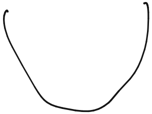
COMPSCI 514: Algorithms for Data Science

Cameron Musco
University of Massachusetts Amherst. Fall 2022.
Lecture 25 (Final Lecture!)

Logistics

- Problem Set 5 is due Dec 12 at 11:59pm.
- Exam is next Wednesday Dec 14, from 10:30am-12:30pm in class.
- I am holding office hours Friday 12/9 2:30-4:30pm and Monday 12/12 10am-12m. Both will be held in LGRC A215.
- It would be really helpful if you could fill out SRTIs for this class (they close Dec 23).
- `http://owl.umass.edu/partners/courseEvalSurvey/uma/`.

Summary



Last Class:

- Analysis of gradient descent for convex and Lipschitz functions.

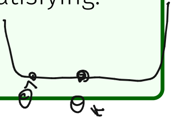
This Class:

- Extend gradient descent to constrained optimization via projected gradient descent.
- Course wrap up and review.

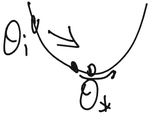
GD Analysis Proof

Theorem – GD on Convex Lipschitz Functions: For convex G-Lipschitz function f, GD run with $t \geq \frac{R^2G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of $\vec{\theta}_*$, outputs $\hat{\theta}$ satisfying:

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



Step 1: For all i, $f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{\|\vec{\theta}_i - \vec{\theta}_*\|_2^2 - \|\vec{\theta}_{i+1} - \vec{\theta}_*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$



GD Analysis Proof

Theorem – GD on Convex Lipschitz Functions: For convex G-Lipschitz function f, GD 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 $\vec{\theta}_*$, outputs $\hat{\theta}$ satisfying:

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

Step 1: For all i, $f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{\|\vec{\theta}_i - \vec{\theta}_*\|_2^2 - \|\vec{\theta}_{i+1} - \vec{\theta}_*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$ $\Longrightarrow$ } convexity Lipschitz

Step 2: $\frac{1}{t}\sum_{i=1}^{t} f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{R^2}{2\eta \cdot t} + \frac{\eta G^2}{2}$. (telescoping sum)

algebra

$\leq \epsilon$

Constrained Convex Optimization

Often want to perform convex optimization with convex constraints.

$$\vec{\theta}^* = \arg\min_{\vec{\theta} \in \mathcal{S}} f(\vec{\theta}),$$

where $\mathcal{S}$ is a convex set.

Constrained Convex Optimization

Often want to perform convex optimization with convex constraints.

$$\vec{\theta}^* = \underset{\vec{\theta} \in \mathcal{S}}{\arg\min} f(\vec{\theta}),$$

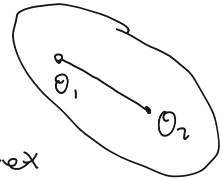
$\mathbb{Z}$ is not convex

$\mathbb{R}$ is convex

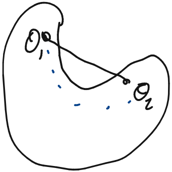
where $\mathcal{S}$ is a convex set.

Definition – Convex Set: A set $\mathcal{S} \subseteq \mathbb{R}^d$ is convex if and only if, for any $\vec{\theta}_1, \vec{\theta}_2 \in \mathcal{S}$ and $\lambda \in [0, 1]$:

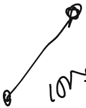
$$(1 - \lambda)\vec{\theta}_1 + \lambda \cdot \vec{\theta}_2 \in \mathcal{S}$$



convex



non convex



convex

Constrained Convex Optimization

Often want to perform convex optimization with convex constraints.

$$\vec{\theta}^* = \arg\min_{\vec{\theta} \in \mathcal{S}} f(\vec{\theta}),$$

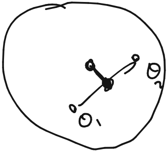
where $\mathcal{S}$ is a convex set.

Definition – Convex Set: A set $\mathcal{S} \subseteq \mathbb{R}^d$ is convex if and only if, for any $\vec{\theta}_1, \vec{\theta}_2 \in \mathcal{S}$ and $\lambda \in [0, 1]$:

$$(1 - \lambda)\vec{\theta}_1 + \lambda \cdot \vec{\theta}_2 \in \mathcal{S}$$

E.g. $\mathcal{S} = \{\vec{\theta} \in \mathbb{R}^d : \|\vec{\theta}\|_2 \leq 1\}$.

$\|\theta_1\|_2 \leq 1 \ , \ \|\theta_2\| \leq 1$



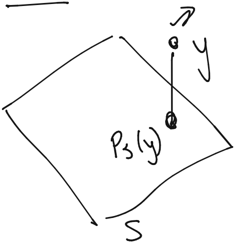
$\|(1-\lambda)\theta_1 + \lambda\theta_2\|_2 \leq 1$

(triangle inequality

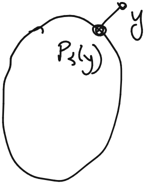
Projected Gradient Descent

For any convex set let $P_{\mathcal{S}}(\cdot)$ denote the projection function onto $\mathcal{S}$.

- $P_{\mathcal{S}}(\vec{y}) = \arg\min_{\vec{\theta} \in \mathcal{S}} \|\vec{\theta} - \vec{y}\|_2$.



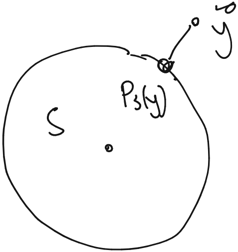
W^T



Projected Gradient Descent

For any convex set let $P_{\mathcal{S}}(\cdot)$ denote the projection function onto $\mathcal{S}$.

- $P_{\mathcal{S}}(\vec{y}) = \arg\min_{\vec{\theta} \in \mathcal{S}} \|\vec{\theta} - \vec{y}\|_2$.
- For $\mathcal{S} = \{\vec{\theta} \in \mathbb{R}^d : \|\vec{\theta}\|_2 \leq 1\}$ what is $P_{\mathcal{S}}(\vec{y})$?



$$P_S(y) = \frac{y}{\|y\|_2}$$

Projected Gradient Descent

For any convex set let $P_{\mathcal{S}}(\cdot)$ denote the projection function onto $\mathcal{S}$.

- $P_{\mathcal{S}}(\vec{y}) = \arg\min_{\vec{\theta} \in \mathcal{S}} \|\vec{\theta} - \vec{y}\|_2$.
- For $\mathcal{S} = \{\vec{\theta} \in \mathbb{R}^d : \|\vec{\theta}\|_2 \leq 1\}$ what is $P_{\mathcal{S}}(\vec{y})$?
- For $\mathcal{S}$ being a k dimensional subspace of $\mathbb{R}^d$, what is $P_{\mathcal{S}}(\vec{y})$?

V has orthonormal cols. spanning S.

$P_S(y) = VV^T y$

Projected Gradient Descent

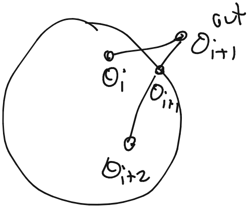
For any convex set let $P_{\mathcal{S}}(\cdot)$ denote the projection function onto $\mathcal{S}$.

- $P_{\mathcal{S}}(\vec{y}) = \arg\min_{\vec{\theta} \in \mathcal{S}} \|\vec{\theta} - \vec{y}\|_2$.
- For $\mathcal{S} = \{\vec{\theta} \in \mathbb{R}^d : \|\vec{\theta}\|_2 \le 1\}$ what is $P_{\mathcal{S}}(\vec{y})$?
- For $\mathcal{S}$ being a k dimensional subspace of $\mathbb{R}^d$, what is $P_{\mathcal{S}}(\vec{y})$?

Projected Gradient Descent

- Choose some initialization $\vec{\theta}_1$ and set $\eta = \frac{R}{G\sqrt{t}}$.
- For $i = 1, \ldots, t-1$
  - $\vec{\theta}_{i+1}^{(out)} = \vec{\theta}_i - \eta \cdot \vec{\nabla} f(\vec{\theta}_i)$
  - $\vec{\theta}_{i+1} = P_{\mathcal{S}}(\vec{\theta}_{i+1}^{(out)})$.
- Return $\hat{\theta} = \arg\min_{\vec{\theta}_i} f(\vec{\theta}_i)$.

never "loses" progress



Convex Projections

Projected gradient descent can be analyzed identically to gradient descent!

Convex Projections

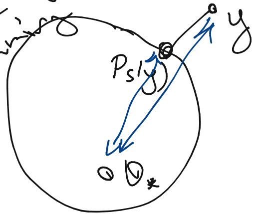
Projected gradient descent can be analyzed identically to gradient descent!

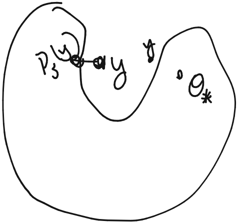
Theorem – Projection to a convex set: For any convex set $\mathcal{S} \subseteq \mathbb{R}^d$, $\vec{y} \in \mathbb{R}^d$, and $\vec{\theta} \in \mathcal{S}$,

$$\|P_{\mathcal{S}}(\vec{y}) - \vec{\theta}\|_2 \leq \|\vec{y} - \vec{\theta}\|_2.$$

linear programming

integer programming





Projected Gradient Descent Analysis

Theorem – Projected GD: For convex G-Lipschitz function f, and convex set $\mathcal{S}$, Projected GD 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 $\vec{\theta}_*$, outputs $\hat{\theta}$ satisfying:

$\theta_1, \ldots, \theta_t \in S$

$$f(\hat{\theta}) \leq f(\vec{\theta}_*) + \epsilon = \min_{\vec{\theta} \in \mathcal{S}} f(\vec{\theta}) + \epsilon$$

Projected Gradient Descent Analysis

Theorem – Projected GD: For convex G-Lipschitz function f, and convex set $\mathcal{S}$, Projected GD run with $t \geq \frac{R^2G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of $\vec{\theta}_*$, outputs $\hat{\theta}$ satisfying:

$$f(\hat{\theta}) \leq f(\vec{\theta}_*) + \epsilon = \min_{\vec{\theta} \in \mathcal{S}} f(\vec{\theta}) + \epsilon$$

Recall: $\vec{\theta}_{i+1}^{(out)} = \vec{\theta}_i - \eta \cdot \vec{\nabla} f(\vec{\theta}_i)$ and $\vec{\theta}_{i+1} = P_{\mathcal{S}}(\vec{\theta}_{i+1}^{(out)})$.

Theorem – Projected GD: For convex G-Lipschitz function f, and convex set $\mathcal{S}$, Projected GD run with $t \geq \frac{R^2G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of $\vec{\theta}_*$, outputs $\hat{\theta}$ satisfying:

$$f(\hat{\theta}) \leq f(\vec{\theta}_*) + \epsilon = \min_{\vec{\theta} \in \mathcal{S}} f(\vec{\theta}) + \epsilon$$

Recall: $\vec{\theta}_{i+1}^{(out)} = \vec{\theta}_i - \eta \cdot \vec{\nabla} f(\vec{\theta}_i)$ and $\vec{\theta}_{i+1} = P_{\mathcal{S}}(\vec{\theta}_{i+1}^{(out)})$.

Step 1: For all i, $f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{\|\vec{\theta}_i - \theta_*\|_2^2 - \|\vec{\theta}_{i+1}^{(out)} - \vec{\theta}_*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$.

Theorem – Projected GD: For convex G-Lipschitz function f, and convex set $\mathcal{S}$, Projected GD run with $t \geq \frac{R^2G^2}{\epsilon^2}$ iterations, $\eta = \frac{R}{G\sqrt{t}}$, and starting point within radius R of $\vec{\theta}_*$, outputs $\hat{\theta}$ satisfying:

$$f(\hat{\theta}) \leq f(\vec{\theta}_*) + \epsilon = \min_{\vec{\theta} \in \mathcal{S}} f(\vec{\theta}) + \epsilon$$

Recall: $\vec{\theta}_{i+1}^{(out)} = \vec{\theta}_i - \eta \cdot \vec{\nabla} f(\vec{\theta}_i)$ and $\vec{\theta}_{i+1} = P_{\mathcal{S}}(\vec{\theta}_{i+1}^{(out)})$.

Step 1: For all i, $f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{\|\vec{\theta}_i - \theta_*\|_2^2 - \|\vec{\theta}_{i+1}^{(out)} - \vec{\theta}_*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$.

Step 1.a: For all i, $f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{\|\vec{\theta}_i - \vec{\theta}_*\|_2^2 - \|\vec{\theta}_{i+1} - \vec{\theta}_*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$.

$* \; \|\theta_{i+1} - \theta_*\|_2^2 \leq \|\theta_{i+1}^{out} - \theta_*\|_2^2$

Projected Gradient Descent Analysis

Theorem – Projected GD: For convex G-Lipschitz function f, and convex set $\mathcal{S}$, Projected GD 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 $\vec{\theta}_*$, outputs $\hat{\theta}$ satisfying:

$$f(\hat{\theta}) \leq f(\vec{\theta}_*) + \epsilon = \min_{\vec{\theta} \in \mathcal{S}} f(\vec{\theta}) + \epsilon$$

Recall: $\vec{\theta}_{i+1}^{(out)} = \vec{\theta}_i - \eta \cdot \vec{\nabla} f(\vec{\theta}_i)$ and $\vec{\theta}_{i+1} = P_{\mathcal{S}}(\vec{\theta}_{i+1}^{(out)})$.

Step 1: For all i, $f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{\|\vec{\theta}_i - \theta_*\|_2^2 - \|\vec{\theta}_{i+1}^{(out)} - \vec{\theta}_*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$.

Step 1.a: For all i, $f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{\|\vec{\theta}_i - \vec{\theta}_*\|_2^2 - \|\vec{\theta}_{i+1} - \vec{\theta}_*\|_2^2}{2\eta} + \frac{\eta G^2}{2}$.

Step 2: $\frac{1}{t}\sum_{i=1}^{t} f(\vec{\theta}_i) - f(\vec{\theta}_*) \leq \frac{R^2}{2\eta \cdot t} + \frac{\eta G^2}{2}$ $\implies$ Theorem.

$\leq \epsilon$

online
grad descent
slides:

Course Review

Randomized Methods

Randomization as a computational resource for massive datasets.

Randomized Methods

Randomization as a computational resource for massive datasets.

- Focus on problems that are easy on small datasets but hard at massive scale – set size estimation, load balancing, distinct elements counting (MinHash), checking set membership (Bloom Filters), frequent items counting (Count-min sketch), near neighbor search (locality sensitive hashing).

Randomized Methods

Randomization as a computational resource for massive datasets.

- Focus on problems that are easy on small datasets but hard at massive scale – set size estimation, load balancing, distinct elements counting (MinHash), checking set membership (Bloom Filters), frequent items counting (Count-min sketch), near neighbor search (locality sensitive hashing).
- Just the tip of the iceberg on randomized streaming/sketching/hashing algorithms. Check out 690RA if you want to learn more.

Randomized Methods

Randomization as a computational resource for massive datasets.

- Focus on problems that are easy on small datasets but hard at massive scale – set size estimation, load balancing, distinct elements counting (MinHash), checking set membership (Bloom Filters), frequent items counting (Count-min sketch), near neighbor search (locality sensitive hashing).
- Just the tip of the iceberg on randomized streaming/sketching/hashing algorithms. Check out 690RA if you want to learn more.
- In the process covered probability/statistics tools that are very useful beyond algorithm design: concentration inequalities, higher moment bounds, law of large numbers, central limit theorem, linearity of expectation and variance, union bound, median as a robust estimator.

Dimensionality Reduction

Methods for working with (compressing) high-dimensional data

Dimensionality Reduction

Methods for working with (compressing) high-dimensional data

- Started with randomized dimensionality reduction and the JL lemma: compression from *any* d-dimensions to $O(\log n/\epsilon^2)$ dimensions while preserving pairwise distances.

Dimensionality Reduction

Methods for working with (compressing) high-dimensional data

- Started with randomized dimensionality reduction and the JL lemma: compression from *any* d-dimensions to $O(\log n/\epsilon^2)$ dimensions while preserving pairwise distances.
- Connections to the weird geometry of high-dimensional space.

Dimensionality Reduction

Methods for working with (compressing) high-dimensional data

- Started with randomized dimensionality reduction and the JL lemma: compression from *any* d-dimensions to $O(\log n/\epsilon^2)$ dimensions while preserving pairwise distances.
- Connections to the weird geometry of high-dimensional space.
- Dimensionality reduction via low-rank approximation and optimal solution with PCA/eigendecomposition/SVD.

Dimensionality Reduction

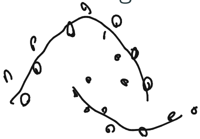
Methods for working with (compressing) high-dimensional data

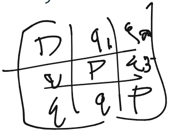
- Started with randomized dimensionality reduction and the JL lemma: compression from *any* d-dimensions to $O(\log n/\epsilon^2)$ dimensions while preserving pairwise distances.
- Connections to the weird geometry of high-dimensional space.
- Dimensionality reduction via low-rank approximation and optimal solution with PCA/eigendecomposition/SVD.
- Low-rank approximation of similarity matrices and entity embeddings (e.g., LSA, word2vec, DeepWalk).

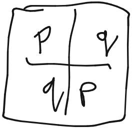
Dimensionality Reduction

Methods for working with (compressing) high-dimensional data

- Started with randomized dimensionality reduction and the JL lemma: compression from *any* d-dimensions to $O(\log n/\epsilon^2)$ dimensions while preserving pairwise distances.
- Connections to the weird geometry of high-dimensional space.
- Dimensionality reduction via low-rank approximation and optimal solution with PCA/eigendecomposition/SVD.
- Low-rank approximation of similarity matrices and entity embeddings (e.g., LSA, word2vec, DeepWalk).
- Spectral graph theory – nonlinear dimension reduction and spectral clustering for community detection.







Dimensionality Reduction

Methods for working with (compressing) high-dimensional data

- Started with randomized dimensionality reduction and the JL lemma: compression from *any* d-dimensions to $O(\log n/\epsilon^2)$ dimensions while preserving pairwise distances.
- Connections to the weird geometry of high-dimensional space.
- Dimensionality reduction via low-rank approximation and optimal solution with PCA/eigendecomposition/SVD.
- Low-rank approximation of similarity matrices and entity embeddings (e.g., LSA, word2vec, DeepWalk).
- Spectral graph theory – nonlinear dimension reduction and spectral clustering for community detection.
- In the process covered linear algebraic tools that are very broadly useful in ML and data science: eigendecomposition, singular value decomposition, projection, norm transformations.

$y^T y = \|y\|_2^2$

$tr(A^T A) = \|A\|_F^2$

Continuous Optimization

Foundations of continuous optimization and gradient descent.

Continuous Optimization

Foundations of continuous optimization and gradient descent.

- Foundational concepts like convexity, convex sets, Lipschitzness, directional derivative/gradient.

Continuous Optimization

Foundations of continuous optimization and gradient descent.

- Foundational concepts like convexity, convex sets, Lipschitzness, directional derivative/gradient.
- How to analyze gradient descent in a simple setting (convex Lipschitz functions).

Continuous Optimization

Foundations of continuous optimization and gradient descent.

- Foundational concepts like convexity, convex sets, Lipschitzness, directional derivative/gradient.
- How to analyze gradient descent in a simple setting (convex Lipschitz functions).
- Simple extension to projected gradient descent for optimization over a convex constraint set.

Continuous Optimization

Foundations of continuous optimization and gradient descent.

- Foundational concepts like convexity, convex sets, Lipschitzness, directional derivative/gradient.
- How to analyze gradient descent in a simple setting (convex Lipschitz functions).
- Simple extension to projected gradient descent for optimization over a convex constraint set.
- Lots that we didn't cover: online and stochastic gradient descent, accelerated methods, adaptive methods, second order methods (quasi-Newton methods), practical considerations. Gave mathematical tools to understand these methods.

Thanks for a great semester!

Final Exam Questions/Review

$$\text{rank}(AB) \le \min(\text{rank}(A), \text{rank}(B))$$

① $\text{rank}(AB) \le \text{rank}(A)$

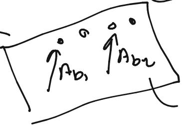
and

$\text{rank}(AB) \le \text{rank}(B)$

$\underline{AB}$

$$\begin{bmatrix} & & \\ & A & \\ & & \end{bmatrix} \underbrace{\begin{bmatrix} | & | & & | \\ b_1 & b_2 & \cdots & b_d \\ | & | & & | \end{bmatrix}}_{B} = \overbrace{\begin{bmatrix} | & | & & | \\ Ab_1 & Ab_2 & \cdots & Ab_d \\ | & | & & | \end{bmatrix}}^{AB}$$

col span A



rank(A) dimensions

$\text{rank}(AI) = \text{rank}(A)$

$$\mathrm{rank}(A+B) \le \mathrm{rank}(A) + \mathrm{rank}(B)$$

0

$$\overset{A}{\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}} + \overset{B}{\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$A = -B$

$A + B = 0$

$$\underbrace{\begin{bmatrix} | & | & & | \\ a_1+b_1 & a_2+b_2 & & \\ | & | & & | \end{bmatrix}}_{A+B}$$

rank(A) dim space

rank(B) dim space

$$\mathrm{rank}(A+B+C) \cancel{\le \mathrm{rank}(A) + \mathrm{rank}(B)} + \mathrm{rank}(C)$$

$\mathrm{rank}(AB) \neq \mathrm{rank}(BA)$

$\underbrace{\mathrm{rank}(VV^T)}_{= k}$ $\underbrace{\mathrm{rank}(V^TV)}_{k \times k \text{ identity}}$

$\mathrm{rank}(XX^T) = \mathrm{rank}(X^TX)$