COMPSCI 514: ALGORITHMS FOR DATA SCIENCE

Andrew McGregor

Lecture 23

SUMMARY

Last Class:

- Analysis of gradient descent for optimizing convex functions.
- Introduction to convex sets and projection functions.
- (The same) analysis of projected gradient descent for optimizing under convex functions under (convex) constraints.

This Class:

- Online learning, regret, and online gradient descent.
- Application to stochastic gradient descent.

ONLINE GRADIENT DESCENT

In reality many learning problems are online.

- Websites optimize ads or recommendations to show users, given continuous feedback from these users.
- Spam filters are incrementally updated and adapt as they see more examples of spam over time.
- Face recognition systems, other classification systems, learn from mistakes over time.

ONLINE GRADIENT DESCENT

In reality many learning problems are online.

- Websites optimize ads or recommendations to show users, given continuous feedback from these users.
- Spam filters are incrementally updated and adapt as they see more examples of spam over time.
- Face recognition systems, other classification systems, learn from mistakes over time.

Want to minimize some global loss $L(\vec{\theta}, \mathbf{X}) = \sum_{i=1}^{n} \ell(\vec{\theta}, \vec{x}_i)$, when data points are presented in an online fashion $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n$ (similar to streaming algorithms)

ONLINE OPTIMIZATION FORMAL SETUP

Online Optimization: In place of a single function f, we see a different objective function at each step:

$$f_1, f_2, \ldots, f_t : \mathbb{R}^d \to \mathbb{R}$$

Online Optimization: In place of a single function f, we see a different objective function at each step:

$$f_1, f_2, \ldots, f_t : \mathbb{R}^d \to \mathbb{R}$$

- At each step, first pick (play) a parameter vector $\vec{\theta}^{(i)}$.
- Then are told f_i and incur cost $f_i(\vec{\theta}^{(i)})$.
- **Goal:** Minimize total cost $\sum_{i=1}^{t} f_i(\vec{\theta}^{(i)})$.

ONLINE OPTIMIZATION FORMAL SETUP

Online Optimization: In place of a single function f, we see a different objective function at each step:

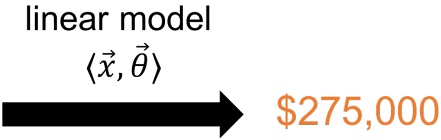
$$f_1, f_2, \ldots, f_t : \mathbb{R}^d \to \mathbb{R}$$

- At each step, first pick (play) a parameter vector $\vec{\theta}^{(i)}$.
- Then are told f_i and incur cost $f_i(\vec{\theta}^{(i)})$.
- **Goal:** Minimize total cost $\sum_{i=1}^t f_i(\vec{\theta}^{(i)})$.

Our analysis will make no assumptions on how $f_1, \ldots, f_t$ are related to each other!

ONLINE OPTIMIZATION EXAMPLE

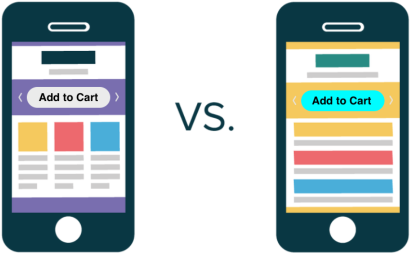
Home pricing tools.



$\vec{x} = [\#baths, \#beds, \#floors \ldots]$

- Parameter vector $\vec{\theta}^{(i)}$: coefficients of linear model at step i.
- Functions $f_1, \ldots, f_t$: $f_i(\vec{\theta}^{(i)}) = (\langle \vec{x}_i, \vec{\theta}^{(i)} \rangle - price_i)^2$ revealed when $home_i$ is listed or sold.
- Want to minimize total squared error $\sum_{i=1}^{t} f_i(\vec{\theta}^{(i)})$ (same as classic least squares regression).

UI design via online optimization.



- Parameter vector $\vec{\theta}^{(i)}$: some encoding of the layout at step i.
- Functions $f_1, \ldots, f_t$: $f_i(\vec{\theta}^{(i)}) = 1$ if user does not click 'add to cart' and $f_i(\vec{\theta}^{(i)}) = 0$ if they do click.
- Want to maximize number of purchases, i.e., minimize $\sum_{i=1}^{t} f_i(\vec{\theta}^{(i)})$.

REGRET

In normal optimization, we seek $\hat{\theta}$ satisfying:

$$f(\hat{\theta}) \leq \min_{\vec{\theta}} f(\vec{\theta}) + \epsilon.$$

REGRET

In normal optimization, we seek $\hat{\theta}$ satisfying:

$$f(\hat{\theta}) \leq \min_{\vec{\theta}} f(\vec{\theta}) + \epsilon.$$

In online optimization we want:

$$\sum_{i=1}^{t} f_i(\vec{\theta}^{(i)}) \leq \sum_{i=1}^{t} f_i(\vec{\theta}^{off}) + \epsilon$$

where $\vec{\theta}^{off} = \arg\min_{\vec{\theta}} \sum_{i=1}^{t} f_i(\vec{\theta})$ and ϵ is called the regret and ϵ/t is the average regret.

REGRET

In normal optimization, we seek $\hat{\theta}$ satisfying:

$$f(\hat{\theta}) \leq \min_{\vec{\theta}} f(\vec{\theta}) + \epsilon.$$

In online optimization we want:

$$\sum_{i=1}^{t} f_i(\vec{\theta}^{(i)}) \leq \sum_{i=1}^{t} f_i(\vec{\theta}^{off}) + \epsilon$$

where $\vec{\theta}^{off} = \arg\min_{\vec{\theta}} \sum_{i=1}^{t} f_i(\vec{\theta})$ and ϵ is called the regret and ϵ/t is the average regret.

- This error metric is a bit unusual: Comparing online solution to best fixed "online" solution in hindsight. ϵ can be negative!

INTUITION CHECK

What if for $i = 1, \ldots, t$, $f_i(\theta) = |\theta - 1000|$ or $f_i(\theta) = |\theta + 1000|$ in an alternating pattern?

How small can the regret ϵ be? $\sum_{i=1}^{t} f_i(\vec{\theta}^{(i)}) \leq \sum_{i=1}^{t} f_i(\vec{\theta}^{off}) + \epsilon$.

INTUITION CHECK

What if for $i = 1, \ldots, t$, $f_i(\theta) = |\theta - 1000|$ or $f_i(\theta) = |\theta + 1000|$ in an alternating pattern?

How small can the regret ϵ be? $\sum_{i=1}^{t} f_i(\vec{\theta}^{(i)}) \leq \sum_{i=1}^{t} f_i(\vec{\theta}^{off}) + \epsilon$.

What if for $i = 1, \ldots, t$, $f_i(\theta) = |\theta - 1000|$ or $f_i(\theta) = |\theta + 1000|$ in no particular pattern? How can any online learning algorithm hope to achieve small regret?

Assume that:

- $f_1, \ldots, f_t$ are all convex.
- Each f_i is G-Lipschitz, i.e., $\|\vec{\nabla} f_i(\vec{\theta})\|_2 \leq G$ for all $\vec{\theta}$.
- $\|\vec{\theta}^{(1)} - \vec{\theta}^{off}\|_2 \leq R$ where $\theta^{(1)}$ is the first vector chosen.

Assume that:

- $f_1, \ldots, f_t$ are all convex.
- Each f_i is G-Lipschitz, i.e., $\|\vec{\nabla} f_i(\vec{\theta})\|_2 \leq G$ for all $\vec{\theta}$.
- $\|\vec{\theta}^{(1)} - \vec{\theta}^{off}\|_2 \leq R$ where $\theta^{(1)}$ is the first vector chosen.

Online Gradient Descent

- Pick some initial $\vec{\theta}^{(1)}$.
- Set step size $\eta = \frac{R}{G\sqrt{t}}$.
- For $i = 1, \ldots, t$
  - Play $\vec{\theta}^{(i)}$ and incur cost $f_i(\vec{\theta}^{(i)})$.
  - $\vec{\theta}^{(i+1)} = \vec{\theta}^{(i)} - \eta \cdot \vec{\nabla} f_i(\vec{\theta}^{(i)})$

ONLINE GRADIENT DESCENT ANALYSIS

Theorem: For convex G-Lipschitz $f_1, \ldots, f_t$, OGD initialized with starting point $\theta^{(1)}$ within radius R of θ^{off}, using step size $\eta = \frac{R}{G\sqrt{t}}$, has regret bounded by:

$$\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] \leq RG\sqrt{t}$$

ONLINE GRADIENT DESCENT ANALYSIS

Theorem: For convex G-Lipschitz $f_1, \ldots, f_t$, OGD initialized with starting point $\theta^{(1)}$ within radius R of θ^{off}, using step size $\eta = \frac{R}{G\sqrt{t}}$, has regret bounded by:

$$\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] \leq RG\sqrt{t}$$

Average regret goes to 0 and $t \to \infty$.

ONLINE GRADIENT DESCENT ANALYSIS

Theorem: For convex G-Lipschitz $f_1, \ldots, f_t$, OGD initialized with starting point $\theta^{(1)}$ within radius R of θ^{off}, using step size $\eta = \frac{R}{G\sqrt{t}}$, has regret bounded by:

$$\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] \leq RG\sqrt{t}$$

Average regret goes to 0 and $t \to \infty$. No assumptions on $f_1, \ldots, f_t$!

ONLINE GRADIENT DESCENT ANALYSIS

Theorem: For convex G-Lipschitz $f_1, \ldots, f_t$, OGD initialized with starting point $\theta^{(1)}$ within radius R of θ^{off}, using step size $\eta = \frac{R}{G\sqrt{t}}$, has regret bounded by:

$$\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] \leq RG\sqrt{t}$$

Average regret goes to 0 and $t \to \infty$. No assumptions on $f_1, \ldots, f_t$!

Step 1: For all i,

$$\nabla f_i(\theta^{(i)})^T(\theta^{(i)} - \theta^{off}) \leq \frac{\|\theta^{(i)} - \theta^{off}\|_2^2 - \|\theta^{(i+1)} - \theta^{off}\|_2^2}{2\eta} + \frac{\eta G^2}{2}$$

Theorem: For convex G-Lipschitz $f_1, \ldots, f_t$, OGD initialized with starting point $\theta^{(1)}$ within radius R of θ^{off}, using step size $\eta = \frac{R}{G\sqrt{t}}$, has regret bounded by:

$$\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] \leq RG\sqrt{t}$$

Average regret goes to 0 and $t \to \infty$. No assumptions on $f_1, \ldots, f_t$!

Step 1: For all i,

$$\nabla f_i(\theta^{(i)})^T(\theta^{(i)} - \theta^{off}) \leq \frac{\|\theta^{(i)} - \theta^{off}\|_2^2 - \|\theta^{(i+1)} - \theta^{off}\|_2^2}{2\eta} + \frac{\eta G^2}{2}$$

Step 2: Convexity implies that for all i,

$$f_i(\theta^{(i)}) - f_i(\theta^{off}) \leq \frac{\|\theta^{(i)} - \theta^{off}\|_2^2 - \|\theta^{(i+1)} - \theta^{off}\|_2^2}{2\eta} + \frac{\eta G^2}{2}.$$

ONLINE GRADIENT DESCENT ANALYSIS

Theorem: For convex G-Lipschitz $f_1, \ldots, f_t$, OGD initialized with starting point $\theta^{(1)}$ within radius R of θ^{off}, using step size $\eta = \frac{R}{G\sqrt{t}}$, has regret bounded by:

$$\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] \leq RG\sqrt{t}$$

Step 2: For all i,

$$f_i(\theta^{(i)}) - f_i(\theta^{off}) \leq \frac{\|\theta^{(i)} - \theta^{off}\|_2^2 - \|\theta^{(i+1)} - \theta^{off}\|_2^2}{2\eta} + \frac{\eta G^2}{2}$$

Theorem: For convex G-Lipschitz $f_1, \ldots, f_t$, OGD initialized with starting point $\theta^{(1)}$ within radius R of θ^{off}, using step size $\eta = \frac{R}{G\sqrt{t}}$, has regret bounded by:

$$\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] \leq RG\sqrt{t}$$

Step 2: For all i,

$$f_i(\theta^{(i)}) - f_i(\theta^{off}) \leq \frac{\|\theta^{(i)} - \theta^{off}\|_2^2 - \|\theta^{(i+1)} - \theta^{off}\|_2^2}{2\eta} + \frac{\eta G^2}{2}$$

Step 3:

$$\begin{aligned}\left[\sum_{i=1}^{t} f_i(\theta^{(i)}) - \sum_{i=1}^{t} f_i(\theta^{off})\right] &\leq \sum_{i=1}^{t} \frac{\|\theta^{(i)} - \theta^{off}\|_2^2 - \|\theta^{(i+1)} - \theta^{off}\|_2^2}{2\eta} + \frac{t \cdot \eta G^2}{2} \\ &= \frac{\|\theta^{(1)} - \theta^{off}\|_2^2 - \|\theta^{(t+1)} - \theta^{off}\|_2^2}{2\eta} + \frac{t \cdot \eta G^2}{2} \\ &\leq R^2/(2\eta) + t\eta G^2/2 = RG\sqrt{t}\end{aligned}$$

Stochastic gradient descent is an efficient offline optimization method, seeking $\hat{\theta}$ with

$$f(\hat{\theta}) \leq \min_{\vec{\theta}} f(\vec{\theta}) + \epsilon$$

STOCHASTIC GRADIENT DESCENT

Stochastic gradient descent is an efficient offline optimization method, seeking $\hat{\theta}$ with

$$f(\hat{\theta}) \leq \min_{\vec{\theta}} f(\vec{\theta}) + \epsilon$$

- The most popular optimization method in modern machine learning.
  Easily analyzed as a special case of online gradient descent!

STOCHASTIC GRADIENT DESCENT

Stochastic gradient descent is an efficient offline optimization method, seeking $\hat{\theta}$ with

$$f(\hat{\theta}) \leq \min_{\vec{\theta}} f(\vec{\theta}) + \epsilon$$

- The most popular optimization method in modern machine learning. Easily analyzed as a special case of online gradient descent!
- Basic Idea: In gradient descent, we set $\vec{\theta}_{i+1} = \vec{\theta}_i - \eta \cdot \vec{\nabla} f(\vec{\theta}_i)$.

STOCHASTIC GRADIENT DESCENT

Stochastic gradient descent is an efficient offline optimization method, seeking $\hat{\theta}$ with

$$f(\hat{\theta}) \leq \min_{\vec{\theta}} f(\vec{\theta}) + \epsilon$$

- The most popular optimization method in modern machine learning. Easily analyzed as a special case of online gradient descent!
- Basic Idea: In gradient descent, we set $\vec{\theta}_{i+1} = \vec{\theta}_i - \eta \cdot \vec{\nabla} f(\vec{\theta}_i)$. In stochastic gradient descent we don't compute $\vec{\nabla} f(\vec{\theta}_i)$ exactly but instead do something random that is correct in expectation. This saves time per step but might increase the number of steps.

STOCHASTIC GRADIENT DESCENT

Assume that:

- f is convex and decomposable as $f(\vec{\theta}) = \sum_{j=1}^{n} f_j(\vec{\theta})$.

Assume that:

- f is convex and decomposable as $f(\vec{\theta}) = \sum_{j=1}^{n} f_j(\vec{\theta})$.
  - For example, trying to minimize a loss function over a data set $\mathbf{X}$, $L(\vec{\theta}, \mathbf{X}) = \sum_{j=1}^{n} \ell(\vec{\theta}, \vec{x}_j)$ that is a sum of losses of element in data set.
- Each f_j is $\frac{G}{n}$-Lipschitz:

Assume that:

- f is convex and decomposable as $f(\vec{\theta}) = \sum_{j=1}^{n} f_j(\vec{\theta})$.
  - For example, trying to minimize a loss function over a data set $\mathbf{X}$, $L(\vec{\theta}, \mathbf{X}) = \sum_{j=1}^{n} \ell(\vec{\theta}, \vec{x}_j)$ that is a sum of losses of element in data set.
- Each f_j is $\frac{G}{n}$-Lipschitz:

$$\|\nabla f(\vec{\theta})\|_2 \leq \| \sum_{j=1}^{n} \nabla f_j(\vec{\theta})\|_2 \leq \sum_{j=1}^{n} \|\nabla f_j(\vec{\theta})\|_2 \leq G \ .$$

- Initialize with $\theta^{(1)}$ satisfying $\|\vec{\theta}^{(1)} - \vec{\theta}^*\|_2 \leq R$.

Assume that:

- f is convex and decomposable as $f(\vec{\theta}) = \sum_{j=1}^{n} f_j(\vec{\theta})$.
  - For example, trying to minimize a loss function over a data set $\mathbf{X}$, $L(\vec{\theta}, \mathbf{X}) = \sum_{j=1}^{n} \ell(\vec{\theta}, \vec{x}_j)$ that is a sum of losses of element in data set.
- Each f_j is $\frac{G}{n}$-Lipschitz:

$$\|\nabla f(\vec{\theta})\|_2 \leq \| \sum_{j=1}^{n} \nabla f_j(\vec{\theta})\|_2 \leq \sum_{j=1}^{n} \|\nabla f_j(\vec{\theta})\|_2 \leq G \ .$$

- Initialize with $\theta^{(1)}$ satisfying $\|\vec{\theta}^{(1)} - \vec{\theta}^*\|_2 \leq R$.

Stochastic Gradient Descent

- Pick some initial $\vec{\theta}^{(1)}$.
- Set step size $\eta = \frac{R}{G\sqrt{t}}$.
- For $i = 1, \ldots, t$
  - Pick random $j_i \in 1, \ldots, n$.
  - $\vec{\theta}^{(i+1)} = \vec{\theta}^{(i)} - \eta \cdot \vec{\nabla} f_{j_i}(\vec{\theta}^{(i)})$
- Return $\hat{\theta} = \frac{1}{t} \sum_{i=1}^{t} \vec{\theta}^{(i)}$.

If $f(x, y) = (x^2 + 3xy) + (x + y)$ then gradient descent updates

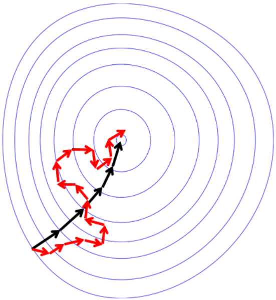
$$\theta^{i+1} = \theta^i - \eta \begin{pmatrix} 2\theta_1^i + 3\theta_2^i + 1 \\ 3\theta_1^i + 1 \end{pmatrix}$$

With probability $1/2$, stochastic gradient descent updates

$$\theta^{i+1} = \theta^i - \eta \begin{pmatrix} 2\theta_1^i + 3\theta_2^i \\ 3\theta_1^i \end{pmatrix}$$

and with probability $1/2$ the update is:

$$\theta^{i+1} = \theta^i - \eta \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$



$$\vec{\theta}^{(i+1)} = \vec{\theta}^{(i)} - \eta \cdot \vec{\nabla} f_{j_i}(\vec{\theta}^{(i)}) \quad \text{vs.} \quad \vec{\theta}^{(i+1)} = \vec{\theta}^{(i)} - \eta \cdot \vec{\nabla} f(\vec{\theta}^{(i)})$$

Note that: $\mathbb{E}[\vec{\nabla} f_{j_i}(\vec{\theta}^{(i)})] = \frac{1}{n}\vec{\nabla} f(\vec{\theta}^{(i)})$.

Analysis extends to any algorithm that takes the gradient step in expectation (minibatch SGD, randomly quantized, measurement noise, differentially private, etc.)

Theorem – SGD on Convex Lipschitz Functions: SGD run with $t \geq \frac{R^2 G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of θ^*, outputs $\hat{\theta}$ satisfying: $\mathbb{E}[f(\hat{\theta})] \leq f(\theta^*) + \epsilon$.

STOCHASTIC GRADIENT DESCENT ANALYSIS

Theorem – SGD on Convex Lipschitz Functions: SGD run with $t \geq \frac{R^2 G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of θ^*, outputs $\hat{\theta}$ satisfying: $\mathbb{E}[f(\hat{\theta})] \leq f(\theta^*) + \epsilon.$

Step 1: $f(\hat{\theta}) - f(\theta^*) \leq \frac{1}{t} \sum_{i=1}^{t} [f(\theta^{(i)}) - f(\theta^*)]$

STOCHASTIC GRADIENT DESCENT ANALYSIS

Theorem – SGD on Convex Lipschitz Functions: SGD run with $t \geq \frac{R^2 G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of θ^*, outputs $\hat{\theta}$ satisfying: $\mathbb{E}[f(\hat{\theta})] \leq f(\theta^*) + \epsilon$.

Step 1: $f(\hat{\theta}) - f(\theta^*) \leq \frac{1}{t}\sum_{i=1}^{t}[f(\theta^{(i)}) - f(\theta^*)]$ since

$$f(\hat{\theta}) = f(\sum_{i=1}^{t} \theta^{(i)}/t) \leq \frac{1}{t}\sum_{i=1}^{t} f(\theta^{(i)}) \text{ by convexity}$$

Theorem – SGD on Convex Lipschitz Functions: SGD run with $t \geq \frac{R^2 G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of θ^*, outputs $\hat{\theta}$ satisfying: $\mathbb{E}[f(\hat{\theta})] \leq f(\theta^*) + \epsilon$.

Step 1: $f(\hat{\theta}) - f(\theta^*) \leq \frac{1}{t}\sum_{i=1}^{t}[f(\theta^{(i)}) - f(\theta^*)]$ since

$$f(\hat{\theta}) = f(\sum_{i=1}^{t} \theta^{(i)}/t) \leq \frac{1}{t}\sum_{i=1}^{t} f(\theta^{(i)}) \text{ by convexity}$$

Step 2: $\mathbb{E}[f(\hat{\theta}) - f(\theta^*)] \leq \frac{n}{t} \cdot \mathbb{E}\left[\sum_{i=1}^{t}[f_{j_i}(\theta^{(i)}) - f_{j_i}(\theta^*)]\right]$

STOCHASTIC GRADIENT DESCENT ANALYSIS

Theorem – SGD on Convex Lipschitz Functions: SGD run with $t \geq \frac{R^2G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of θ^*, outputs $\hat{\theta}$ satisfying: $\mathbb{E}[f(\hat{\theta})] \leq f(\theta^*) + \epsilon$.

Step 1: $f(\hat{\theta}) - f(\theta^*) \leq \frac{1}{t}\sum_{i=1}^{t}[f(\theta^{(i)}) - f(\theta^*)]$ since

$$f(\hat{\theta}) = f(\sum_{i=1}^{t} \theta^{(i)}/t) \leq \frac{1}{t}\sum_{i=1}^{t} f(\theta^{(i)}) \text{ by convexity}$$

Step 2: $\mathbb{E}[f(\hat{\theta}) - f(\theta^*)] \leq \frac{n}{t} \cdot \mathbb{E}\left[\sum_{i=1}^{t}[f_{j_i}(\theta^{(i)}) - f_{j_i}(\theta^*)]\right]$ since

$$\mathbb{E}[f_{j_i}(\vec{\theta})] = \frac{1}{n} f(\vec{\theta}) \text{ since } f(\vec{\theta}) = \textstyle\sum_{j=1}^{n} f_j(\vec{\theta})$$

Theorem – SGD on Convex Lipschitz Functions: SGD run with $t \geq \frac{R^2G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of θ^*, outputs $\hat{\theta}$ satisfying: $\mathbb{E}[f(\hat{\theta})] \leq f(\theta^*) + \epsilon$.

Step 1: $f(\hat{\theta}) - f(\theta^*) \leq \frac{1}{t}\sum_{i=1}^{t}[f(\theta^{(i)}) - f(\theta^*)]$ since

$$f(\hat{\theta}) = f(\sum_{i=1}^{t} \theta^{(i)}/t) \leq \frac{1}{t}\sum_{i=1}^{t} f(\theta^{(i)}) \text{ by convexity}$$

Step 2: $\mathbb{E}[f(\hat{\theta}) - f(\theta^*)] \leq \frac{n}{t} \cdot \mathbb{E}\left[\sum_{i=1}^{t}[f_{j_i}(\theta^{(i)}) - f_{j_i}(\theta^*)]\right]$ since

$$\mathbb{E}[f_{j_i}(\vec{\theta})] = \frac{1}{n} f(\vec{\theta}) \text{ since } f(\vec{\theta}) = \textstyle\sum_{j=1}^{n} f_j(\vec{\theta})$$

Step 3: $\mathbb{E}[f(\hat{\theta}) - f(\theta^*)] \leq \frac{n}{t} \cdot \underbrace{R \cdot \frac{G}{n} \cdot \sqrt{t}}_{\text{OGD bound}} = \frac{RG}{\sqrt{t}}$.

SGD VS. GD: TIME PER ITERATION

Stochastic gradient descent generally makes more iterations than gradient descent.

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

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

- Foundational concepts like convexity (line between any two points on curve is above the curve), convex sets (line between any two points in set in the set), directional derivative (slope of curve if we move in particular direction), and Lipschitzness (slope is bounded).

- Foundational concepts like convexity (line between any two points on curve is above the curve), convex sets (line between any two points in set in the set), directional derivative (slope of curve if we move in particular direction), and Lipschitzness (slope is bounded).
- Gradient descent greedily tries to find the min value of function $f : \mathbb{R}^d \to \mathbb{R}$ by maintaining a vector $\vec{\theta} \in \mathbb{R}^d$ and at each step moving $\vec{\theta}$ "downhill", i.e., in the direction that minimizes directional derivative

CONTINUOUS OPTIMIZATION

- Foundational concepts like convexity (line between any two points on curve is above the curve), convex sets (line between any two points in set in the set), directional derivative (slope of curve if we move in particular direction), and Lipschitzness (slope is bounded).
- Gradient descent greedily tries to find the min value of function $f : \mathbb{R}^d \to \mathbb{R}$ by maintaining a vector $\vec{\theta} \in \mathbb{R}^d$ and at each step moving $\vec{\theta}$ "downhill", i.e., in the direction that minimizes directional derivative
- Bounded the number of steps required if f is convex and Lipschitz.

CONTINUOUS OPTIMIZATION

- Foundational concepts like convexity (line between any two points on curve is above the curve), convex sets (line between any two points in set in the set), directional derivative (slope of curve if we move in particular direction), and Lipschitzness (slope is bounded).
- Gradient descent greedily tries to find the min value of function $f : \mathbb{R}^d \to \mathbb{R}$ by maintaining a vector $\vec{\theta} \in \mathbb{R}^d$ and at each step moving $\vec{\theta}$ "downhill", i.e., in the direction that minimizes directional derivative
- Bounded the number of steps required if f is convex and Lipschitz.
- Simple extensions for optimization over a convex constraint set or online optimization.

CONTINUOUS OPTIMIZATION

- Foundational concepts like convexity (line between any two points on curve is above the curve), convex sets (line between any two points in set in the set), directional derivative (slope of curve if we move in particular direction), and Lipschitzness (slope is bounded).
- Gradient descent greedily tries to find the min value of function $f : \mathbb{R}^d \to \mathbb{R}$ by maintaining a vector $\vec{\theta} \in \mathbb{R}^d$ and at each step moving $\vec{\theta}$ "downhill", i.e., in the direction that minimizes directional derivative
- Bounded the number of steps required if f is convex and Lipschitz.
- Simple extensions for optimization over a convex constraint set or online optimization.
- Can typically speed up offline optimization via stochastic gradient descent: requires more iterations but each iteration is faster.

CONTINUOUS OPTIMIZATION

- Foundational concepts like convexity (line between any two points on curve is above the curve), convex sets (line between any two points in set in the set), directional derivative (slope of curve if we move in particular direction), and Lipschitzness (slope is bounded).
- Gradient descent greedily tries to find the min value of function $f : \mathbb{R}^d \to \mathbb{R}$ by maintaining a vector $\vec{\theta} \in \mathbb{R}^d$ and at each step moving $\vec{\theta}$ "downhill", i.e., in the direction that minimizes directional derivative
- Bounded the number of steps required if f is convex and Lipschitz.
- Simple extensions for optimization over a convex constraint set or online optimization.
- Can typically speed up offline optimization via stochastic gradient descent: requires more iterations but each iteration is faster.
- Lots that we didn't cover: accelerated methods, adaptive methods, second order methods (quasi-Newton methods). Gave mathematical tools to understand these methods. See CS 690OP for more!

EQUIVALENT DEFINITIONS FOR CONVEXITY

We defined convexity of $f : \mathbb{R}^d \to \mathbb{R}$ in two ways:

(1) For all $x, y \in \mathbb{R}^d, \lambda \in [0, 1]$, $\lambda f(x) + (1 - \lambda) f(y) \geq f(\lambda x + (1 - \lambda) y)$.

(2) For all $x, y \in \mathbb{R}^d$, $f(x) \leq f(y) + \langle \nabla f(x), x - y \rangle$

EQUIVALENT DEFINITIONS FOR CONVEXITY

We defined convexity of $f : \mathbb{R}^d \to \mathbb{R}$ in two ways:

(1) For all $x, y \in \mathbb{R}^d, \lambda \in [0,1]$, $\lambda f(x) + (1-\lambda)f(y) \geq f(\lambda x + (1-\lambda)y)$.

(2) For all $x, y \in \mathbb{R}^d$, $f(x) \leq f(y) + \langle \nabla f(x), x - y \rangle$

To see (1) implies (2)

$$
\begin{aligned}
\langle \nabla f(x), y - x \rangle = \lim_{\epsilon \to 0} \frac{f(x + \epsilon(y - x)) - f(x)}{\epsilon} &= \lim_{\epsilon \to 0} \frac{f((1-\epsilon)x + \epsilon y) - f(x)}{\epsilon} \\
&\leq \lim_{\epsilon \to 0} \frac{(1-\epsilon)f(x) + \epsilon f(y) - f(x)}{\epsilon} \\
&= f(y) - f(x)
\end{aligned}
$$

EQUIVALENT DEFINITIONS FOR CONVEXITY

We defined convexity of $f : \mathbb{R}^d \to \mathbb{R}$ in two ways:

(1) For all $x, y \in \mathbb{R}^d, \lambda \in [0, 1]$, $\lambda f(x) + (1 - \lambda)f(y) \geq f(\lambda x + (1 - \lambda)y)$.

(2) For all $x, y \in \mathbb{R}^d$, $f(x) \leq f(y) + \langle \nabla f(x), x - y \rangle$

EQUIVALENT DEFINITIONS FOR CONVEXITY

We defined convexity of $f : \mathbb{R}^d \to \mathbb{R}$ in two ways:

(1) For all $x, y \in \mathbb{R}^d, \lambda \in [0, 1]$, $\lambda f(x) + (1 - \lambda)f(y) \geq f(\lambda x + (1 - \lambda)y)$.

(2) For all $x, y \in \mathbb{R}^d$, $f(x) \leq f(y) + \langle \nabla f(x), x - y \rangle$

To see (2) implies (1)

$$f(\lambda x + (1 - \lambda)y) \leq f(x) + \langle \nabla f(\lambda x + (1 - \lambda)y), \lambda x + (1 - \lambda)y - x \rangle$$

$$f(\lambda x + (1 - \lambda)y) \leq f(y) + \langle \nabla f(\lambda x + (1 - \lambda)y), \lambda x + (1 - \lambda)y - y \rangle$$

λ times the first equation plus $(1 - \lambda)$ times the second equation gives

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