# Introduction to Deep Learning

## 22. Optimization, Gradient Descent

## STAT 157, Spring 2019, UC Berkeley

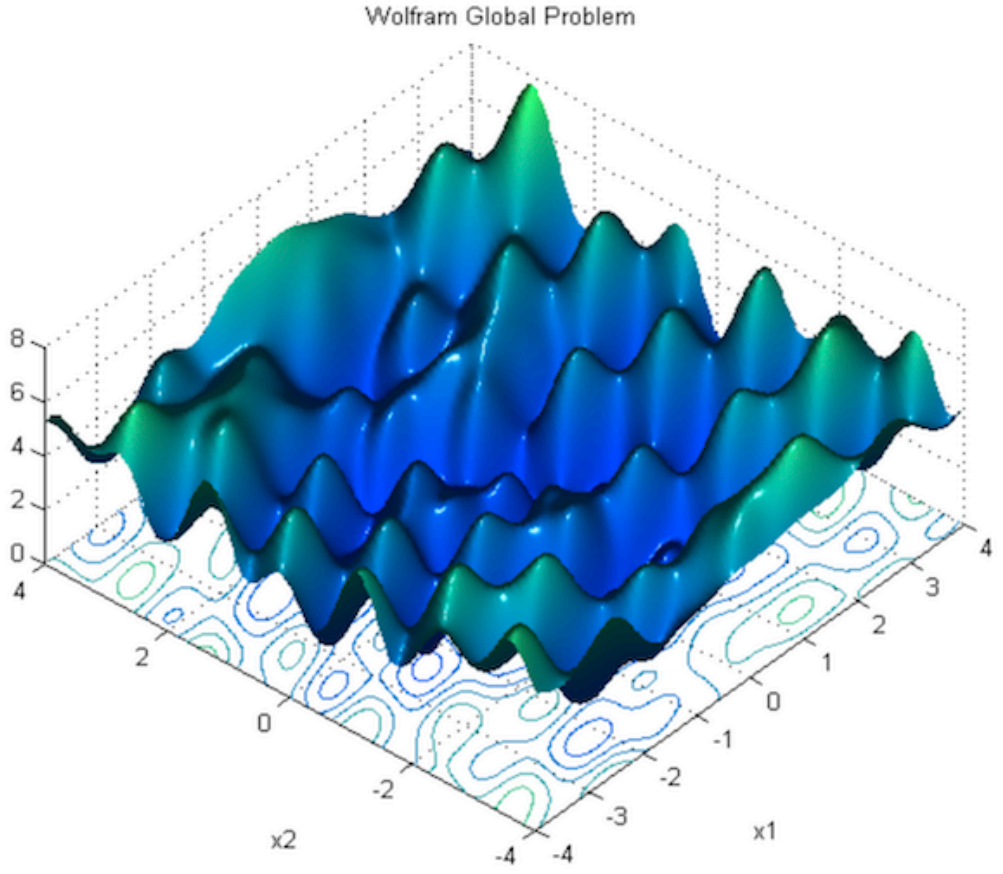
**Alex Smola and Mu Li**

courses.d2l.ai/berkeley-stat-157

© 2018, Amazon Web Services, Inc. or its Affiliates. All rights reserved.

aws

# Optimization



# Optimization Problems

- General form:

$$\text{minimize } f(\mathbf{x}) \quad \text{subject to } \mathbf{x} \in C$$

  - Cost function $f : \mathbb{R}^n \to \mathbb{R}$
  - Constraint set example

$$C = \left\{\mathbf{x} \,|\, h_1(\mathbf{x}) = 0, \ldots, h_m(\mathbf{x}) = 0,\ g_1(\mathbf{x}) \leq 0, \ldots, g_r(\mathbf{x}) \leq 0\right\}$$

  - Unconstraint if $C = \mathbb{R}^n$



aws

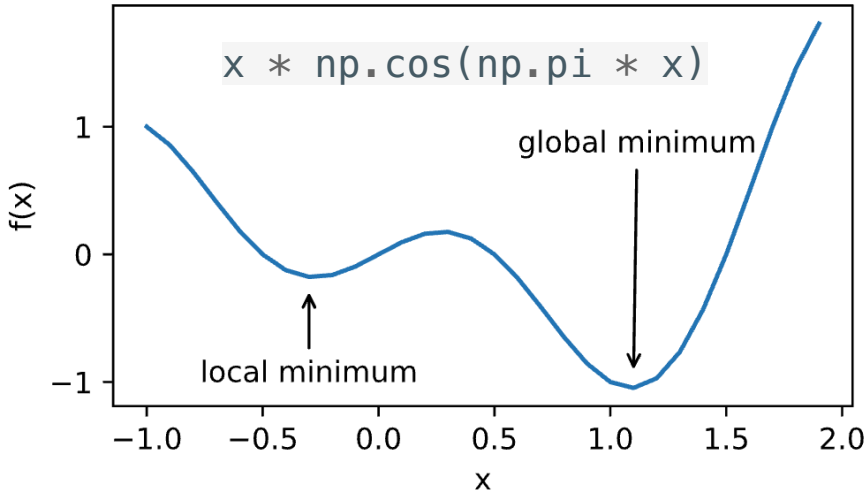
# Local Minima and Global Minima

- Most optimization problems have no close form solution
- We then aim to find a minima through iterative methods
- Global minima $\mathbf{x}^*$

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \quad \forall \mathbf{x} \in C$$

- Local minima $\mathbf{x}^*$, there exists $\varepsilon$

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \quad \forall \mathbf{x} : \|\mathbf{x} - \mathbf{x}^*\| \leq \varepsilon$$

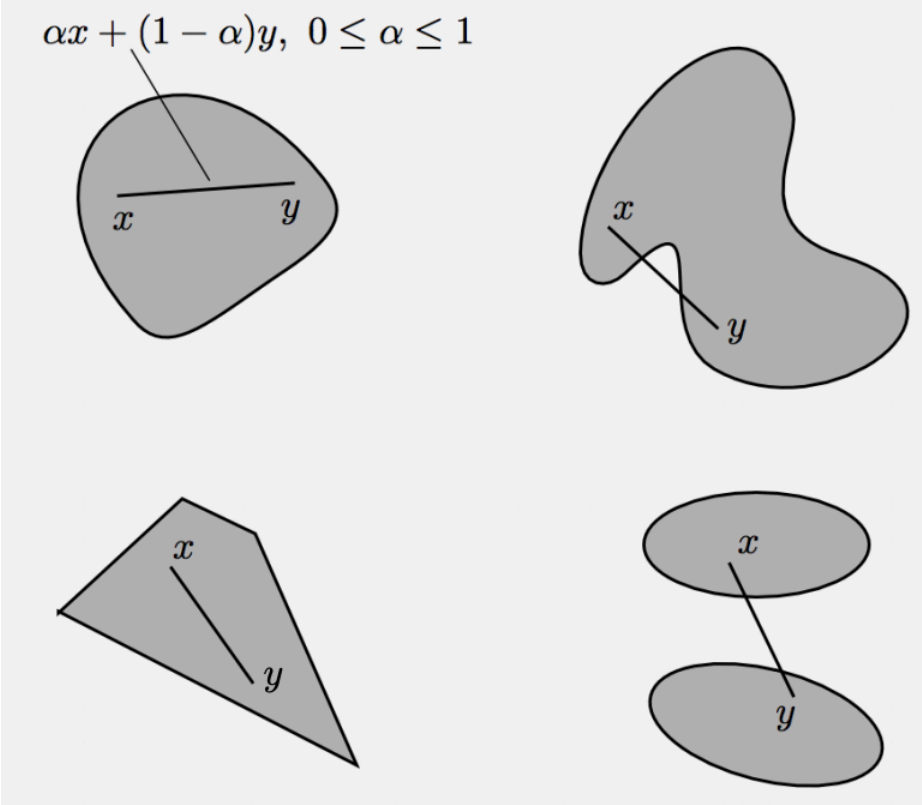


# Convex Set

- A subset $C$ of $\mathbb{R}^n$ is called convex if

$$\alpha \mathbf{x} + (1-\alpha)\mathbf{y} \in C$$

$$\forall \alpha \in [0,1] \; \forall \mathbf{x}, \mathbf{y} \in C$$






aws

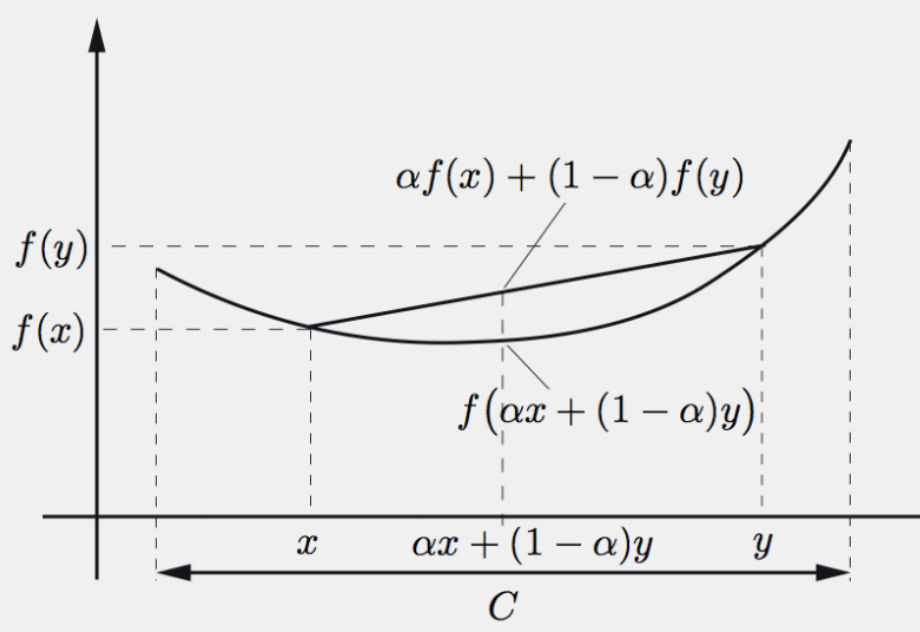
# Convex Function

- $f : C \to \mathbb{R}$ is called convex if

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

$$\forall \alpha \in [0,1] \quad \forall \mathbf{x}, \mathbf{y} \in C$$

- If the inequality is strict whenever $\alpha \in (0,1)$ and $\mathbf{x} \neq \mathbf{y}$, then *f* is called strictly convex

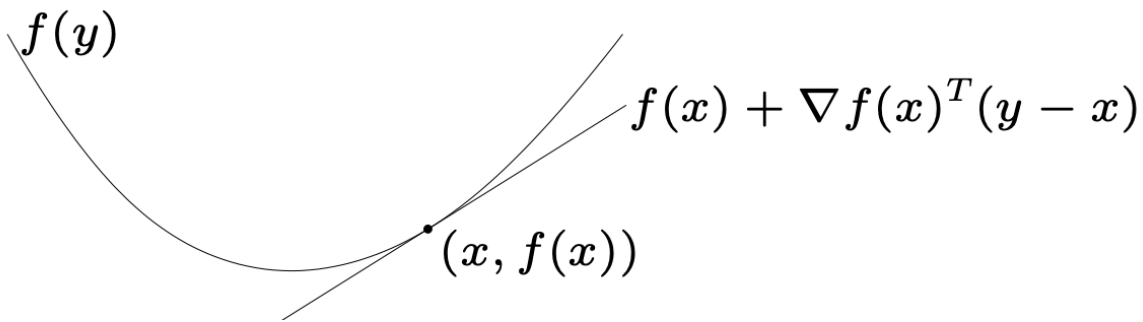


# First-order condition

- *f* is convex if and only if

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) \quad \forall \mathbf{x}, \mathbf{y} \in C$$

- If the inequality is strict, then *f* is strictly convex



# Second-order conditions

- *f* is convex if and only if

$$\nabla^2 f(\mathbf{x}) \succeq 0 \quad \forall \mathbf{x} \in C$$

- *f* is strictly convex if and only if

$$\nabla^2 f(\mathbf{x}) \succ 0 \quad \forall \mathbf{x} \in C$$



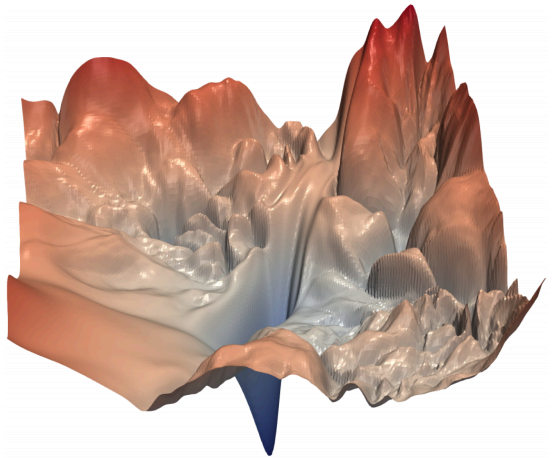
aws

# Convex and Non-convex Examples

- Convex
  - Linear regression $f(\mathbf{x}) = \|\mathbf{W}\mathbf{x} - \mathbf{b}\|_2^2$

$$\nabla f(\mathbf{x}) = 2\mathbf{W}^T(\mathbf{W}\mathbf{x} - \mathbf{b}), \quad \nabla^2 f(\mathbf{x}) = 2\mathbf{W}^T\mathbf{W}$$

  - Softmax regression
- Non-convex
  - Multi-layer perception
  - Convolution neural networks
  - Recurrent neural networks





aws

# Convex Optimization

- If *f* is a convex function, and *C* is a convex set, then the problem is called a convex problem
- Any local minima is a global minima
- Unique global minima if strictly convex




aws

# Proof

- Assume local minima $\mathbf{x}$, if exists a global minima $\mathbf{y}$
  - Choose $\alpha \leq 1 - \varepsilon / |\mathbf{x} + \mathbf{y}|$ and $\mathbf{z} = \alpha \mathbf{x} + (1 - \alpha)\mathbf{y}$
  - Then $\|\mathbf{x} - \mathbf{z}\| = (1 - \alpha)\|\mathbf{x} + \mathbf{y}\| \leq \varepsilon$
  - Due to $\mathbf{y}$ is a global minima, so $f(\mathbf{y}) < f(\mathbf{x})$

$$f(\mathbf{z}) \leq \alpha f(\mathbf{x}) + (1 - a) f(\mathbf{z}) < \alpha f(\mathbf{x}) + (1 - a) f(\mathbf{x}) = f(\mathbf{x})$$

  - It contradicts $\mathbf{x}$ is a local minima



aws

# Gradient Descent

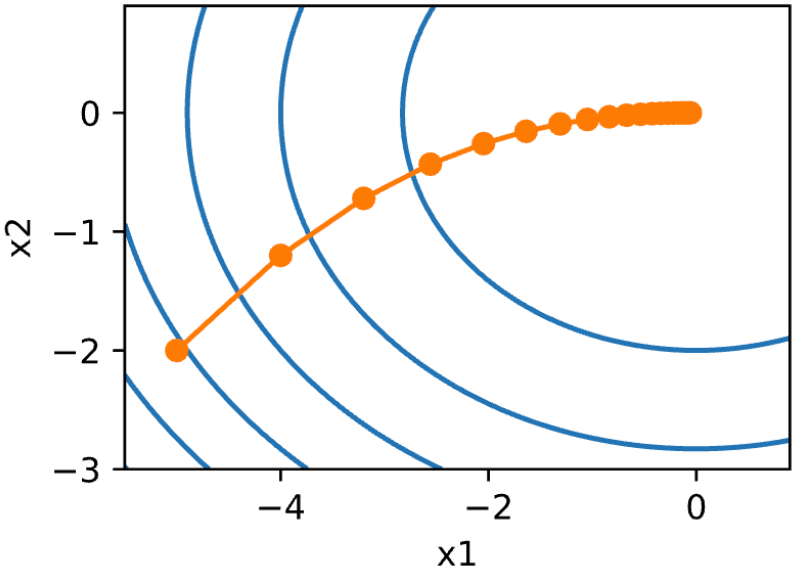

courses.d2l.ai/berkeley-stat-157

# Algorithm

- Choose initial $\mathbf{x}_0$
- At time $t = 1, \ldots, T$

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \eta \nabla f(\mathbf{x}_{t-1})$$

  - $\eta$ is called learning rate



# The Choice of Learning Rate

- Given $\|\boldsymbol{\Delta}\| < \varepsilon$, for any *f,* by the Taylor expansion

$$f(\mathbf{x} + \boldsymbol{\Delta}) \approx f(\mathbf{x}) + \boldsymbol{\Delta}^T \nabla f(\mathbf{x})$$

- Choose small enough learning rate $\eta \leq \varepsilon / \|\nabla f(\mathbf{x})\|$

$$\| - \eta \nabla f(\mathbf{x})\| \leq \varepsilon$$

$$f(\mathbf{x} - \eta \nabla f(\mathbf{x})) \approx f(\mathbf{x}) - \eta \|\nabla f(\mathbf{x})\|^2 \leq f(\mathbf{x})$$



aws

# Convergence Rate

- Assume *f* is convex, and its gradient is Lipschitz continuous with constant *L*

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$$

Gradient does not change dramatically

- If use learning rate $\eta \leq 1/L$, after *T* steps

$$f(\mathbf{x}_T) - f(\mathbf{x}^*) \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\|^2}{2\eta T}$$

  - Convergence rate $O(1/T)$
  - To get $f(\mathbf{x}_T) - f(\mathbf{x}^*) \leq \epsilon$, needs $O(1/\epsilon)$ iterations



aws

# Proof

- Gradient L-Lipschitz means

$$f(\mathbf{y}) \leq f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|^2$$

- Plug in $\mathbf{y} = \mathbf{x} - \eta \nabla f(\mathbf{x})$

$$f(\mathbf{y}) \leq f(\mathbf{x}) - \left(1 - \frac{L\eta}{2}\right)\eta\|\nabla f(\mathbf{x})\|^2$$

- Take $0 < \eta \leq 1/L$

$$f(\mathbf{y}) \leq f(\mathbf{x}) - \frac{\eta}{2}\|\nabla f(\mathbf{x})\|^2$$

*f* decreases every time



aws

# Proof II

- By the convexity: $f(\mathbf{x}) \leq f(\mathbf{x}^*) + \nabla f(\mathbf{x})^T(\mathbf{x} - \mathbf{x}^*)$
- Plug in to $f(\mathbf{y}) \leq f(\mathbf{x}) - \frac{\eta}{2}\|\nabla f(\mathbf{x})\|^2$

$$f(\mathbf{y}) \leq f(\mathbf{x}^*) + \nabla f(\mathbf{x})^T(\mathbf{x} - \mathbf{x}^*) - \frac{\eta}{2}\|\nabla f(\mathbf{x})\|^2$$

$$\begin{aligned}
f(\mathbf{y}) - f(\mathbf{x}^*) &\leq \left(2\eta\nabla f(\mathbf{x})^T(\mathbf{x} - \mathbf{x}^*) - \eta^2\|\nabla f(\mathbf{x})\|^2\right)/2\eta \\
&= \left(\|\mathbf{x} - \mathbf{x}^*\|^2 + 2\eta\nabla f(\mathbf{x})^T(\mathbf{x} - \mathbf{x}^*) - \eta^2\|\nabla f(\mathbf{x})\|^2 - \|\mathbf{x} - \mathbf{x}^*\|^2\right)/2\eta \\
&= \left(\|\mathbf{x} - \mathbf{x}^*\|^2 - \|\mathbf{x} - \eta\nabla f(\mathbf{x}) - \mathbf{x}^*\|^2\right)/2\eta \\
&= \left(\|\mathbf{x} - \mathbf{x}^*\|^2 - \|\mathbf{y} - \mathbf{x}^*\|^2\right)/2\eta
\end{aligned}$$



aws

# Proof III

- Sum all $T$ steps

$$\sum_{t=1}^{T} f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq \sum_{t=1}^{T} \left( \|\mathbf{x}_{t-1} - \mathbf{x}^*\|^2 - \|\mathbf{x}_t - \mathbf{x}^*\|^2 \right) / 2\eta$$

$$= \left( \|\mathbf{x}_0 - \mathbf{x}^*\|^2 - \|\mathbf{x}_T - \mathbf{x}^*\|^2 \right) / 2\eta \leq \|\mathbf{x}_0 - \mathbf{x}^*\|^2 / 2\eta$$

- $f$ is decreasing every time:

$$f(\mathbf{x}_T) - f(\mathbf{x}^*) \leq \frac{1}{T} \sum_{t=1}^{T} f(\mathbf{x}_t) - f(\mathbf{x}^*) \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\|^2}{2\eta T}$$



aws

# Apply to Deep Learning

- *f* is the sum of loss over all training data, **x** is the learnable parameters

$$f(\mathbf{x}) = \frac{1}{n}\sum_{i=0}^{n} \ell_i(\mathbf{x}) \qquad \ell_i(\mathbf{x}) \text{ the loss for the } i\text{-th example}$$

- *f* is often not convex, so the convergence analysis before cannot be applied



aws

# Stochast Gradient Descent


Singapore Dollar (SGD) 1000
~740 USD

# Algorithm

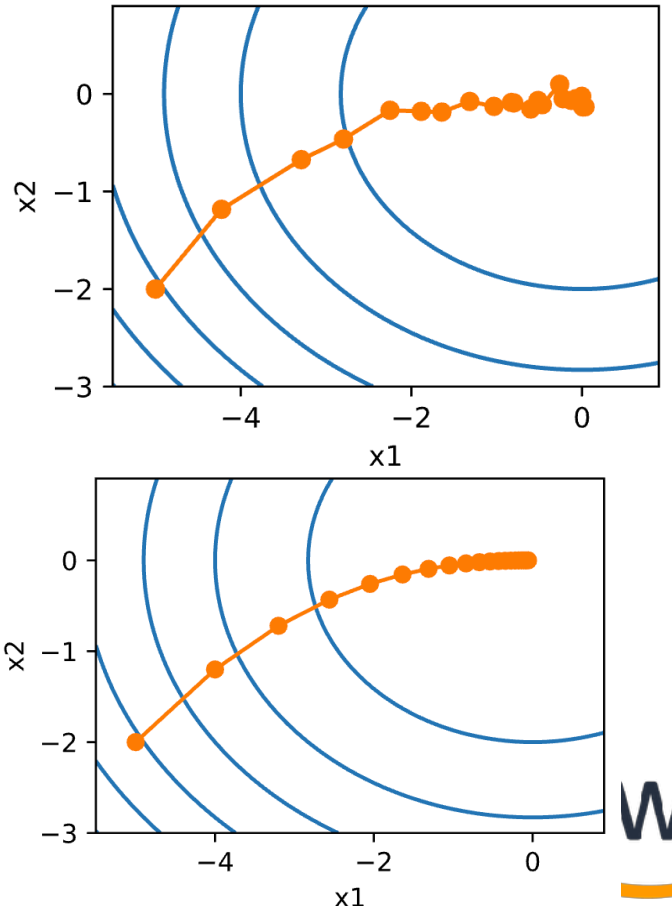
- At time $t$ , sample example $t_i$

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \eta_t \nabla \ell_{t_i}(\mathbf{x}_{t-1})$$

- Compare to gradient descent

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \eta \nabla f(\mathbf{x}_{t-1})$$

$$f(\mathbf{x}) = \frac{1}{n} \sum_{i=0}^{n} \ell_i(\mathbf{x})$$



# Sample Example

- Two rules to sample example it at time t
  - Random rule: choose $i_t \in \{1,\ldots,n\}$ uniformly at random
  - Cyclic rule: choose $i_t = 1,2,\ldots,n,1,2,\ldots,n$
    - Often called incremental gradient descent
- Randomized rule is more common in practice

$$\mathbb{E}\left[\nabla \ell_{t_i}(\mathbf{x})\right] = \mathbb{E}[\nabla f(\mathbf{x})]$$

  - An unbiased estimate of the gradient



aws

# Convergence Rate

- Assume *f* is convex with a diminishing $\eta_t$ , e.g. $\eta_t = O(1/t)$

$$\mathbb{E}[f(\mathbf{x}_T)] - f(\mathbf{x}^*) = O(1/\sqrt{T})$$

- Under the same assumption, for gradient descent

$$f(\mathbf{x}_T) - f(\mathbf{x}^*) = O(1/\sqrt{T})$$

- Assume gradient L-Lipschitz and fixed $\eta$

$$f(\mathbf{x}_T) - f(\mathbf{x}^*) = O(1/T)$$

  - Does not improve for SGD



aws

# In Practice

- Does not diminish the learning rate so dramatically
  - We don’t care about optimizing to high accuracy
- Despite converging slower, SGD is way faster on computing the gradient than GD in each iteration
  - Specially for deep learning with complex models and large-scale datasets



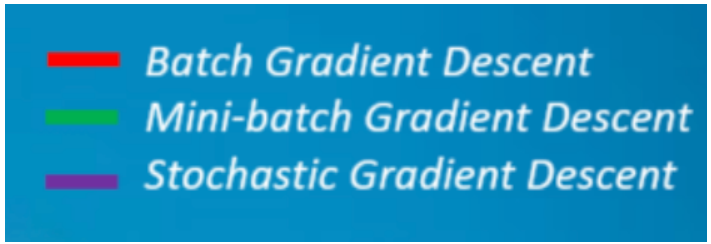
aws

# Code…



aws

# Mini-batch SGD




courses.d2l.ai/berkeley-stat-157

# Algorithm

- At time $t$, sample a random subset $I_t \subset \{1,\ldots,n\}$ with $|I_t| = b$

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \frac{\eta_t}{b} \sum_{i \in I_t} \nabla \ell_i(\mathbf{x}_{t-1})$$

- Again, it's an unbiased estimate

$$\mathbb{E}\left[\frac{1}{b} \sum_{i \in I_t} \nabla \ell_i(\mathbf{x})\right] = \nabla f(\mathbf{x})$$

  - Reduces variance by a factor of 1/b compared to SGD



aws

# Code…



aws