COMS 4771 Lecture 10

1. Solving convex optimization problems

Solving convex optimization problems

Standard form of convex optimization problem

$$\begin{aligned} \min_{\boldsymbol{x}\in\mathbb{R}^d} \quad & f_0(\boldsymbol{x}) \\ \text{s.t.} \quad & f_i(\boldsymbol{x}) \leq 0 \quad i = 1, 2, \ldots, n \end{aligned}$$

(for convex functions $f_0, f_1, \ldots, f_n \colon \mathbb{R}^d \to \mathbb{R}$).

Unconstrained convex optimization problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \quad f(\boldsymbol{x})$$

(f is the convex objective function).

Unconstrained convex optimization problem

$$\min_{\boldsymbol{x} \in \mathbb{R}^d} \qquad f(\boldsymbol{x})$$

(f is the convex objective function).

Unconstrained convex optimization

Unconstrained convex optimization problem

$$\min_{\boldsymbol{x} \in \mathbb{R}^d} \qquad f(\boldsymbol{x})$$

(f is the convex objective function).

Optimality condition for differentiable convex objectives

$\boldsymbol{x}$ is a global minimizer if and only if $\nabla f(\boldsymbol{x}) = \mathbf{0}$.

Unconstrained convex optimization problem

$$\min_{\boldsymbol{x} \in \mathbb{R}^d} \qquad f(\boldsymbol{x})$$

(f is the convex objective function).

Optimality condition for differentiable convex objectives

$\boldsymbol{x}$ is a global minimizer if and only if $\nabla f(\boldsymbol{x}) = \mathbf{0}$.

Unfortunately, can't always find closed-form solution to system of equations $\nabla f(\boldsymbol{x}) = \mathbf{0}$. $\longrightarrow$ Resort to iterative methods to find a solution.

Local optimization for convex objectives

Iterative local optimization

Main idea: locally change $\boldsymbol{x} \to \boldsymbol{x} + \boldsymbol{\delta}$ to improve its objective value $f(\boldsymbol{x}) \to f(\boldsymbol{x} + \boldsymbol{\delta})$.

Iterative local optimization

Main idea: locally change $\boldsymbol{x} \to \boldsymbol{x} + \boldsymbol{\delta}$ to improve its objective value $f(\boldsymbol{x}) \to f(\boldsymbol{x} + \boldsymbol{\delta})$. **What should δ be?**

Iterative local optimization

Main idea: locally change $\boldsymbol{x} \to \boldsymbol{x} + \boldsymbol{\delta}$ to improve its objective value $f(\boldsymbol{x}) \to f(\boldsymbol{x} + \boldsymbol{\delta})$. **What should δ be?**

By convexity of f: $\quad f(\boldsymbol{x} + \boldsymbol{\delta}) \geq f(\boldsymbol{x}) + \langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle.$

Iterative local optimization

Main idea: locally change $\boldsymbol{x} \to \boldsymbol{x} + \boldsymbol{\delta}$ to improve its objective value $f(\boldsymbol{x}) \to f(\boldsymbol{x} + \boldsymbol{\delta})$. **What should δ be?**

By convexity of f: $\quad f(\boldsymbol{x} + \boldsymbol{\delta}) \geq f(\boldsymbol{x}) + \langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle.$

If $\langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle \geq 0$, then

$$f(\boldsymbol{x} + \boldsymbol{\delta}) \geq f(\boldsymbol{x}).$$

Local optimization for convex objectives

Iterative local optimization

Main idea: locally change $\boldsymbol{x} \to \boldsymbol{x} + \boldsymbol{\delta}$ to improve its objective value $f(\boldsymbol{x}) \to f(\boldsymbol{x} + \boldsymbol{\delta})$. **What should δ be?**

By convexity of f: $\quad f(\boldsymbol{x} + \boldsymbol{\delta}) \geq f(\boldsymbol{x}) + \langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle.$

If $\langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle \geq 0$, then

$$f(\boldsymbol{x} + \boldsymbol{\delta}) \geq f(\boldsymbol{x}). \quad \text{Clearly a bad direction.}$$

Iterative local optimization

Main idea: locally change $\boldsymbol{x} \to \boldsymbol{x} + \boldsymbol{\delta}$ to improve its objective value $f(\boldsymbol{x}) \to f(\boldsymbol{x} + \boldsymbol{\delta})$. **What should $\boldsymbol{\delta}$ be?**

By convexity of f: $$f(\boldsymbol{x} + \boldsymbol{\delta}) \geq f(\boldsymbol{x}) + \langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle.$$

deferentiable

If $\langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle \geq 0$, then

$$f(\boldsymbol{x} + \boldsymbol{\delta}) \geq f(\boldsymbol{x}).$$ Clearly a bad direction.

Moral: to be useful, the change $\boldsymbol{\delta}$ must satisfy

$$\langle \nabla f(\boldsymbol{x}), \boldsymbol{\delta} \rangle < 0.$$

For example, $\boldsymbol{\delta} := -\eta \nabla f(\boldsymbol{x})$ for some $\eta > 0$:

just opposite direction.
don't make assumption
over delta(f(x)) is
positive and negative

$$\langle \nabla f(\boldsymbol{x}), -\eta \nabla f(\boldsymbol{x}) \rangle = -\eta \|\nabla f(\boldsymbol{x})\|_2^2 < 0$$

as long as $\nabla f(\boldsymbol{x}) \neq \mathbf{0}$.

Gradient descent for differentiable objectives

- Start with some initial $\boldsymbol{x}^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Compute gradient of f at $\boldsymbol{x}^{(t)}$:

$$\boldsymbol{\lambda}^{(t)} := \nabla f(\boldsymbol{x}^{(t)}).$$

  - Update:

$$\boldsymbol{x}^{(t+1)} := \boldsymbol{x}^{(t)} - \eta_t \boldsymbol{\lambda}^{(t)}.$$

Gradient descent for differentiable objectives

- Start with some initial $\boldsymbol{x}^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Compute gradient of f at $\boldsymbol{x}^{(t)}$:
  $$\boldsymbol{\lambda}^{(t)} := \nabla f(\boldsymbol{x}^{(t)}).$$
  - Update:
  $$\boldsymbol{x}^{(t+1)} := \boldsymbol{x}^{(t)} - \eta_t \boldsymbol{\lambda}^{(t)}.$$

Here, $\eta_1, \eta_2, \ldots > 0$ are the **step sizes**.

Gradient descent for differentiable objectives

- Start with some initial $\boldsymbol{x}^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Compute gradient of f at $\boldsymbol{x}^{(t)}$:
  $$\boldsymbol{\lambda}^{(t)} := \nabla f(\boldsymbol{x}^{(t)}).$$
  - Update:
  $$\boldsymbol{x}^{(t+1)} := \boldsymbol{x}^{(t)} - \eta_t \boldsymbol{\lambda}^{(t)}.$$

Here, $\eta_1, \eta_2, \ldots > 0$ are the **step sizes**. Common choices include:

1. Set $\eta_t := c$ for some constant $c > 0$.
2. Set $\eta_t := c/\sqrt{t}$ for some constant $c > 0$.
3. Set η_t using a line search procedure.

Backtracking line search

Goal: given $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{\lambda} = \nabla f(\boldsymbol{x}) \in \mathbb{R}^d$, find $\eta > 0$ so that $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) < f(\boldsymbol{x})$ by a reasonable amount.

- Start with $\eta := 1$.
- While $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) > f(\boldsymbol{x}) - \frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2$: Set $\eta := \frac{1}{2}\eta$.

Backtracking line search

Goal: given $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{\lambda} = \nabla f(\boldsymbol{x}) \in \mathbb{R}^d$, find $\eta > 0$ so that $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) < f(\boldsymbol{x})$ by a reasonable amount.

- Start with $\eta := 1$.
- While $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) > f(\boldsymbol{x}) - \frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2$: Set $\eta := \frac{1}{2}\eta$.

Main idea: $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) \approx f(\boldsymbol{x}) - \eta\|\boldsymbol{\lambda}\|_2^2$ when η is small, so can optimistically hope to decrease value by about $\eta\|\boldsymbol{\lambda}\|_2^2$.

Backtracking line search

Goal: given $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{\lambda} = \nabla f(\boldsymbol{x}) \in \mathbb{R}^d$, find $\eta > 0$ so that $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) < f(\boldsymbol{x})$ by a reasonable amount.

- Start with $\eta := 1$.
- While $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) > f(\boldsymbol{x}) - \frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2$: Set $\eta := \frac{1}{2}\eta$.

Main idea: $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) \approx f(\boldsymbol{x}) - \eta\|\boldsymbol{\lambda}\|_2^2$ when η is small, so can optimistically hope to decrease value by about $\eta\|\boldsymbol{\lambda}\|_2^2$.

Settle for decreasing by $\frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2$: upon termination of while-loop,

$$f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) \;\leq\; f(\boldsymbol{x}) - \frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2.$$

Backtracking line search

Goal: given $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{\lambda} = \nabla f(\boldsymbol{x}) \in \mathbb{R}^d$, find $\eta > 0$ so that $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) < f(\boldsymbol{x})$ by a reasonable amount.

- Start with $\eta := 1$. guess
- While $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) > f(\boldsymbol{x}) - \frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2$: Set $\eta := \frac{1}{2}\eta$.

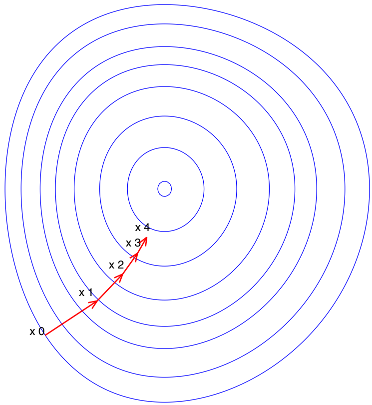
Main idea: $f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) \approx f(\boldsymbol{x}) - \eta\|\boldsymbol{\lambda}\|_2^2$ when η is small, so can optimistically hope to decrease value by about $\eta\|\boldsymbol{\lambda}\|_2^2$.

Settle for decreasing by $\frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2$: upon termination of while-loop,

$$f(\boldsymbol{x} - \eta\boldsymbol{\lambda}) \;\leq\; f(\boldsymbol{x}) - \frac{1}{2}\eta\|\boldsymbol{\lambda}\|_2^2.$$

Many other line search methods are possible.

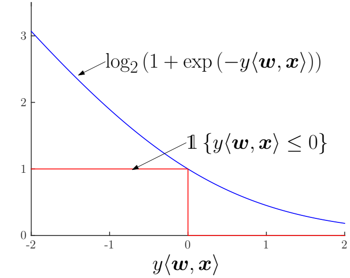
Illustration of gradient descent



If f is convex (and satisfies some other smoothness and curvature conditions), then $f(\boldsymbol{x}^{(t)})$ converges to the optimal value at a geometric rate.

EXAMPLE: (UNCONSTRAINED) LOGISTIC REGRESSION

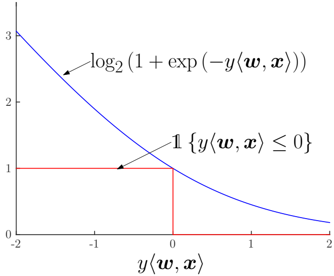
$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \qquad f(\boldsymbol{w}) := \frac{1}{|S|}\sum_{(\boldsymbol{x},y)\in S} \ln(1+\exp(-y\langle\boldsymbol{w},\boldsymbol{x}\rangle))$$



We've already established that objective $f(\boldsymbol{w})$ is convex.

Example: (unconstrained) logistic regression

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \qquad f(\boldsymbol{w}) := \frac{1}{|S|} \sum_{(\boldsymbol{x}, y) \in S} \ln(1 + \exp(-y\langle \boldsymbol{w}, \boldsymbol{x} \rangle))$$



We've already established that objective $f(\boldsymbol{w})$ is convex.

Question: How do we compute its gradient at a given point $\boldsymbol{w} \in \mathbb{R}^d$?

Example: (unconstrained) logistic regression

Gradient of f at $\boldsymbol{w}$:

$$\nabla f(\boldsymbol{w}) = -\frac{1}{|S|} \sum_{(\boldsymbol{x},y)\in S} \frac{1}{1+e^{y\langle \boldsymbol{w},\boldsymbol{x}\rangle}} y\boldsymbol{x}$$

Example: (unconstrained) logistic regression

Gradient of f at $\boldsymbol{w}$:

$$\nabla f(\boldsymbol{w}) = -\frac{1}{|S|} \sum_{(\boldsymbol{x},y)\in S} \frac{1}{1+e^{y\langle \boldsymbol{w},\boldsymbol{x}\rangle}} y\boldsymbol{x}$$

Gradient descent algorithm for logistic regression:

- Start with some initial $\boldsymbol{w}^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.

$$\begin{aligned} \boldsymbol{w}^{(t+1)} &:= \boldsymbol{w}^{(t)} - \eta_t \nabla f(\boldsymbol{w}^{(t)}) \\ &= \boldsymbol{w}^{(t)} + \eta_t \frac{1}{|S|} \sum_{(\boldsymbol{x},y)\in S} \frac{1}{1+e^{y\langle \boldsymbol{w},\boldsymbol{x}\rangle}} y\boldsymbol{x}. \end{aligned}$$

Stopping condition

- In many applications of (convex) optimization, care about solving problems to very high precision.

  Example: stop when gradient is close enough to zero ($\|\nabla f(\boldsymbol{x})\|_2 \leq \epsilon$ for some small parameter $\epsilon > 0$).

Stopping condition

- In many applications of (convex) optimization, care about solving problems to very high precision.

  Example: stop when gradient is close enough to zero ($\|\nabla f(\boldsymbol{x})\|_2 \leq \epsilon$ for some small parameter $\epsilon > 0$).

- For machine learning applications: optimization problem based on training data often just a means-to-an-end.

Stopping condition

- In many applications of (convex) optimization, care about solving problems to very high precision.

  Example: stop when gradient is close enough to zero ($\|\nabla f(\boldsymbol{x})\|_2 \leq \epsilon$ for some small parameter $\epsilon > 0$).

- For machine learning applications: optimization problem based on training data often just a means-to-an-end.

  We really just care about true error.

Stopping condition

- In many applications of (convex) optimization, care about solving problems to very high precision.

  Example: stop when gradient is close enough to zero ($\|\nabla f(\boldsymbol{x})\|_2 \leq \epsilon$ for some small parameter $\epsilon > 0$).

- For machine learning applications: optimization problem based on training data often just a means-to-an-end.

  We really just care about true error.

- Running gradient descent to convergence not strictly necessary: **may be beneficial to stop early (e.g., when hold-out error starts to increase significantly).**

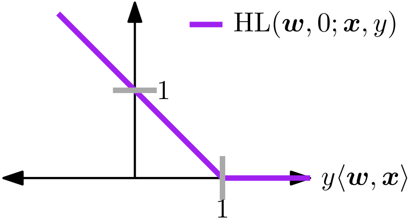
when objective value is different
from hold-out error

Non-differentiable convex objectives

Some convex functions f are not differentiable everywhere;
gradient descent not even well-specified for these problems.

Example: hinge loss

$$f(\boldsymbol{w}) = \mathrm{HL}(\boldsymbol{w}, 0; \boldsymbol{x}, y) = \Big[1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle\Big]_+ .$$



Not differentiable at $\boldsymbol{w} \in \mathbb{R}^d$ where $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = 1$.

Subgradients

Although not every function f is differentiable everywhere, every **convex function** f has **subgradients** *everywhere*†.

We say $\boldsymbol{\lambda} \in \mathbb{R}^d$ is a **subgradient** of a function $f\colon \mathbb{R}^d \to \mathbb{R}$ at $\boldsymbol{x}_0 \in \mathbb{R}^d$ if

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_0) + \langle \boldsymbol{\lambda}, \boldsymbol{x} - \boldsymbol{x}_0 \rangle \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

†some technical conditions apply.

Subgradients

Although not every function f is differentiable everywhere,
every **convex function** f has **subgradients** *everywhere*[†].

We say $\boldsymbol{\lambda} \in \mathbb{R}^d$ is a **subgradient** of a function $f\colon \mathbb{R}^d \to \mathbb{R}$ at $\boldsymbol{x}_0 \in \mathbb{R}^d$ if

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_0) + \langle \boldsymbol{\lambda}, \boldsymbol{x} - \boldsymbol{x}_0 \rangle \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

In other words, a subgradient of a convex function f at a point $\boldsymbol{x}_0$ specifies an **affine lower bound** on the function ...

[†]some technical conditions apply.

Subgradients

Although not every function f is differentiable everywhere, every **convex function** f has **subgradients** *everywhere*†.

We say $\boldsymbol{\lambda} \in \mathbb{R}^d$ is a **subgradient** of a function $f \colon \mathbb{R}^d \to \mathbb{R}$ at $\boldsymbol{x}_0 \in \mathbb{R}^d$ if

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_0) + \langle \boldsymbol{\lambda}, \boldsymbol{x} - \boldsymbol{x}_0 \rangle \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

In other words, a subgradient of a convex function f at a point $\boldsymbol{x}_0$ specifies an **affine lower bound** on the function ...

... **just like the gradient in the case of a differentiable function**:

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_0) + \langle \nabla f(\boldsymbol{x}_0), \boldsymbol{x} - \boldsymbol{x}_0 \rangle \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

†some technical conditions apply.

Subgradients

Although not every function f is differentiable everywhere, every **convex function** f has **subgradients** *everywhere*[†].

We say $\boldsymbol{\lambda} \in \mathbb{R}^d$ is a **subgradient** of a function $f \colon \mathbb{R}^d \to \mathbb{R}$ at $\boldsymbol{x}_0 \in \mathbb{R}^d$ if

$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_0) + \langle \boldsymbol{\lambda}, \boldsymbol{x} - \boldsymbol{x}_0 \rangle \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

In other words, a subgradient of a convex function f at a point $\boldsymbol{x}_0$ specifies an **affine lower bound** on the function ...

... **just like the gradient in the case of a differentiable function**:

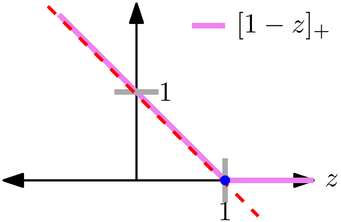
$$f(\boldsymbol{x}) \geq f(\boldsymbol{x}_0) + \langle \nabla f(\boldsymbol{x}_0), \boldsymbol{x} - \boldsymbol{x}_0 \rangle \quad \forall \boldsymbol{x} \in \mathbb{R}^d.$$

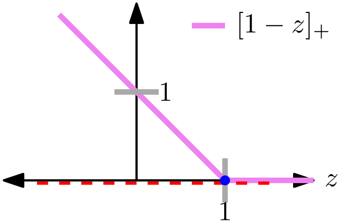
There might be many subgradients at a given point $\boldsymbol{x}_0$—i.e., many affine lower bounds: call the entire set the **subdifferential of f at $\boldsymbol{x}_0$**, $\partial f(\boldsymbol{x}_0)$.

[†]some technical conditions apply.

Example: subgradient of hinge loss (sorta)

Consider one-dimensional function $f(z) := [1 - z]_+ = \max\{0, 1 - z\}$.



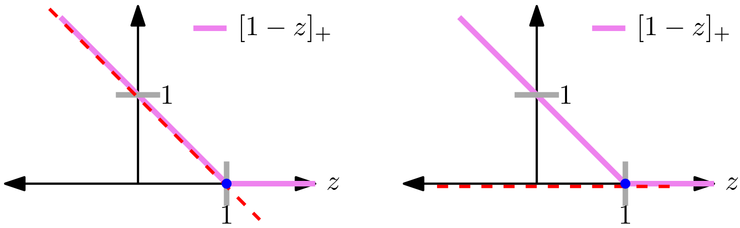


Two subgradients of f at $z = 1$: -1 and 0.

$$f(z) \;\geq\; f(1) + (-1) \cdot (z - 1) \;=\; 1 - z \;;$$
$$f(z) \;\geq\; f(1) + (0) \cdot (z - 1) \;=\; 0.$$

Example: subgradient of hinge loss (sorta)

Consider one-dimensional function $f(z) := [1-z]_+ = \max\{0, 1-z\}$.



as long as r ~ [-1, 0].
there are infinitely
subgradients!!!

Two subgradients of f at $\boxed{z = 1}$: -1 and 0.

$$f(z) \;\geq\; f(1) + (-1)\cdot(z-1) \;=\; 1-z\;;$$
$$f(z) \;\geq\; f(1) + (0)\cdot(z-1) \;=\; 0.$$

Actually, **infinitely-many subgradients at** $z = 1$: all $\lambda \in [-1, 0]$ satisfy

$$f(z) \;\geq\; f(1) + \lambda\cdot(z-1).$$

Subgradient calculus

Suppose g, g_1, g_2 are convex functions.
Below, sufficient conditions under which f is convex, and corresponding subdifferential:

- **Addition**: If $f(\boldsymbol{x}) = g_1(\boldsymbol{x}) + g_2(\boldsymbol{x})$, then

$$\partial f(\boldsymbol{x}) = \{\boldsymbol{\lambda} + \boldsymbol{\nu} : \boldsymbol{\lambda} \in \partial g_1(\boldsymbol{x}),\ \boldsymbol{\nu} \in \partial g_2(\boldsymbol{x})\}.$$

Subgradient calculus

Suppose g, g_1, g_2 are convex functions.
Below, sufficient conditions under which f is convex, and corresponding subdifferential:

- **Addition**: If $f(\boldsymbol{x}) = g_1(\boldsymbol{x}) + g_2(\boldsymbol{x})$, then

$$\partial f(\boldsymbol{x}) = \{\boldsymbol{\lambda} + \boldsymbol{\nu} : \boldsymbol{\lambda} \in \partial g_1(\boldsymbol{x}),\ \boldsymbol{\nu} \in \partial g_2(\boldsymbol{x})\}.$$

- **Positive scaling**: If $f(\boldsymbol{x}) = \alpha \cdot g(\boldsymbol{x})$ for some $\alpha > 0$, then

$$\partial f(\boldsymbol{x}) = \{\alpha\boldsymbol{\lambda} : \boldsymbol{\lambda} \in \partial g(\boldsymbol{x})\}.$$

Suppose g, g_1, g_2 are convex functions.
Below, sufficient conditions under which f is convex, and corresponding subdifferential:

- **Addition**: If $f(\boldsymbol{x}) = g_1(\boldsymbol{x}) + g_2(\boldsymbol{x})$, then

$$\partial f(\boldsymbol{x}) = \{\boldsymbol{\lambda} + \boldsymbol{\nu} : \boldsymbol{\lambda} \in \partial g_1(\boldsymbol{x}),\ \boldsymbol{\nu} \in \partial g_2(\boldsymbol{x})\}.$$

- **Positive scaling**: If $f(\boldsymbol{x}) = \alpha \cdot g(\boldsymbol{x})$ for some $\alpha > 0$, then

$$\partial f(\boldsymbol{x}) = \{\alpha \boldsymbol{\lambda} : \boldsymbol{\lambda} \in \partial g(\boldsymbol{x})\}.$$

- **Affine composition**: If $f(\boldsymbol{x}) = g(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b})$, then

$$\partial f(\boldsymbol{x}) = \left\{\boldsymbol{A}^\top \boldsymbol{\lambda} : \boldsymbol{\lambda} \in \partial g(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b})\right\}.$$

Suppose g_1, g_2 are convex functions.
Below, sufficient conditions under which f is convex, and corresponding subdifferential:

- **Max of convex functions**: If $f(\boldsymbol{x}) = \max\{g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \ldots, g_n(\boldsymbol{x})\}$, then

$$\partial f(\boldsymbol{x}) = \begin{cases} \partial g_1(\boldsymbol{x}) & \text{if } g_1(\boldsymbol{x}) > g_2(\boldsymbol{x}); \\ \partial g_2(\boldsymbol{x}) & \text{if } g_1(\boldsymbol{x}) < g_2(\boldsymbol{x}); \\ \boxed{\operatorname{conv}(\partial g_1(\boldsymbol{x}) \cup \partial g_2(\boldsymbol{x})) \quad \text{if } g_1(\boldsymbol{x}) = g_2(\boldsymbol{x}).} \end{cases}$$

How about connecting those regions together!!!

Text

Example: subgradient of hinge loss (really)

Hinge loss function: $f(\boldsymbol{w}) := [1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle]_+ = \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle\}$.

Example: subgradient of hinge loss (really)

Hinge loss function: $f(\boldsymbol{w}) := \left[1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle\right]_+ = \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle\}$.

$f(\boldsymbol{w}) = \max\{g_1(\boldsymbol{w}),\ g_2(\boldsymbol{w})\}$ where $g_1(\boldsymbol{w}) = 0$ and $g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle$.

Example: subgradient of hinge loss (really)

Hinge loss function: $f(\boldsymbol{w}) := [1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle]_+ = \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle\}$.

$f(\boldsymbol{w}) = \max\{g_1(\boldsymbol{w}),\ g_2(\boldsymbol{w})\}$ where $g_1(\boldsymbol{w}) = 0$ and $g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle$.

- $\partial g_1(\boldsymbol{w}) = \{\mathbf{0}\}$ and $\partial g_2(\boldsymbol{w}) = \{-y\boldsymbol{x}\}$.

Example: subgradient of hinge loss (really)

Hinge loss function: $f(\boldsymbol{w}) := [1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle]_+ = \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle\}$.

$f(\boldsymbol{w}) = \max\{g_1(\boldsymbol{w}),\ g_2(\boldsymbol{w})\}$ where $g_1(\boldsymbol{w}) = 0$ and $g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle$.

- $\partial g_1(\boldsymbol{w}) = \{\boldsymbol{0}\}$ and $\partial g_2(\boldsymbol{w}) = \{-y\boldsymbol{x}\}$.
- If $f(\boldsymbol{w}) = g_1(\boldsymbol{w}) = 0 > 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle = g_2(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x} \rangle > 1$), then

$$\partial f(\boldsymbol{w}) = \{\boldsymbol{0}\}.$$

Text

Example: subgradient of hinge loss (really)

Hinge loss function: $f(\boldsymbol{w}) := [1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle]_+ = \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle\}$.

$f(\boldsymbol{w}) = \max\{g_1(\boldsymbol{w}),\, g_2(\boldsymbol{w})\}$ where $g_1(\boldsymbol{w}) = 0$ and $g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle$.

- $\partial g_1(\boldsymbol{w}) = \{\boldsymbol{0}\}$ and $\partial g_2(\boldsymbol{w}) = \{-y\boldsymbol{x}\}$.
- If $f(\boldsymbol{w}) = g_1(\boldsymbol{w}) = 0 > 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle = g_2(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x} \rangle > 1$), then
$$\partial f(\boldsymbol{w}) = \{\boldsymbol{0}\}.$$
- If $f(\boldsymbol{w}) = g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle > 0 = g_1(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x} \rangle < 1$), then
$$\partial f(\boldsymbol{w}) = \{-y\boldsymbol{x}\}.$$

Example: subgradient of hinge loss (really)

Hinge loss function: $f(\boldsymbol{w}) := [1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle]_+ = \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle\}$.

$f(\boldsymbol{w}) = \max\{g_1(\boldsymbol{w}),\ g_2(\boldsymbol{w})\}$ where $g_1(\boldsymbol{w}) = 0$ and $g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle$.

- $\partial g_1(\boldsymbol{w}) = \{\mathbf{0}\}$ and $\partial g_2(\boldsymbol{w}) = \{-y\boldsymbol{x}\}$.
- If $f(\boldsymbol{w}) = g_1(\boldsymbol{w}) = 0 > 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = g_2(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle > 1$), then
$$\partial f(\boldsymbol{w}) = \{\mathbf{0}\}.$$
- If $f(\boldsymbol{w}) = g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle > 0 = g_1(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle < 1$), then
$$\partial f(\boldsymbol{w}) = \{-y\boldsymbol{x}\}.$$
- If $f(\boldsymbol{w}) = g_1(\boldsymbol{w}) = g_2(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = 1$), then
$$\partial f(\boldsymbol{w}) = \operatorname{conv}\{\mathbf{0},\ -y\boldsymbol{x}\}.$$

Great!!!

Example: subgradient of hinge loss (really)

Hinge loss function: $f(\boldsymbol{w}) := [1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle]_+ = \max\{0, 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle\}$.

$f(\boldsymbol{w}) = \max\{g_1(\boldsymbol{w}),\ g_2(\boldsymbol{w})\}$ where $g_1(\boldsymbol{w}) = 0$ and $g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle$.

- $\partial g_1(\boldsymbol{w}) = \{\boldsymbol{0}\}$ and $\partial g_2(\boldsymbol{w}) = \{-y\boldsymbol{x}\}$.
- If $f(\boldsymbol{w}) = g_1(\boldsymbol{w}) = 0 > 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = g_2(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle > 1$), then
$$\partial f(\boldsymbol{w}) = \{\boldsymbol{0}\}.$$
- If $f(\boldsymbol{w}) = g_2(\boldsymbol{w}) = 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle > 0 = g_1(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle < 1$), then
$$\partial f(\boldsymbol{w}) = \{-y\boldsymbol{x}\}.$$
- If $f(\boldsymbol{w}) = g_1(\boldsymbol{w}) = g_2(\boldsymbol{w})$ (i.e., if $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = 1$), then
$$\partial f(\boldsymbol{w}) = \operatorname{conv}\{\boldsymbol{0},\ -y\boldsymbol{x}\}.$$

Can also derive using affine composition rule.

Subgradient descent for general convex objectives

- Start with some initial $\boldsymbol{x}^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Compute *any* subgradient $\boldsymbol{\lambda}^{(t)} \in \partial f(\boldsymbol{x}^{(t)})$.
  - Update:

$$\boldsymbol{x}^{(t+1)} := \boldsymbol{x}^{(t)} - \eta_t \boldsymbol{\lambda}^{(t)}.$$

Example: soft-margin SVM (without offset)

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \qquad f(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{|S|}\sum_{(\boldsymbol{x},y)\in S}[1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle]_+$$

f is the sum of convex functions, and hence is convex.

Example: soft-margin SVM (without offset)

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \qquad f(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{|S|} \sum_{(\boldsymbol{x},y) \in S} [1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle]_+$$

f is the sum of convex functions, and hence is convex.

Question: How do we compute *a* subgradient $\boldsymbol{g}$ of f at a given point $\boldsymbol{w} \in \mathbb{R}^d$?

Example: soft-margin SVM (without offset)

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \qquad f(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{|S|}\sum_{(\boldsymbol{x},y)\in S} [1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle]_+$$

f is the sum of convex functions, and hence is convex.

Question: How do we compute *a* subgradient $\boldsymbol{g}$ of f at a given point $\boldsymbol{w} \in \mathbb{R}^d$?

$$\partial f(\boldsymbol{w}) \ni \boldsymbol{g} = \lambda \boldsymbol{w} + \frac{1}{|S|}\sum_{(\boldsymbol{x},y)\in S} \begin{cases} \boldsymbol{0} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle < 0; \\ -y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle \geq 0. \end{cases}$$

EXAMPLE: SOFT-MARGIN SVM (WITHOUT OFFSET)

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} \qquad f(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{|S|} \sum_{(\boldsymbol{x},y) \in S} [1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle]_+$$

f is the sum of convex functions, and hence is convex.

Question: How do we compute *a* subgradient $\boldsymbol{g}$ of f at a given point $\boldsymbol{w} \in \mathbb{R}^d$?

$$\partial f(\boldsymbol{w}) \ni \boldsymbol{g} = \lambda \boldsymbol{w} + \frac{1}{|S|} \sum_{(\boldsymbol{x},y) \in S} \begin{cases} \mathbf{0} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle < 0; \\ -y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle \geq 0. \end{cases}$$

Also fine:

$$\partial f(\boldsymbol{w}) \ni \boldsymbol{g} = \lambda \boldsymbol{w} + \frac{1}{|S|} \sum_{(\boldsymbol{x},y) \in S} \begin{cases} \mathbf{0} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle < 0; \\ -\frac{1}{3}y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle = 0; \\ -y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle > 0. \end{cases}$$

Example: soft-margin SVM (without offset)

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \qquad f(\boldsymbol{w}) := \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2 + \frac{1}{|S|}\sum_{(\boldsymbol{x},y)\in S}[1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle]_+$$ not differentable

f is the sum of convex functions, and hence is convex.

Question: How do we compute *a* subgradient $\boldsymbol{g}$ of f at a given point $\boldsymbol{w} \in \mathbb{R}^d$?

$$\partial f(\boldsymbol{w}) \ni \boldsymbol{g} = \lambda\boldsymbol{w} + \frac{1}{|S|}\sum_{(\boldsymbol{x},y)\in S}\begin{cases}\boldsymbol{0} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle < 0; \\ -y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle \geq 0.\end{cases}$$

Also fine:

many subgradients pick on

$$\partial f(\boldsymbol{w}) \ni \boldsymbol{g} = \lambda\boldsymbol{w} + \frac{1}{|S|}\sum_{(\boldsymbol{x},y)\in S}\begin{cases}\boldsymbol{0} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle < 0; \\ -\frac{1}{3}y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = 0; \\ -y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x}\rangle > 0.\end{cases}$$

In practice, usually don't have examples with $y\langle \boldsymbol{w}, \boldsymbol{x}\rangle = 1$ *exactly* anyway.

Example: soft-margin SVM (without offset)

Subgradient descent algorithm for soft-margin SVM:

- Start with some initial $\boldsymbol{w}^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.

$$\boldsymbol{w}^{(t+1)} := \boldsymbol{w}^{(t)} - \eta_t \left(\lambda \boldsymbol{w}^{(t)} + \frac{1}{|S|} \sum_{(\boldsymbol{x},y) \in S} \begin{cases} \mathbf{0} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle < 0; \\ -y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle \geq 0 \end{cases} \right)$$

$$= (1 - \lambda\eta_t)\boldsymbol{w}^{(t)} + \eta_t \frac{1}{|S|} \sum_{\substack{(\boldsymbol{x},y) \in S: \\ y\langle \boldsymbol{w}, \boldsymbol{x} \rangle \leq 1}} y\boldsymbol{x}.$$

Example: soft-margin SVM (without offset)

Subgradient descent algorithm for soft-margin SVM:

- Start with some initial $\boldsymbol{w}^{(1)} \in \mathbb{R}^d$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.

$$\boldsymbol{w}^{(t+1)} := \boldsymbol{w}^{(t)} - \eta_t \left(\lambda \boldsymbol{w}^{(t)} + \frac{1}{|S|} \sum_{(\boldsymbol{x},y) \in S} \begin{cases} \boldsymbol{0} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle < 0; \\ -y\boldsymbol{x} & \text{if } 1 - y\langle \boldsymbol{w}, \boldsymbol{x} \rangle \geq 0 \end{cases} \right)$$

$$= (1 - \lambda\eta_t)\boldsymbol{w}^{(t)} + \eta_t \frac{1}{|S|} \sum_{\substack{(\boldsymbol{x},y) \in S: \\ y\langle \boldsymbol{w}, \boldsymbol{x} \rangle \leq 1}} y\boldsymbol{x}.$$

Note effect of regularization term $\frac{\lambda}{2}\|\boldsymbol{w}\|_2^2$ (whenever $\eta_t < 1/\lambda$):
Shrink $\boldsymbol{w}^{(t)}$ by a factor $1 - \lambda\eta_t$ before updating with subgradient of loss term – tries to prevent length of $\boldsymbol{w}^{(t)}$ from becoming too large.

Convergence of subgradient descent

In general, subgradient descent (with possibly non-differentiable convex objectives) can converge relatively slowly.

Convergence of subgradient descent

In general, subgradient descent (with possibly non-differentiable convex objectives) can converge relatively slowly.

Optimality condition for general convex objectives

$\boldsymbol{x}$ is a global minimizer if and only if $\mathbf{0} \in \partial f(\boldsymbol{x})$.

Convergence and optimality conditions

Convergence of subgradient descent

In general, subgradient descent (with possibly non-differentiable convex objectives) can converge relatively slowly.

Optimality condition for general convex objectives

$$\boldsymbol{x} \text{ is a global minimizer if and only if } \mathbf{0} \in \partial f(\boldsymbol{x}).$$

Unfortunately, does not suggest a good stopping criterion for subgradient descent (since difficult to check all possible subgradients of f).

Convergence of subgradient descent

In general, subgradient descent (with possibly non-differentiable convex objectives) can converge relatively slowly.

Optimality condition for general convex objectives

$$\boldsymbol{x} \text{ is a global minimizer if and only if } \mathbf{0} \in \partial f(\boldsymbol{x}).$$

Unfortunately, does not suggest a good stopping criterion for subgradient descent (since difficult to check all possible subgradients of f).

But for machine learning purposes, we can use alternative criteria for stopping (e.g., hold-out error).

Constrained convex optimization

$$\begin{aligned} \min_{\boldsymbol{x} \in \mathbb{R}^d} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & \boldsymbol{x} \in A \end{aligned}$$

for convex function f and convex feasible set A (possibly a strict subset of $\mathbb{R}^d$).

Constrained convex optimization

$$\begin{array}{ll} \min_{\boldsymbol{x} \in \mathbb{R}^d} & f(\boldsymbol{x}) \\ \text{s.t.} & \boldsymbol{x} \in A \end{array}$$

for convex function f and convex feasible set A (possibly a strict subset of $\mathbb{R}^d$).

Projected subgradient descent

- Start with some initial $\boldsymbol{x}^{(1)} \in A$.
- For $t = 1, 2, \ldots$ until some stopping condition is satisfied.
  - Compute *any* subgradient $\boldsymbol{\lambda}^{(t)} \in \partial f(\boldsymbol{x}^{(t)})$.
  - Update:
  $$\boldsymbol{x}^{(t+0.5)} := \boldsymbol{x}^{(t)} - \eta_t \boldsymbol{\lambda}^{(t)}.$$
  - Project:
  $$\boldsymbol{x}^{(t+1)} := \operatorname{Proj}_A(\boldsymbol{x}^{(t+0.5)}).$$

Projection onto convex set A: $\operatorname{Proj}_A(\boldsymbol{x}) := \arg\min_{\boldsymbol{x}' \in A} \|\boldsymbol{x} - \boldsymbol{x}'\|_2^2$
(another convex optimization problem, but hopefully simpler objective!).

Other solvers

Many other algorithms for solving convex optimization problems

- *Newton-Raphson*: use Hessian to pick better descent directions.
- *Quasi-Newton methods* (e.g., conjugate gradient, "BFGS", "L-BFGS"): use efficient approximations of Hessians.
- Techniques for dealing with constraints:
  - *Barrier methods*: add penalties for constraint violations, slowly relax.
  - *Primal-dual methods*: start with dual-feasible point, iteratively improve until corresponding primal point is feasible.
  - ...
- *Stochastic gradient methods*: a lot like Perceptron
- ...

But remember: end goal in machine learning is *not* to minimize training error (let alone training surrogate loss).