

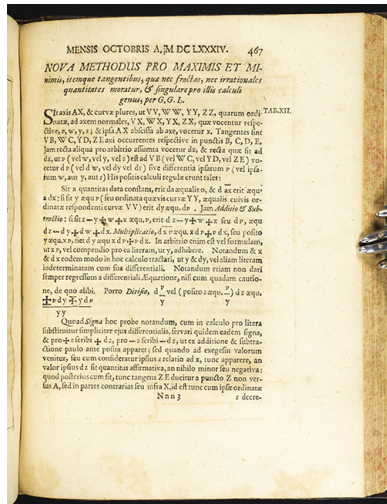
Optimizasyon ve Yapay Öğrenme

İlker Birbil

Sabancı Üniversitesi
Mühendislik ve Doğa Bilimleri Fakültesi
Endüstri Mühendisliği Programı

Veri Bilim - Yapay Öğrenme Yaz Okulu
Matematiksel Temeller ve Vaka Çalışmaları
Eylül, 2017 - İstanbul

- Önceden belirlenmiş kısıtlar altında bir fonksiyonun en büyük ya da en küçük değerini bulmak.
- Bu konuşmada problemin *düzgünce* tanımlı olduğunu varsayacağız. Yani bir minimum (ya da maksimum) noktasının olduğu problemlerle ilgileneceğiz.



Şekil: Leibniz'in 1684 makalesi

TANIMLAR

Matematiksel Programlama Modeli

Genel bir **optimizasyon (eniyleme) problemi** şu şekilde gösterilebilir:

$$\begin{array}{ll} \text{enküçüle} & f(x) \\ \text{öyle ki} & c_j(x) = 0, \quad j \in \mathcal{E}, \\ & c_j(x) \geq 0, \quad j \in \mathcal{I}. \end{array} \quad (1)$$

Burada $x \in \mathbb{R}^n$ vektörü ile **karar değişkenleri (bilinmeyenler)**, $f : \mathbb{R}^n \mapsto \mathbb{R}$ ile **amaç fonksiyonu**, $c_j : \mathbb{R}^n \mapsto \mathbb{R}$, $j \in \mathcal{E} \cup \mathcal{I}$ ile de **kısıtlar** gösterilmiştir.

Not

Bir enbüyükleme problemi kolayca enküçükleme problemine dönüştürülebilir:

$$\max f(x) = -\min\{-f(x)\}.$$

İki kısıtlı bir matematiksel model şu şekilde¹ verilmiş olsun:

$$\begin{array}{ll} \text{enküçüle} & f(x) \\ \text{öyle ki} & x_1^2 - x_2 \leq 0, \\ & x_1 + x_2 \leq 2. \end{array}$$

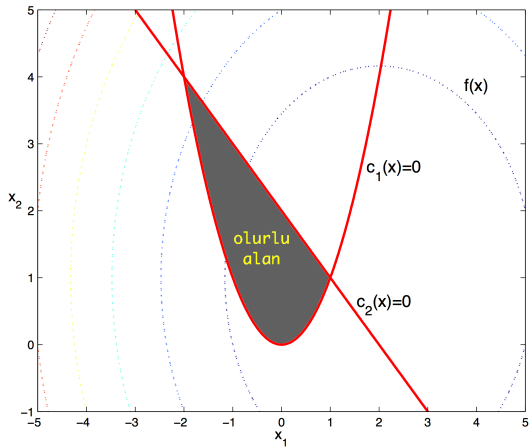
Burada

$$f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad c(x) = \begin{bmatrix} c_1(x) \\ c_2(x) \end{bmatrix} = \begin{bmatrix} -x_1^2 + x_2 \\ -x_1 - x_2 + 2 \end{bmatrix}, \quad \mathcal{I} = \{1, 2\}, \quad \mathcal{E} = \emptyset.$$

¹ Nocedal, J., Wright, S. J., Numerical Optimization, 2. Basım, New York:Springer, 2006.

Örnek (devam)



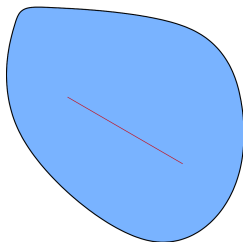
Dışbükey Küme

Tanım

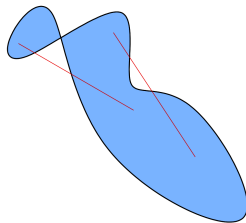
$S \in \mathbb{R}^n$ kümesinin dışbükey (convex) olması için S kümesinden herhangi iki noktayı birleştiren doğru parçasının tamamının S kümesinde olması gerekir. Matematiksel olarak gösterirsek, her $x, y \in S$ çifti ve tüm $\alpha \in [0, 1]$ değerleri için

$$\alpha x + (1 - \alpha)y \in S$$

olmalıdır.



Dışbükey küme



Dışbükey olmayan küme

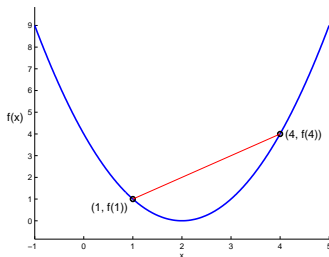
Dışbükey Fonksiyon

Tanım

$f(\cdot)$ ile gösterilen bir fonksiyonun **dışbükey** olması için tanım kümesinin dışbükey olması ve bu kümeden seçilen herhangi x, y çifti için $f(\cdot)$ grafiğinin $(x, f(x))$ ve $(y, f(y))$ noktalarını birleştiren doğru parçasının altında kalması gerekir. Matematiksel olarak ifade edersek, her x, y ve tüm $\alpha \in [0, 1]$ değerleri için

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

eşitsizliği sağlanmalıdır.



Dışbükeylik Hakkında

- ▶ Bir $f(\cdot)$ fonksiyonunun içbükey (concave) olması, $-f(\cdot)$ fonksiyonunun dışbükey olduğunu gösterir.
- ▶ Dışbükey bir fonksiyon ile yazılan kısıtsız problemlerde lokal minimum noktası, global minimum noktası olur.
- ▶ Eğer bir kısıt \leq şeklinde bir eşitsizlikse ve dışbükey fonksiyonlar ile oluşturulmuşsa, ortaya çıkan olurlu alan da dışbükey bir kümedir.
- ▶ Kabaca söylemek gerekirse, pek çok durumda dışbükey fonksiyonlar ile çalışıldığında minimum noktasını belirlemek için kullanılan gerek şartlar aynı zamanda yeter şartlar olurlar.
- ▶ Dışbükey fonksiyonlar, lokal olarak daha karmaşık ve dışbükey olmayan fonksiyonların yaklaşık gösteriminde kullanılırlar.

Dışbükeylik Hakkında (devam)

En başta (1) ile gösterdiğimiz matematiksel programlama modelinde

- ▶ amaç fonksiyonu $f(\cdot)$ dışbükeyse,
- ▶ eşitlik $c_j(\cdot), j \in \mathcal{E}$ kısıtları doğrusalsa,
- ▶ ve eşitsizlik fonksiyonları $c_j(\cdot), j \in \mathcal{I}$ içbükeyse,

elde edilen model **dışbükey optimizasyon** modeli olur.

enküçükle

$f(x)$

öyle ki

$$c_j(x) = 0, \quad j \in \mathcal{E},$$

$$c_j(x) \geq 0, \quad j \in \mathcal{I}.$$

Kısıtsız Optimizasyon

Kısıtsız optimizasyon problemlerinde eşitlikler ve eşitsizlikler yoktur. Yani (1) modelinde $\mathcal{I} \equiv \mathcal{E} \equiv \emptyset$ olarak alınır. Bu durumda model kısaca

$$\min_{x \in \mathbb{R}^n} f(x)$$

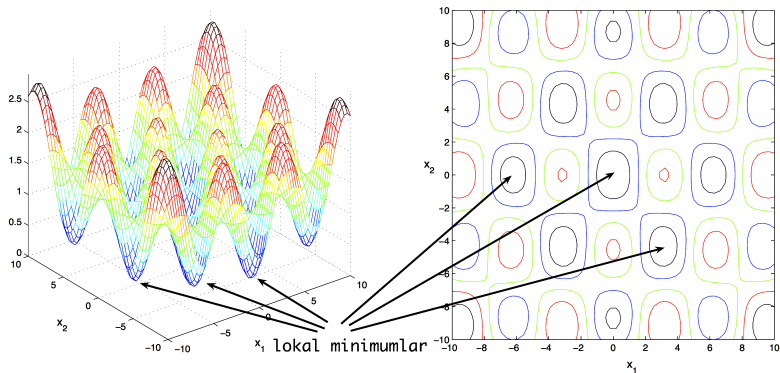
olarak yazılabilir. Birinci türevi elde etmek için n boyutun her birine göre kısmi türev alınarak **gradyant (gradient) vektörü** elde edilir:

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \frac{\partial f(x)}{\partial x_2} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix}.$$

İkinci türevleri ise **Hesyan (Hessian) matrisini** verecektir:

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix}.$$

Lokal ve Global Optimizasyon



ÖRNEK YAPAY ÖĞRENME PROBLEMLERİ

Elimizde N tane veri noktası olduğunu düşünelim; $x^{(k)} \in \mathbb{R}^n$, $k = 1, \dots, N$. Her veri noktası için iki etiketten biri verilmiş; $y^{(k)} \in \{-1, 1\}$, $k = 1, \dots, N$.

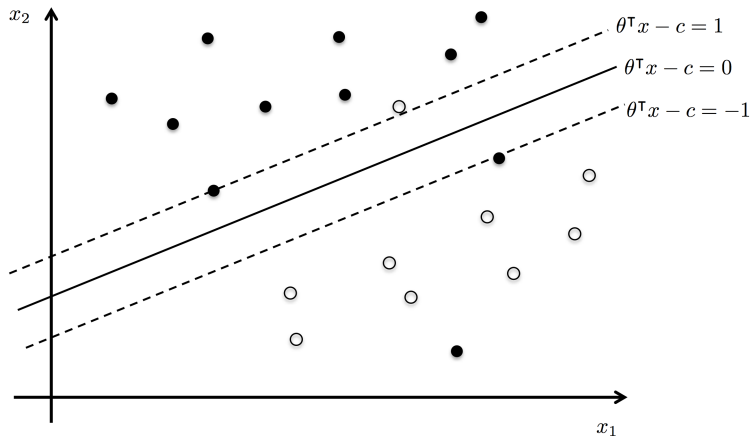
Amaç bu veriyi iki kümeye ayıracak şekilde çok boyutlu bir düzlem bulmak. Her $k = 1, \dots, N$ için şu ifadeleri kullanalım:

$$\left. \begin{array}{ll} \theta^\top x^{(k)} - c \leq -1 & \implies y^{(k)} = -1 \\ \theta^\top x^{(k)} - c \geq 1 & \implies y^{(k)} = 1 \end{array} \right\} (\theta^\top x^{(k)} - c)y^{(k)} = 1.$$

Bu durumda yapmamız gereken, verilen bir c değerine göre θ vektörünü hesaplamaktır.

Kümeleme - İkili Sınıflandırma

$$\left. \begin{array}{l} \theta^T x^{(k)} - c \leq -1 \implies y^{(k)} = -1 \\ \theta^T x^{(k)} - c \geq 1 \implies y^{(k)} = 1 \end{array} \right\} (\theta^T x^{(k)} - c)y^{(k)} = 1.$$



Kümeleme - İkili Sınıflandırma (devam)

Optimizasyon modelimiz için önce amaç fonksiyonunu oluşturalım:

$$l_k(\theta) = \max\{0, 1 - (\theta^\top x^{(k)} - c)y^{(k)}\}.$$

Bu fonksiyona literatürde **menteşe kayıp fonksiyonu (hinge loss function)** da denmektedir. Modelimiz

$$\min J_\lambda(\theta) = \frac{1}{N} \sum_{k=1}^N l_k(\theta) + \frac{\lambda}{2} \|\theta\|^2.$$

haline gelir. Modelin en sonuna eklediğimiz terim **aşırı uyum (overfitting)** sorunundan kaçınmak için eklenmiştir. Burada λ değeri dışarıdan verilen bir parametredir.

Bu problem amaç fonksiyonundaki \max işleci yüzünden türevlenebilir değildir. Ancak bu model, kısıtlar yardımıyla dışbükey optimizasyon problemine dönüştürülebilir.

Kümeleme - İkili Sınıflandırma (devam)

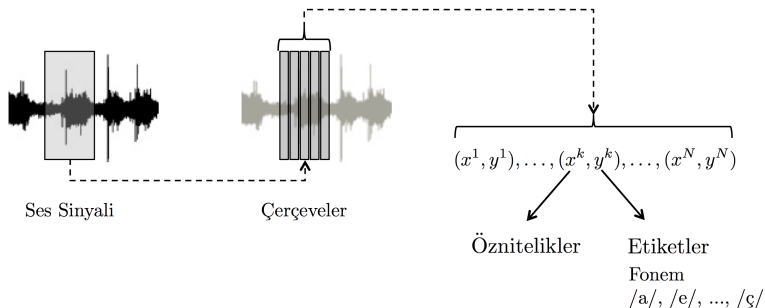
Dışbükey optimizasyon modeli için önce yardımcı değişkenler

$$z_k = \max\{0, 1 - (\theta^\top x^{(k)} - c)y^{(k)}\}, \quad k = 1, \dots, N$$

olarak tanımlanır. Ardından kısıtlı modelimiz şu şekilde yazılır:

$$\begin{aligned} \text{enküçüle} \quad & \frac{1}{N} \sum_{k=1}^N z_k + \frac{\lambda}{2} \|\theta\|^2 \\ \text{öyle ki} \quad & z_k \geq 1 - (\theta^\top x^{(k)} - c)y^{(k)}, \quad k = 1, \dots, N; \\ & z_k \geq 0, \quad k = 1, \dots, N. \end{aligned}$$

Literatürde bu ikili sınıflandırma yaklaşımına **destek vektör makinesi (support vector machine)** denmektedir.



- **Veri:** Milisaniyelik kayıtlar (çerçeveler); $(x^{(k)}, y^{(k)})$, $k = 1, \dots, N$. Burada $x^{(k)} \in \mathbb{R}^n$ öznitelikler, $y^{(k)} \in \mathbb{C}$ ses etiketleri.
- **Amaç:** Öznitelikleri bilinen yeni bir kaydı doğru şekilde etiketlemek.

- ▶ Her etikete bir ağırlık vektörü, $\theta^l \in \mathbb{R}^n$, $l \in \mathbb{C}$ verilir.
- ▶ Kolaylık olması için $|\mathbb{C}| \times n$ boyutlarında bir θ matrisi tanımlarız. Her k çerçevesine j etiketi atanması için hesaplanan olasılık

$$\mathbf{P} \left\{ y^{(k)} = j \mid x^{(k)}; \theta \right\} = \frac{\exp((\theta^j)^\top x^{(k)})}{\sum_{l \in \mathbb{C}} \exp((\theta^l)^\top x^{(k)})}$$

olarak verilir. Bu durumda ölçekli log-benzerlik fonksiyonu

$$\frac{1}{N} \sum_{k=1}^N \underbrace{\sum_{j \in \mathbb{C}} \mathbf{1}\{y^{(k)} = j\} \log \frac{\exp((\theta^j)^\top x^{(k)})}{\sum_{l \in \mathbb{C}} \exp((\theta^l)^\top x^{(k)})}}_{l_k(\theta)},$$

şeklinde yazılır. Buradaki $\mathbf{1}$ gösterge işleci, içerisindeki ifade doğru ise 1, aksi halde 0 çevirir.

Kümeleme - Ses İşleme (devam)

Şimdi **örneklem ortalaması yaklaşımı (sample average approximation)** fonksiyonunu yazabiliriz:

$$J(\theta) = \frac{1}{N} \sum_{k=1}^N l_k(\theta).$$

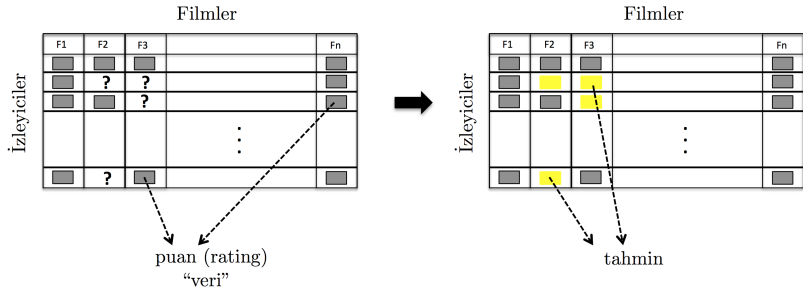
Bu durumda **maksimum benzerlik kestirimcisi (maximum likelihood estimator)** ise

$$\theta^* = \arg \max_{\theta \in \mathbb{R}^{|\mathcal{C}| \times n}} J(\theta)$$

olarak bulunur. Enbüyükleme probleminden, enküçüklemeye geçerken de basitçe fonksiyonu -1 ile çarpabiliriz. Kısacası, çözmemiz gereken model şu şekilde yazılabilir:

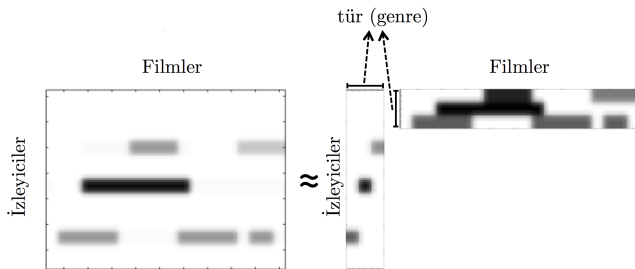
$$\min_{\theta \in \mathbb{R}^{|\mathcal{C}| \times n}} -J(\theta).$$

Tavsiye Sistemi



- ▶ **Veri:** İzleyicilerin farklı filmlere verdikleri puanları gösteren bir **seyrek (sparse)** matris (Y).
- ▶ **Amaç:** İzleyiciler ile filmleri belirli sayıda türe göre gruplamak ve ilgilerine göre izleyicilere film tavsiye etmek.

Tavsiye Sistemi - Matrisleri Çarpanlarına Ayırma



$$Y \approx X_1 \times X_2$$

Tavsiye Sistemi - Matrisleri Çarpanlarına Ayırma (devam)

	F1	F2	F3	F4
Ali	5	2	?	2
Berna	4	?	?	3
Cemal	1	1	?	4
Deniz	2	?	4	5
Esra	?	2	?	4

$$\underbrace{\begin{bmatrix} 5 & 2 & ? & 2 \\ 4 & ? & ? & 3 \\ 1 & 1 & ? & 4 \\ 2 & ? & 4 & 5 \\ ? & 2 & ? & 4 \end{bmatrix}}_Y \approx \underbrace{\begin{bmatrix} * & * \\ * & * \\ * & * \\ * & * \\ * & * \end{bmatrix}}_{X_1} \underbrace{\begin{bmatrix} * & * & * & * \\ * & * & * & * \end{bmatrix}}_{X_2}$$

Tavsiye Sistemi - Matrisleri Çarpanlarına Ayırma (devam)

$$\underbrace{\begin{bmatrix} 5 & 2 & ? & 2 \\ 4 & ? & ? & 3 \\ 1 & 1 & ? & 4 \\ 2 & ? & 4 & 5 \\ ? & 2 & ? & 4 \end{bmatrix}}_Y \approx \underbrace{\begin{bmatrix} * & * \\ * & * \\ * & * \\ * & * \\ * & * \end{bmatrix}}_{X_1} \underbrace{\begin{bmatrix} * & * & * & * \\ * & * & * & * \end{bmatrix}}_{X_2}$$

Model

$$(X_1^*, X_2^*) = \arg \min_{X_U, X_M} \|Y - X_U X_M\|_F^2.$$

$$\underbrace{\begin{bmatrix} 1.99 & 0.56 \\ 1.45 & 1.04 \\ 0.00 & 1.64 \\ 0.33 & 2.01 \\ 1.18 & 1.50 \end{bmatrix}}_{X_1^*} \underbrace{\begin{bmatrix} 2.32 & 0.85 & 1.04 & 0.32 \\ 0.61 & 0.62 & 1.82 & 2.42 \end{bmatrix}}_{X_2^*} = \underbrace{\begin{bmatrix} 4.95 & 2.05 & 3.08 & 2.00 \\ 3.99 & 1.88 & 3.39 & 2.99 \\ 1.01 & 1.02 & 2.97 & 3.97 \\ 2.00 & 1.53 & 3.99 & 4.98 \\ 3.65 & 1.93 & 3.94 & 4.00 \end{bmatrix}}_{\hat{Y}}$$

Tavsiye Sistemi - Matrisleri Çarpanlarına Ayırma (devam)

Bu yaklaşım ile çözeceğimiz optimizasyon modeli şu şekilde yazılır:

$$\min_{X_1, X_2} \|Y - X_1 X_2\|_F^2.$$

İlk bakışta karışık gözükse de, aslında amaç fonksiyonu her veri noktası için karesel sapmaların toplamına karşılık gelmektedir. Üç izleyici ve iki filmli basit bir örnek bu noktayı gösterecektir:

$$\min_{x_1, \dots, x_5} \left\| \underbrace{\begin{pmatrix} y_1 & y_2 \\ y_3 & y_4 \\ y_5 & y_6 \end{pmatrix}}_Y - \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{X_1} \underbrace{\begin{pmatrix} x_4 & x_5 \end{pmatrix}}_{X_2} \right\|_F^2 = \min_{x_1, \dots, x_5} (y_1 - x_1 x_4)^2 + \dots + (y_6 - x_3 x_5)^2$$

SIK KULLANILAN ÇÖZÜM YÖNTEMLERİ

Şu ana kadar konuştuğumuz kısıtsız yapay öğrenme modelleri, genel bir formda

$$\min_{x \in \mathbb{R}^n} f(x) = \min_{x \in \mathbb{R}^n} \sum_{k=1}^N f_k(x)$$

şeklinde yazılabilirler.

Bu formda bir optimizasyon modeli çözmeyi gerektiren diğer yapay öğrenme yaklaşımlarına birkaç örnek verebiliriz:

- ▶ Lojistik bağlantım (regression)
- ▶ Derin öğrenme (deep learning)
- ▶ Çok katmanlı yapay sinir ağları (multilayer artificial neural networks)

Problemlerin Ortak Yapısı (devam)

Amaç fonksiyonumuzu tekrar yazalım

$$f(x) = \sum_{k=1}^N f_k(x).$$

Bu fonksiyonun minimum noktasını bulmak için bir önceki dersteki çözüm yöntemlerini kullanabiliriz. Bunun için amaç fonksiyonunun türevine ihtiyacımız olacak:

$$\nabla f(x) = \sum_{k=1}^N \nabla f_k(x).$$

Yapay öğrenme problemlerinin önemli bir kısmında N değeri veri boyutuna bağlıdır. O nedenle N kolayca oldukça büyük bir sayı olur. Dolayısıyla her seferinde türev hesabı yapmanın hesaplama zamanı açısından maliyeti yüksektir.

Kısıtsız Optimizasyon Algoritmaları

- ▶ Bu algoritmalarda bir dizi adım hesaplanır: x_0, x_1, x_2, \dots
- ▶ Çoğu zaman başlangıç noktası x_0 kullanıcı tarafından belirlenir.
- ▶ Her adımda algoritma $f(\cdot)$, $\nabla f(\cdot)$ ya da $\nabla^2 f(\cdot)$ fonksiyonlarını kullanır.
- ▶ Pek çok algoritmada $\{f(x_i)\}_{i=0}^{\infty}$ dizisi monoton olarak azalır. Ancak monoton olmayan algoritmalar da mevcuttur.
- ▶ Kısıtsız optimizasyon için iki temel strateji vardır: **doğru arama (line search)** ve **güven bölgesi (trust region)**.

Doğru Arama Algoritmaları

Gradyant İniş

$$x_{k+1} = x_i - \alpha_i \nabla f(x_i)$$

Newton Algoritması

$$x_{k+1} = x_i - \alpha_i \nabla^2 f(x_i)^{-1} \nabla f(x_i)$$

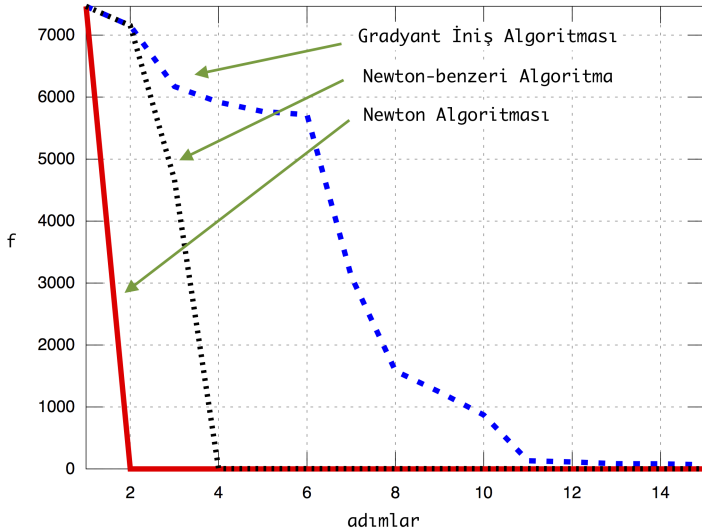
Newton-benzeri Algoritmalar

$$x_{k+1} = x_i - \alpha_i B_i^{-1} \nabla f(x_i)$$

"Hesaplamalı Tarifler I: Newton ve Benzeri Metodlar," Matematik Dünyası, 2016

Burada α_i ile gösterilen adım boyu (step-length), yapay öğrenme camiasında **öğrenme hızı** (learning rate) olarak bilinir.

Çözümeye Yakınsama Hızları



Rassal Gradyant Yöntemleri

Rassal yöntemler, her seferinde türevin tamamını hesaplamak yerine sadece bir kısmını rassal olarak seçip hesaplarlar.

Rassal olarak seçilen kısımları $\mathcal{K} \subseteq \{1, \dots, N\}$ kümesi olarak gösterirsek, gradyant iniş algoritmasının adımları şu şekilde yazılabilir:

$$x_{i+1} = x_i - \alpha_i \sum_{k \in \mathcal{K}} \nabla f_k(x)$$

Literatürde \mathcal{K} kümesindeki eleman sayısına göre farklı yöntemler denenmiştir:

- ▶ $|\mathcal{K}| = 1$, **rassal gradyant iniş (stochastic gradient descent)**
- ▶ $|\mathcal{K}| < N$, **mini-yığın rassal gradyant iniş (mini-batch gradient descent)**
- ▶ $|\mathcal{K}| = N$, **yığın gradyant iniş (batch gradient descent)**.

Dikkat edilirse son seçenekte rassalık yok ve bu şekilde koşturulan algoritma radyant iniş algoritmasının aynısı.

Rassal Gradyant Yöntemleri (devam)

Rassal gradyant yöntemlerini uygulamak için

$$x_{i+1} = x_i - \alpha_i \sum_{k \in \mathcal{K}} \nabla f_k(x)$$

döngüsünde adım boyu α_i değerini de belirlememiz gerek. Bu yöntemler adım boyunu arama algoritmaları ile hesaplamak yerine, adım boyunu azalan bir dizi olarak düşünürler.

Yakınsaklık analizleri adım boyu dizisinin şu şartları sağlaması gerektiğini göstermiştir:

$$\alpha_i \xrightarrow{i \uparrow \infty} 0 \text{ ve } \sum_{i=1}^{\infty} \alpha_i = \infty.$$

Uygulamada

$$\alpha_i = \frac{\epsilon}{\sqrt{i}}$$

dizisi için $\epsilon = 10^{-4}$ olarak alınabilir.



Hessian
Approximated
Multiple
Subsets
Iteration

MAKALE: <https://arxiv.org/abs/1509.01698>
KOD: <https://github.com/spartensor/hamsi-mf>



Cornell University
Library

[arXiv.org](https://arxiv.org) > [stat](#) > [arXiv:1509.01698](https://arxiv.org/abs/1509.01698)

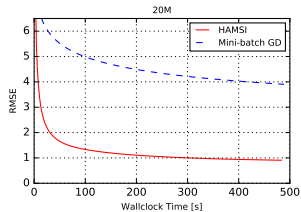
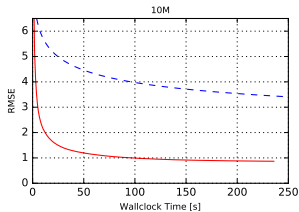
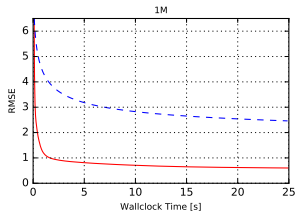
Statistics > Machine Learning

HAMSI: A Parallel Incremental Optimization Algorithm Using Quadratic Approximations for Solving Partially Separable Problems

MB-GD \iff HAMSİ

Dataset	Algorithm	schedule	Average Final RMSE Value				
			HOGWILD	COLOR	COLOR-B	STRATA	STRATA-B
1M – 6040 – 3883 ratings users movies (25 seconds)	mb-GD	det	3.1074	3.1061	3.0845	2.5315	2.4588
		stoc	3.1433	3.1470	3.1003	2.5325	2.4650
	HAMSİ	det	0.6901	0.6955	0.7102	0.6133	0.6022
		stoc	0.6900	0.7987	0.8017	0.6088	0.5994
10M – 71567 – 10681 ratings users movies (250 seconds)	mb-GD	det	4.3167	4.2676	4.2617	4.0029	3.4088
		stoc	4.3009	4.2863	4.2801	4.0035	3.4094
	HAMSİ	det	0.9279	1.0181	0.8941	0.8923	0.8643
		stoc	0.9207	1.1357	1.1229	0.8988	0.8652
20M – 138493 – 26744 ratings users movies (500 seconds)	mb-GD	det	4.8655	4.8051	4.8000	4.8093	3.8890
		stoc	4.8641	4.8279	4.8142	4.8091	3.8975
	HAMSİ	det	1.0170	1.1117	0.9521	1.0113	0.9042
		stoc	1.0112	1.2944	1.2220	1.0231	0.9035

MB-GD \iff HAMSİ



- ▶ Bertsekas, D. P., **Nonlinear Programming**, 2. basım, Athena Scientific, Belmont, Massachusetts, 2003.
- ▶ Bazaara, M. S., Sherali, H. D. ve Shetty, C. M., **Nonlinear Programming Theory and Algorithms**, 2. basım, Wiley, N.Y., 1993.
- ▶ Hiriart-Urruty, J.-B. ve Lemaréchal, **Convex Analysis and Minimization Algorithms I and II**, Springer-Verlag, Berlin, Heidelberg, 1993.
- ▶ Luenberger, D. G., **Introduction to Linear and Nonlinear Programming**, 2. basım, Addison-Wesley, Reading, MA, 1984.
- ▶ Boyd, S. ve Vandenberghe, I., **Convex Optimization**, Cambridge University Press, Cambridge, UK, 2004.
- ▶ Faigle, U., Kern, W. ve Still, G., **Algorithmic Principles of Mathematical Programming**, Dordrecht, Boston, Kluwer Academic Publishers, 2002.