Georgia Tech
College of Computing

CS4641 Spring 2025

Brief Intro to Optimization

Bo Dai
School of CSE, Georgia Tech
bodai@cc.gatech.edu

Office Hours

https://bo-dai.github.io/CS4641-spring2025/

Machine Learning for Apartment Hunting

- Suppose you are to move to Atlanta
- And you want to find the most reasonably priced apartment satisfying your needs:

Living area (ft^2)	# bedroom	Monthly rent ($)
230	1	900
506	2	1800
433	2	1500
190	1	800
…		
150	1	?
270	1.5	?

Linear Regression Model

- Assume y is a linear function of x (features) plus noise ϵ

$$y = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n + \epsilon$$

where ϵ is an error model as Gaussian $N(0, \sigma^2)$ Probability

- Let $\theta = (\theta_0, \theta_1, \dots, \theta_n)^\top$, and augment data by one dimension

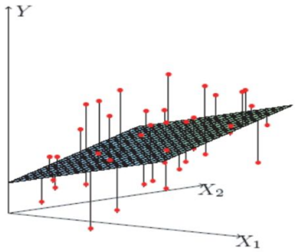
$$x \leftarrow (1, x)^\top$$

Linear algebra

Then $y = \theta^\top x + \epsilon$

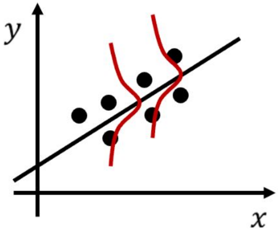
Linear algebra

Gaussian Likelihood



- Assume y is a linear in x plus noise ϵ

$$y = \theta^{\top} x + \epsilon$$



- Assume ϵ follows a Gaussian $N(0, \sigma)$

$$p(y^i \mid x^i; \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left(y^i - \theta^{\top} x^i\right)^2}{2\sigma^2}\right)$$

- By independence assumption, likelihood is

$$L(\theta) = \prod_i^m p(y^i \mid x^i; \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m \left(y^i - \theta^{\top} x^i\right)^2}{2\sigma^2}\right)$$

Probability

MLE

$$L(\theta) = \prod_{i}^{m} p(y^i | x^i; \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m \left(y^i - \theta^\top x^i\right)^2}{2\sigma^2}\right)$$

MLE

$$L(\theta) = \prod_{i}^{m} p(y^i|x^i;\theta) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^m \exp\left(-\frac{\sum_i^m (y^i - \theta^\top x^i)^2}{2\sigma^2}\right)$$

$$\max_\theta \ \log L(\theta) = -\frac{1}{2\sigma^2}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2 - m\log(\sqrt{2\pi}\sigma)$$

Optimization

Least Mean Square for Linear Regression

Optimization Problem

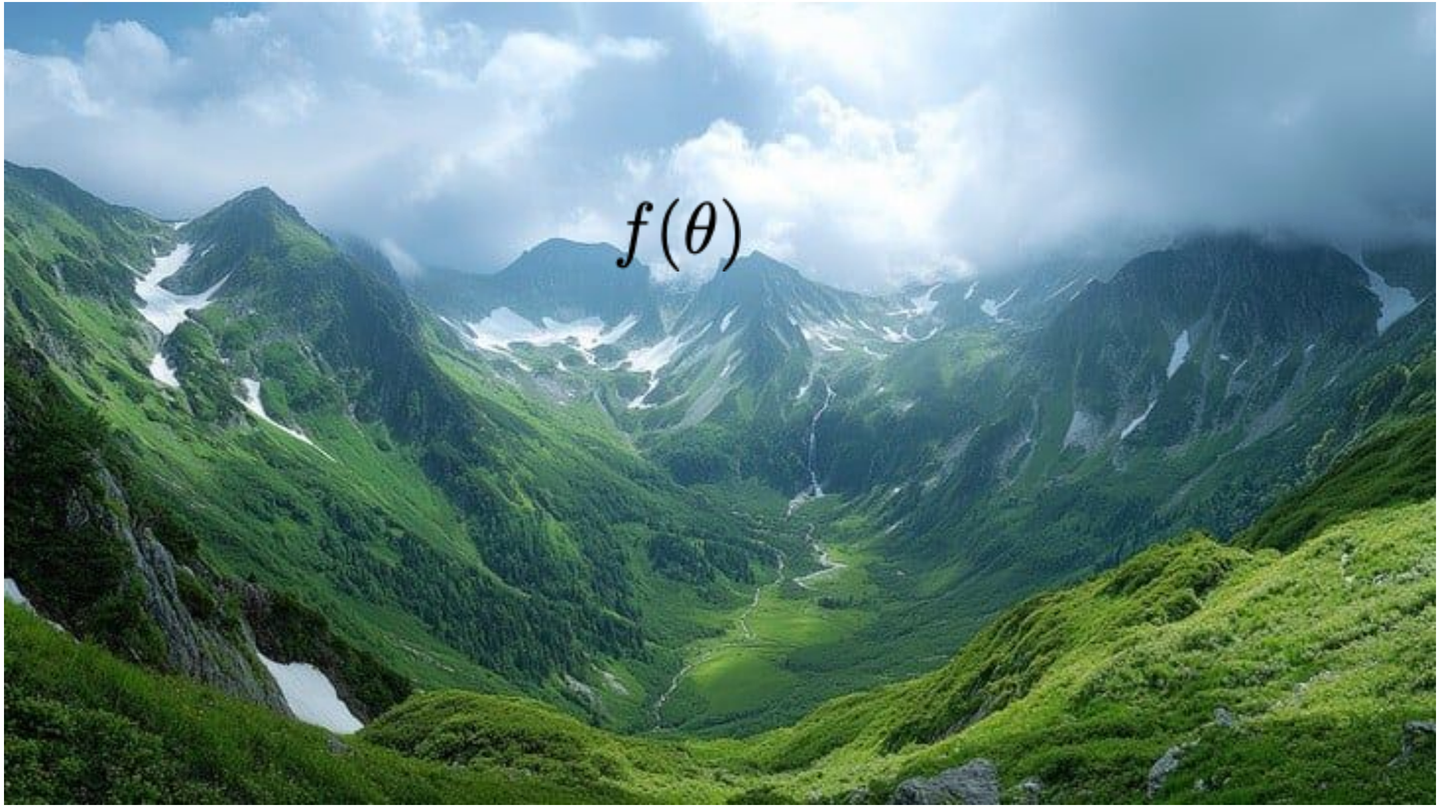
$$\text{minimize}\ f(\theta)$$

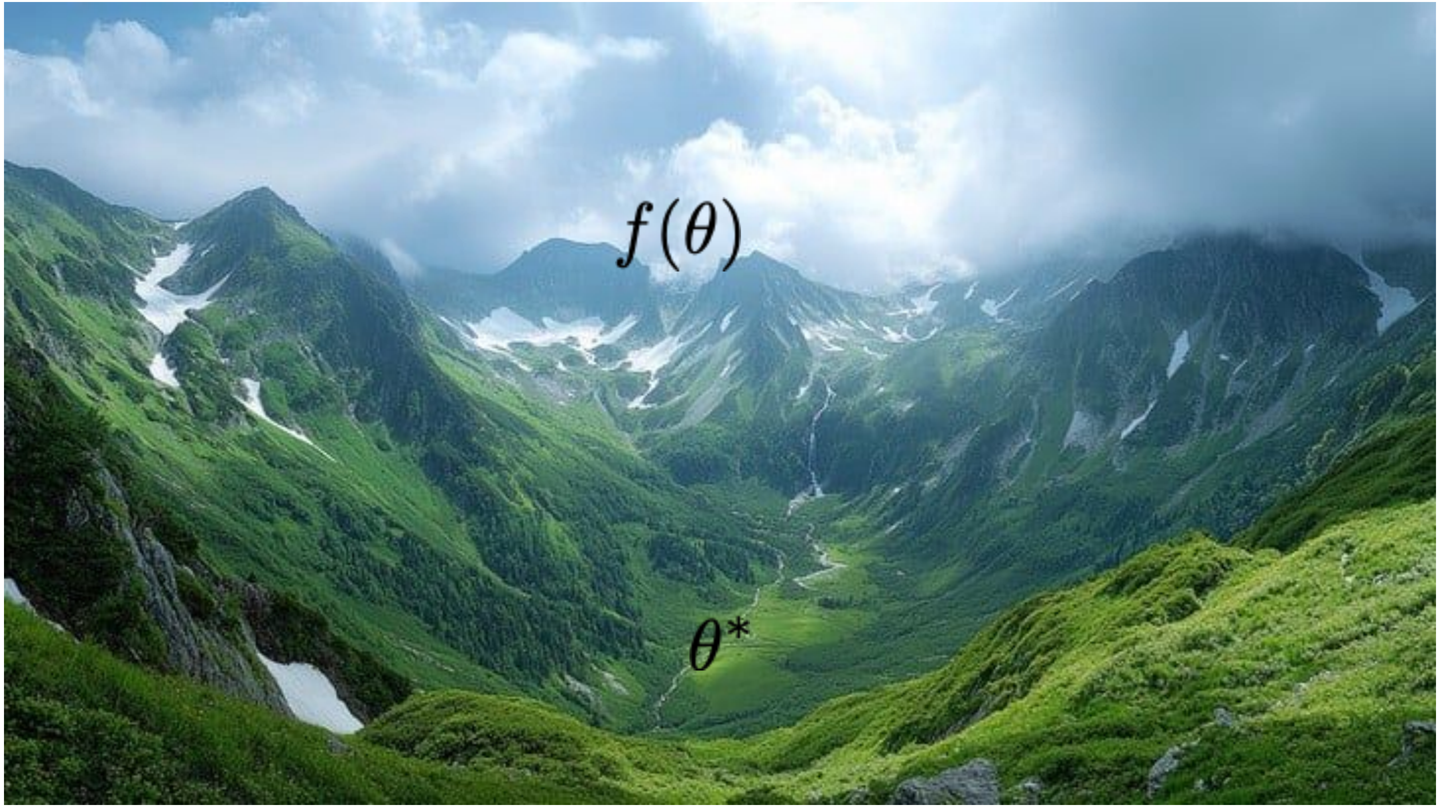
- $\theta \in \mathbf{R}^d$ is the variable or decision variable
- $f: \mathbf{R}^d \to \mathbf{R}$ is the objective function
- goal is to choose θ to minimize f

Optimization Problem

$$\text{minimize } f(\theta)$$

- $\theta \in \mathbf{R}^d$ is the variable or decision variable
- $f: \mathbf{R}^d \to \mathbf{R}$ is the objective function
- goal is to choose θ to minimize f
- $\theta^\star$ is optimal means that for all $\theta, f(\theta) \geq f(\theta^\star)$ $\qquad \theta^* = \arg\min_\theta \; f(\theta)$
- $f^\star = f(\theta^\star)$ is the optimal value of the problem $\qquad f^* = \min_\theta \; f(\theta)$


$f(\theta)$


$f(\theta)$
θ^*

Random Search ?!

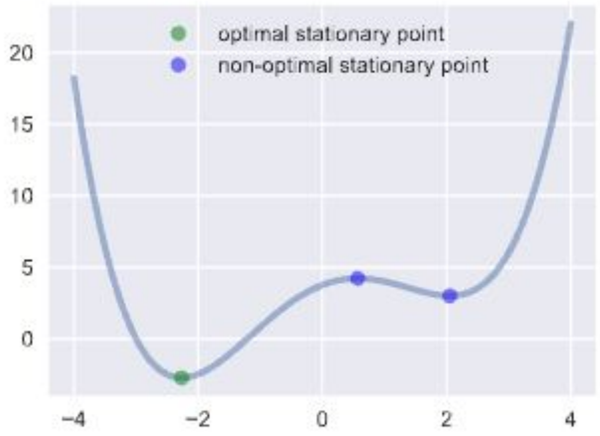
Random Search ?!

Random Search ?!

Random Search ?!

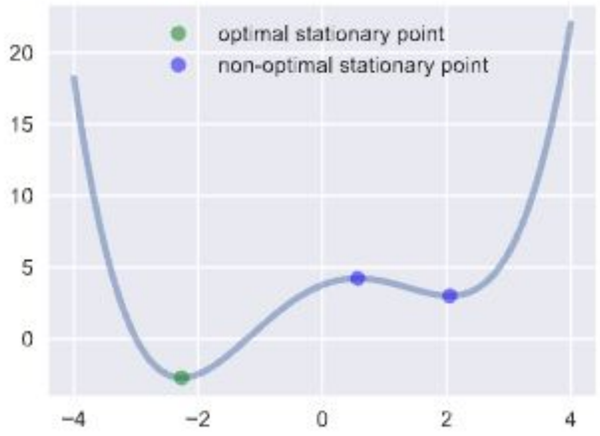
Random Search ?!

Optimality Condition



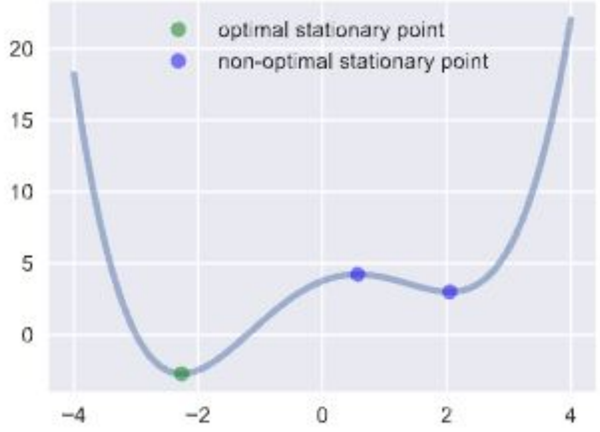
- let's assume that f is differentiable, i.e., partial derivatives $\frac{\partial f(\theta)}{\partial \theta_i}$ exist
- if $\theta^\star$ is optimal, then $\nabla f(\theta^\star) = 0$
- $\nabla f(\theta) = 0$ is called the optimality condition for the problem

Optimality Condition



- let's assume that f is differentiable, i.e., partial derivatives $\frac{\partial f(\theta)}{\partial \theta_i}$ exist
- if $\theta^\star$ is optimal, then $\nabla f(\theta^\star) = 0$
- $\nabla f(\theta) = 0$ is called the optimality condition for the problem
- there can be points that satisfy $\nabla f(\theta) = 0$ but are not optimal
- we call points that satisfy $\nabla f(\theta) = 0$ stationary points
- not all stationary points are optimal

Optimality Condition



- let's assume that f is differentiable, i.e., partial derivatives $\frac{\partial f(\theta)}{\partial \theta_i}$ exist
- if $\theta^\star$ is optimal, then $\nabla f(\theta^\star) = 0$
- $\nabla f(\theta) = 0$ is called the optimality condition for the problem
- there can be points that satisfy $\nabla f(\theta) = 0$ but are not optimal
- we call points that satisfy $\nabla f(\theta) = 0$ stationary points
- not all stationary points are optimal

Solving Optimization Problems

- in some cases, we can solve the problem analytically

Solving Optimization Problems

- in some cases, we can solve the problem analytically
- e.g., least squares: minimize $f(\theta) = \|X\theta - y\|_2^2$

Solving Optimization Problems

- in some cases, we can solve the problem analytically
- e.g., least squares: minimize $f(\theta) = \|X\theta - y\|_2^2$
- optimality condition is $\nabla f(\theta) = 2X^\top(X\theta - y) = 0$

Solving Optimization Problems

- in some cases, we can solve the problem analytically
- e.g., least squares: minimize $f(\theta) = \|X\theta - y\|_2^2$
- optimality condition is $\nabla f(\theta) = 2X^\top(X\theta - y) = 0$
  this has unique solution $\theta^\star = (X^\top X)^1 X^\top y = X^\dagger y$ (when columns of X are linearly independent)

Solving Optimization Problems

- in some cases, we can solve the problem analytically
- e.g., least squares: minimize $f(\theta) = \|X\theta - y\|_2^2$
- optimality condition is $\nabla f(\theta) = 2X^\top(X\theta - y) = 0$
  this has unique solution $\theta^\star = (X^\top X)^1 X^\top y = X^\dagger y$ (when columns of X are linearly independent)

What if optimality condition is difficult to be solved?

Local Search

Local Search

Local Search

Local Search

Local Search

Local Search

Iterative Algorithm

- iterative algorithm computes a sequence $\theta^1, \theta^2, \ldots$
- θ^k is called the k th iterate
- θ^1 is called the starting point

$$f(\theta^{k+1}) < f(\theta^k), k = 1,2, \ldots$$

i.e., each iterate is better than the previous one

Iterative Algorithm

- iterative algorithm computes a sequence $\theta^1, \theta^2, \ldots$
- θ^k is called the k th iterate
- θ^1 is called the starting point

$$f(\theta^{k+1}) < f(\theta^k), k = 1,2, \ldots$$

i.e., each iterate is better than the previous one

- this means that $f(\theta^k)$ converges, but not necessarily to $f^\star$

Local Search

Local Search

Local Search

Local Search

Local Search

Iterative Algorithm

- iterative algorithm computes a sequence $\theta^1, \theta^2, \ldots$
- θ^k is called the k th iterate
- θ^1 is called the starting point

$$f(\theta^{k+1}) < f(\theta^k), k = 1,2, \ldots$$

i.e., each iterate is better than the previous one

- this means that $f(\theta^k)$ converges, but not necessarily to $f^\star$

Local Search

Gradient Descent

Gradient Descent

Gradient Descent

Iterative Algorithm

- iterative algorithm computes a sequence $\theta^1, \theta^2, \ldots$
- θ^k is called the k th iterate
- θ^1 is called the starting point
- many iterative algorithms are descent methods, which means

$$f(\theta^{k+1}) < f(\theta^k), k = 1,2, \ldots$$

i.e., each iterate is better than the previous one

- this means that $f(\theta^k)$ converges, but not necessarily to $f^\star$

Gradient Method Summary

choose an initial $\theta^1 \in \mathbf{R}^d$ and $h^1 > 0$ (e.g., $\theta^1 = 0, h^1 = 1$)
for $k = 1,2,\ldots,k^{\max}$

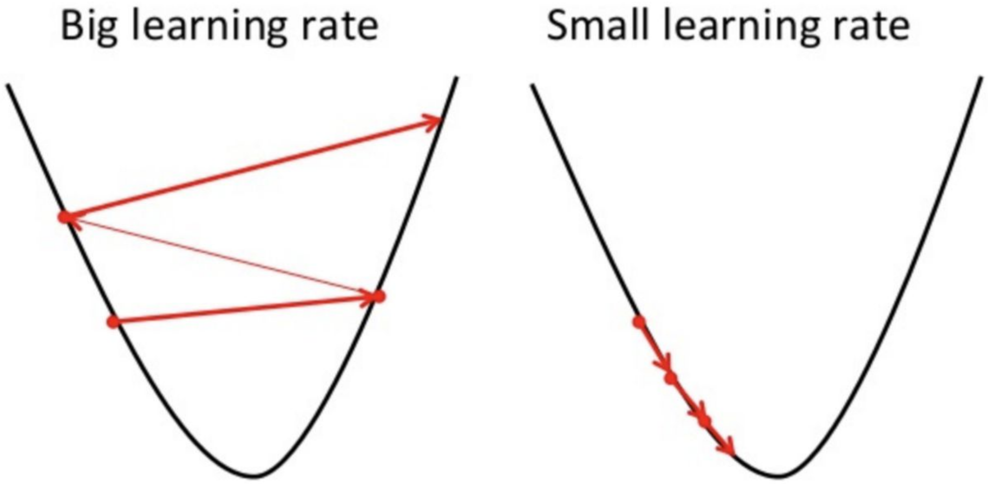
1. compute $\nabla f(\theta^k)$; quit if $\|\nabla f(\theta^k)\|_2$ is small enough
2. form tentative update $\theta^{\text{tent}} = \theta^k - h^k \nabla f(\theta^k)$
3. if $f(\theta^{\text{tent}}) < f(\theta^k)$, set $\theta^{k+1} = \theta^{\text{tent}}, h^{k+1} = 1.2h^k$
4. else set $h^k := 0.5h^k$ and go to step 2

Gradient Method Summary

choose an initial $\theta^1 \in \mathbf{R}^d$ and $h^1 > 0$ (e.g., $\theta^1 = 0, h^1 = 1$)
for $k = 1,2,\dots,k^{\max}$

1. compute $\nabla f(\theta^k)$; quit if $\|\nabla f(\theta^k)\|_2$ is small enough
2. form tentative update $\theta^{\text{tent}} = \theta^k - h^k \nabla f(\theta^k)$
3. $\boxed{\text{if } f(\theta^{\text{tent}}) < f(\theta^k)\text{, set } \theta^{k+1} = \theta^{\text{tent}}, h^{k+1} = 1.2h^k}$
4. else set $h^k := 0.5h^k$ and go to step 2

Step-size Matters



Stopping Criterion

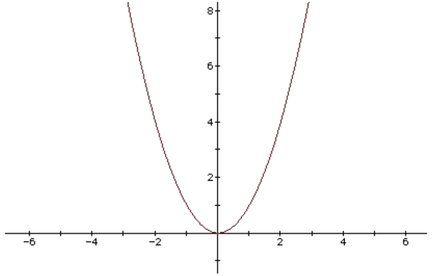
- in practice, we stop after a finite number K of steps
- typical stopping criterion: stop if $\left\|\nabla f\left(\theta^k\right)\right\|_2 \leq \epsilon$ or $k = k^{\max}$
- ϵ is a small positive number, the stopping tolerance
- $k^{\max}$ is the maximum number of iterations

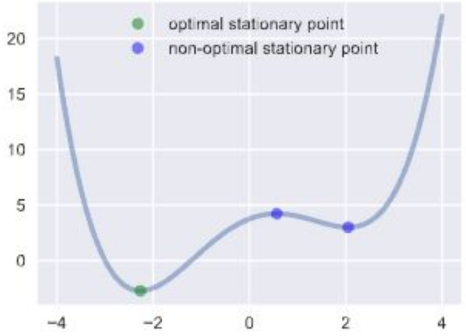
Gradient Method Convergence

- (assuming some technical conditions hold) we have

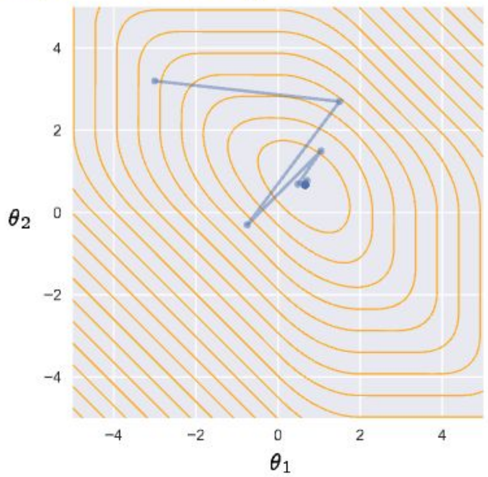
$$\|\nabla f(\theta^k)\|_2 \to 0 \text{ as } k \to \infty$$

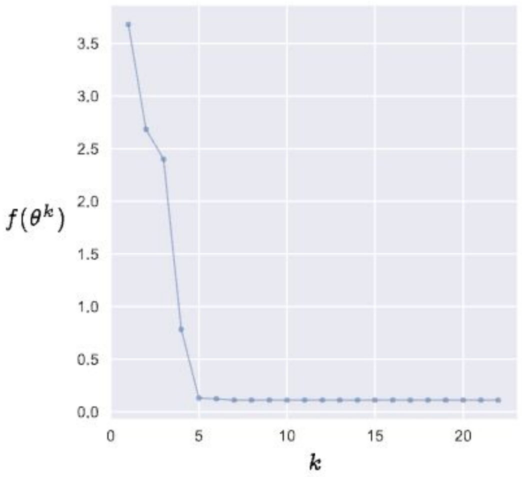
- i.e., the gradient method always finds a stationary point
- for convex problems
  - gradient method is non-heuristic
  - for any starting point $\theta^1, f(\theta^k) \to f^\star$ as $k \to \infty$
- for non-convex problems
  - gradient method is heuristic
  - we can (and often do) have $f(\theta^k) \nrightarrow f^\star$





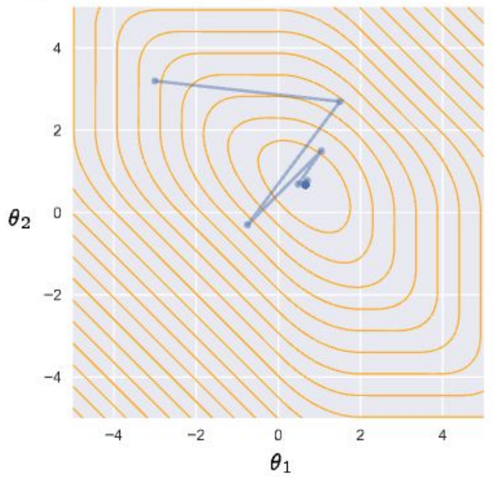
Example: Convex Objective

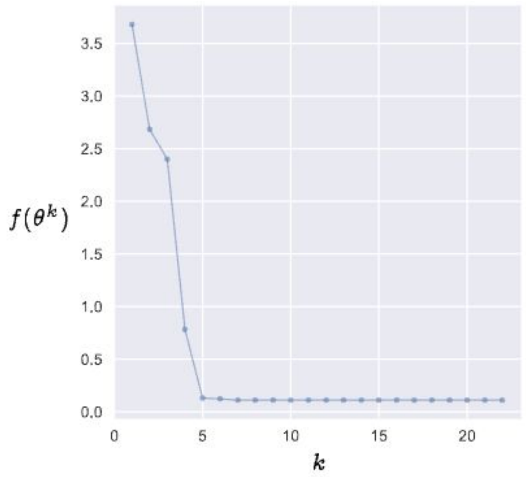




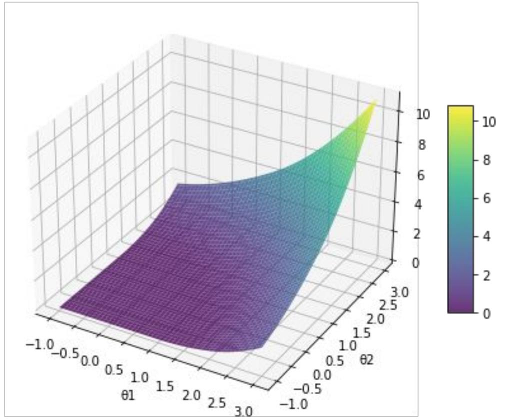
- f is convex

Example: Convex Objective

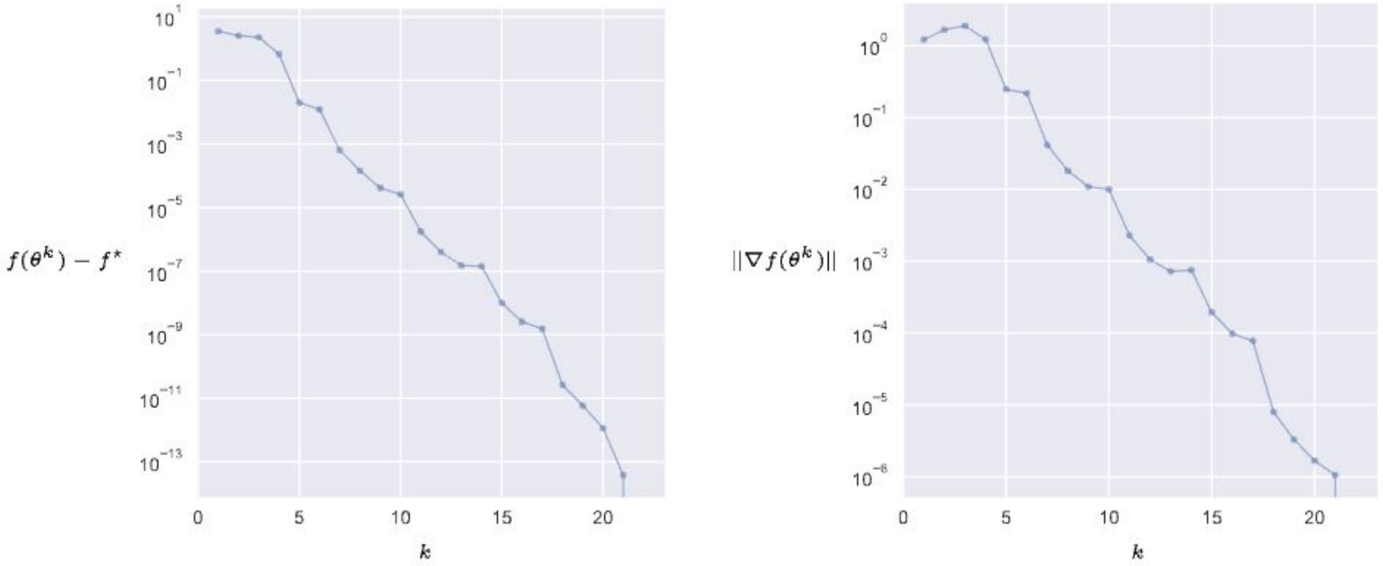




- $f(\theta) = \frac{1}{3}\left(p^{\text{hub}}(\theta_1 - 1) + p^{\text{hub}}(\theta_2 - 1) + p^{\text{hub}}(\theta_1 + \theta_2 - 1)\right)$
- f is convex
- optimal point is $\theta^\star = (2/3, 2/3)$, with $f^\star = 1/9$

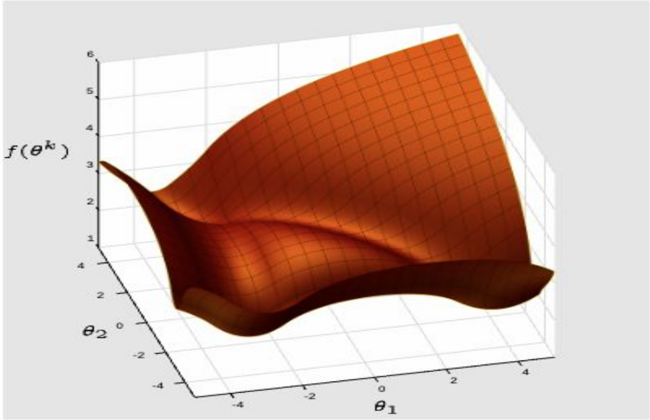


Example: Convex Objective



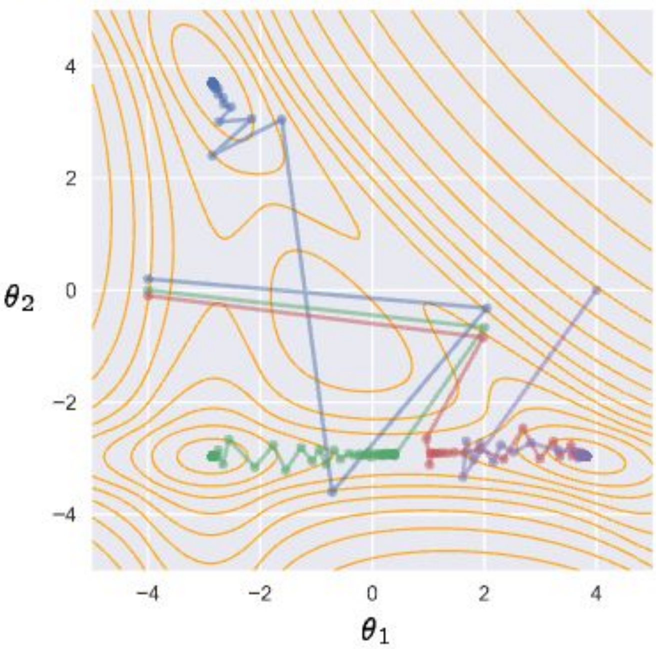
- $f(\theta^k)$ is a decreasing function of k, (roughly) exponentially
- $\|\nabla f(\theta^k)\| \to 0$ as $k \to \infty$

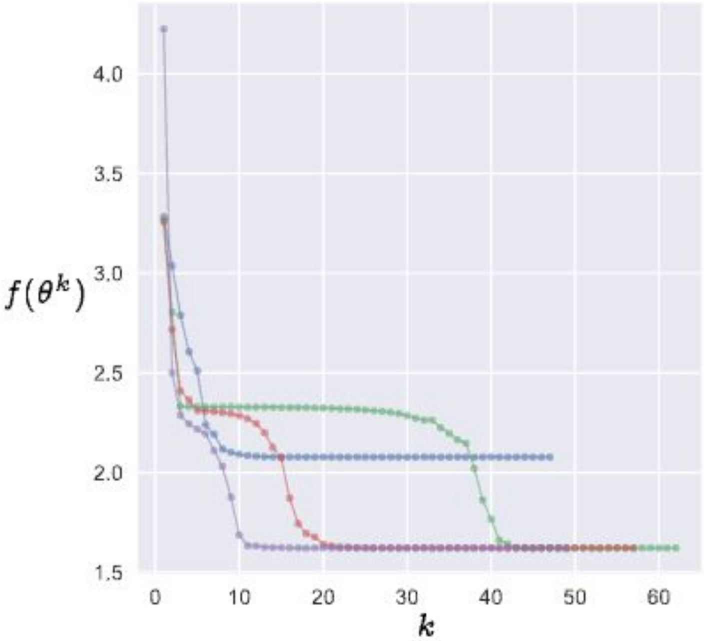
Example: Non-Convex Objective



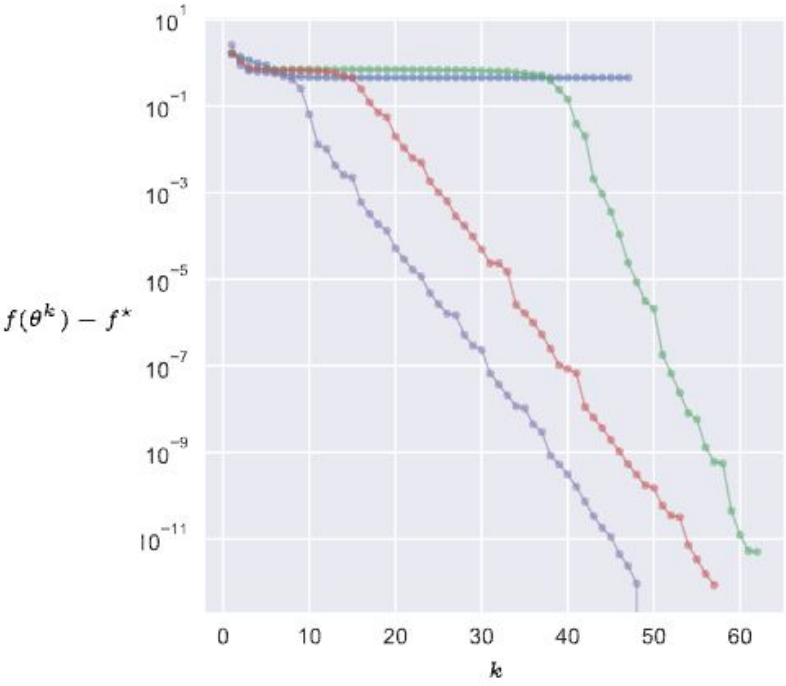
- gradient algorithm converges, but limit depends on initial guess

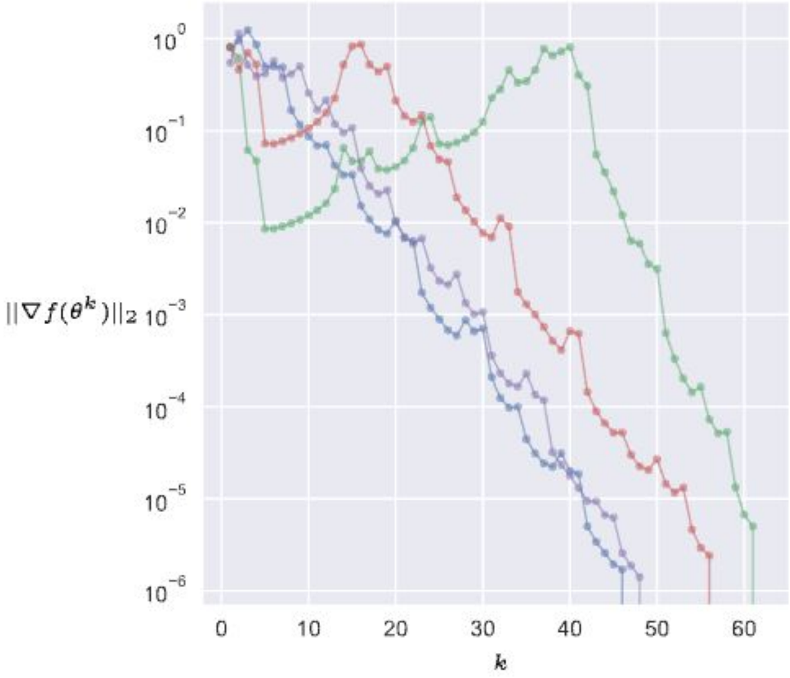
Example: Non-Convex Objective





Example: Non-Convex Objective

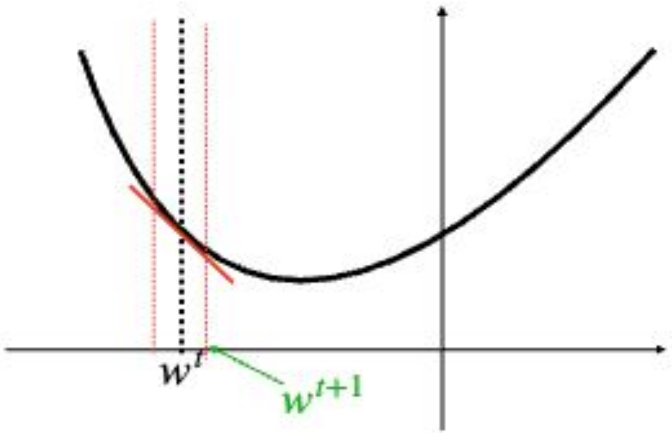




Recap on Gradient Descent

Gradient descent minimizes $\ell(w)$ iteratively:

$$w^{t+1} = w^t - \eta \nabla \ell(w)|_{w=w_t}$$



Stochastic Gradient Descent

$$\text{Goal: minimize } \ell(w) = \frac{1}{n}\sum_{i=1}^{n} \ell(x_i, y_i; w)$$

Initialize $w^0 \in \mathbb{R}^d$ randomly
Iterate until convergence:

1. Randomly sample a point (x_i, y_i) from the n data points
2. Compute noisy gradient $\tilde{g}^t = \nabla \ell(x_i, y_i; w)|_{w=w^t}$
3. Update (GD): $w^{t+1} = w^t - \eta \tilde{g}^t$

Intuition of why Stochastic GD can work

Claim: the random noisy gradient is an unbiased estimate of the true gradient

Note the point (x_i, y_i) is uniformly random sampled from n data points, we have:

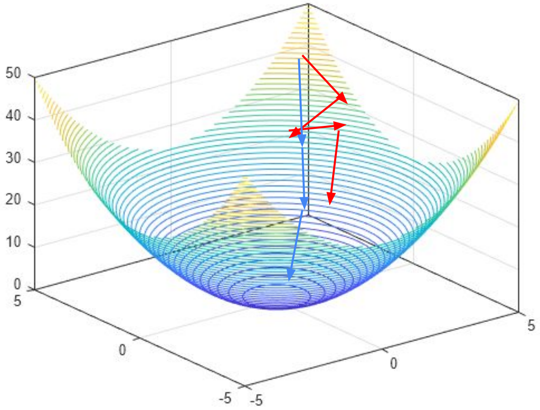
$$\mathbb{E}\nabla\ell(x_i, y_i; w)$$

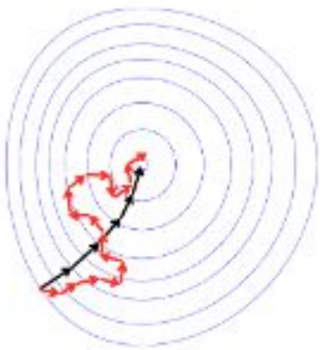
$$= \frac{1}{n}\sum_{i=1}^{n} \nabla\ell(x_i, y_i; w) = \nabla\underbrace{[\frac{1}{n}\sum_{i=1}^{n} \ell(x_i, y_i; w)]}_{\ell(w)}{}_{\ell(w)} = \nabla\ell(w)$$

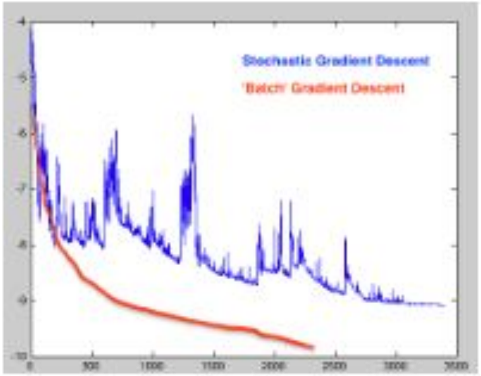
Stochastic gradient descent generally makes more iterations than gradient descent.

Each iteration is much cheaper (by a factor of n).

$$\vec{\nabla} f(\vec{\theta}) = \vec{\nabla} \sum_{j=1}^{n} f_j(\vec{\theta}) \text{ vs. } \vec{\nabla} f_j(\vec{\theta})$$







Apply GD and SGD to LMS

$$\max_{\theta} \ \log L(\theta) = -\frac{1}{2\sigma^2}\sum_{i=1}^{m}(y^i - \theta^\top x^i)^2 - m\log(\sqrt{2\pi}\sigma)$$

- The gradient of LMS is

$$\frac{1}{m}\nabla_\theta \log L(\theta) = \frac{1}{m\sigma^2}\sum_{i=1}^{m}(y^i - \theta^\top x^i)\, x^i$$

- The stochastic gradient of LMS is

$$\nabla_\theta \log L(\theta)\Big|_{\text{sample } i} = \frac{1}{\sigma^2}\left(y^i - \theta^\top x^i\right) x^i$$

Summary

- Random Search
- Closed-form
- Iterative methods:
    - Local Search
    - Gradient Descent
    - Stochastic Gradient Descent

- Homework 1 is released
- Due: 11:59PM EST, 01/29/2025

Q&A