Optimization for Data Science Lecture 10: Stochastic Gradient and Sub-Gradient

Kimon Fountoulakis

School of Computer Science
University of Waterloo

10/10/2019

Outline of this lecture

- Recap: ML loss functions, stochastic gradient, iteration complexity
- Sub-gradients, sub-gradient calculus
- Stochastic sub-gradient

Data

- We are given n data points (a_i, b_i) $i = 1,\ldots,n$.

- $a_i \in \mathbb{R}^d$ is a sample/vector of length d, i.e., for each sample we have d features.

- b_i are the given labels. It can be real or binary or more generally an integer.

Finite-sum optimization problems

- We consider the following optimization problem

$$\min_{x \in \mathbb{R}^n} \underbrace{\frac{1}{n} \sum_{i=1}^{n} f_i(x)}_{f(x)} = \mathbb{E}[f_i(x)] = f(x)$$

Examples of Loss Functions

- **Squared error:** $f_i(x) := \frac{1}{2}(a_i^T x - b_i)^2$, where $a_i \in \mathbb{R}^d$ is the i-th sample with d features and $b_i \in \mathbb{R}$ is the label.

- **Absolute error:** $|a_i^T x - b_i|$, robust to outliers.

- **Hinge loss:** $\max\{0, 1 - b_i a_i^T x\}$, better for binary labels b_i

- **Logistic loss:** $\log(1 + \exp(-b_i a_i^T x))$, better for binary labels b_i and also it's a smooth function.

Stochastic Gradient Method

- Assuming that each f_i is differentiable then stochastic gradient is equivalent to:

- Pick randomly a sample i

- $x_{k+1} := x_k - \alpha_k \nabla f_i(x)$

- iteration cost is now independent of n, but how many iterations?

Assumptions for analyzing stochastic gradient for non-convex functions

- f is bounded below
- Each f_i is differentiable
- $\nabla f(x)$ is Lipschitz continuous
- $\mathbb{E}[\|\nabla f_i(x_k)\|_2^2] \leq B^2$, where B^2 is a constant.

Assumptions for analyzing stochastic gradient for convex functions

- f is convex
- Each f_i is differentiable
- $\mathbb{E}[\|\nabla f_i(x_k)\|_2^2] \leq B^2$, where B^2 is a constant.

Convergence rate of stochastic gradient for smooth functions

- For non-convex and convex functions we have the following:
- If $\alpha_k = 1/k$, then we get rate $\mathcal{O}(1/\log t)$
- If $\alpha_k = 1/\sqrt{k}$, then we get rate $\mathcal{O}(\log t/\sqrt{t})$
- If $\alpha_k = \alpha$ for some constant α, then we get rate $\mathcal{O}(1/t + \alpha)$. This means implies that the minimum expected norm of the gradient will never go to zero and then algorithm only converges to a neighborhood of a stationary point. Also, this rate appears to be converging sub-linearly initially, but then stagnates!

Assumptions for analyzing stochastic gradient for strongly-convex

- f is strongly-convex
- Each f_i is differentiable
- $\mathbb{E}[\|\nabla f_i(x_k)\|_2^2] \leq B^2$, where B^2 is a constant.
- The objective function is Lipschitz continuous $|f(x) - f(y)| \leq G\|x - y\|_2$, where $G > 0$ is a constant. This assumption also implies that $\|\nabla f(x)\|_2 \leq G$. This assumption is used only for the result with step-size $\alpha_k = 1/(\delta k)$.

Convergence rate of stochastic gradient for smooth functions

- For δ-strongly-convex functions we have the following:

- If $\alpha_k = 1/(\delta k)$, then we get rate $\mathcal{O}(1/t)$. This implies that you have to know the strong-convexity constant.

- If $\alpha_k = \alpha$ for some constant α, then we get rate $\mathcal{O}((1-(\delta/L))^t + \alpha)$. This means that the algorithm converges to a neighborhood of a minimizer. Also, this rate appears to be converging linearly initially, but then stagnates!

Iteration complexity for smooth functions

- The iteration complexity results for stochastic gradient in the next slide correspond to step-size $\alpha_k := 1/k$ for non-convex and convex functions. Different iteration complexity results are obtained if you use other techniques for setting α_k.

- For δ-strongly convex functions and stochastic gradient the result in the next slide is obtained by setting $\alpha_k = 1/(\delta k)$.

Iteration complexity for smooth functions

	Gradient Descent	Accelerated Gradient	Stochastic Gradient
Non-convex	$\mathcal{O}\left(\frac{L}{\epsilon}\right)$	-	$\mathcal{O}\left(e^{\frac{LB^2}{\epsilon}}\right)$
Convex	$\mathcal{O}\left(\frac{L}{\epsilon}\right)$	$\mathcal{O}\left(\sqrt{\frac{L}{\epsilon}}\right)$	$\mathcal{O}\left(e^{\frac{B^2}{\epsilon}}\right)$
Strongly convex	$\mathcal{O}\left(\frac{L}{\delta}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\sqrt{\frac{L}{\delta}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{GB^2}{\delta^2}\frac{1}{\epsilon}\right)$

Output of stochastic gradient for convex and strongly-convex functions

- For convex and strongly-convex functions theory suggests that the output of stochastic gradient should be the average over all iterations, i.e., $\bar{x}_k := \dfrac{1}{k+1}\displaystyle\sum_{i=0}^{k} x_i$.

Why would we consider stochastic methods?

- For two reasons
  - Extremely cheap iteration cost. Very good when we have millions (or more) data points and each data point is low-dimensional. This is good when we do not want to access all data points at each iteration.
  - Fast convergence to low accuracy. For most AI/ML problems a low accuracy solution to the optimization problem is good enough to obtain high accuracy w.r.t domain metrics of measuring performance, e.g., precision/recall.

Preferred step-sizes

- Practitioners prefer to use constant step-sizes for two reasons:
  - Fast convergence to low accuracy solutions. Remember the rate is $\mathcal{O}(1/t + \alpha)$ for non-convex and convex functions, and $\mathcal{O}((1 - (\delta/L))^t + \alpha)$ for δ-strongly convex.
  - For most AI/ML problems a low accuracy solution to the optimization problem is good enough to obtain high accuracy w.r.t domain metrics of measuring performance, e.g., precision/recall.

Termination Criterion of Stochastic Gradient

- Usually we terminate an algorithm if the norm of the gradient is small.
- However, in this algorithm we only access one sample per iteration. This means that we cannot compute the norm of the whole gradient.
- If we do compute the norm of the whole gradient at each iteration, then this defeats the purpose of stochastic gradient.

Some options for terminating stochastic gradient

- Predefined maximum number of iterations
- Predefined upper bound for the running time
- Stop when the norm of the gradient of the chosen sample is small.
- Measure the norm of the whole gradient every n iterations, if it is small then terminate.
- Use your validation data and terminate the algorithm when precision/recall (or your preferred metric) are large enough.
- Measure validation error every n iterations. Stop if the validation error starts overfitting

Non-smooth loss functions

- What if the loss functions f_i in the optimization problem below are non-smooth?

$$\min_{x\in\mathbb{R}^n} \underbrace{\frac{1}{n}\sum_{i=1}^{n} f_i(x)}_{f(x)} = \mathbb{E}[f_i(x)] = f(x)$$

- We have two options
  - Smooth the loss functions and then use stochastic-gradient (previous lectures)
  - Use stochastic sub-gradient method (this lecture)

Basic inequality

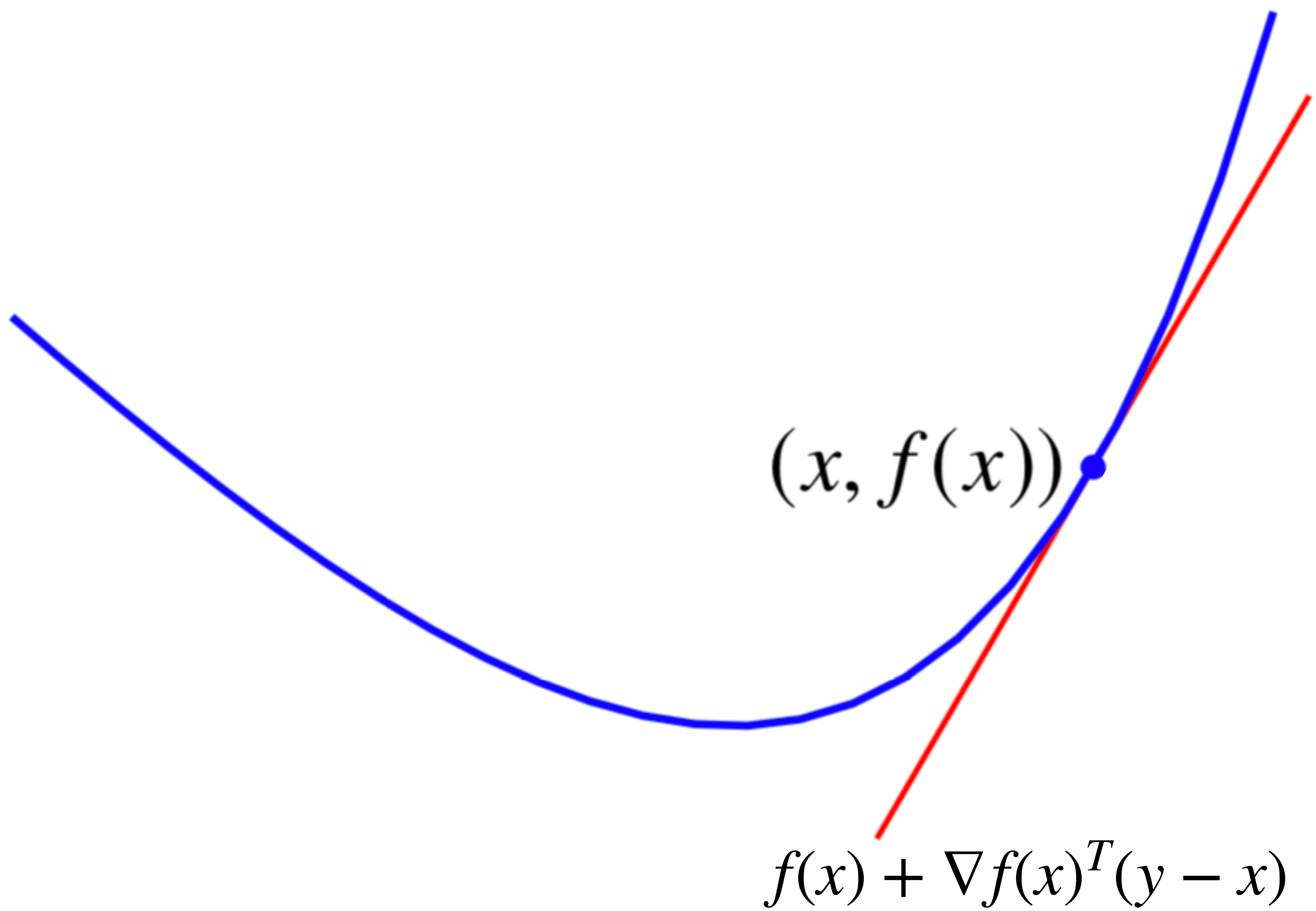
- Recall that the definition of convexity is:
  $f(y) \geq f(x) + \nabla f(x)^T (y - x)$

- First-order approximation of f (right-hand-side) is a global underestimator.

- What if f is not differentiable?

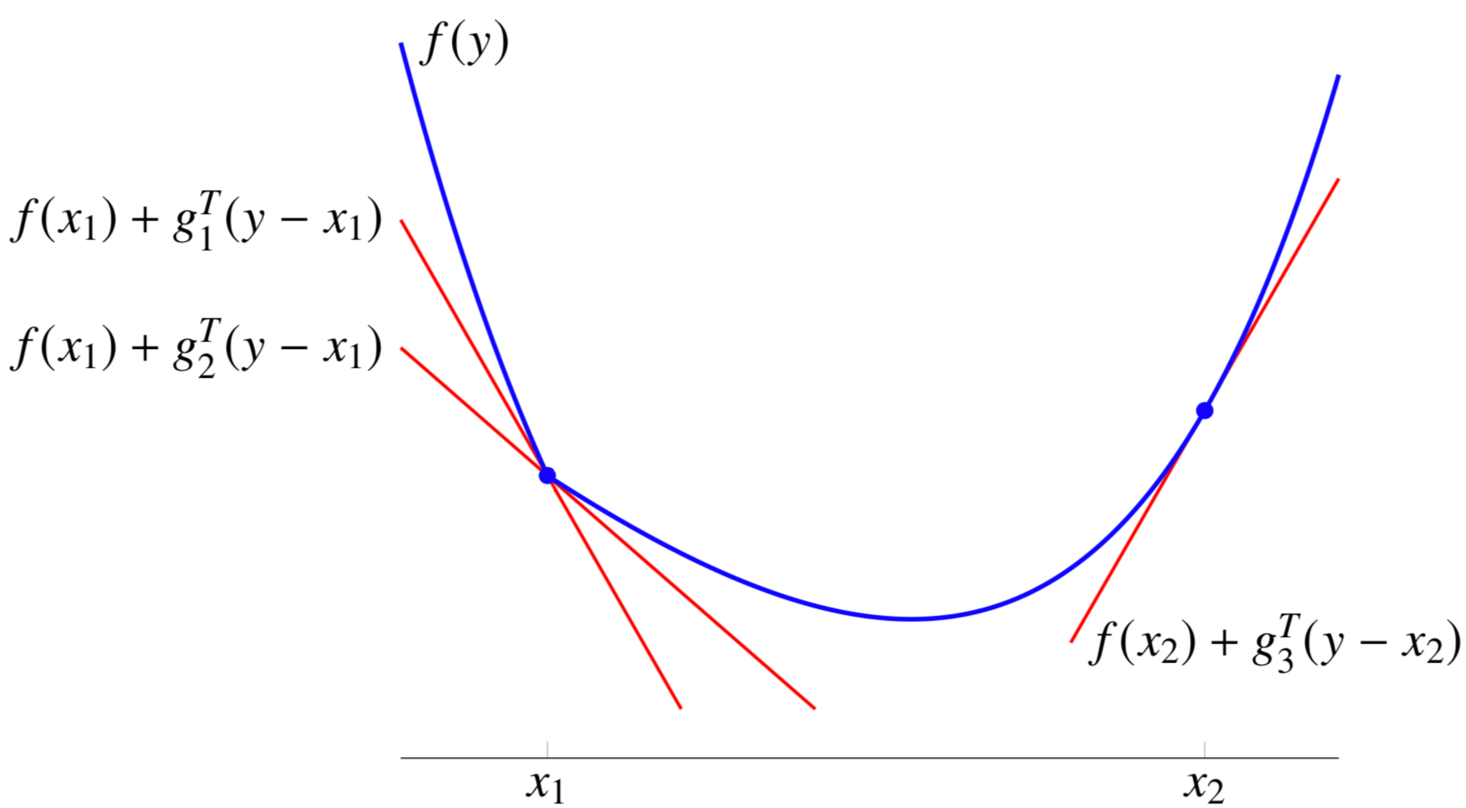
This part of the lecture is mostly based on L. Vandenberghe and S. Boyd lecture on sub-gradients

Visualization: lower bound of convex smooth function



Sub-gradient of function

- A vector g is a sub-gradient of a function f at point x if $f(y) \geq f(x) + g^T(y - x) \quad \forall y \in \mathbb{R}^n$

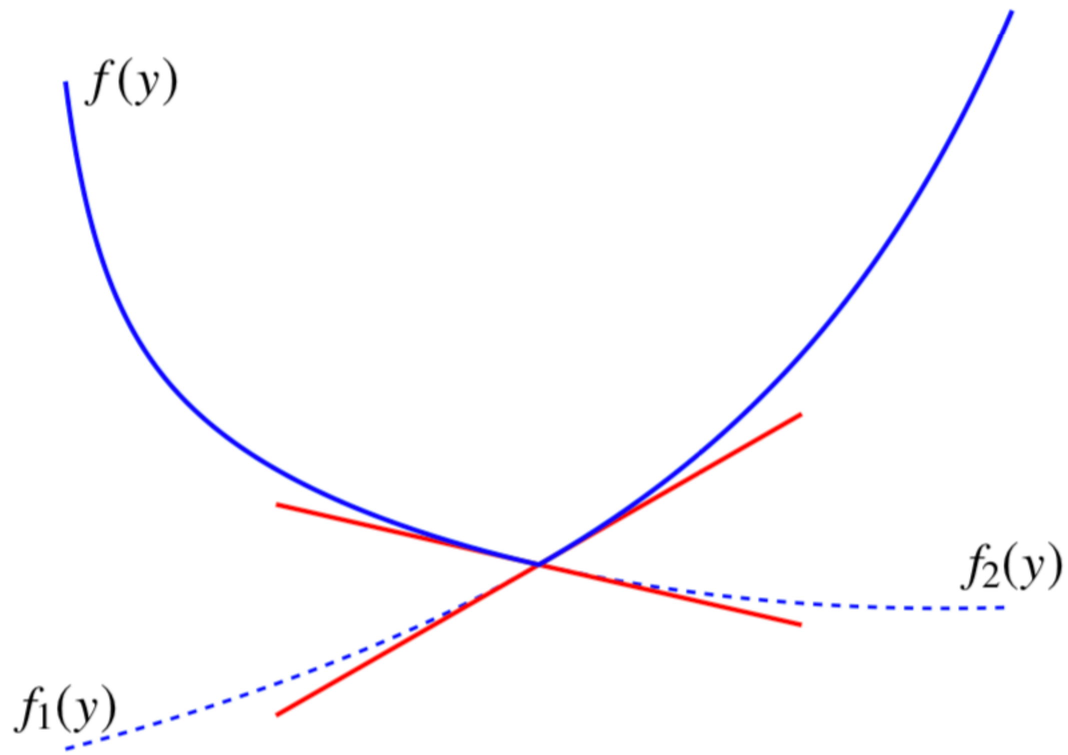


Sub-differential for convex functions

- We define the **sub-differential set at x as:** $\partial f(x) := \{g \in \mathbb{R}^n \mid f(y) \geq f(x) + g^T(y - x) \forall y \in \mathbb{R}^n\}$

Example

- Define $f(x) := \max(f_1(x), f_2(x))$, where f_1, f_2 are convex and differentiable.



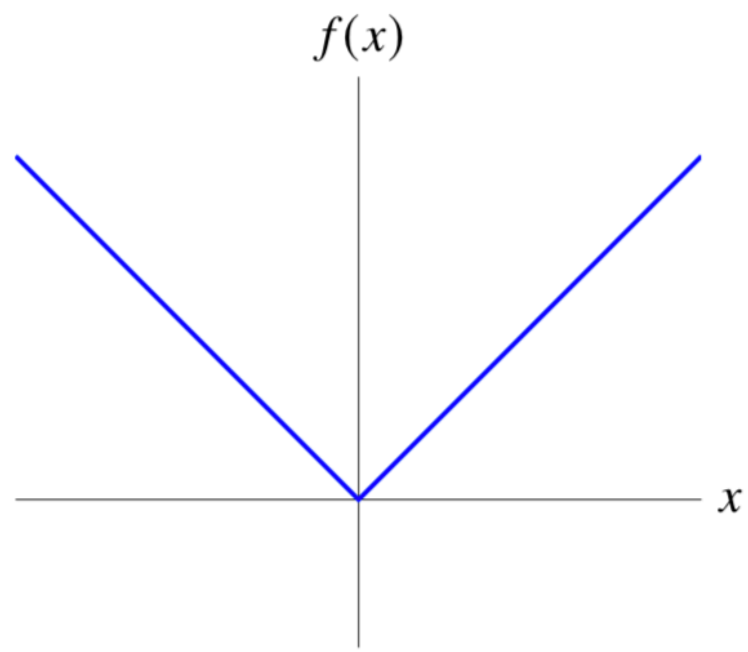
- If $f_1(x) = f_2(x)$ for some x, then
  $\partial f(x) := \left\{ g \in \mathbb{R}^n \mid t\,\nabla f_1(x) + (1-t)\,\nabla f_2(x)\ \forall t \in [0,1] \right\}$

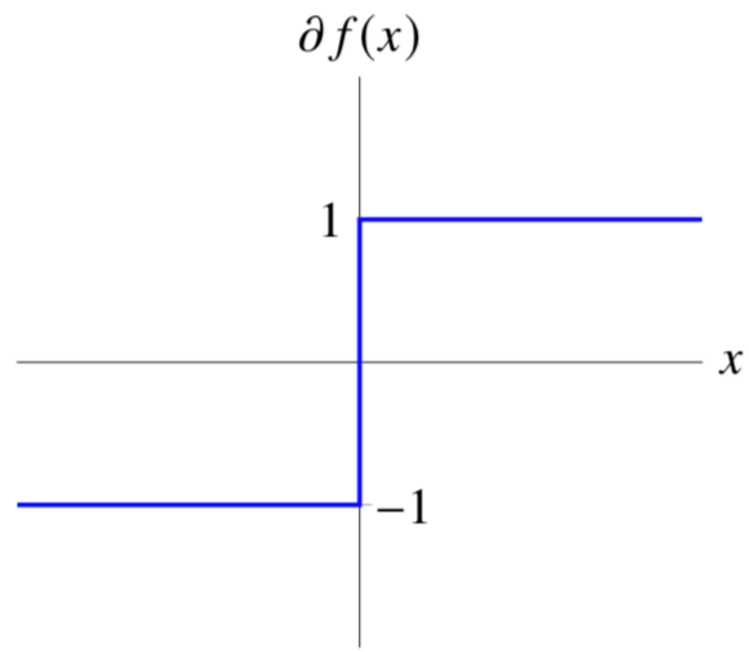
- If $f_1(x) > f_2(x)$ for some x, then $\partial f(x) := \nabla f_1(x)$

- If $f_1(x) < f_2(x)$ for some x, then $\partial f(x) := \nabla f_2(x)$

Example

- Define $f(x) := |x|$





- $\partial f(x) = 1$ if $x > 0$
- $\partial f(x) = -1$ if $x < 0$
- $\partial f(x) \in [-1,1]$ if $x = 0$

Convex and strongly-convex non-smooth functions

- A non-smooth function f is convex if
$$f(y) \geq f(x) + g(x)^T(y - x) \quad \forall x, y \in \mathbb{R}^n \text{ and } \forall g(x) \in \partial f(x)$$

- A non-smooth function f is δ-strongly convex if
$$f(y) \geq f(x) + g(x)^T(y - x) + \frac{\delta}{2}\|y - x\|_2^2 \quad \forall x, y \in \mathbb{R}^n \text{ and } \forall g(x) \in \partial f(x)$$

First-order conditions for non-smooth convex functions

- A point x^* is a minimizer iff $0 \in \partial f(x^*)$.
- This means that there exists a sub-gradient at x^* which is equal to zero.
- This follows directly from the definition of sub-gradient:
$f(y) \geq f(x^*) + 0^T(y - x^*)\ \forall y \iff 0 \in \partial f(x^*)$

Sub-gradient calculus

- **Weak sub-gradient calculus:** rules for finding one sub-gradient
  - Sufficient for most algorithms for non-smooth convex problems

- **Strong sub-gradient calculus:** rules for finding $\partial f(x)$ (all sub-gradients)
  - Some algorithms, optimality conditions need the entire sub-differential.
  - Usually, it is difficult to compute

Basic rules

- Smooth functions $\partial f(x) := \nabla f(x)$

- Non-negative linear combination: if $f(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x)$ with $\alpha_1, \alpha_2 \geq 0$, then $\partial f(x) := \alpha_1 \partial f_1(x) + \alpha_2 \partial f_2(x)$

- Affine transformation of variables: if $f(x) = h(Ax + b)$, then $\partial f(x) := A^T \partial h(Ax + b)$ (so far we used ∂h to denote a set, here we abuse notation to denote a sub-gradient in ∂h).

Pointwise maximum

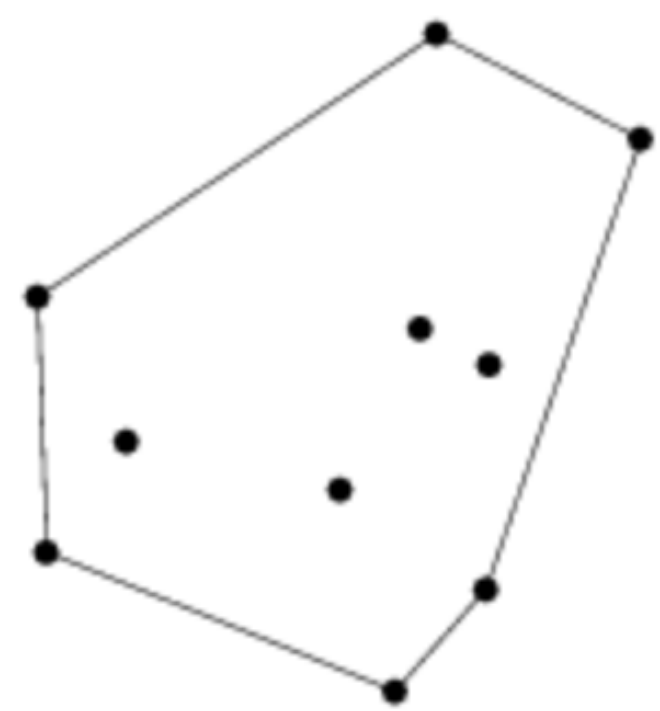
- Let $f(x) := \max\{f_1(x), f_2(x), \ldots, f_m(x)\}$ and f_i be differentiable for any i.

- Define $I(x) := \{i \mid f_i(x) = f(x)\}$, which denotes the active function f_i at x

- **Weak result:** To compute a sub-gradient at x, choose any $i \in I(x)$, the sub-gradient is $\nabla f_i(x)$.

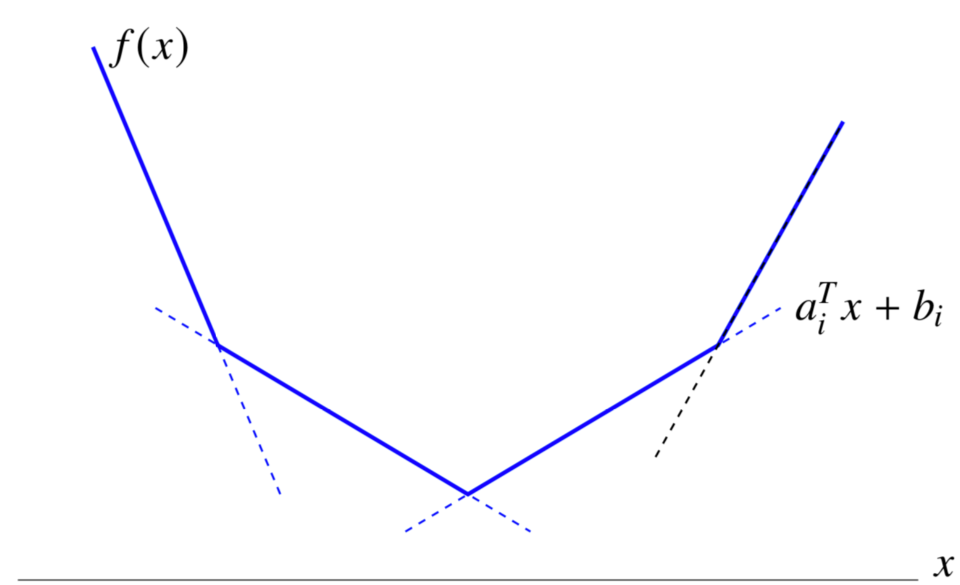
Pointwise maximum

- **Strong result:** $\partial f(x) := \text{conv}\{\ \nabla f_i(x) \mid i \in I(x)\}$
- The convex hull of a set of points is defined as all the points inside the polytope:



Example: piece-wise linear function

- Let $f(x) := \max_{i=1,\ldots,m} \{a_i^T x + b_i\}$



- $\partial f(x) := \text{conv}\{\alpha_i \mid i \in I(x)\}$

- where $I(x) = \{i \mid a_i^T x + b_i = f(x)\}$

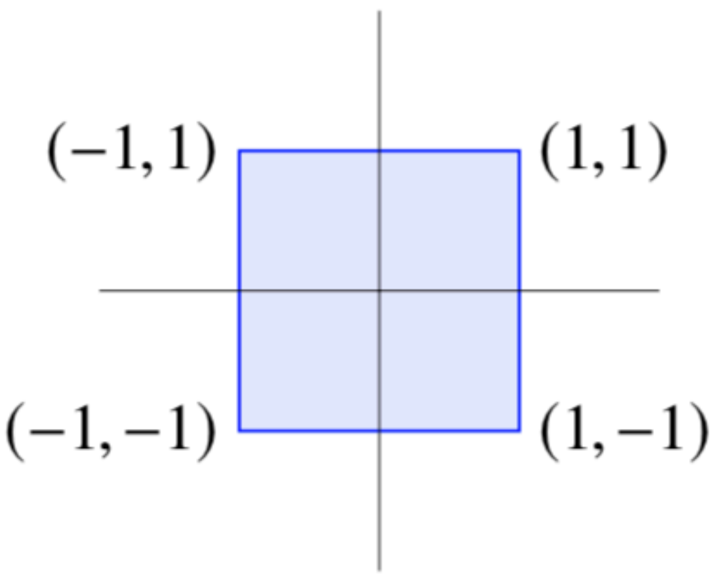
Example: l1-norm

- Let $f(x) := \|x\|_1$
- The sub-differential is a product of intervals:

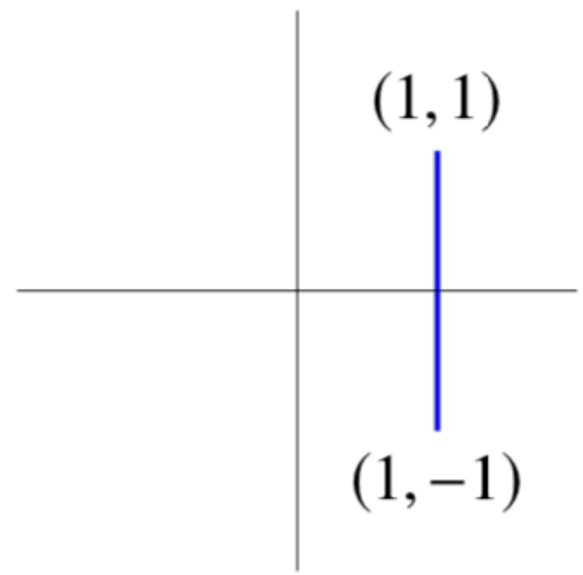
$$\partial f(x) := J_1 \times J_2 \times \cdots \times J_n$$

$$J_i := \begin{cases} [-1,1] & \textbf{if } x_i = 0 \\ 1 & \textbf{if } x_i > 0 \\ -1 & \textbf{if } x_i < 0 \end{cases} \quad \forall i = 1,\ldots,n$$

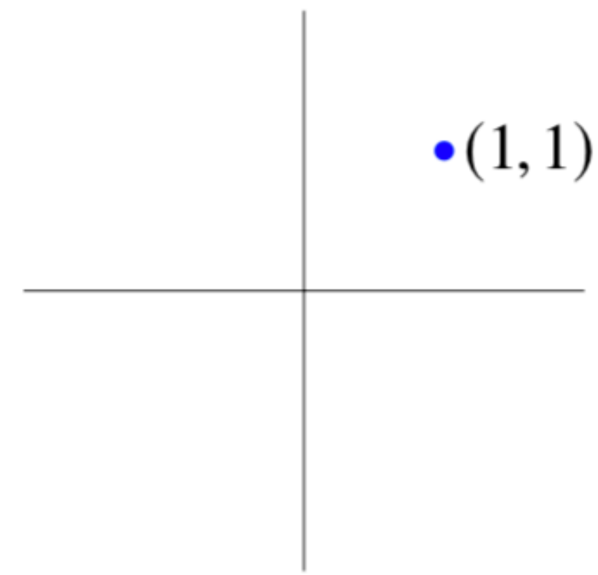
Example: l1-norm



$\partial f(0,0) = [-1,1] \times [-1,1]$



$\partial f(1,0) = \{1\} \times [-1,1]$



$\partial f(1,1) = \{(1,1)\}$

Example: expectation

- Let $f(x) := \mathbb{E}[h(x,u)]$, u is the random variable h is convex w.r.t x for any u.

- For a chosen u, a sub-gradient of h is $g(x,u) \in \partial h(x,u)$.

- Since $g(x,u)$ is a random variable then the sub-gradient is $g := \mathbb{E}[g(x,u)] \in \partial f(x)$.

- Proof:

- $f(x) := \mathbb{E}[h(x,u)] \geq \mathbb{E}[h(y,u) + g(u)^T(x-y)]$

- which is further lower bounded by $f(y) + g^T(x-y)$. This shows that g is a sub-gradient of f.

Stochastic sub-gradient

- We consider the following optimization problem

$$\min_{x\in\mathbb{R}^n} \underbrace{\frac{1}{n}\sum_{i=1}^{n} f_i(x)}_{f(x)} = \mathbb{E}[f_i(x)] = f(x)$$

- where we assume that f_i are non-smooth and f is convex.

- The **stochastic sub-gradient method is**:

  - Pick randomly a sample i

  - $x_{k+1} := x_k - \alpha_k g_i(x)$, where $g_i(x) \in \partial f_i(x)$

Assumptions for analyzing stochastic sub-gradient for convex functions

- f is convex
- $\mathbb{E}[\|g_i(x_k)\|_2^2] \leq \sigma^2$, where σ^2 is a constant.

Convergence rate of stochastic sub-gradient

- For convex functions we have the following:
- If $\alpha_k = 1/k$, then we get rate $\mathcal{O}(1/\log t)$
- If $\alpha_k = 1/\sqrt{k}$, then we get rate $\mathcal{O}(\log t/\sqrt{t})$
- If $\alpha_k = \alpha$ for some constant α, then we get rate $\mathcal{O}(1/t + \alpha)$. This means implies that the minimum expected norm of the gradient will never go to zero and then algorithm only converges to a neighborhood of a stationary point. Also, this rate appears to be converging sub-linearly initially, but then stagnates!

Assumptions for analyzing stochastic gradient for strongly-convex

- f is strongly-convex
- $\mathbb{E}[\|g_i(x_k)\|_2^2] \leq \sigma^2$, where σ^2 is a constant.
- The objective function is Lipschitz continuous $|f(x) - f(y)| \leq G\|x - y\|_2$, where $G > 0$ is a constant. This assumption also implies that $\|g(x)\|_2 \leq G$. This assumption is used only for the result with step-size $\alpha_k = 1/(\delta k)$.

Convergence rate of stochastic sub-gradient

- For δ-strongly-convex functions we have the following:
- If $\alpha_k = 1/(\delta k)$, then we get rate $\mathcal{O}(1/t)$. This implies that you have to know the strong-convexity constant.
- If $\alpha_k = \alpha$ for some constant $0 < \alpha < 2/\delta$, then we get rate $\mathcal{O}((1 - 2\alpha\delta)^t + \alpha)$. This means that the algorithm converges to a neighborhood of a minimizer. Also, this rate appears to be converging linearly initially, but then stagnates!

Iteration complexity for smooth functions

- The iteration complexity results for stochastic gradient in the next slide correspond to step-size $\alpha_k := 1/k$ for convex functions. Different iteration complexity results are obtained if you use other techniques for setting α_k.

- For δ-strongly convex functions and stochastic gradient the result in the next slide is obtained by setting $\alpha_k = 1/(\delta k)$.

Iteration complexity: non-smooth functions

- D is a constant due to smoothing. σ^2 is variance.

	Gradient Descent	**Accelerated Gradient**	**Stochastics Sub-Gradient**
Non-convex	$\mathcal{O}\left(\frac{D}{\epsilon^2}\right)$	-	$\mathcal{O}\left(\frac{1}{\epsilon^4}\right)$ **Just appeared 2018**
Convex	$\mathcal{O}\left(\frac{D}{\epsilon^2}\right)$	$\mathcal{O}\left(\frac{\sqrt{D}}{\epsilon}\right)$	$\mathcal{O}\left(e^{\frac{\sigma^2}{\epsilon}}\right)$
Strongly convex	$\mathcal{O}\left(\frac{D}{\delta\epsilon}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\sqrt{\frac{D}{\delta\epsilon}}\log\frac{1}{\epsilon}\right)$	$\mathcal{O}\left(\frac{G\sigma^2}{\delta^2}\frac{1}{\epsilon}\right)$

Comments on iteration complexity

- For convex and strongly-convex functions, gradient descent and stochastic sub-gradient have similar complexity, but each iteration of stochastic sub-gradient is n times less expensive!!

- The result for non-convex functions just appeared in 2018 (Damek and Drusvyatskiy 2018). The result is much worse than the one of gradient descent, but practitioners still prefer stochastic sub-gradient.

Output of stochastic sub-gradient for convex and strongly-convex functions

- For convex and strongly-convex functions theory suggests that the output of stochastic gradient should be the average over all iterations, i.e., $\bar{x}_k := \dfrac{1}{k+1}\displaystyle\sum_{i=0}^{k} x_i$.

Next lecture

- advanced stochastic gradient methods
- Practical versions of stochastic gradient