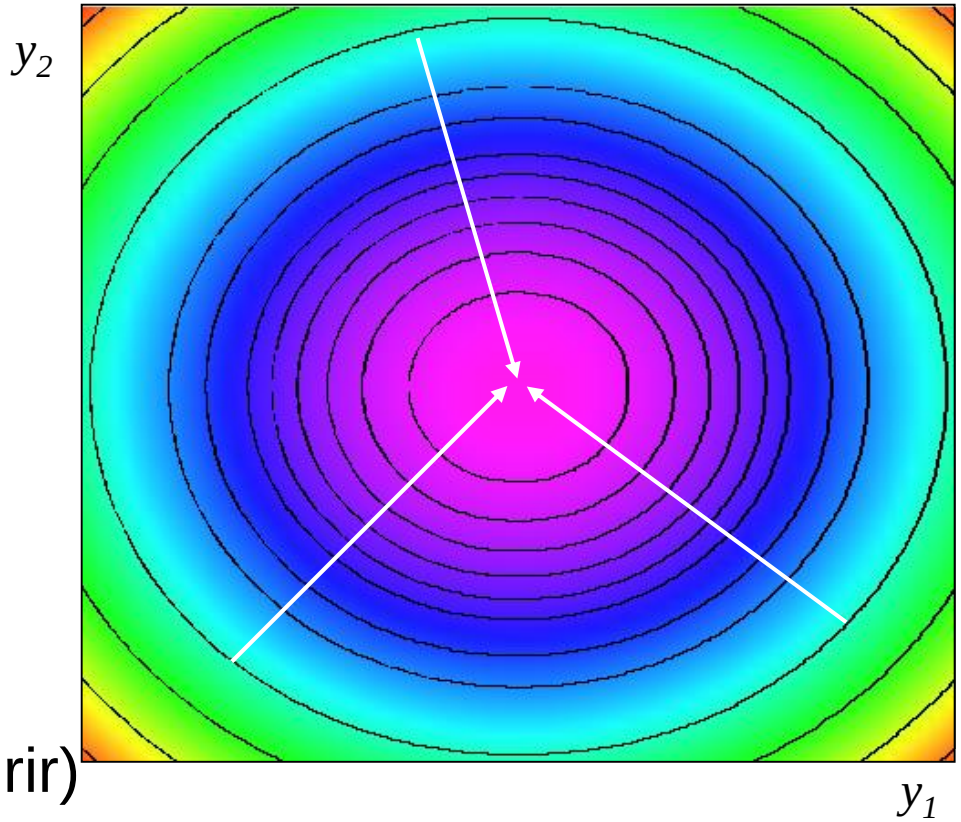
Ölçekleme etkisi

$$f = x_1^{\ 2} + 16 x_2^{\ 2}$$

$$y_1 = x_1, \quad y_2 = 4x_2 \Rightarrow$$

$$f = y_1^{\ 2} + y_2^{\ 2}$$

İdeal ölçeklemenin belirlenmesi zordur (Hessian bilgisi gerektirir)

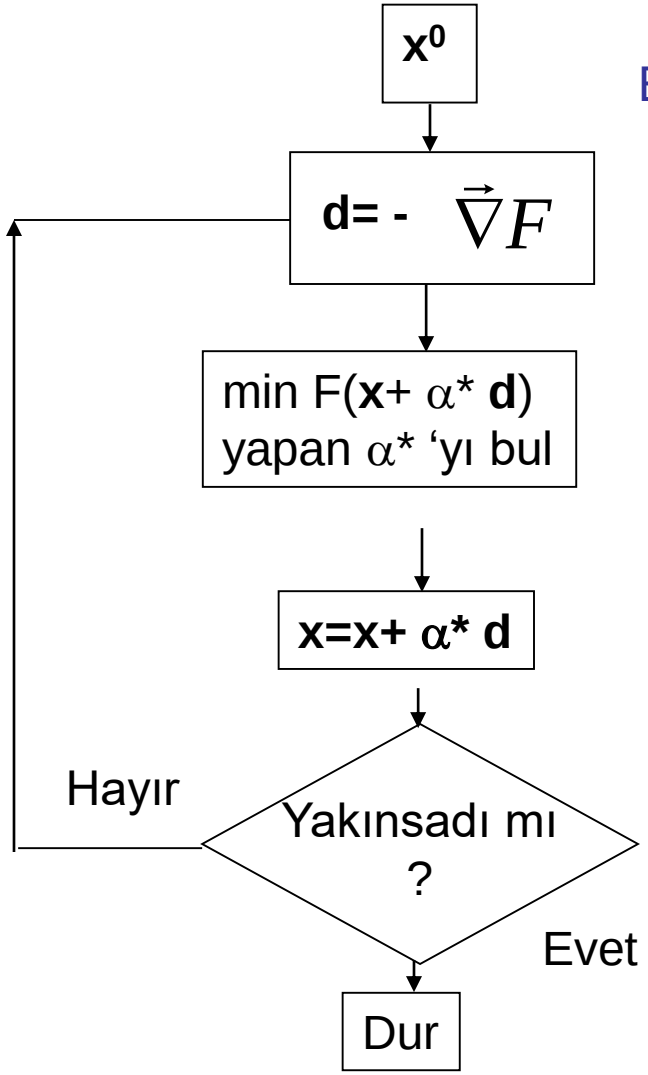


ÖZET

DİK İNİŞ YÖNTEMİ (Steepest Descent): Bu yöntemin algoritması şu şekildedir:

1. Başlangıç parametre değerleri $\mathbf{x}^{(0)}$ belirlenir ve iterasyon sayacı k=0 alınır. Yakınsama değeri $\varepsilon > 0$ olarak seçilir.
2. $\mathbf{x}^{(k)}$ noktasında f(x) fonksiyonunun gradyanı hesaplanır. $\mathbf{c}^{(k)} = \nabla f(\mathbf{x}^{(k)})$
3. $\|\mathbf{c}^{(k)}\|$ hesaplanır. Eğer $\|\mathbf{c}^{(k)}\| < \varepsilon$ ise iterasyon **bitirilir**, optimum çözüm $\mathbf{x}^* = \mathbf{x}^{(k)}$ dır, değilse iterasyona **devam edilir**.
4. $\mathbf{x}^{(k)}$ noktası için arama yönü $\mathbf{d}^{(k)} = -\mathbf{c}^{(k)}$ ile hesaplanır.
5. $f(\mathbf{x}^{(k)} + \alpha \mathbf{d}^{(k)})$ fonksiyonunu minimum yapan α_k değeri bulunur. Bunun için gerek ve yeter şartlar yazılabilir veya tek boyutlu arama algoritmaları kullanılabilir.
6. $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}$ ile değerler güncellenir, k değeri artırılır ($k = k + 1$) ve 2. adıma gidilir.

Algoritma



En dik iniş yönünü belirle

En dik iniş yönünde
tek boyutlu minimizasyon
uygula

Tasarımı güncelle

STEEPEST DESCENT METODU

Performans degerlendirmesi:

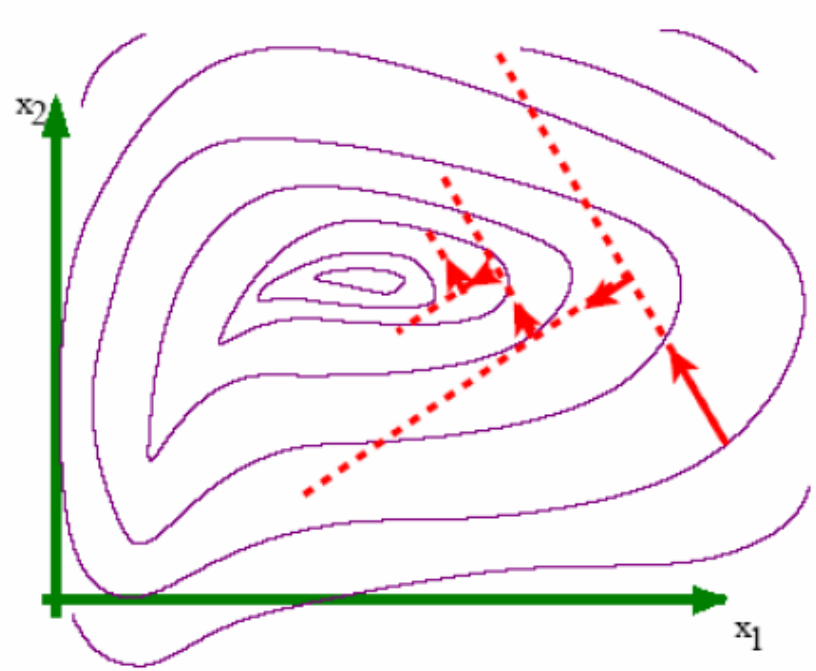
- Steepest descent metodunda her yön bir önceki yöne diktir.

- Bu nedenle optimum, bir çok arastirma yönü kullanildiktan sonra (zikzakli bir yoldan sonra) bulunur.

- Yönlerdeki adim uzunluklari, optimizasyon iterasyonlari ilerledikçe küçülür.

- Adim uzunluklari o kadar küçülebilirki, algoritma yakinsadigini sanabilir.

- Genelde algoritmanin yakinsam hizi düsüktür.



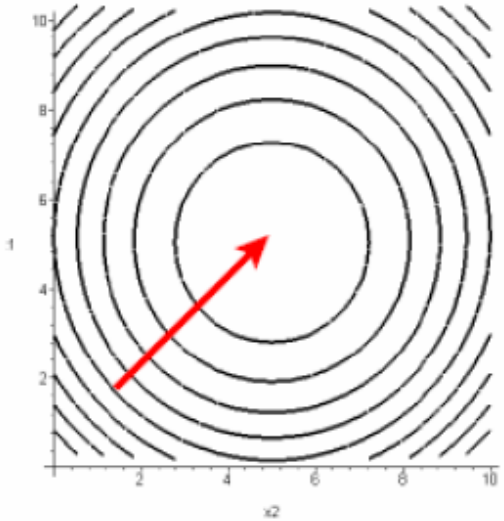
STEEPEST DESCENT METODU

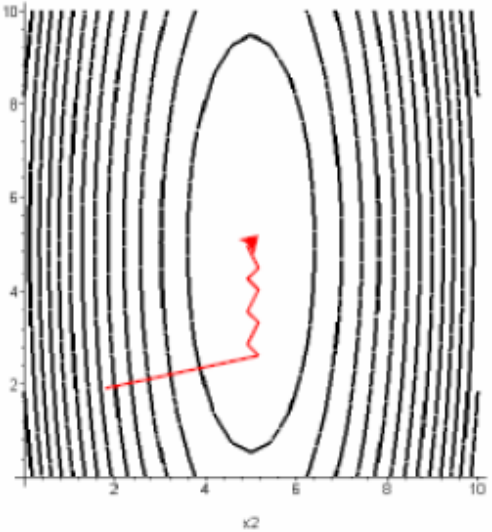
<u>Performans degerlendirmesi (devam)</u>:

Degiskenlerin *ayni mertebede* oldugu problemlerde Steepest descent metodu hizli yakinsar.

Örnegin amaç fonksiyonu konturu dairesel ise, Steepest descent metodu optimumu sadece bir arastirma yönü ile (bir iterasyonda) bulur.

Fakat, amaç fonksiyonu konturu çarpik ise, metodun yakinsamasi bir çok iterasyon gerektirebilir, hatta yakinsama olmayabilir.





Dik iniş yönteminin algoritması her ne kadar basit ve yakınsama açısından problemsiz olmasına rağmen bazı dezavantajları mevcuttur:

1. Yakınsama garantisi olsa da minimum değer için gerekli iterasyon sayısı oldukça **fazladır**.
2. Önceki iterasyonlardaki bilgiler **kullanılmaz**, her iterasyon yeni baştan başlar.
3. Fonksiyonlar hakkında sadece birinci dereceden bilgiler kullanıldığından iterasyon sayısı **yüksektir**.
4. Pratik örnekler göstermiştir ki ilk iterasyonlarda optimum noktaya yaklaşmak hızlı ilerlenirken iterasyonlar ilerledikçe optimum noktaya daha **yavaş** yaklaşılır.
5. Yerel anlamda dik iniş yönü avantajlı görünse de global anlamda **verimsizdir**.

Örnek: Aşağıdaki fonksiyonun minimum noktasını dik iniş yöntemi ile hesaplayınız.

$$\min \quad f(x_1, x_2) = x_1^2 + x_2^2 - 2x_1x_2$$

1. Başlangıç noktası $\mathbf{x}^{(0)} = (1, 0)$,
2. $\mathbf{c}^{(0)} = (2x_1 - 2x_2,\ 2x_2 - 2x_1) = (2, -2)$.,
3. $\|\mathbf{c}^{(0)}\| = 2\sqrt{2} \neq 0$,
4. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = (-2, 2)$,
5. Min $f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)})$, $\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = (1 - 2\alpha, 2\alpha)$,

$$f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = (1-2\alpha)^2 + (2\alpha)^2 - 2(1-2\alpha)(2\alpha)$$
$$= 16\alpha^2 - 8\alpha + 1 = f(\alpha)$$

$$\frac{df(\alpha)}{d\alpha} = 0 \qquad 32\alpha - 8 = 0 \qquad \alpha_0 = 0.25 \qquad \frac{d^2 f(\alpha)}{d\alpha^2} = 32 > 0.$$

6. Güncelleme $(\mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)})$ $x_1^{(1)} = 1 - 0.25(2) = 0.5,\ x_2^{(1)} = 0 + 0.25(2) = 0.5$
7. $\mathbf{c}^{(1)} = (0, 0)$ olduğundan algoritma durdurulur. Optimum değerler (0.5,0.5) ve fonksiyon değeri f*=0 olarak bulunur.

Örnek: Aşağıdaki fonksiyonun minimum noktasını dik iniş yöntemi ile hesaplayınız.

$$\min\ f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3$$

1. $\mathbf{x}^{(0)} = (2, 4, 10)$, ε=0.0001 olsun,
2. $\mathbf{c} = \nabla f = (2x_1 + 2x_2, 4x_2 + 2x_1 + 2x_3, 4x_3 + 2x_2)$; $\mathbf{c}^{(0)} = (12, 40, 48)$
3. $\|\mathbf{c}^{(0)}\| = \sqrt{4048} = 63.6 > \varepsilon$
4. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = (-12, -40, -48)$
5. $f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)})$; $\alpha_0 = 0.1587$
6. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = (0.0956, -2.348, 2.381)$
7. $\mathbf{c}^{(1)} = (-4.5, -4.438, 4.828)$, $\|\mathbf{c}^{(1)}\| = 7.952 > \varepsilon$.
8. İterasyonlara devam edilerek algoritmanın durma şartına kadar devam edilir

40 iterasyon sonunda optimum değerler 8.04787E–03, –6.81319E–03, 3.42174E–03 ve fonksiyon değeri 2.473 47E–05 olarak bulunur.

Örnek : Aşağıda verilen iki değişkenli fonksiyonu minimum yapacak değişken değerlerini yukarıda temel adımları verilen en-dik iniş algoritmasını kullanarak bulunuz (Arora, 1989).

$f(x_1, x_2) = x_1^2 + x_2^2 - 2x_1x_2$

Adım 1. $x^{(0)} = (1, 0)$ (Başlangıç çözümü)

Adım 2. $g^{(0)} = (2x_1-2x_2 , 2x_2-2x_1) = (2, -2)$; $\|g^{(0)}\| = 2\sqrt{2} \neq 0$.

Adım 3. $d^{(0)} = -g^{(0)} = (-2 , 2)$ olarak tayin et.

Adım 4. α'yı $f(x^{(0)}+\alpha d^{(0)})$ fonksiyonunu enküçük yapacak şekilde hesapla. Burada, $x^{(0)}+\alpha d^{(0)} = (1-2\alpha, 2\alpha)$ ve

$$f(x^{(0)}+\alpha d^{(0)}) = (1-2\alpha)^2 + (2\alpha)^2 - 2(1-2\alpha)(2\alpha)$$

$$= 16\alpha^2 - 8\alpha + 1 = f(\alpha)$$

$f(\alpha)$, α'nın tek değişkenli bir fonksiyonu olduğundan optimum adım büyüklüğünü belirlemek için optimalite şartlarını kullanabilir. Analitik yaklaşım kullanarak,

$$\frac{df(\alpha)}{d\alpha} = 0 \quad \text{ve } 32\alpha - 8 = 0 \text{ veya } \alpha_0 = 0.25$$

$$\frac{df^2(\alpha)}{d\alpha^2} = 32 > 0$$ olduğundan, minimumluk için yeter şart sağlanmıştır.

Adım 5. Çözümü ($x^{(0)}+\alpha_0 d^{(0)}$) bağıntısı aracılığıyla yenile.

$x_1^{(1)} = 1-0.25\,(2) = 0.5$

$x_2^{(1)} = 0+0.25\,(2) = 0.5$

İkinci adımdaki ifade $g^{(1)}$ için çözülürse $g^{(1)} = (0, 0)$ elde edilir ve bu da durdurma kriterini sağlar. Bundan dolayı (0.5, 0.5) verilen problem için minimum noktadır.

Örnek: Aşağıdaki problem en dik artış yöntemi ile çözünüz.

enb$f(x_1,x_2)=-(x_1-2)^2-x_1-x_2^2 \qquad (x_1,x_2)\in R^n$

Başlangıç noktası olarak $v_0=(2.5,1.5)$ seçilmiş olsun. Önce fonksiyonun gradyant vektörünü hesaplayalım. $\nabla f(x_1,x_2)$ aşağıda gösterilmiştir.

$\nabla f(x_1,x_2) \;=(-2x_1+3,-2x_2)$

$\nabla f(x_1,x_2)$ 'in v_0'daki değeri $\nabla f(v_0)= \nabla f(2.5,1.5)=(-2,-3)$ olur.

$f(t_0)$ aşağıda gösterilmiştir.

$f(t_0)=f(v_0+t_0\nabla f(v_0))=f(2.5,1.5)+t_0(-2,-3)=f(-2t_0+2.5,-3t_0+1.5)$

$$=-13t_0^2+13t_0-5$$

$f(t_0)$'I en büyük yapan t_0 değerini bulmak için, fonksiyonun birinci türevini alalım. Fonksiyonun t_0 'a göre birinci türevi aşağıda gösterilmiştir.

$\frac{\partial f(t0)}{\partial t0}=-26t_0+13$ olur.

$-26t_0+13=0$ ise buradan $t_0=0.5$ dolayısıyla $v_1=v_0+ t_0\nabla f(v_0)=(2.5,1.5)+0.5(-2,-3)=(1.5,0)$ olarak elde edilir. İlk adımda ulaşılan bu noktada, $\nabla f(1.5,0)=(0,0)$ yeterince küçük olduğundan işlemlere son verilir. En iyi çözüm $x_1=1.5,x_2=0$.

$Z_{enb}=-1.75$ olarak elde edilmiştir.

Örnek

- Use steepest descent method to minimize

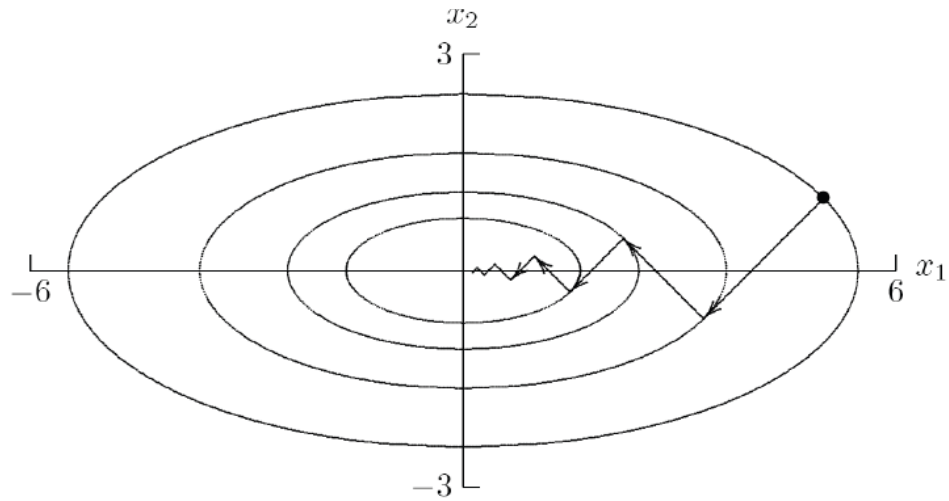
$$f(\boldsymbol{x}) = 0.5x_1^2 + 2.5x_2^2$$

- Gradient is given by $\nabla f(\boldsymbol{x}) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix}$
- Taking $\boldsymbol{x}_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix}$, we have $\nabla f(\boldsymbol{x}_0) = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$
- Performing line search along negative gradient direction,

$$\min_{\alpha_0} f(\boldsymbol{x}_0 - \alpha_0 \nabla f(\boldsymbol{x}_0))$$

exact minimum along line is given by $\alpha_0 = 1/3$, so next approximation is $\boldsymbol{x}_1 = \begin{bmatrix} 3.333 \\ -0.667 \end{bmatrix}$

Örnek Devam



$\boldsymbol{x}_k$		$f(\boldsymbol{x}_k)$	$\nabla f(\boldsymbol{x}_k)$	
5.000	1.000	15.000	5.000	5.000
3.333	−0.667	6.667	3.333	−3.333
2.222	0.444	2.963	2.222	2.222
1.481	−0.296	1.317	1.481	−1.481
0.988	0.198	0.585	0.988	0.988
0.658	−0.132	0.260	0.658	−0.658
0.439	0.088	0.116	0.439	0.439
0.293	−0.059	0.051	0.293	−0.293
0.195	0.039	0.023	0.195	0.195
0.130	−0.026	0.010	0.130	−0.130

ESLENIK GRADYENT METODU

Eslenik gradyent (conjugate gradient) metotlarini gelistirmedeki iki temel sebep:
1- Steepest descent metodundan daha hizli yakinsayan bir metot gelistirmek.
2- Daha hizli yakinsayan Newton metotlari kadar fazla hesap ve hafiza kullanmayan bir metot gelistirmek.

Bu nedenle Eslenik gradyent metotlari, Steepest descent ve Newton metotlarinin her ikisinin pozitif özelliklerini birlestiren metotlar olarak degerlendirilir.

Eslenik gradyent metotlarinin hizli yakinsamasinin sebebi geçmisteki bilgileri kullanmasidir.

Eslenik gradyent metotlari geçmis arastirma yönlerini yeni arastirma yönü hesabinda kullanirlar.

ESLENIK GRADYENT METODU

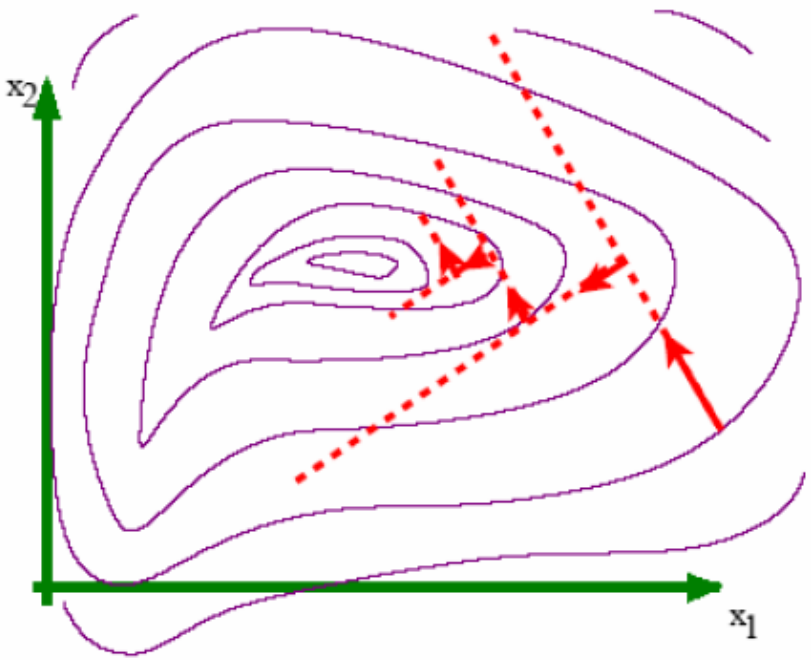
Arastirma yönü seçimi:

- Eslenik gradyent metodu, Steepest descent metodundaki zikzakli yakinsama sorununu arastirma yönlerini farkli sekilde hesaplayarak (eski arastirma yönlerini dikkate alarak) ortadan kaldirir.

- Eslenik gradyent metodu ile arastirma yönü (search direction) hesabi:

$$d^{(k)} = -\nabla f(x^{(k)}) + \beta^{(k)} d^{(k-1)}$$

$$\beta^{(k)} = \left(\frac{\left|\nabla f(x^{(k)})\right|}{\left|\nabla f(x^{(k-1)})\right|} \right)^2$$



ESLENIK GRADYENT METODU

<u>Arastirma yönü seçimi</u> (devam):

- Eslenik gradyent metodunda arastirma yönü, Steepst descent yönünün biraz modifiye edilmesiyle elde edilir.

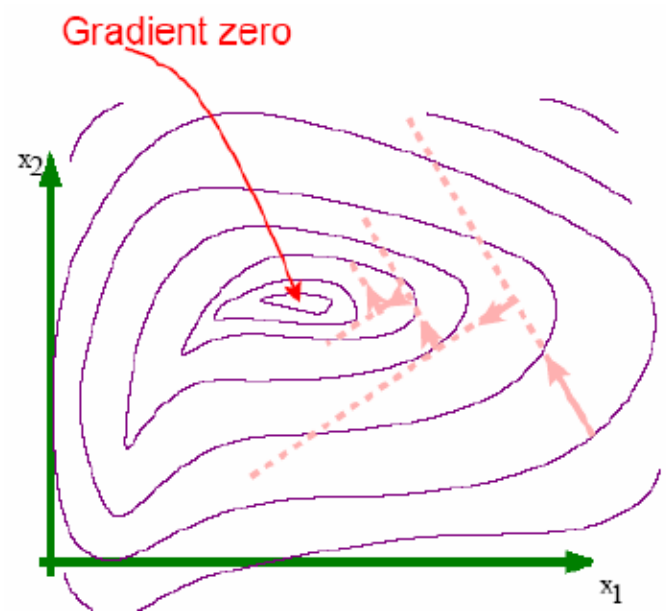
- Modifiye miktari gradyent degerlerindeki degisimle dogru orantilidir. Optimuma dogru yaklastikça gradyent degerleri sifira yaklasir ve modifiye miktari azalir. Yani optimuma dogru yaklastikça, arastirma yönleri Steepest descent yönlerine benzemeye baslar.

Steepest descent

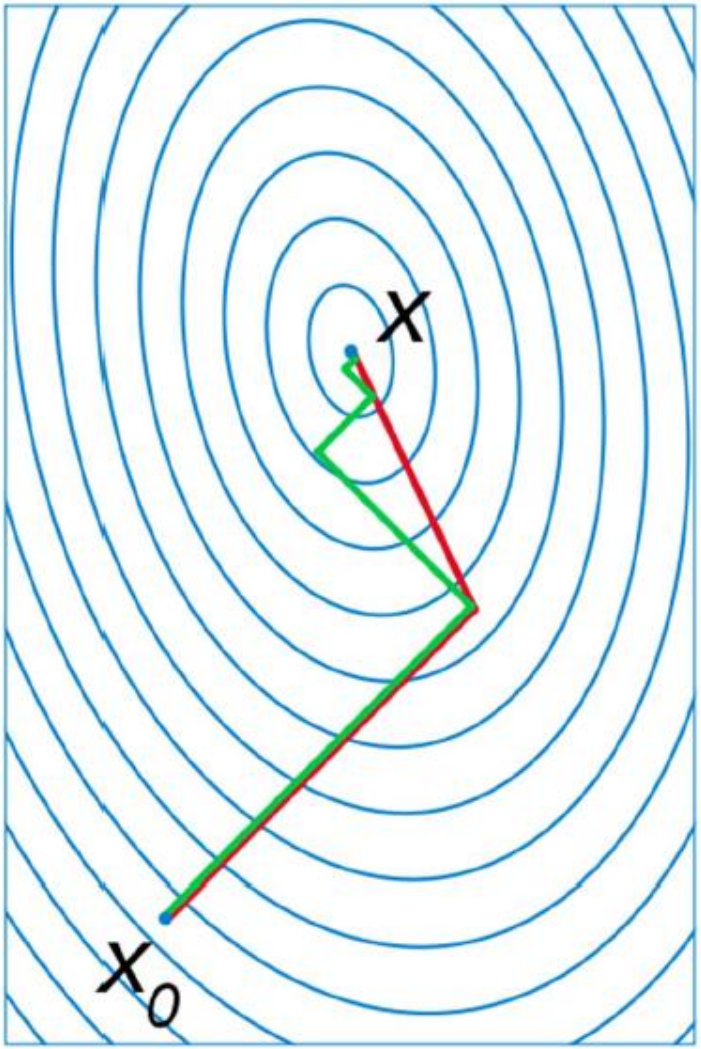
$$d^{(k)} = -\nabla f(x^{(k)}) + \beta^{(k)} d^{(k-1)}$$

$$\beta^{(k)} = \left(\frac{\left|\nabla f(x^{(k)})\right|}{\left|\nabla f(x^{(k-1)})\right|} \right)^2$$

Correction



EŞLENİK GRADYAN METODU (Conjugate Gradient): Dik iniş yöntemine benzer ama **daha verimli** bir algoritmadır. Dik iniş yönteminde iki adım arasındaki arama yönü birbirine **diktir**. Yani dik iniş yöntemindeki adımlar birbirine dik yönlerde ilerlemektedir. Eşlenik gradyan yönteminde ise iki dik yönün bileşkesi yönünde hareket edilir. Böylece **daha kısa iterasyonda** optimum değere ulaşılır.



Yeşil çizgiler : Dik iniş iterasyonları

Kırmızı çizgiler: Eşlenik gradyan iterasyonları

EŞLENİK GRADYAN Yöntemi Algoritması: Bu yöntemin algoritması şu şekildedir:

1. Başlangıç parametre değerleri $\mathbf{x}^{(0)}$ belirlenir ve iterasyon sayacı k=0 alınır. Yakınsama değeri $\varepsilon > 0$ olarak seçilir. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = -\nabla f(\mathbf{x}^{(0)})$ Değeri hesaplanır.
   Eğer $\|\mathbf{c}^{(0)}\| < \varepsilon$ ise iterasyon bitirilir değilse 4. adıma gidilir.
2. $\mathbf{x}^{(k)}$ noktasında f(x) fonksiyonunun gradyanı hesaplanır. $\mathbf{c}^{(k)} = \nabla f(\mathbf{x}^{(k)})$
3. $\|\mathbf{c}^{(k)}\|$ hesaplanır. Eğer $\|\mathbf{c}^{(k)}\| < \varepsilon$ ise iterasyon **bitirilir**, optimum çözüm $\mathbf{x}^* = \mathbf{x}^{(k)}$ dır, değilse ıterasyona **devam edilir**.
4. Yeni eşlenik yön hesaplanır. $\mathbf{d}^{(k)} = -\mathbf{c}^{(k)} + \beta_k \mathbf{d}^{(k-1)}; \qquad \beta_k = (\|\mathbf{c}^{(k)}\| / \|\mathbf{c}^{(k-1)}\|)^2$
5. $f(\mathbf{x}^{(k)} + \alpha \mathbf{d}^{(k)})$ fonksiyonunu minimum yapan α_k değeri bulunur. Bunun için gerek ve yeter şartlar yazılabilir veya tek boyutlu arama algoritmaları kullanılabilir.
6. $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}$ ile değerler güncellenir, k değeri artırılır ($k = k + 1$) ve 2. adıma gidilir.

Örnek: Aşağıdaki fonksiyonun minimum noktasını eşlenik gradyan yöntemi ile hesaplayınız.

$$\min \; f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3$$

1. $\mathbf{x}^{(0)} = (2, 4, 10)$. ε=0.0001

   $\mathbf{c}^{(0)} = (12, 40, 48); \quad \|\mathbf{c}^{(0)}\| = 63.6, \quad f(\mathbf{x}^{(0)}) = 332.0$

   $$\mathbf{x}^{(1)} = (0.0956, -2.348, 2.381)$$

2. $\mathbf{c}^{(1)} = (-4.5, -4.438, 4.828), \quad f(\mathbf{x}^{(1)}) = 10.75$
3. $\|\mathbf{c}^{(1)}\| = 7.952 > \varepsilon$ olduğundan iterasyona devam edilir.
4. $\beta_1 = [\|\mathbf{c}^{(1)}\| / \mathbf{c}^{(0)}]^2 = (7.952/63.3)^2 = 0.015633$

   $$\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1 \mathbf{d}^{(0)} = \begin{bmatrix} 4.500 \\ 4.438 \\ -4.828 \end{bmatrix} + (0.015633) \begin{bmatrix} -12 \\ -40 \\ -48 \end{bmatrix} = \begin{bmatrix} 4.31241 \\ 3.81268 \\ -5.57838 \end{bmatrix}$$

5. $\mathbf{d}^{(1)}$ yönündeki adım büyüklüğü $\alpha = 0.3156$.
6. yeni noktalar:

   $$\mathbf{x}^{(2)} = \begin{bmatrix} 0.0956 \\ -2.348 \\ 2.381 \end{bmatrix} + \alpha \begin{bmatrix} 4.31241 \\ 3.81268 \\ -5.57838 \end{bmatrix} = \begin{bmatrix} 1.4566 \\ -1.1447 \\ 0.6205 \end{bmatrix}$$

$\mathbf{c}^{(2)} = (0.6238, -0.4246, 0.1926)$.

$\|\mathbf{c}^2\| = 0.7788 > \varepsilon$ olduğundan iterasyona devam edilmelidir.

4 iterasyon sonunda optimum değerler $-6.4550\text{E}{-10}, -5.8410\text{E}{-10}, 1.3150\text{E}{-10}$.
ve fonksiyon değeri $6.8520\text{E}{-20}$. olarak bulunur.

Örnek : Aşağıda verilen üç değişkenli fonksiyonu minimum yapacak değişken değerlerini bulmak için yukarıda temel adımları verilen eşlenik gradyent algoritmasını iki iterasyon için gerçekleştiriniz (Arora, 1989).

$f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3$

Adım 1. $x^{(0)} = (2, 4, 10)$ (Başlangıç çözümü)

Adım 2. $g^{(0)} = (12, 40, 48)$; $\| g^{(0)} \| = 63.6$, $f(x^{(0)}) = 332.0$

1. iterasyon sonunda (Bu iterasyon en-dik iniş algoritması ile aynıdır) çözüm olarak

$x^{(1)} = (0.0956, -2.348, 2.381)$

değeri bulunur (İkinci iterasyon algoritmanın ikinci adımından başlar).

$g^{(1)} = (-4.5, -4.438, 4.828)$, $f(x^{(1)}) = 10.75$

$\|g^{(1)}\| = 7.952 > \varepsilon$ olduğundan araştırmaya devam et.

Adım 3. $\beta^{(1)} = [\|g^{(1)}\| / \|g^{(0)}\|]^2$

$= (7.952/ 63.6)^2 = 0.015633$

$d^{(1)} = g^{(1)} + \beta^{(1)} d^{(0)}$

$$= \begin{bmatrix} 4.500 \\ 4.438 \\ -4.828 \end{bmatrix} + (0.015633) \begin{bmatrix} -12 \\ -40 \\ -48 \end{bmatrix} = \begin{bmatrix} 4.31241 \\ 3.81268 \\ -5.57838 \end{bmatrix}$$

dolayısıyla yeni çözüm aşağıdaki gibi ifade edilir.

$$x^{(2)} = \begin{bmatrix} 0.0956 \\ -2.348 \\ 2.381 \end{bmatrix} + \alpha \begin{bmatrix} 4.31241 \\ 3.81268 \\ -5.57838 \end{bmatrix}$$

Adım 4. $f(x^{(1)} + \alpha d^{(1)})$ ifadesini minimize edecek adım büyüklüğü değeri

$\alpha = 0.3156$ olarak hesaplanır ve yeni çözüm

$x^{(2)} = (1.4566, -1.1447, 0.6205)$

olarak bulunur. Bu noktada gradyent değeri

$g^{(2)} = (0.6238, -0.4246, 0.1926)$ dir.

Adım 5. $\|g^{(2)}\| = 0.7788 > \varepsilon$ olduğundan ikinci adıma git ve yeni iterasyona başla.

ESLENIK GRADYENT METODU

<u>Diger eslenik gradyent metotlar</u>:
Bu metotlar, sadece ß katsayisinin hesaplanmasi konusunda farklilik gösteririler.

Fletcher – Reeves Metodu

$$\beta_k = \frac{(\nabla f_{k+1})^T (\nabla f_{k+1})}{(\nabla f_k)^T (\nabla f_k)}$$

Polak - Robiee Metodu

$$\beta_k = \frac{(\nabla f_{k+1})^T (\nabla f_{k+1} - \nabla f_k)}{(\nabla f_k)^T (\nabla f_k)}$$

ESLENIK GRADYENT METODU

Degerlendirme:

- Arastirma yönü hesabi basittir, sadece çok basit olan Steepest descent yönünden biraz karisiktir.
- Arastirma yönleri eslenik gradyenttir. Bu da yakinsamayi hizlandiran bir özelliktir.
- Arastirma yönü hesabinda türev bilgileri (gradyent degerleri) kullanildigindan hizli yakinsar.
- Hesaplamalarda az hafiza kullanir.

NEWTON METODU

En-dik iniş metodunda, araştırma yönünü belirlemek amacıyla maliyet fonksiyonun sadece birinci dereceden türev bilgisi kullanılmaktadır. Eğer ikinci dereceden türev bilgisi var ise maliyet yüzeyini bu tür bilgiyi kullanarak daha doğru ve hassas olarak temsil etmek suretiyle, araştırma yönü daha doğru olarak belirlenebilir. İkinci dereceden bilginin kullanılmasıyla daha iyi bir yakınsama oranı sağlanabilir. Newton metodunun temel fikri, mevcut nokta etrafında fonksiyonun ikinci dereceden Taylor seri açılımını kullanmaktır. Bu fikir, yeni çözüme ulaşmak amacıyla eski çözümde yapılması gereken değişimi tanımlamak için karesel bir ifade üretir. Newton metodunda birinci ve ikinci dereceden türevlerin var olduğu kabul edilir. İlk önce gradyent (g) ve Hessian (H) hesaplanır ve bunlar Denklem (1.7)'de kullanılarak yeni çözüm elde edilir.

$$x^{(t+1)} = x^{(t)} + \Delta x^{(t)} \text{ ve } \Delta x^{(t-1)} = -\alpha[H(x^{(t)})]^{-1} g(x^{(t)})$$

Burada H, Hessian matrisi temsil etmektedir ve bu matris tekil değildir.

Klasik Newton metodları, gradyent tabanlı bilgi aracılığıyla belirlenen doğrultuda adım büyüklüğünü 1 alır. Bundan dolayı klasik Newton metodu her problem için yakınsamayı garanti edemez. Bu dezavantajı ortadan kaldırmak için Değiştirilmiş Newton metodu önerilmiştir.

Aşağıda Değiştirilmiş Newton araştırma algoritmasının temel adımları verilmektedir.

Adım 1. Bir başlangıç çözümü ($x^{(0)}$) al ve İterasyon sayacını sıfırla ($t=0$) ve durdurma kriteri için bir tolerans (ε) seç.

Adım 2. $x^{(t)}$'de gradyent $g^{(t)}$ ve $\|g^{(t)}\|$'yi hesapla.

Şayet, ($\|g^{(t)}\|$ tolerans (ε) dan küçükse araştırmayı durdur.

Yoksa, devam et.

Adım 3. Hessian matrisi $H(x^{(t)})$'yi hesapla.

Adım 4. $d^{(t)} = -H^{-1}g^{(t)}$ ifadesinden faydalanarak araştırmanın yönünü belirle.

Adım 5. $x^{(t+1)} = x^{(t)} + \alpha^{(t)} d^{(t)}$ ifadesinden faydalanarak yeni çözümü hesapla. (Burada $\alpha^{(t)}$, $f(x^{(t)} + \alpha^{(t)} d^{(t)})$ ifadesini minimum yapmak için hesaplanmış adım büyüklüğüdür).

Adım 6. $t = t+1$ olarak al ve ikinci adıma git.

NEWTON METODU

$$X = \begin{Bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{Bmatrix}$$ veya $X = [x_1, x_2, \ldots, x_n]^T$ olmak üzere ;

Taylor Serisi açilimi:

$$f(X) = f(X_0) + (X - X_0)\nabla f(X_0) + \frac{1}{2}(X - X_0)^T H(X_0)(X - X_0) + \ldots$$ H: Hessian Matrisi

Optimum: $$X_{k+1} = X_k - H^{-1}(X_k)\nabla f(X_k)$$

Newton Metodu özelligi:

- Çözüme yakin noktadan baslanirsa hizli yakinsar.
- Uzak noktalardan baslanirsa yakinsamasi için modifiye edilmesi lazim.
- Hessian matrisin tersini hesaplamak maliyetli bir istir.

Örnek : Aşağıda verilen iki değişkenli fonksiyonu minimum yapacak değişken değerlerini yukarda temel adımları verilen Değiştirilmiş Newton algoritmasını kullanarak bulunuz (Arora, 1989).

$f(\mathbf{x}) = 3x_1^2 + 2x_1x_2 + 2x_2^2 + 7$

$\varepsilon = 0.0001$

Adım 1. $x^{(0)} = (5, 10)$ (Başlangıç çözümü)

Adım 2. Bu noktada gradyent değeri

$g^{(0)} = (6x_1+2x_2, 2x_1+4x_2) = (50, 50)$ olarak hesaplanır ve

$\| g^{(0)} \| = \sqrt{(50^2+50^2)} = 50\sqrt{2} > \varepsilon$

Adım 3. (5, 10) noktası için Hessian matrisi

$$H^{(0)} = \begin{bmatrix} 6 & 2 \\ 2 & 4 \end{bmatrix}$$

olarak bulunur. Hessian matris pozitif tanımlı olduğu için bulunacak araştırma yönü, maliyet fonksiyonun değerini azaltacak şekildedir.

Adım 4. Araştırma yönü ise

$$d^{(0)} = -H^{-1} g^{(0)} = \frac{-1}{20}\begin{bmatrix} 4 & -2 \\ -2 & 6 \end{bmatrix}\begin{bmatrix} 50 \\ 50 \end{bmatrix} = \begin{bmatrix} -5 \\ -10 \end{bmatrix}$$

olarak hesaplanır.

Adım 5. α adım büyüklüğü değeri, $f(x^{(0)}+\alpha d^{(0)})$ fonksiyonunu minimum yapacak şekilde hesaplandığında

$$x^{(1)} = x^{(0)}+\alpha d^{(0)}$$

$$= \begin{bmatrix} 5 \\ 10 \end{bmatrix} + \alpha \begin{bmatrix} -5 \\ -10 \end{bmatrix} = \begin{bmatrix} 5-5\alpha \\ 10-10\alpha \end{bmatrix}$$

$$\frac{df}{d\alpha} = 0 \quad \text{veya} \quad \nabla f(x^{(1)}).d^{(0)} = 0$$

elde edilir. Adım 2 deki değerler kullanılarak $\nabla f(x^{(1)})$ için

$$\nabla f(x^{(1)}) = \begin{bmatrix} 6(5-5\alpha)+2(10-10\alpha) \\ 2(5-5\alpha)+4(10-10\alpha) \end{bmatrix} = \begin{bmatrix} 50-50\alpha \\ 50-50\alpha \end{bmatrix}$$

bulunur. Bundan dolayı;

$$\nabla f(x^{(1)}).d^{(0)} = (50-50\alpha, 50-50\alpha)\begin{bmatrix} -5 \\ -10 \end{bmatrix} = 0$$

veya

$$-5(50-50\alpha) - 10(50-50\alpha) = 0$$

bu denklemi çözerek $\alpha = 1$ elde edilir.

Netice olarak,

$$x^{(1)} = \begin{bmatrix} 5-5\alpha \\ 10-10\alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$g^{(1)} = \begin{bmatrix} 50-50\alpha \\ 50-50\alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

bulunur.

Adım 6. $\|g^{(1)}\| < \varepsilon$ olduğu için Newton metodu çözümü sadece bir iterasyonda bulmuştur.

Örnek

- Use Newton's method to minimize

$$f(\boldsymbol{x}) = 0.5x_1^2 + 2.5x_2^2$$

- Gradient and Hessian are given by

$$\nabla f(\boldsymbol{x}) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix} \quad \text{and} \quad \boldsymbol{H}_f(\boldsymbol{x}) = \begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix}$$

- Taking $\boldsymbol{x}_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix}$, we have $\nabla f(\boldsymbol{x}_0) = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$
- Linear system for Newton step is $\begin{bmatrix} 1 & 0 \\ 0 & 5 \end{bmatrix} \boldsymbol{s}_0 = \begin{bmatrix} -5 \\ -5 \end{bmatrix}$, so $\boldsymbol{x}_1 = \boldsymbol{x}_0 + \boldsymbol{s}_0 = \begin{bmatrix} 5 \\ 1 \end{bmatrix} + \begin{bmatrix} -5 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, which is exact solution for this problem, as expected for quadratic function

QUASI-NEWTON METODU

Newton metodu ile optimum nokta:

$$\boxed{X_{k+1} = X_k - H^{-1}(X_k)\nabla f(X_k)}$$

Quasi-Newton metodu ile optimum nokta:

$$X_{k+1} = X_k - B_k \nabla f(X_k) \qquad B_k \cong H^{-1}(X_k)$$

- Hessian matrisin tersinin hesabi ya pratik degildir veya çok maliyetlidir.
- Quasi-Newton metotlari optimum nokta hesabinda Hessian tersi yerine, bunun *yaklasikligini* kullanirlar.
- Quasi-Newton metotlari birbirinden yaklasikliklarin *farkli* olmasi ile ayrilirlar.
- Quasi-Newton metotlari non lineer kisitsiz problemlerin çözümünde en etkili metotlar olarak kabul edilirler.

QUASI-NEWTON METODU

Quasi-Newton metodun varyasyonlari baslica iki tanedir :

- Davidon – Fletcher – Powel (DFP)
- Broyden – Fletcher – Goldfarb – Shanno Formülü (BFGS)

$$B = |H|^{-1} \qquad q_k = \nabla f_k \qquad p_k = x_k - x_{k-1}$$

- **DFP** ile Hessian tersi yaklasimi:

$$B_{k+1} = B_k + \frac{p_k p_k^T}{p_k^T q_k} - \frac{B_k q_k q_k^T B_k}{q_k^T B_k q_k}$$

- **BFGS** ile Hessian tersi yaklasimi:

$$\Rightarrow B_{k+1} = B_k + \left(\frac{1 + q_k^T B_k q_k}{q_k^T p_k}\right)\frac{p_k p_k^T}{p_k^T q_k} - \frac{p_k q_k^T B_k + B_k q_k p_k^T}{q_k^T p_k}$$

Yarı-Newton metotlar Newton metoduna dayalı metotlardır. Aralarındaki fark Hessian matrisin elde edilişi ile ilgilidir. Hessian matrisinin hesaplanması ve tersinin alınması uzun işlem gerektirdiği için, Yarı-Newton metotlar da bu işlemlerin tam olarak yapılması yerine, sadece birinci dereceden türevler kullanılarak ikinci dereceden türevlerin yaklaşıklıkları elde edilir. Yani, Hessian matrisin yaklaşık olarak hesabı birinci derece türevlerden yapılır. Temel fikir, çözüm ve grandyent vektörü ile ilgili değişimleri kullanarak mevcut kullanılan yaklaşıklığı yenilemektir. Böylece H^{-1} matrisi, r iterasyon sonra H^{-1}'in yaklaşıklığı olan H^{r} ile temsil edilir. Bu sınıfın yaygın olarak kullanılan algoritmalarından biri Davidon-Fletcher-Powell (DFP) algoritmasıdır (Fletcher ve Powell, 1963).

QUASI-NEWTON METODU

Degerlendirme:

- BFGS formülü DFP den biraz daha karmasiktir.
- DFP formülü BFGS ’nin özel halidir, BFGS’den türetilebilir.
- Bir çok optimizasyon probleminin çözümü, BFGS performansinin DFP’den daha iyi oldugunu göstermistir. BFGS bu yüzden daha çok tercih edilir.
- Her iki metotta, hem kisitsiz problemlerin çözümünde hem de kisitli problemlerin (kisitli problemleri esdeger kisitsiz problemlere dönüstürerek) çözümünde yaygin olarak kullanilmaktadir.

Kaynaklar

- Hasan KURTARAN, “Mühendislik tasarımların optimizasyonu”, 2005
- Edwin K.P. Chong Stanislaw H.Zak,“An Introduction to Optimization”, 2001, John Wiley&Sons
- Jorge Nocedal, Stephan J. Wright, “Numerical Optimization”, 1999, Springer-Verlag
- R. Fletcher, “Practical Methods of Optimization”, 1987, John Wiley&Sons
- David G.Luenberger, “Introduction to Linear and Nonlinear Programming”, 1987, Addison-Wesley
- S.G. Nash, A. Sofer, “Linear and Nonlinear programming”, 1996, McGraw Hill
- Dimitri P. Bertsekas, “Constrained Optimization and Lagrange Multiplier Methods”, 1982, Academic Press
- Abbas Azimli, “Matematiksel Optimizasyon”, 2011, Papatya Yayınevi
- Derviş Karaboğa: “Yapay Zeka Optimizasyon Algoritmaları”, 2006, Nobel Yayınevi