Machine Learning and Optimization

Lecture 9: Subdifferential Calculus Rules, Optimality Conditions, and Subgradient Descent

CSCI 4140/6960 — Spring 2026

Course website: `cs.rpi.edu/~gittea/.../mlandopt.html`

Learning Objectives

By the end of this lecture, you should be able to:

1. **Apply subdifferential calculus rules** (sum, chain, and max rules) to derive subdifferentials for **complex nonsmooth objectives** like ℓ_1-regularized loss functions.
2. **Characterize the Normal Cone** $\mathcal{N}_C(x)$ for various convex sets and understand its role as the subdifferential of the **convex indicator function** ι_C.
3. **Utilize Fermat's Condition** ($0 \in \partial f(x^*)$) to identify optimal solutions for both **unconstrained and constrained** nonsmooth optimization problems.
4. **Explain the geometry of Subgradient Descent**, including why negative subgradients are not always descent directions and how **non-expansiveness** ensures convergence.

Outline

Subdifferential Calculus Rules

Example Subdifferential Calculation: ℓ_1-Regularized Hinge Loss

Subdifferentials of Convex Indicator Functions and Normal Cones

Fermat's Condition for Nonsmooth Convex Optimization

Subgradient Descent

Projections onto Convex Sets

Projected Subgradient Descent

Capstone: Denoising

Capstone: Image Denoising via Projected Subgradient Descent

Summary and Looking Ahead

Calculus Rules for Subdifferential

Computing $\partial f(x)$ from the definition requires checking *all* y:

$$v \in \partial f(x) \iff f(y) \geq f(x) + \langle v, y - x \rangle \quad \forall\, y$$

In practice, we build up subdifferentials of complex functions from simple pieces, just as in smooth calculus.

The key rules: sum, positive scaling, affine transformation, chain rule, and max rule.

Sum Rule

If $f_1, \ldots, f_m$ are proper, convex, and have the same domain[1]:

$$\partial(f_1 + \cdots + f_m)(x) = \partial f_1(x) + \cdots + \partial f_m(x)$$

where $A + B = \{a + b : a \in A, b \in B\}$ is the **Minkowski sum** of sets.

Example:

$$\partial(|x| + |x|)(0) = [-1, 1] + [-1, 1] = [-2, 2]$$

[1] This is not a necessity, but is the simplest sufficient condition.

When the Sum Rule Can Fail (Why Regularity Matters)

The sum rule can fail when domains only touch on a boundary (no interior overlap).

1D counterexample:
Let $f(x) = \iota_{[0,\infty)}(x)$ and $g(x) = \iota_{(-\infty,0]}$.
At $x = 0$:

$$\partial f(0) = (-\infty, 0], \quad \partial g(0) = [0, \infty),$$

so $\partial f(0) + \partial g(0) = \mathbb{R}$,
but $f + g = 0$, so

$$\partial(f + g)(0) = \{0\} \quad \neq \quad \mathbb{R} = \partial f(0) + \partial g(0)$$

Takeaway: We assume interior overlap / mild regularity in all future uses of the sum rule.

Positive Scalar Multiplication

For $a > 0$:

$$\partial(af)(x) = a\,\partial f(x)$$

Example:

$$\partial(3|x|)(0) = 3 \cdot [-1, 1] = [-3, 3]$$

Combined with the sum rule, if $\{f_i\}$ are convex and a_i are positive, then:

$$\partial\left(\sum_i a_i f_i(x)\right) = \sum_i a_i \partial f_i(x)$$

Differentiability Rule

If f is differentiable at x:

$$\partial f(x) = \{\nabla f(x)\}$$

This connects the subdifferential to classical calculus.

Practical use: In composite objectives like $\frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1$, the smooth part contributes its gradient and only the nonsmooth part requires subdifferential computation.

By the sum rule:

$$\partial\left(\frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1\right) = \{A^T(Ax - b)\} + \lambda\,\partial\|x\|_1$$

Affine Rule

Let f be a convex function and define $g(x) = f(Ax + b)$, then

$$\partial g(x) = A^\top \partial f(Ax + b).$$

Example:

Let $f(t) = |t|$ and define

$$g(x) = |a^T x + b| = f(a^T x + b), \qquad a \in \mathbb{R}^d.$$

We know

$$\partial |t| = \begin{cases} \{1\}, & t > 0 \\ [-1, 1], & t = 0 \\ \{-1\}, & t < 0 \end{cases}$$

By the affine rule:

$$\partial g(x) = \begin{cases} \{+a\}, & a^\top x + b > 0 \\ \{\lambda a : \lambda \in [-1, 1]\}, & a^\top x + b = 0 \\ \{-a\}, & a^\top x + b < 0 \end{cases}$$

Chain Rule (Nondecreasing Convex Composition)

If $h = g \circ f$ where:

- $g : \mathbb{R} \to \mathbb{R}$ is nondecreasing, differentiable, and convex
- $f : \mathbb{R}^d \to \mathbb{R}$ is convex

Then:

$$\partial h(x) = g'(f(x))\, \partial f(x)$$

Example: For $h(x) = |x|^p$ with $p \geq 1$, write $h = g \circ f$ where $f(x) = |x|$ and $g(t) = t^p$:

$$\partial h(x) = p|x|^{p-1} \operatorname{sign}(x)$$

Key requirement: g must be nondecreasing. Without this, the composition $g \circ f$ need not be convex.

Max Rule

$$h(x) = \max_{i=1,\dots,m} f_i(x)$$

$$\partial h(x) = \text{conv}\left(\bigcup_{i \in I(x)} \partial f_i(x)\right)$$

where $I(x) = \{i : f_i(x) = h(x)\}$ is the **active set** and conv denotes convex hull.

Intuition: Only the functions achieving the maximum contribute to the subdifferential.

Max Rule: Recovering $\partial|x|$

Example: $f(x) = \max\{x, -x\} = |x|$

- At $x = 0$: both active, $I(0) = \{1, 2\}$

$$\partial f(0) = \text{conv}(\{1\} \cup \{-1\}) = \text{conv}(\{-1, 1\}) = [-1, 1] \quad \checkmark$$

- At $x > 0$: only $f_1(x) = x$ active, $I(x) = \{1\}$

$$\partial f(x) = \{1\} \quad \checkmark$$

- At $x < 0$: only $f_2(x) = -x$ active, $I(x) = \{2\}$

$$\partial f(x) = \{-1\} \quad \checkmark$$

Summary: Subdifferential Calculus Toolkit

Rule	**Formula**
Sum	$\partial(f_1 + f_2)(x) = \partial f_1(x) + \partial f_2(x)$
Scaling	$\partial(af)(x) = a\,\partial f(x) \quad (a > 0)$
Smooth	$\partial f(x) = \{\nabla f(x)\}$
Affine	$\partial(f \circ (Ax + b))(x) = A^T \partial f(Ax + b)$
Chain	$\partial(g \circ f)(x) = g'(f(x))\,\partial f(x)$
Max	$\partial \max_i f_i(x) = \operatorname{conv}\big(\bigcup_{i \in I(x)} \partial f_i(x)\big)$

These rules, combined with the basic examples ($|x|$, $\|\cdot\|_1$, $\|\cdot\|_2$, ι_C), suffice for many problems in machine learning.

Example: Subdifferential of ℓ_1-Regularized Hinge Loss

$$f(\omega) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}(1 - y_i\omega^T x_i)_+}_{\text{hinge loss}} + \underbrace{\lambda\|\omega\|_1}_{\ell_1 \text{ regularizer}}$$

- $y_i \in \{-1, +1\}$: labels
- $x_i \in \mathbb{R}^d$: features
- $(1 - y_i\omega^T x_i)_+$: penalizes margin violations
- $\lambda\|\omega\|_1$: promotes sparsity

This is the **hinge loss with ℓ_1 regularization**, related to linear SVMs with feature selection.

Goal: Compute $\partial f(\omega)$ using our calculus rules.

Step 1: Subdifferential of the Regularizer

We already computed:

$$\partial(\lambda\|\omega\|_1) = \lambda\,\partial\|\omega\|_1$$

By the separable sum structure:

$$[\partial(\lambda\|\omega\|_1)]_j = \lambda\ \mathsf{sign}(\omega_j) = \begin{cases} \{-\lambda\} & \omega_j < 0 \\ [-\lambda, \lambda] & \omega_j = 0 \\ \{\lambda\} & \omega_j > 0 \end{cases}$$

Step 2: Subdifferential of a Single Hinge Term

Consider $g_i(\omega) = (1 - y_i \omega^T x_i)_+$.

Write $g_i(\omega) = h(y_i \omega^T x_i)$ where $h(z) = (1 - z)_+$.

From an example last lecture (or use the max rule),

$$\partial h(z) = \begin{cases} \{-1\} & z < 1 \\ [-1, 0] & z = 1 \\ \{0\} & z > 1 \end{cases}$$

The map $\omega \mapsto y_i \omega^T x_i$ is affine: $z = A\omega$ with $A = y_i x_i^T$.
By the **affine transformation rule**:

$$\partial g_i(\omega) = (y_i x_i) \cdot \partial h(y_i \omega^T x_i)$$

Step 2 (continued): Three Regimes

$$\partial g_i(\omega) = y_i x_i \cdot \begin{cases} \{-1\} & y_i \omega^T x_i < 1 \\ [-1, 0] & y_i \omega^T x_i = 1 \\ \{0\} & y_i \omega^T x_i > 1 \end{cases}$$

Interpretation:

- $y_i \omega^T x_i < 1$: margin violated $\Rightarrow$ contributes $-y_i x_i$
  Pushes ω in direction $y_i x_i$ to improve classification
- $y_i \omega^T x_i = 1$: on the margin boundary $\Rightarrow$ contributes any $v \in [-y_i x_i, 0]$
- $y_i \omega^T x_i > 1$: correctly classified with margin $\Rightarrow$ contributes 0
  Inactive constraint, no influence on the subdifferential

Step 3: Combine Using the Sum Rule

By the sum rule and positive scaling:

$$\partial f(\omega) = \frac{1}{n}\sum_{i=1}^{n} \partial g_i(\omega) \; + \; \lambda \, \partial \|\omega\|_1$$

Expanding:

$$\partial f(\omega) = \frac{1}{n}\sum_{i=1}^{n} y_i x_i \cdot \underbrace{\begin{cases} \{-1\} & y_i \omega^T x_i < 1 \\ [-1, 0] & y_i \omega^T x_i = 1 \\ \{0\} & y_i \omega^T x_i > 1 \end{cases}}_{\text{contribution from example } i} + \lambda \sum_{j=1}^{d} e_j \cdot \text{sign}(\omega_j)$$

Computing $\partial \iota_C(x)$

We saw that constrained convex optimization problems can be converted to unconstrained ones using convex indicator functions

$$\min_{x \in C} f(x) \Leftrightarrow \min_{x} f(x) + \iota_C(x).$$

The optimality conditions for constrained optimization problems are stated in terms of $\partial \left(f(x) + \iota_C(x) \right)$.

Using the sum rule,

$$\partial \left(f(x) + \iota_C(x) \right) = \partial f(x) + \partial \iota_C(x)$$

We'll now compute $\partial \iota_C(x)$. It turns out to have a compelling geometric interpretation

Step 1: When x is not in C

Suppose $x \notin C$. We want to find a subgradient v such that:

$$\iota_C(y) \geq \iota_C(x) + \langle v, y - x \rangle \quad \forall y \in \mathbb{R}^n$$

- By definition, since $x \notin C$, then $\iota_C(x) = \infty$.
- Pick any $y \in C$, so $\iota_C(y) = 0$. The requirement becomes:

$$0 \geq \infty + \langle v, y - x \rangle$$

- No finite vector v can satisfy this, as ∞ plus any real number remains ∞.

Conclusion: For any $x \notin C$, the subdifferential $\partial \iota_C(x) = \emptyset$.

Step 2: When x is in C

If $x \in C$, then $\iota_C(x) = 0$. To find $v \in \partial\iota_C(x)$, we check two cases for y:

1. **Case $y \notin C$:** $\iota_C(y) = \infty$. The condition $\infty \geq 0 + \langle v, y - x \rangle$ is always true for any v.
2. **Case $y \in C$:** $\iota_C(y) = 0$. The condition $0 \geq 0 + \langle v, y - x \rangle$ must hold.

This leaves us with one requirement for v to be a subgradient:

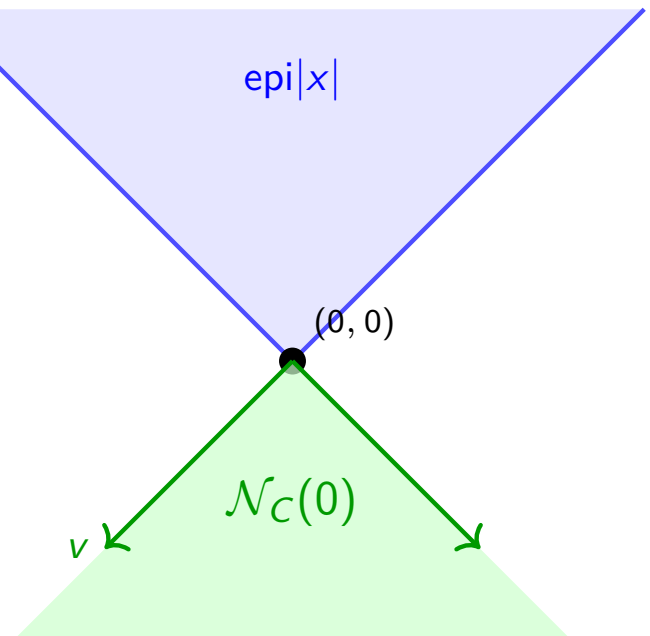
$$\langle v, y - x \rangle \leq 0 \quad \text{for all } y \in C$$

Geometric Interpretation: This means v must make an **obtuse angle** with every direction pointing from x into the set C (aka every **feasible direction**).

Step 3: The Normal Cone $\mathcal{N}_C(x)$

We define $\partial \iota_C(x)$ for $x \in C$ as the **Normal Cone** $\mathcal{N}_C(x)$

- **Cone:** If v satisfies the inequality, so does αv for any $\alpha \geq 0$.
- **Normal:** If v satisfies the inequality, then it forms an obtuse angle (is perpendicular, or more) to any feasible direction from x.



Subdifferential of the Convex Indicator ι_C

Putting it all together, for a convex C:

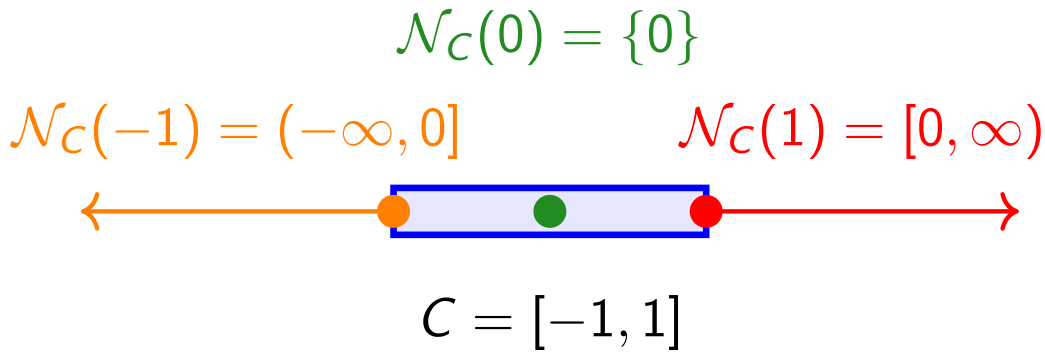
$$\partial \iota_C(x) = \begin{cases} \emptyset & x \notin C \\ \mathcal{N}_C(x) & x \in C \end{cases}$$

where $\mathcal{N}_C(x) = \{v : \langle v, y - x \rangle \leq 0 \text{ for all } y \in C\}$ is the **normal cone**.

Geometric interpretation: $\mathcal{N}_C(x)$ consists of all vectors pointing "outward" from C at x. It is the set of normals to all the supporting hyperplanes of C at x.

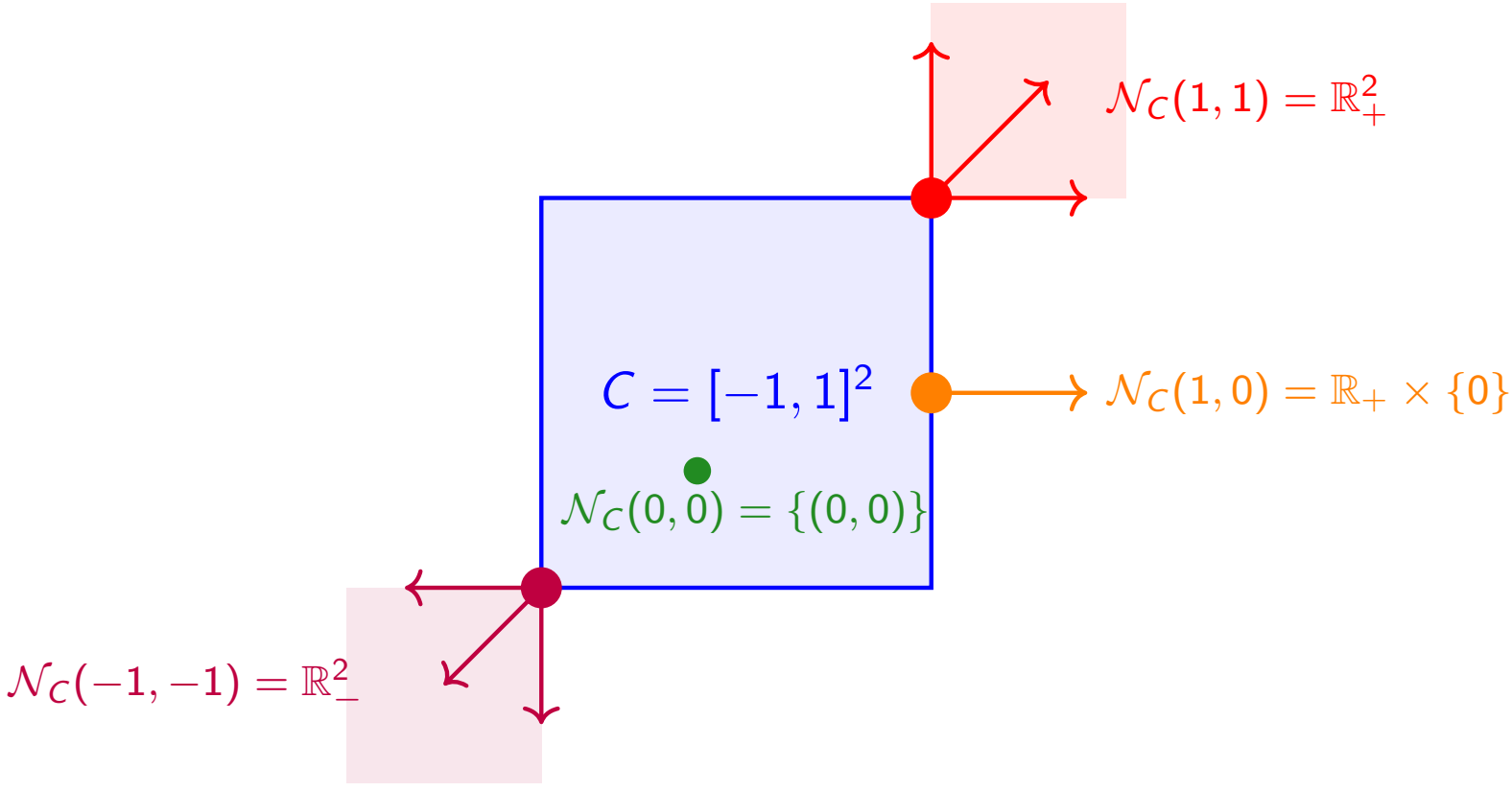
Normal Cone of the Interval [-1, 1]

For $C = [-1, 1]$ on the real line:



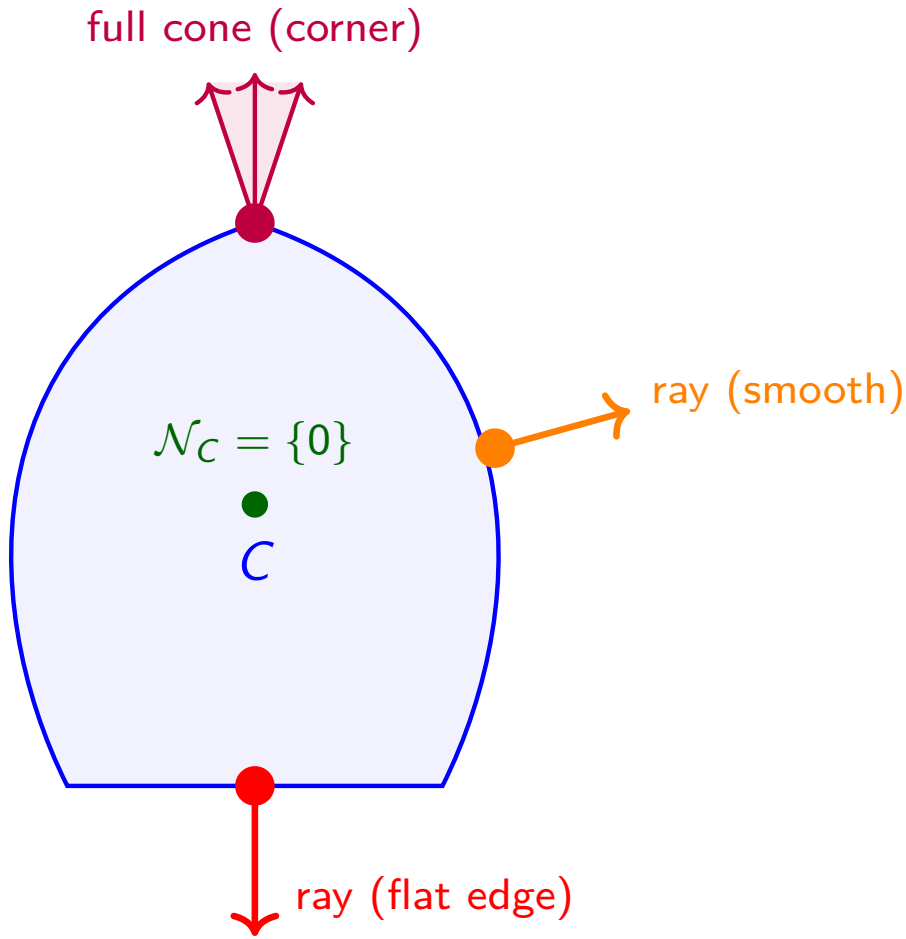
Interior points: only the zero vector is "outward."
Boundary points: half-lines pointing away from C.

Normal Cone of the Square $[-1,1]^2$



Corners get conic normal cones; edges get ray normal cones; interior gets $\{0\}$.

Normal Cones on a General Convex Set



Takeaway: At smooth points or flat edges, the normal cone is just a **ray** (one direction). At sharp corners, the normal cone has **interior** (multiple directions). At interior points, the subdifferential is just $\{0\}$.

Classical Case: Differentiable Optimality

For differentiable convex f, Fermat's condition:

$$x^* \in \operatorname*{argmin}_x f(x) \iff \nabla f(x^*) = 0$$

Geometric meaning: x^* is optimal iff the tangent hyperplane to the graph of f at x^* is **horizontal**.

What replaces this when f is not differentiable?

Nonsmooth Fermat's Condition (Unconstrained)

Theorem (Fermat's Condition)

For proper, lsc, convex f:

$$x^* \in \operatorname*{argmin}_x f \iff 0 \in \partial f(x^*)$$

Geometric interpretation: x^* is optimal iff there exists a **horizontal supporting hyperplane** at x^*.

Proof. By definition, x^* is a global minimizer iff for every $y \in \operatorname{dom}(f)$,

$$f(y) \geq f(x^*) \Leftrightarrow f(y) \geq f(x^*) + \langle 0, y - x^* \rangle \Leftrightarrow 0 \in \partial f(x^*).$$

□

Fermat's Condition: Constrained Problems

For $\min_{x \in C} f(x)$ with C closed convex, rewrite as $\min_x f(x) + \iota_C(x)$.

Apply Fermat's condition with the sum rule:

$$0 \in \partial f(x^*) + \partial \iota_C(x^*) = \partial f(x^*) + \mathcal{N}_C(x^*)$$

When f is differentiable, this simplifies to

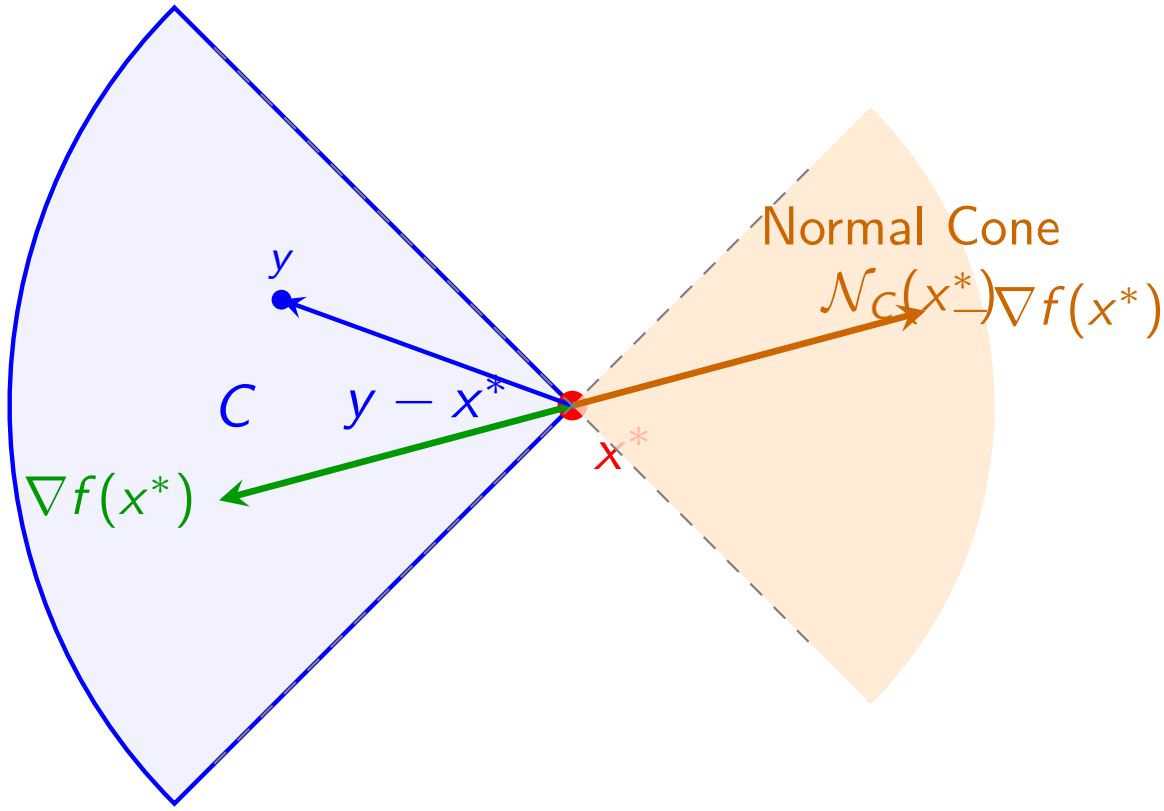
$$-\nabla f(x^*) \in \mathcal{N}_C(x^*)$$

which means:

$$\langle \nabla f(x^*), y - x^* \rangle \geq 0 \quad \forall\, y \in C$$

This is the **variational inequality** characterization of constrained optimality.

Geometric Picture of Constrained Optimality



$$-\nabla f(x^*) \in \mathcal{N}_C(x^*) \iff \langle \nabla f(x^*), y - x^* \rangle \geq 0 \quad \forall y \in C$$

Intuition: $\nabla f(x^*)$ points into the set, ensuring that all feasible moves from x^* increase f.

Example: Constrained Quadratic

Problem: Minimize $f(x) = x^2$ over $C = [1, 2]$.

Unconstrained minimum: $x = 0 \notin C$, so the optimum is on the boundary. (**Justify this claim!**)

Check Fermat's condition at $x^* = 1$:

- $\nabla f(1) = 2$, so $-\nabla f(1) = -2$
- $\mathcal{N}_{[1,2]}(1) = (-\infty, 0]$
- Check: $-2 \in (-\infty, 0]$ ✓

Check at $x = 2$:

- $-\nabla f(2) = -4$
- $\mathcal{N}_{[1,2]}(2) = [0, \infty)$
- Check: $-4 \notin [0, \infty)$ × (not optimal)

So $x^* = 1$ is the unique minimizer, confirmed by subdifferential calculus.

The Big Picture: Smooth vs Nonsmooth Optimality

	Gradient $\nabla f(x)$	**Subdifferential** $\partial f(x)$
Exists when	f differentiable	f proper, lsc, convex $x \in \text{int}(\text{dom}(f))$
Size	single vector	closed convex set
Optimality	$\nabla f(x^*) = 0$	$0 \in \partial f(x^*)$
Constrained	$-\nabla f(x^*) \in \mathcal{N}_C(x^*)$	$0 \in \partial f(x^*) + \mathcal{N}_C(x^*)$
Algorithm	gradient descent	subgradient descent

The subdifferential framework **strictly generalizes** smooth calculus: when f is differentiable, all subdifferential results reduce to their gradient counterparts.

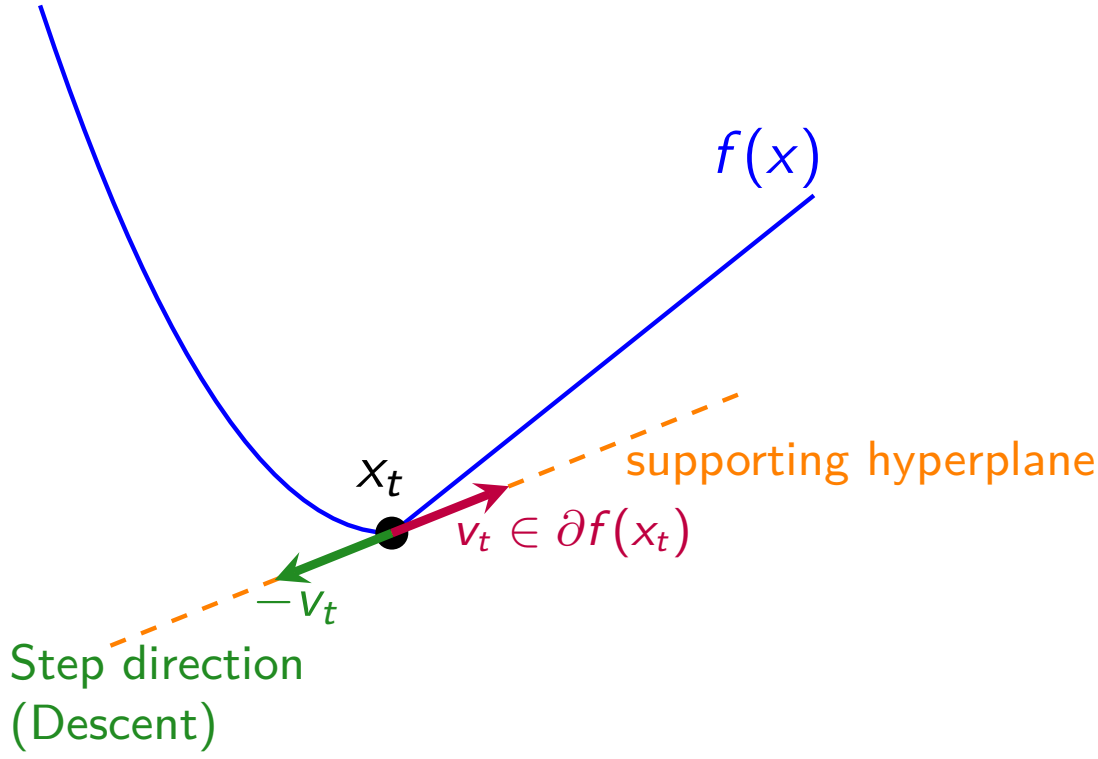
Algorithmic Consequence

In practice: Algorithms like subgradient descent select an element $v_t \in \partial f(\omega_t)$ at each iteration and update the iterate by a multiple of $-v_t$.

Computational note: Subdifferentials are sets, algorithms typically work with **a single subgradient** at each step.

- The choice of element can affect algorithm behavior
- But *any* choice is valid theoretically

Why Subgradient Steps Make Sense (Geometry)



The subgradient v_t defines a **linear lower bound**. While $-v_t$ is not necessarily a descent direction, it moves the iterate closer to the set of minimizers:

$$0 \geq f(x^*) - f(x_t) \geq \langle v_t, x^* - x_t \rangle \Rightarrow \langle -v_t, x^* - x_t \rangle \geq 0.$$

Subgradient Descent Algorithm

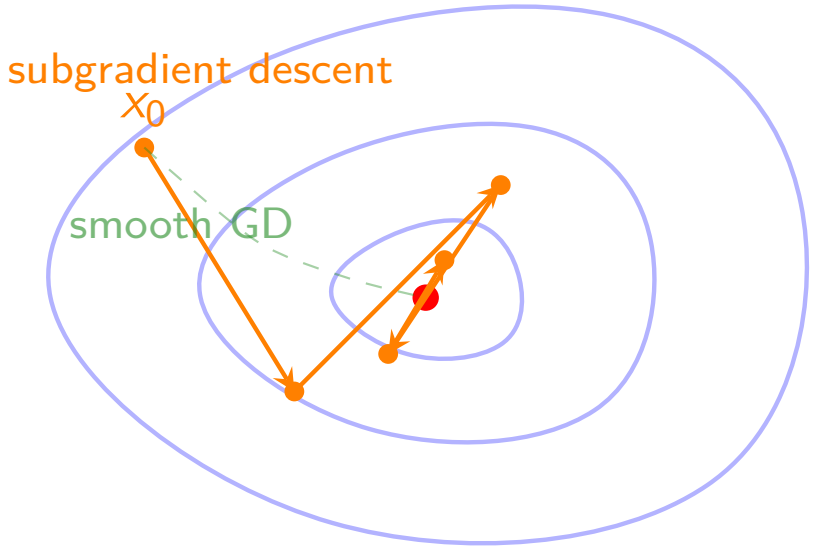
Subgradient descent is the natural extension of gradient descent to nonsmooth convex functions.

Algorithm:

1. Initialize $x_0 \in \mathbb{R}^d$
2. For $t = 0, 1, 2, \ldots$:
   - Pick any $v_t \in \partial f(x_t)$
   - Update: $x_{t+1} = x_t - \eta_t v_t$

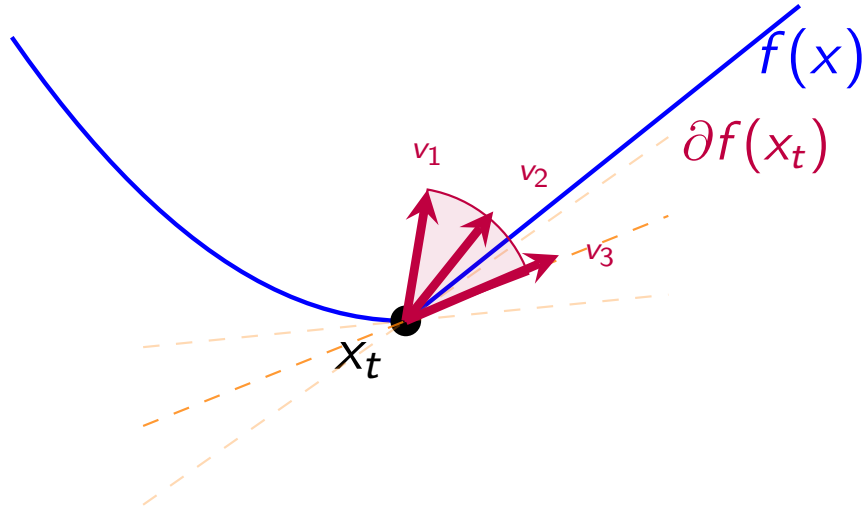
v_t is not unique when f is nonsmooth, but *any* choice from $\partial f(x_t)$ is valid.

Subgradient Descent Is Slow



Nonsmoothness prevents curvature exploitation.
Explains why rates are $O(1/\sqrt{T})$ instead of $O(1/T)$.

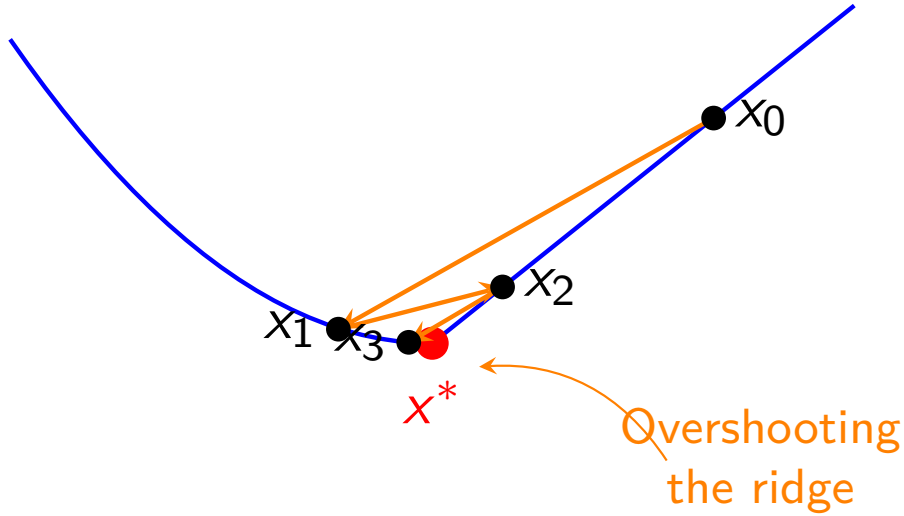
Geometry of Kinks



Nonsmoothness Causes Bouncing

- **Non-Uniqueness:** $\partial f(x_t)$ is the *set* of all slopes that stay below the function.
- **Inconsistent Steps:** Since the algorithm picks only one $v \in \partial f(x_t)$, the direction may not point toward the minimizer.
- **Curvature:** Without a smooth gradient to steer, the iterates oscillate across the ridge.

The Zig-Zag Path to Convergence



- **Bouncing:** Because the subgradient changes sign abruptly at the kink, the algorithm hunts for the minimum by jumping across it.
- **Step Size is Key:** Unlike smooth GD, the subgradient doesn't go to zero at the minimum. We **must** diminish the step size ($\alpha_t \to 0$) to actually stay at x^*. **Subgradients give direction, not length.**
- **Efficiency:** This oscillation is why we observe a slower $O(1/\sqrt{T})$ convergence rate.

Subgradient Descent Convergence Rate

Theorem (Subgradient Descent for Convex Functions)

Let f be convex and assume $\|g\| \leq G$ for all $g \in \partial f(x)$. Assume $\|x_0 - x^*\| \leq R$.
If we run subgradient descent with step sizes

$$\eta_t = \frac{R}{G\sqrt{T}} \quad \text{for } t = 1, \ldots, T,$$

then

$$\min_{t \leq T} f(x_t) - f(x^*) \leq \frac{RG}{\sqrt{T}}.$$

Compare to smooth convex GD: $O(1/T)$ with constant step size.
Nonsmoothness costs a factor of $\sqrt{T}$. This is the optimal rate.

What Changes When We Add Constraints?

So far, our iterates live in $\mathbb{R}^d$:

$$x_{t+1} = x_t - \eta_t v_t, \qquad v_t \in \partial f(x_t).$$

Now suppose the problem is constrained:

$$\min_{x \in C} f(x), \qquad C \text{ closed, convex.}$$

Immediate problem:

- The update $x_t - \eta_t v_t$ generally leaves C
- Once we leave C, the objective $f(x)$ is no longer even defined (via ι_C)

We need a mechanism that:

- Enforces feasibility ($x_{t+1} \in C$)
- Preserves the geometry needed in the convergence proof
- Is computationally simple for common sets

Projections– a new primitive

This mechanism is: projection onto C.

$$x_{t+1} = \min_{z \in C} \|\tilde{x}_{t+1} - z\|_2^2$$

Projections serve three roles:

1. **Algorithmic:** enforce feasibility in constrained optimization
2. **Geometric:** find the closest feasible point to a given point
3. **Analytic:** non-expansive mapping $\Rightarrow$ convergence proofs

Projections connect geometry, computation, and theory.

Projection onto a Convex Set

Definition

The **projection** of x onto a closed convex set C is:

$$P_C(x) = \operatorname*{argmin}_{z \in C} \|x - z\|_2^2$$

Geometric interpretation: Find the closest point in C to x.

The objective $\frac{1}{2}\|x - z\|_2^2$ is strongly convex in z, so the projection always exists and is unique.

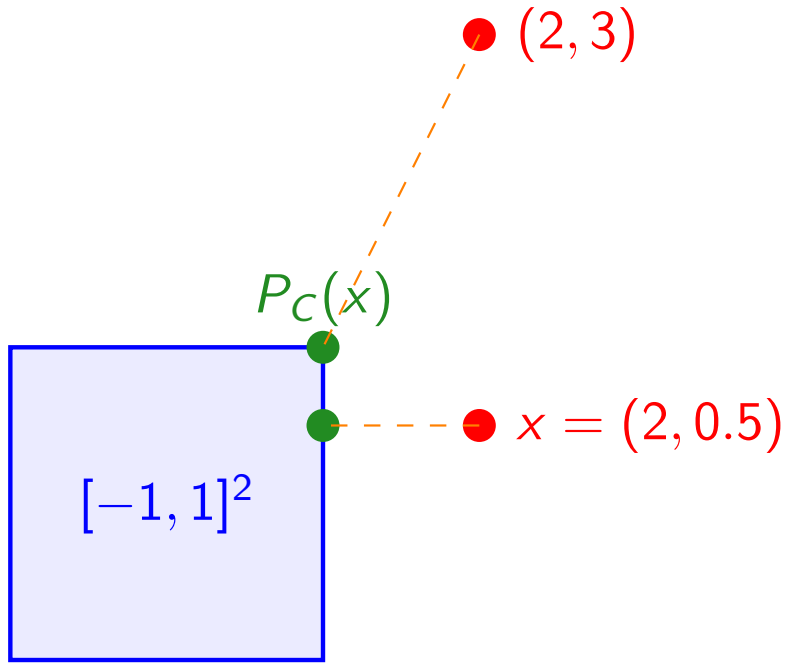
Strategy: Reformulate as unconstrained, apply Fermat's condition, solve.

Example: Projection onto the ℓ_∞ Ball

Problem:

$$P_C(x) = \underset{\|z\|_\infty \leq 1}{\operatorname{argmin}} \|x - z\|_2^2$$

where $C = \{z : \|z\|_\infty \leq 1\} = [-1, 1]^d$.



Converting to Unconstrained Form

$$z^* = P_C(x) = \operatorname*{argmin}_z \frac{1}{2}\|x - z\|_2^2 + \iota_C(z)$$

Fermat's condition (sum rule):

$$0 \in \nabla_z \left(\frac{1}{2}\|x - z\|_2^2 \right) \Big|_{z=z^*} + \partial \iota_C(z^*)$$

$$0 \in (z^* - x) + \mathcal{N}_C(z^*)$$

Since $C = [-1, 1]^d$ is a product set and the objective separates coordinate-wise, we can solve **independently for each coordinate**:

$$0 \in z_i^* - x_i + \mathcal{N}_{[-1,1]}(z_i^*)$$

Normal Cone of $[-1, 1]$ (Recall)

$$\mathcal{N}_{[-1,1]}(z) = \begin{cases} (-\infty, 0] & z = -1 \\ \{0\} & z \in (-1, 1) \\ [0, \infty) & z = 1 \end{cases}$$

The condition $0 \in z_i^* - x_i + \mathcal{N}_{[-1,1]}(z_i^*)$ gives three cases to check.

Solution by Cases

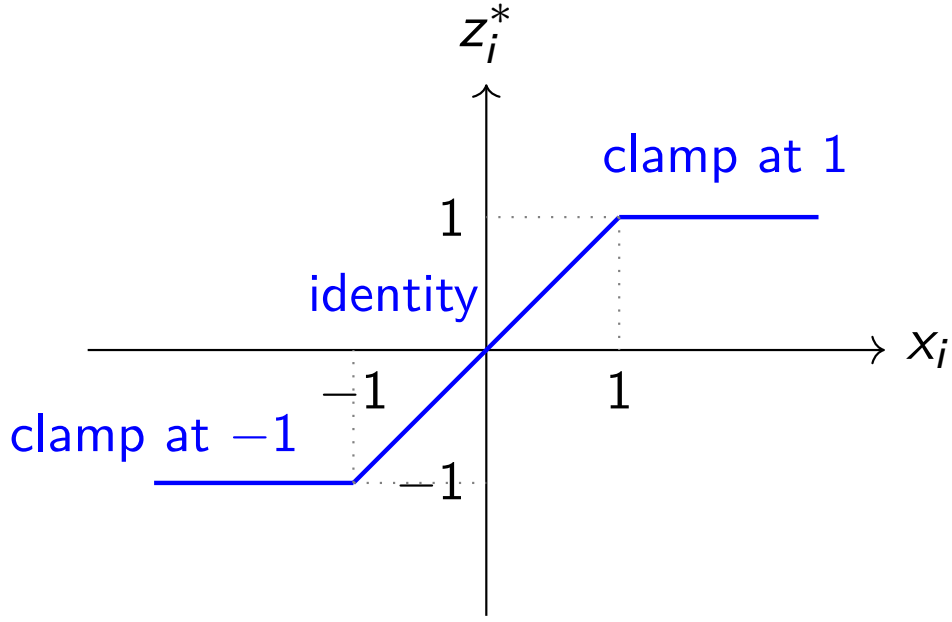
Case 1: $z_i^* \in (-1, 1)$ (interior)
$\mathcal{N}_{[-1,1]}(z_i^*) = \{0\}$, so $0 = z_i^* - x_i$, giving $z_i^* = x_i$.
This requires $x_i \in (-1, 1)$.

Case 2: $z_i^* = 1$ (right boundary)
$0 \in 1 - x_i + [0, \infty) \iff x_i - 1 \geq 0 \iff x_i \geq 1.$

Case 3: $z_i^* = -1$ (left boundary)
$0 \in -1 - x_i + (-\infty, 0] \iff -1 - x_i \geq 0 \iff x_i \leq -1.$

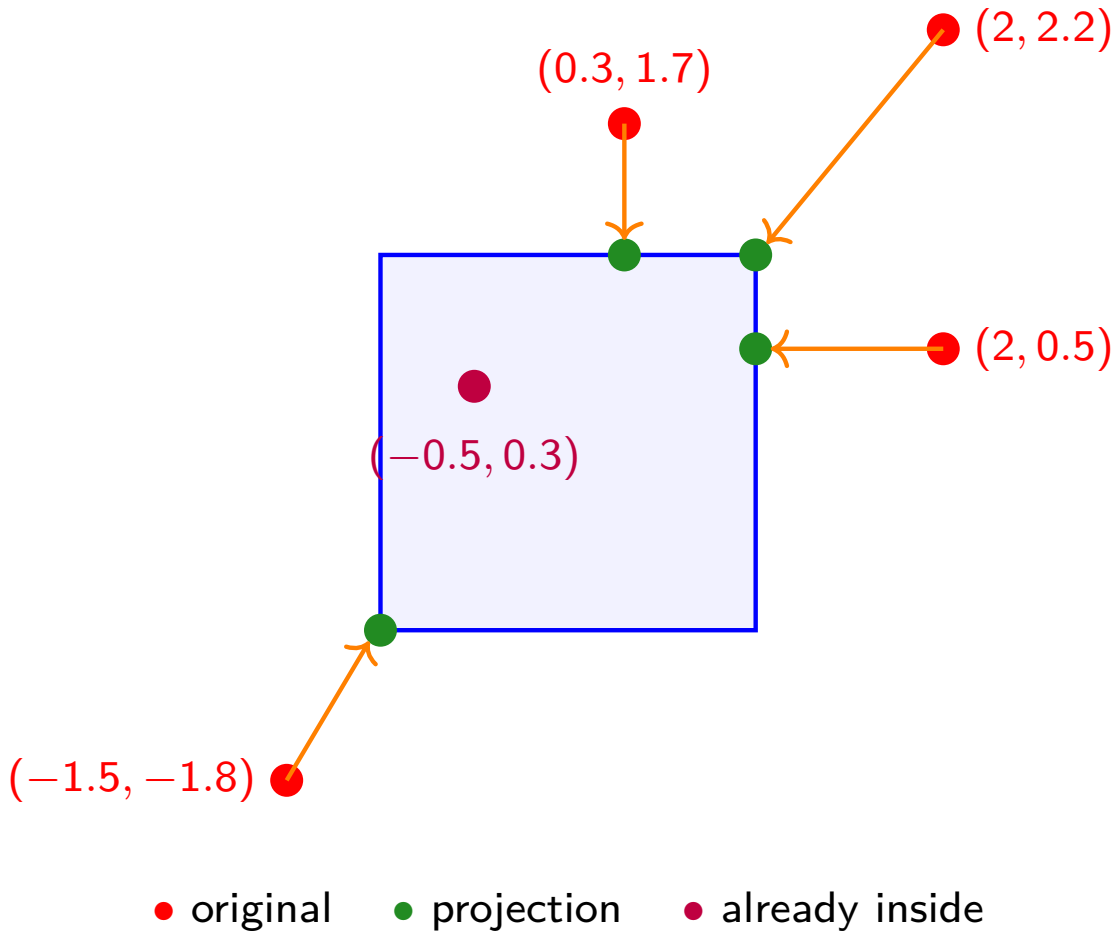
Result: Coordinate-wise Clipping

$$z_i^* = [P_C(x)]_i = \begin{cases} -1 & x_i < -1 \\ x_i & x_i \in [-1, 1] \\ 1 & x_i > 1 \end{cases} = \text{clip}(x_i, -1, 1)$$



Fermat's condition + subdifferential calculus $\Rightarrow$ **closed-form solution** for the projection!

Projection onto ℓ_∞ Ball: 2D Visualization



Projected Subgradient Descent Algorithm

For constrained problems $\min_{x \in C} f(x)$:

Algorithm:

1. Initialize $x_0 \in C$
2. For $t = 0, 1, 2, \ldots$:
   - Pick any $v_t \in \partial f(x_t)$
   - Unconstrained update: $\tilde{x}_{t+1} = x_t - \eta_t v_t$
   - Project back: $x_{t+1} = P_C(\tilde{x}_{t+1})$

Take a subgradient step, then project back onto the feasible set.
We assume the projection step can be solved easily.

Convergence of Projected Subgradient Descent

Theorem

Let f be convex and assume $\|g\| \leq G$ for all $g \in \partial f(x)$ and all $x \in C$. Assume C is closed, convex, and $\|x_0 - x^*\| \leq R$.
Run projected subgradient descent:

$$x_{t+1} = P_C(x_t - \eta_t g_t), \qquad g_t \in \partial f(x_t).$$

With step sizes

$$\eta_t = \frac{R}{G\sqrt{T}} \quad \text{for } t = 1, \ldots, T,$$

we have

$$\min_{s \leq T} f(x_s) - f(x^*) \;\leq\; \frac{RG}{\sqrt{T}}.$$

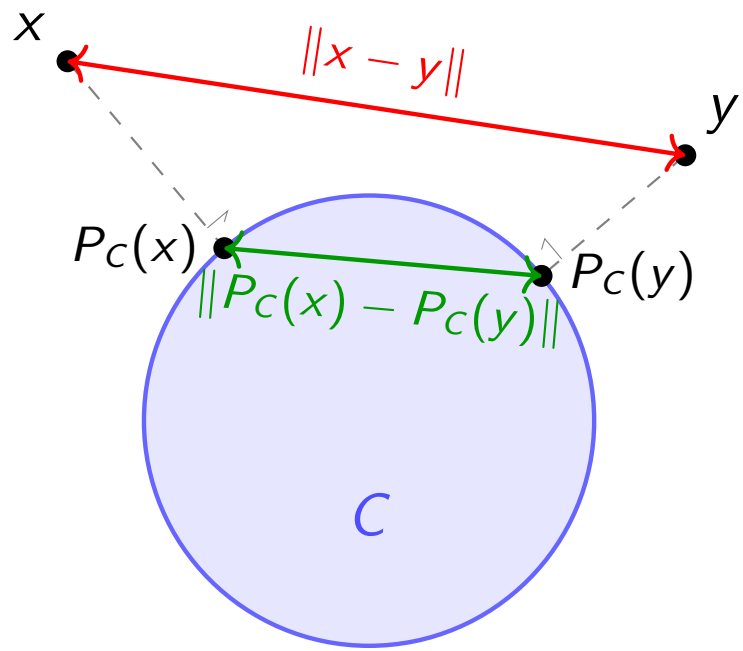
Projection does *not* worsen the rate!

Projections shrink distances

Proposition (Non-expansiveness)

For any $x, y \in \mathbb{R}^d$ and closed convex C:

$$\|P_C(x) - P_C(y)\|_2 \leq \|x - y\|_2$$



Intuition: Projecting two points onto C cannot increase their distance. So the projection step doesn't hurt convergence to minimizer.

The Denoising Problem

Setup: We observe a noisy image $y = x_{\text{true}} + \varepsilon$, where $\varepsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$.

Natural images are approximately **piecewise constant** — most neighboring pixels have similar intensities, with sharp transitions at edges.

This motivates the **total variation (TV) denoising** problem:

$$\min_{x \in [0,1]^d} \underbrace{\frac{1}{2}\|x - y\|^2}_{\text{fidelity}} + \underbrace{\lambda \sum_{(i,j)\ \text{adjacent}} |x_i - x_j|}_{\text{anisotropic TV}}$$

- **Fidelity:** Stay close to the observed data y
- **TV penalty:** Promote piecewise-constant structure (sparse differences)
- **Box constraint:** Pixel intensities must lie in $[0, 1]$

Computing the subdifferential

1. **Smooth part:** $\frac{1}{2}\|x-y\|^2$ is differentiable $\Rightarrow$ contributes gradient $(x-y)$
2. **TV part:** $\sum_{i\sim j}|x_i - x_j| = \sum_{i\sim j}|a_{ij}^T x|$ where $a_{ij} = e_j - e_i$
   By the **sum rule** + **affine transformation rule**:

$$\sum_{i\sim j} \text{sign}(a_{ij}^T x)\, a_{ij} \in \partial \text{TV}(x)$$

3. **Box constraint:** $x \in [0,1]^d \Rightarrow$ projection is **coordinate clipping**
   (See the derivation of Fermat's condition on the ℓ_∞ ball earlier)
4. **Overall:** By the **sum rule**, the full subgradient is

$$v = \underbrace{(x-y)}_{\nabla \text{fidelity}} + \underbrace{\lambda \cdot g_{\text{TV}}}_{\in \partial(\lambda \text{TV}(x))}$$

The Algorithm

Projected subgradient descent with Robbins–Monro step sizes:

1. Initialize: $x_0 = \text{clip}(y, 0, 1)$
2. For $t = 0, 1, 2, \ldots$:
   - Compute $v_t = (x_t - y) + \lambda \cdot g_{\text{TV}}(x_t)$
   - Update: $\tilde{x}_{t+1} = x_t - \eta_t v_t$ where $\eta_t = \frac{c}{(t+1)^\alpha}$
   - Project: $x_{t+1} = \text{clip}(\tilde{x}_{t+1}, 0, 1)$
3. Return: $x_{\text{best}} = \arg\min_{s \le T} f(x_s)$

Important: We track the **best iterate**, not the last one!
The objective is **not monotonically decreasing** — subgradient descent oscillates, and the best value may occur at any iterate.

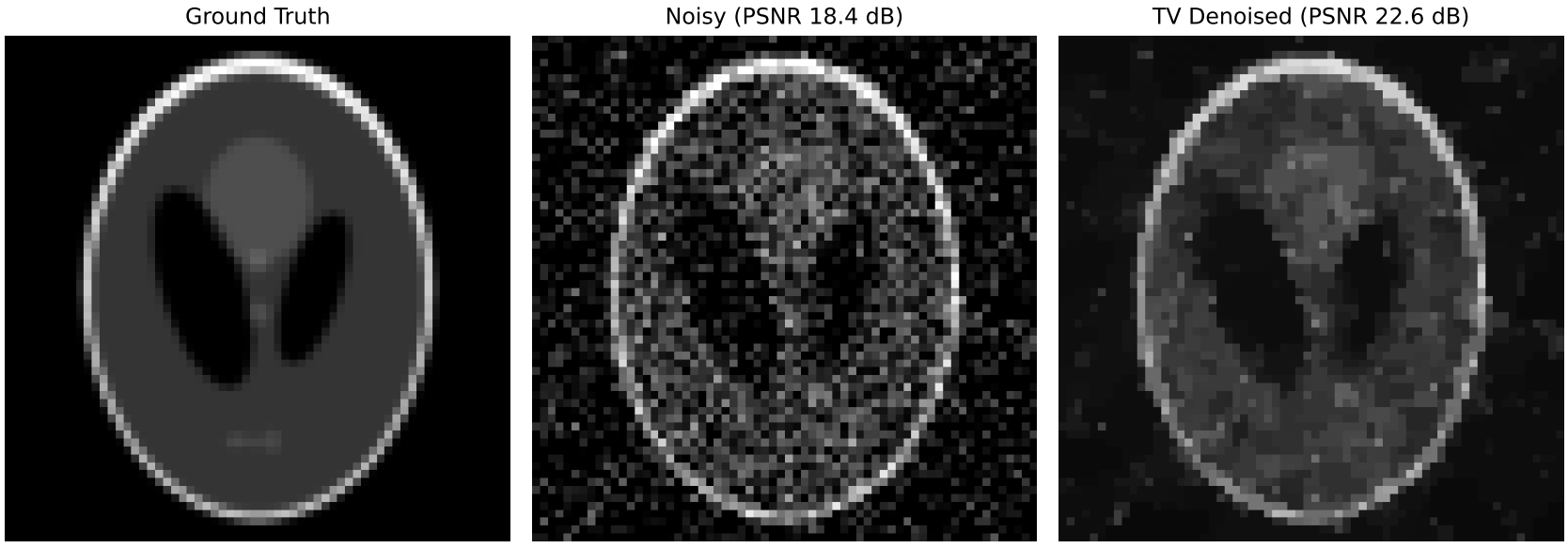
Test Problem: Shepp–Logan Phantom

The **Shepp–Logan phantom** is a standard test image in medical imaging, designed to mimic a cross-section of a human head.

- 64×64 image ($d = 4096$ pixels, 8064 edges)
- Gaussian noise: $\sigma = 0.15$
- Regularization: $\lambda = 0.05$
- Step size: $\eta_t = 0.05/(t+1)^{0.51}$
- Budget: $T = 300{,}000$ iterations

The phantom is piecewise constant with elliptical boundaries — an ideal target for TV denoising.

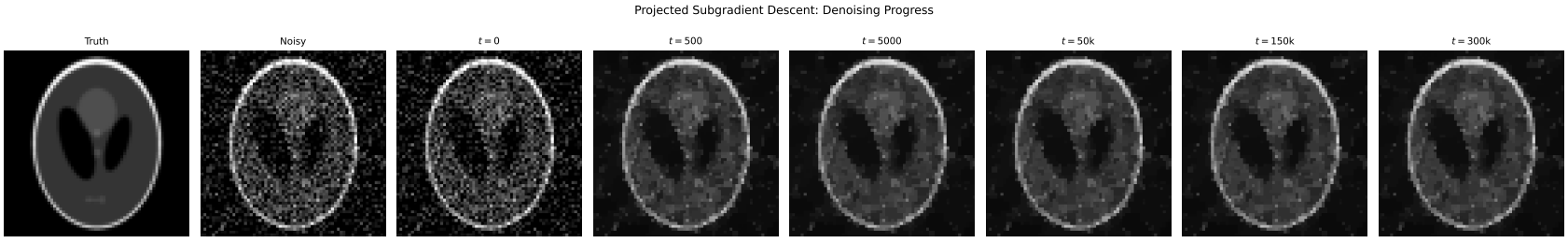
Result: Denoising the Shepp–Logan Phantom



	MSE	**PSNR**
Noisy input	0.01446	18.4 dB
TV denoised	0.00550	22.6 dB

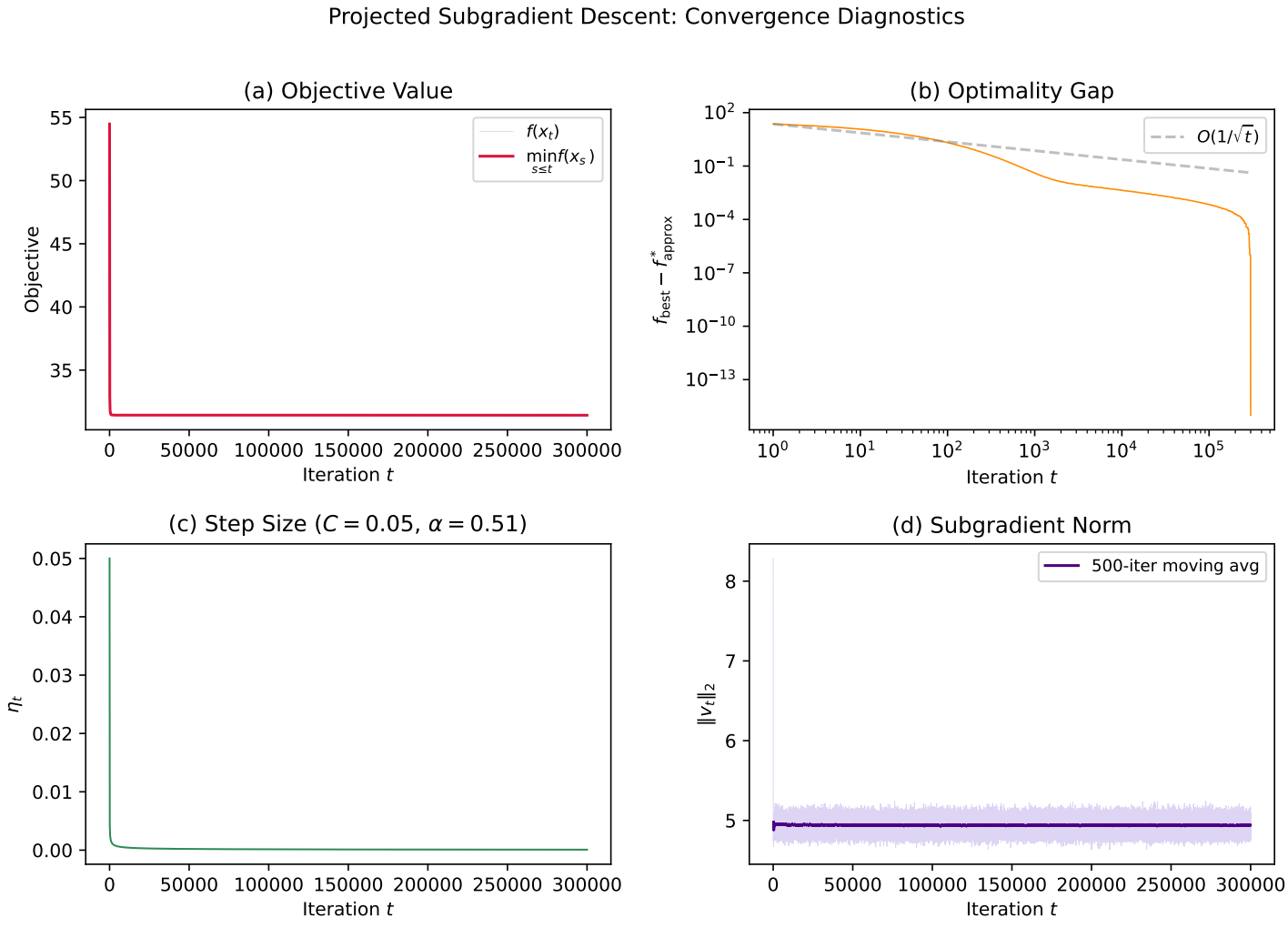
62% MSE reduction and +4.2 dB PSNR gain, using only the tools from this lecture.

Watching Convergence



Structure emerges gradually. Most improvement happens in the first 5000 iterations; the remaining 295,000 refine details. This is characteristic of the $O(1/\sqrt{T})$ rate.

Convergence Diagnostics



What the Diagnostics Reveal

1. **Panel (b):** The optimality gap decays **faster than** $O(1/\sqrt{T})$ — the theory gives a worst-case bound, but this problem is better-behaved. (The smooth fidelity term helps.)
2. **Panel (c):** The step size $\eta_t = C/(t+1)^\alpha$ decays slowly ($\alpha = 0.51 \approx 1/2$), satisfying both Robbins–Monro conditions.
3. **Panel (d):** The subgradient norm $\|v_t\|$ stays **bounded** (≈ 5), consistent with the convergence theorem's assumption $\|v\| \leq G$.

Why So Slow?

300,000 iterations for a 64×64 image is a lot!

The bottleneck: Subgradient descent does not exploit the smoothness of the fidelity term $\frac{1}{2}\|x - y\|^2$. It treats the entire objective as nonsmooth.

Key observation: Our objective is $f(x) = \underbrace{g(x)}_{\text{smooth}} + \underbrace{h(x)}_{\text{nonsmooth}}$

More sophisticated **composite optimization** algorithms take advantage of this structure, so can leverage the fact that g is smooth.

They can recover the $O(1/T)$ rate of smooth gradient descent while handling nonsmoothness. The same denoising problem could converge in $\sim$100 iterations instead of 300,000.

What We Learned

1. **Subdifferential calculus rules** (sum, scalar, affine, chain, max) enable practical computation of subdifferentials for composite objectives
2. **Normal cones** and the subdifferential of convex indicator functions
3. **Fermat's condition** $0 \in \partial f(x^*)$ characterizes optimality, even for constrained nonsmooth problems via normal cones
4. **Subgradient descent** converges at $O(1/\sqrt{T})$ rate for convex functions
5. **Projections** onto convex sets are non-expansive and enable constrained optimization

Questions?

Next time: line search methods and Newton's method

Appendix Contents

1. Proof of Non-expansiveness of Convex Projections
2. Proof of Convergence Theorem for Projected Gradient Descent

Non-expansiveness of Euclidean Projection

Theorem

Let $C \subseteq \mathbb{R}^n$ be a closed convex set. The projection $P_C : \mathbb{R}^n \to C$ is non-expansive:

$$\|P_C(y) - P_C(z)\| \leq \|y - z\| \quad \textit{for all } y, z \in \mathbb{R}^n.$$

Intuition: We'll use the optimality condition for the projection: the vector from the projection to any other point in C forms an obtuse angle with the vector from the projection back to the original point.

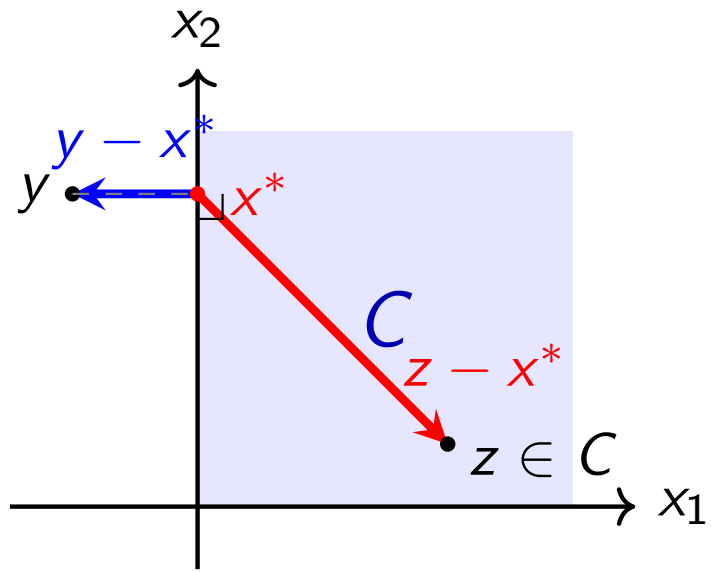
Characterization of Projection

Lemma (First-order optimality condition for projection)

A point $x^* \in C$ is the projection of y onto C if and only if

$$\langle y - x^*, z - x^* \rangle \leq 0 \quad \textit{for all } z \in C.$$

Geometric interpretation: For any other point $z \in C$, the angle between $(y - x^*)$ and $(z - x^*)$ is at least 90.



Proof Strategy

Goal: Show $\|P_C(y) - P_C(z)\| \leq \|y - z\|$ for arbitrary $y, z \in \mathbb{R}^n$.

Plan:

1. Let $y^* = P_C(y)$ and $z^* = P_C(z)$.
2. Apply the optimality condition twice: once for y^* (testing against z^*), and once for z^* (testing against y^*).
3. Add these two inequalities to get a relationship between $\|y^* - z^*\|^2$ and $\langle y - z, y^* - z^* \rangle$.
4. Use Cauchy-Schwarz to conclude.

Proof: Applying the optimality conditions

Proof.

Let $y^* = P_C(y)$ and $z^* = P_C(z)$.

Since $y^* = P_C(y)$ and $z^* \in C$, the optimality condition gives:

$$\langle y - y^*, z^* - y^* \rangle \leq 0.$$

Similarly, since $z^* = P_C(z)$ and $y^* \in C$:

$$\langle z - z^*, y^* - z^* \rangle \leq 0.$$

□

Proof: Rearranging the inequalities

Proof (continued).

Rewrite the first inequality:

$$\langle y - y^*, z^* - y^* \rangle \leq 0 \quad \Rightarrow \quad \langle y, z^* - y^* \rangle \leq \langle y^*, z^* - y^* \rangle.$$

Rewrite the second inequality:

$$\langle z - z^*, y^* - z^* \rangle \leq 0 \quad \Rightarrow \quad \langle z, y^* - z^* \rangle \leq \langle z^*, y^* - z^* \rangle.$$

Note that $\langle z, y^* - z^* \rangle = -\langle z, z^* - y^* \rangle$. □

Proof: Combining the inequalities

Proof (continued).

Add the two inequalities:

$$\langle y, z^* - y^* \rangle + \langle z, y^* - z^* \rangle \leq \langle y^*, z^* - y^* \rangle + \langle z^*, y^* - z^* \rangle.$$

The left side simplifies:

$$\langle y, z^* - y^* \rangle - \langle z, z^* - y^* \rangle = \langle y - z, z^* - y^* \rangle.$$

The right side:

$$\langle y^*, z^* - y^* \rangle + \langle z^*, y^* - z^* \rangle = \langle y^* - z^*, z^* - y^* \rangle = -\|y^* - z^*\|^2.$$

□

Proof: Final estimate

Proof (continued).

We have established:

$$\langle y - z, z^* - y^* \rangle \leq -\|y^* - z^*\|^2.$$

Equivalently (multiplying by -1), $\langle y - z, y^* - z^* \rangle \geq \|y^* - z^*\|^2$.
By Cauchy-Schwarz:

$$\|y^* - z^*\|^2 \leq \langle y - z, y^* - z^* \rangle \leq \|y - z\| \cdot \|y^* - z^*\|.$$

Dividing by $\|y^* - z^*\|$ (if nonzero):

$$\|y^* - z^*\| \leq \|y - z\|.$$



Theorem: Convergence rate for Projected Gradient Descent

Theorem

Let f be convex and assume $\|g\| \leq G$ for all $g \in \partial f(x)$ and all $x \in C$. Assume C is closed, convex, and $\|x_0 - x^*\| \leq R$. Run projected subgradient descent:

$$x_{t+1} = P_C(x_t - \eta_t g_t), \qquad g_t \in \partial f(x_t).$$

With step sizes

$$\eta_t = \frac{R}{G\sqrt{T}} \quad \text{for } t = 1, \ldots, T,$$

we have

$$\min_{s \leq T} f(x_s) - f(x^*) \;\leq\; \frac{RG}{\sqrt{T}}.$$

Proof Strategy

We use a **distance-based** approach:

Step 1: Relate the distance of x_{t+1} from a minimizer x^* to that of $\tilde{x}_{t+1} = x_t - \eta_t g_t$ using the non-expansiveness of P_C.

Step 2: Use the subgradient inequality to estimate the alignment of the subgradient with $x_t - x^*$ by the function value difference.

Step 3: Telescope the resulting inequalities to get a bound on the minimum objective value achieved over the iterates.

Step 4: Plug in the given step size to establish the claimed $O(1/\sqrt{T})$ rate

Proof: Step 1 – Non-expansiveness of projection

Proof (Step 1).

Since P_C is the Euclidean projection onto the closed convex set C, it is non-expansive:

$$\|P_C(y) - P_C(z)\| \leq \|y - z\| \quad \text{for all } y, z \in \mathbb{R}^n.$$

Setting $\tilde{x}_{t+1} = x_t - \eta_t g_t$ and using $x_{t+1} = P_C(\tilde{x}_{t+1})$:

$$\begin{aligned} \|x_{t+1} - x^*\|^2 &= \|P_C(\tilde{x}_{t+1}) - P_C(x^*)\|^2 \leq \|\tilde{x}_{t+1} - x^*\|^2 \\ &= \|x_t - \eta_t g_t - x^*\|^2. \end{aligned}$$

Proof: Step 2 – Expanding the squared distance

Proof (Step 2).

Expand the right-hand side:

$$\|x_t - \eta_t g_t - x^*\|^2 = \|x_t - x^*\|^2 - 2\eta_t \langle g_t, x_t - x^* \rangle + \eta_t^2 \|g_t\|^2.$$

By the **subgradient inequality**, since $g_t \in \partial f(x_t)$:

$$f(x^*) \geq f(x_t) + \langle g_t, x^* - x_t \rangle.$$

Rearranging:

$$\langle g_t, x_t - x^* \rangle \geq f(x_t) - f(x^*).$$

Proof: Step 2 (continued) – Combining the bounds

Proof (Step 2, continued).

Substituting the subgradient bound and using $\|g_t\| \leq G$:

$$\|x_{t+1} - x^*\|^2 \leq \|x_t - x^*\|^2 - 2\eta_t(f(x_t) - f(x^*)) + \eta_t^2 G^2.$$

Rearranging to isolate the objective gap:

$$2\eta_t(f(x_t) - f(x^*)) \leq \|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2 + \eta_t^2 G^2.$$

Proof: Step 3 – Telescoping sum

Proof (Step 3).

Sum the inequality over $t = 1, \dots, T$:

$$2\sum_{t=1}^{T} \eta_t (f(x_t) - f(x^*)) \le \sum_{t=1}^{T} \left(\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2 \right) + G^2 \sum_{t=1}^{T} \eta_t^2.$$

The left sum **telescopes**:

$$\sum_{t=1}^{T} \left(\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2 \right) = \|x_1 - x^*\|^2 - \|x_{T+1} - x^*\|^2.$$

Since $\|x_{T+1} - x^*\|^2 \ge 0$ and $\|x_1 - x^*\| \le R$:

$$2\sum_{t=1}^{T} \eta_t (f(x_t) - f(x^*)) \le R^2 + G^2 \sum_{t=1}^{T} \eta_t^2.$$

Proof: Step 3 (continued) – Extracting the minimum

Proof (Step 3, continued).

Let $f_{\min} = \min_{t \le T} f(x_t)$. Then:

$$\sum_{t=1}^{T} \eta_t (f(x_t) - f(x^*)) \ge \sum_{t=1}^{T} \eta_t (f_{\min} - f(x^*)) = (f_{\min} - f(x^*)) \sum_{t=1}^{T} \eta_t.$$

Therefore:

$$2(f_{\min} - f(x^*)) \sum_{t=1}^{T} \eta_t \le R^2 + G^2 \sum_{t=1}^{T} \eta_t^2.$$

Dividing both sides by $2\sum_{t=1}^{T} \eta_t$:

$$f_{\min} - f(x^*) \le \frac{R^2}{2\sum_{t=1}^{T} \eta_t} + \frac{G^2 \sum_{t=1}^{T} \eta_t^2}{2\sum_{t=1}^{T} \eta_t}.$$

Proof: Step 4 – Plugging in the step size

Proof (Step 4).

With $\eta_t = \frac{R}{G\sqrt{T}}$ for all $t = 1, \ldots, T$:

$$\sum_{t=1}^{T} \eta_t = T \cdot \frac{R}{G\sqrt{T}} = \frac{R\sqrt{T}}{G},$$

$$\sum_{t=1}^{T} \eta_t^2 = T \cdot \frac{R^2}{G^2 T} = \frac{R^2}{G^2}.$$

Substituting into our bound:

$$f_{\min} - f(x^*) \leq \frac{R^2 + G^2 \cdot \frac{R^2}{G^2}}{2 \cdot \frac{R\sqrt{T}}{G}} = \frac{2R^2}{2R\sqrt{T}/G} = \frac{RG}{\sqrt{T}}.$$

□