Optimization and Online Learning

David Martínez-Rubio
Carlos III University of Madrid, Spain

August 8
Machine Learning Summer School 2025
Arequipa, Perú

I am looking for PhD students and Postdocs. Tell your friends and/or contact me.

Overview

I am looking for PhD students and Postdocs. Tell your friends and/or contact me.

Convexity

A set $\mathcal{X} \subseteq \mathbb{R}^d$ is convex if for any $x, y \in \mathcal{X}$ and $\lambda \in [0,1]$ then $\lambda x + (1-\lambda)y \in \mathcal{X}$.
For functions:,

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y) \text{ for } x, y \in \mathbb{R}^d, \ \lambda \in [0,1].$$

Subgradient: $g \in \partial f(x)$ satisfies $f(y) \geq f(x) + \langle g, y - x \rangle \ \forall y$.

Convex Functions

$f : \mathbb{R}^d \to \mathbb{R}$ is convex iff $\mathrm{epi}(f)$ is convex. It is *closed* (lsc) iff $\mathrm{epi}(f)$ is closed.

- Supremum of convex closed $\to$ convex closed.
- Sum of convex functions is convex.
- Differentiable f is convex $\iff f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle$.
- For twice-differentiable f, if $\nabla^2 f \succeq 0$ then f is convex.

Examples: $\|x\|$, $(x, y) \mapsto \|x - y\|$, $\|x\|^2$, $\exp(x)$, $-\log x$, $x \log x$, $x^\top A x$ for PSD A.

First-Order Optimality & Subgradients

- If f convex differentiable on convex $\mathcal{X}$ with a minimizer at x^*, then

$$x^* \in \arg\min_{x \in \mathcal{X}} f(x) \iff \langle \nabla f(x^*), x^* - x \rangle \leq 0 \quad \text{for all } x \in \mathcal{X}.$$

  Thus, for unconstrained problems: $\nabla f(x^*) = 0$.

- If subdifferentiable:

$$\langle g, x^* - x \rangle \leq 0 \quad \text{for all } x \in \mathcal{X} \text{ and all } g \in \partial f(x^*)$$

  and $0 \in \partial f(x^*)$, even with constraints.

Convexity in Machine Learning

Many problems:

- Linear regression $\min_x \{\|Ax - b\|_2^2\}$, compressed sensing $\min_x \{\|Ax - b\|_2^2 \mid \|x\|_1 \leq \rho\}$.
- Logistic Regression.
- SVMs.
- MLE for many problems.
- Convex Relaxations: $\min_X \|X\|_*$, subject to $X_{ij} = M_{ij}$, for $ij \in \text{SomeSet}$.
- Discrete Optimal Transport.
- Distributional Robust Optimization.
- ...

Convexity in Machine Learning

Many problems:

- Linear regression $\min_x \{\|Ax - b\|_2^2\}$, compressed sensing $\min_x \{\|Ax - b\|_2^2 \mid \|x\|_1 \leq \rho\}$.
- Logistic Regression.
- SVMs.
- MLE for many problems.
- Convex Relaxations: $\min_X \|X\|_*$, subject to $X_{ij} = M_{ij}$, for $ij \in \text{SomeSet}$.
- Discrete Optimal Transport.
- Distributional Robust Optimization.
- ...

Also useful for:

- Understanding convex problems is the first step to understand the non-convex world.
- Optimal algorithms for non-convex problems consist sometimes of the convex algorithms.
- Convex algorithms are helpful in practice for deep learning.

Other Regularity Properties

Lipschitz Continuity. $f : \mathcal{X} \to \mathbb{R}$ is G-Lipschitz w.r.t. $\|\cdot\|$ if

$$|f(x) - f(y)| \leq G\|x - y\|, \quad \forall x, y \in \mathcal{X}.$$

Implied for subdifferentiable f, if all subgradients g satisfy $\|g\|_* \leq G$.

Other Regularity Properties

Lipschitz Continuity. $f : \mathcal{X} \to \mathbb{R}$ is G-Lipschitz w.r.t. $\|\cdot\|$ if

$$|f(x) - f(y)| \;\le\; G\,\|x - y\|, \quad \forall x, y \in \mathcal{X}.$$

Implied for subdifferentiable f, if all subgradients g satisfy $\|g\|_* \le G$.
Smoothness. A differentiable function f is L-smooth w.r.t. $\|\cdot\|$ in $\mathcal{X}$ if

$$f(y) \;\le\; f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|y - x\|^2, \quad \forall x, y \in \mathcal{X}.$$

Implied by ∇f being L-Lipschitz: $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|$. Equivalent when f is convex.
It is implied by $v^T \nabla^2 f(z) v \le L\|v\|^2$ for all z. So for $\|\cdot\|_2$, it is implied if $\nabla^2 f(x) \preccurlyeq L$.

Other Regularity Properties

Lipschitz Continuity. $f : \mathcal{X} \to \mathbb{R}$ is G-Lipschitz w.r.t. $\|\cdot\|$ if

$$|f(x) - f(y)| \;\le\; G\,\|x - y\|, \quad \forall x, y \in \mathcal{X}.$$

Implied for subdifferentiable f, if all subgradients g satisfy $\|g\|_* \le G$.
Smoothness. A differentiable function f is L-smooth w.r.t. $\|\cdot\|$ in $\mathcal{X}$ if

$$f(y) \;\le\; f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|y - x\|^2, \quad \forall x, y \in \mathcal{X}.$$

Implied by ∇f being L-Lipschitz: $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|$. Equivalent when f is convex.
It is implied by $v^T \nabla^2 f(z) v \le L\|v\|^2$ for all z. So for $\|\cdot\|_2$, it is implied if $\nabla^2 f(x) \preccurlyeq L$.

Strong Convexity. f is μ-strongly convex w.r.t. $\|\cdot\|$ if

$$f(y) \;\ge\; f(x) + \langle \nabla f(x), y - x\rangle + \frac{\mu}{2}\|y - x\|^2, \quad \forall x, y \in \mathcal{X}.$$

Implies a unique minimizer and $\frac{\mu}{2}\|x - x^*\|^2 \le f(x) - f(x^*)$. Implied by $v^T \nabla^2 f(z) v \ge L\|v\|^2, \;\; \forall z$.

Towards Online Convex Optimization: Statistical Learning Theory

- Collect i.i.d. $Z_1, \dots, Z_n \sim \mathcal{P}$ in a set $\mathcal{Z}$.
- Choose action/prediction $a = a(Z_1, \dots, Z_n) \in \mathcal{A}$.
- Loss $\ell(a, Z) \geq 0$, define the *excess risk*

$$r_n = \mathbb{E}\left[\ell\big(a(Z_{1:n}), Z\big)\right] - \inf_{a \in \mathcal{A}} \mathbb{E}\left[\ell(a, Z)\right].$$

- Study upper/lower bounds on the risk r_n, and algorithms to attain them.

Examples

- Linear regression: $\ell((w, b), (x, y)) = (w^\top x + b - y)^2$ w/ random (x, y).
- Binary classification for data in $R^n \times \{-1, 1\}$. Zero-one loss, logistic, hinge...

Towards Online Convex Optimization

- Data may arrive in a stream.
- We may need to take actions sequentially.
- Measure cumulative excess risk:

$$\sum_{t=1}^{T} \mathbb{E}\left[\ell(a_t, Z)\right] - T \inf_{a \in \mathcal{A}} \mathbb{E}\left[\ell(a, Z)\right], \text{ where } a_t = a(Z_{1:t-1}).$$

Towards Online Convex Optimization

- Data may arrive in a stream.
- We may need to take actions sequentially.
- Measure cumulative excess risk:

$$\sum_{t=1}^{T} \mathbb{E}\left[\ell(a_t, Z)\right] - T \inf_{a \in \mathcal{A}} \mathbb{E}\left[\ell(a, Z)\right], \text{ where } a_t = a(Z_{1:t-1}).$$

- However, what if the distribution shifts with time? What if data is adversarial?
- We still can say obtain non-trivial bounds in that case!

Adversarial Online Learning Game: Regret minimization

1. Adversary picks loss $\ell_t : \mathcal{X} \to \mathbb{R}$
2. Learner picks $x_t \in \mathcal{X}$.
3. Learner suffers $\ell_t(x_t)$ and observes full, i.e. $\ell_t(\cdot)$, or partial feedback.

Adversarial Online Learning Game: Regret minimization

1. Adversary picks loss $\ell_t : \mathcal{X} \to \mathbb{R}$
2. Learner picks $x_t \in \mathcal{X}$.
3. Learner suffers $\ell_t(x_t)$ and observes full, i.e. $\ell_t(\cdot)$, or partial feedback.

Static Regret:

$$\text{Regret}_T(u) = \sum_{t=1}^{T} \ell_t(x_t) - \sum_{t=1}^{T} \ell_t(u), \qquad \text{Regret}_T \stackrel{\text{def}}{=} \sup_{u \in \mathcal{X}} \text{Regret}_T(u).$$

Generalizes statistical bound by choosing $\ell_t(x) = \mathbb{E}\left[\ell(x, Z)\right]$.

Adversarial Online Learning Game: Regret minimization

1. Adversary picks loss $\ell_t : \mathcal{X} \to \mathbb{R}$
2. Learner picks $x_t \in \mathcal{X}$.
3. Learner suffers $\ell_t(x_t)$ and observes full, i.e. $\ell_t(\cdot)$, or partial feedback.

Static Regret:

$$\text{Regret}_T(u) = \sum_{t=1}^{T} \ell_t(x_t) - \sum_{t=1}^{T} \ell_t(u), \qquad \text{Regret}_T \stackrel{\text{def}}{=} \sup_{u \in \mathcal{X}} \text{Regret}_T(u).$$

Generalizes statistical bound by choosing $\ell_t(x) = \mathbb{E}[\ell(x, Z)]$.

Dynamic Regret:

$$\text{Regret}_T(u_1, \ldots, u_T) = \sum_{t=1}^{T} \ell_t(x_t) - \sum_{t=1}^{T} \ell_t(u_t).$$

Typical bounds for dynamic regret depend on how much comparators move, like $\sum_{t=1}^{T-1} \|u_i - u_{i+1}\|$.

Peeking into the future

One-step-ahead regret. If you can see one step into the future, you may not regret anything.

Let $x_t^* = \arg\min_{x \in \mathcal{X}} \sum_{i=1}^{t} \ell_i(x)$, assuming it exists. Then by induction $\sum_{t=1}^{T} \ell_t(x_t^*) \leq \sum_{t=1}^{T} \ell_t(x_T^*)$.

Indeed, $T = 0$ is trivial, and if it holds for $T \geq 0$:

$$\sum_{t=1}^{T} \ell_t(x_t^*) + \ell_{T+1}(x_{T+1}^*) \overset{(IH)}{\leq} \sum_{t=1}^{T} \ell_t(x_T^*) + \ell_{T+1}(x_{T+1}^*) \leq \sum_{t=1}^{T} \ell_t(x_{T+1}^*) + \ell_{T+1}(x_{T+1}^*),$$

and so $\text{Regret}_T \leq 0$.

Example: Squared-Loss Prediction

Adversary plays

$$\ell_t(x) = (x - y_t)^2, \quad \text{for } y_t \in [0, 1].$$

Play

$$x_t^* = \frac{1}{t} \sum_{i=1}^{t-1} y_i.$$

Then

$$\text{Regret}_T \leq 2 \sum_{t=1}^{T} |x_t^* - x_{t-1}^*| \leq 2 \sum_{t=1}^{T} \frac{1}{t} = O(\log T).$$

- Sublinear regret on T, no stochastic assumption needed.
- But works well in the stochastic case too.

Betting on a coin

- Coin result can is $y_t \in \{-1, 1\}$. You play a percentage x_t of your wealth (with sign).
- **Win**: you get $2x_t$. **Lose**: you get nothing.
- Randomize: the adversary knows your distribution but not what you'll play.

Betting on a coin

- Coin result can is $y_t \in \{-1, 1\}$. You play a percentage x_t of your wealth (with sign).
- **Win**: you get $2x_t$. **Lose**: you get nothing.
- Randomize: the adversary knows your distribution but not what you'll play.
- In expectation, you win $y_t(x_t W_t)$, so your new expected wealth W_{t+1} is

$$W_{t+1} = W_t + y_t x_t W_t = W_t(1 + y_t x_t) = W_0 \prod_{i=1}^{t} (1 + y_i x_i).$$

To simplify, consider you bet on a result $[-1, 1]$. The above holds. Taking logs:

$$\log(W_{t+1}) = \log(W_0) + \sum_{i=1}^{t} \log(1 + y_i x_i).$$

Betting on a coin

- Coin result can is $y_t \in \{-1, 1\}$. You play a percentage x_t of your wealth (with sign).
- **Win**: you get $2x_t$. **Lose**: you get nothing.
- Randomize: the adversary knows your distribution but not what you'll play.
- In expectation, you win $y_t(x_t W_t)$, so your new expected wealth W_{t+1} is

$$W_{t+1} = W_t + y_t x_t W_t = W_t(1 + y_t x_t) = W_0 \prod_{i=1}^{t} (1 + y_i x_i).$$

To simplify, consider you bet on a result $[-1, 1]$. The above holds. Taking logs:

$$\log(W_{t+1}) = \log(W_0) + \sum_{i=1}^{t} \log(1 + y_i x_i).$$

- **Scenario**: adversary (or nature), played heads 85% of the time.
- You are as good $\{\text{heads}, \text{heads}, \dots\}$ up to the regret in log wealth.
- But no prior knowledge required.

The Experts Problem

- k experts, losses $g_t \in [-G, G]^k$.
- Actions in the simplex: $x_t \in \Delta^{k-1} = \{x \in \mathbb{R}^k_{\geq 0} \mid \sum_{i=1}^k x_i = 1\}$.
- Loss $\ell_t(x_t) = \langle g_t, x_t \rangle$.
- The regret is:

$$\text{Regret}_T = \sum_{t=1}^{T} \langle g_t, x_t \rangle - \min_{x \in \Delta^{k-1}} \sum_{t=1}^{T} \langle g_t, x \rangle = \sum_{t=1}^{T} \langle g_t, x_t \rangle - \min_{i \in [k]} \sum_{t=1}^{T} g_{t,i}.$$

The Experts Problem

- k experts, losses $g_t \in [-G, G]^k$.
- Actions in the simplex: $x_t \in \Delta^{k-1} = \{x \in \mathbb{R}^k_{\geq 0} \mid \sum_{i=1}^k x_i = 1\}$.
- Loss $\ell_t(x_t) = \langle g_t, x_t \rangle$.
- The regret is:

$$\text{Regret}_T = \sum_{t=1}^T \langle g_t, x_t \rangle - \min_{x \in \Delta^{k-1}} \sum_{t=1}^T \langle g_t, x \rangle = \sum_{t=1}^T \langle g_t, x_t \rangle - \min_{i \in [k]} \sum_{t=1}^T g_{t,i}.$$

- Many problems reduce to experts with linear losses. Examples:
  - **Boosting**. k points, weak-learner error $\leq \frac{1}{2} - \gamma$.
  - Each point is an expert, play distribution x_t.
  - Loss $\ell_t(x_t) = 1 - \text{err}(h_t, x_t) \geq \frac{1}{2} + \gamma$.
  - $\text{Regret}_T / T < \gamma \implies$ strong learner.
  - **Linear Feasibility**. m constraints $A_i x \geq b_i - \varepsilon$.
  - Experts on constraints, distribution $x_t \in \Delta^{m-1}$.
  - Oracle tests $\langle x_t^\top A, y \rangle \geq x_t^\top b$.
  - Loss $\langle Ay_t - b, x_t \rangle$.
  - $\text{Regret}_T / T \leq \varepsilon \implies \bar{y}$ is ε-feasible.

Online Learning in Optimization: Online-to-Batch Conversion

- We can optimize by running OL algorithms on linear losses given by subgradients:

$$f\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) - f(x^*) \le \frac{1}{T}\sum_{t=1}^{T}(f(x_t) - f(x^*)) \le \frac{1}{T}\sum_{t=1}^{T}\langle g_t, x_t - x^*\rangle = \frac{\text{Regret}_T(x^*)}{T} \le \frac{\text{Regret}_T}{T}.$$

Online Learning in Optimization: Online-to-Batch Conversion

- We can optimize by running OL algorithms on linear losses given by subgradients:

$$f\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) - f(x^*) \le \frac{1}{T}\sum_{t=1}^{T}(f(x_t) - f(x^*)) \le \frac{1}{T}\sum_{t=1}^{T}\langle g_t, x_t - x^*\rangle = \frac{\text{Regret}_T(x^*)}{T} \le \frac{\text{Regret}_T}{T}.$$

- Non-uniform average version. Denote $A_t \stackrel{\text{def}}{=} \sum_{i=1}^{t} a_i$, for $a_i > 0$. Loss $\ell_t(\cdot) \equiv \langle a_t g_t, \cdot\rangle$. Then

$$f\left(\frac{\sum_{t=1}^{T} a_t x_t}{A_T}\right) - f(x^*) \le \frac{\sum_{t=1}^{T} a_t(f(x_t) - f(x^*))}{A_T} \le \frac{\sum_{t=1}^{T}\langle a_t g_t, x_t - x^*\rangle}{A_T} \le \frac{\text{Regret}_T}{A_T}.$$

  Often, recent losses are given more weight. They are (sub)gradients at partially optimized points.

Online Learning in Optimization: Online-to-Batch Conversion

- We can optimize by running OL algorithms on linear losses given by subgradients:

$$f\left(\frac{1}{T}\sum_{t=1}^{T} x_t\right) - f(x^*) \le \frac{1}{T}\sum_{t=1}^{T}(f(x_t) - f(x^*)) \le \frac{1}{T}\sum_{t=1}^{T}\langle g_t, x_t - x^*\rangle = \frac{\text{Regret}_T(x^*)}{T} \le \frac{\text{Regret}_T}{T}.$$

- Non-uniform average version. Denote $A_t \stackrel{\text{def}}{=} \sum_{i=1}^{t} a_i$, for $a_i > 0$. Loss $\ell_t(\cdot) \equiv \langle a_t g_t, \cdot\rangle$. Then

$$f\left(\frac{\sum_{t=1}^{T} a_t x_t}{A_T}\right) - f(x^*) \le \frac{\sum_{t=1}^{T} a_t (f(x_t) - f(x^*))}{A_T} \le \frac{\sum_{t=1}^{T}\langle a_t g_t, x_t - x^*\rangle}{A_T} \le \frac{\text{Regret}_T}{A_T}.$$

  Often, recent losses are given more weight. They are (sub)gradients at partially optimized points.

- In expectation. Sampling one point $\hat{x}$ uniformly at random in $\{x_1, \dots, x_T\}$ (or w/ the average; or in last iterate if you apply the trick in (Nesterov and Shikhman, 2015; Cutkosky, 2019)):

$$\mathbb{E}\left[f(\hat{x}) - f(x^*)\right] \le \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T}\langle g_t, x_t - x^*\rangle\right] = \frac{\mathbb{E}\left[\text{Regret}_T(x^*)\right]}{T} \le \frac{\mathbb{E}\left[\text{Regret}_T\right]}{T}.$$

Anytime Online-to-Batch Conversion

Run an OL algorithm $\mathcal{A}$ on linearized losses, but play averaged points:

$$\bar{x}_t = \frac{1}{A_t} \sum_{i=1}^{t} a_i \, x_i, \quad A_t = \sum_{i=1}^{t} a_i.$$

Send to $\mathcal{A}$ the loss

$$\ell_t(x) = \big\langle a_t \, \nabla f(\bar{x}_t), \, x \big\rangle.$$

Algorithm $\mathcal{A}$ plays x_{t+1}.
Then one shows

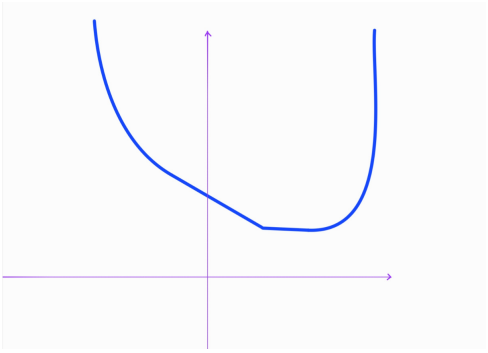
$$f(\bar{x}_T) - f(u) \;\leq\; \frac{R_T(u)}{A_T} - \sum_{t=1}^{T} \frac{A_{t-1}}{A_T} \, D_f(\bar{x}_{t-1}, \bar{x}_t).$$

Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.

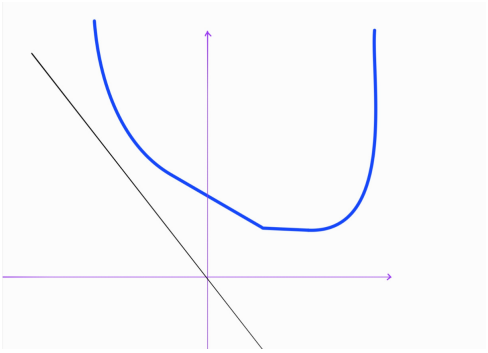


Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.

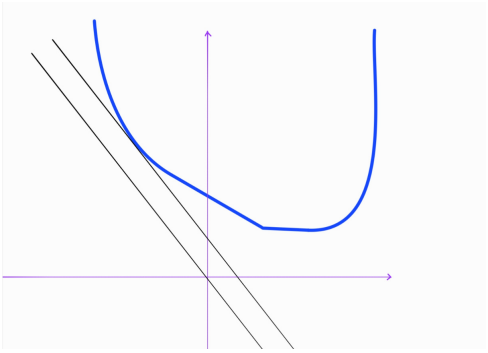


Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.

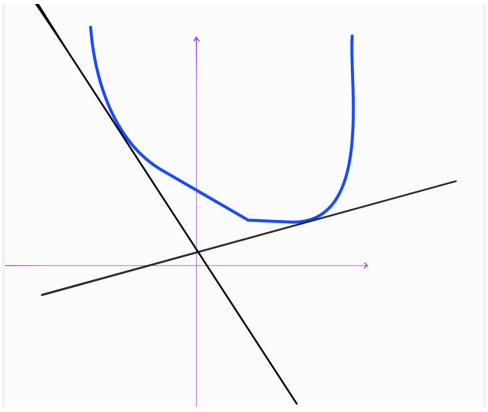


Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.

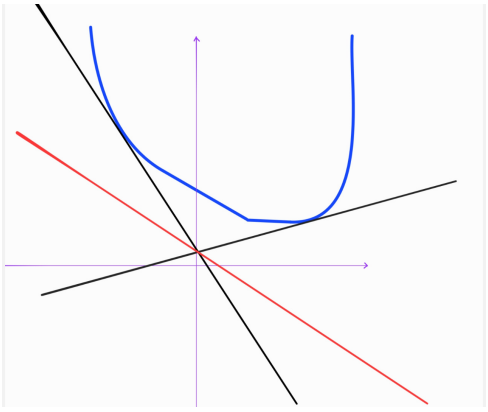


Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.

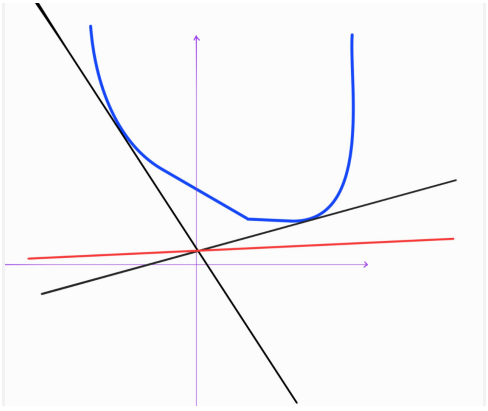


Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.

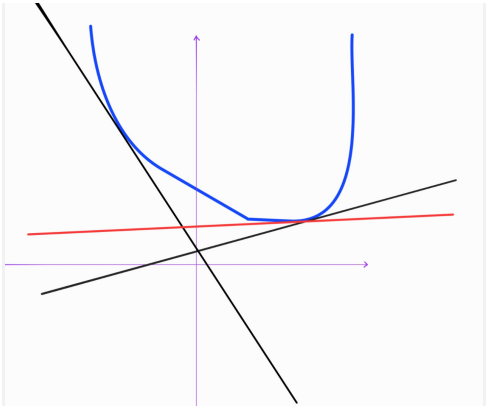


Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.



Fenchel Dual

$$f^*(u) \stackrel{\text{def}}{=} \sup_{x \in \mathbb{R}^d} \{\langle u, x \rangle - f(x)\}.$$

$f(x) + f^*(u) \geq \langle u, x \rangle$ with equality $\iff$ $u \in \partial f(x)$.

If f is convex, closed and proper then $f^{**} = f$.

Example: The Fenchel dual of $\frac{1}{2}\|x\|_p^2$ is $\frac{1}{2}\|x\|_{p^*}^2$, where $\frac{1}{p} + \frac{1}{p^*} = 1$. Including $p = 1$, $p^* = \infty$.

Legendre Functions

Definition

A closed convex proper function f is *Legendre* if

- strictly convex, differentiable on $\operatorname{int}(\operatorname{dom} f) \neq \emptyset$,
- $\|\nabla f(x)\| \to \infty$ as $x \to \partial \operatorname{dom} f$.

Example

- $f(x) = x \log x - x + \mathcal{I}_{[0,\infty)}(x)$,
- $f'(x) = \log(x)$ for $x \in (0, \infty)$, $\quad f''(x) = \frac{1}{x} > 0$,
- $(f^*)'(u) = \exp(u)$ for $u \in \mathbb{R}$, $\quad (f^*)''(u) = \exp(u) > 0$,
- $f^*(u) = \exp(u)$ for $u \in \mathbb{R}$.

Bregman Divergence

For differentiable convex $\psi : \mathcal{X} \to \mathbb{R}$,

$$D_\psi(x, y) = \psi(x) - \psi(y) - \langle \nabla\psi(y), x - y \rangle.$$

- **Non-negativity:** $D_\psi(x, y) \geq 0$, equals 0 iff $x = y$.
- **Three-point identity:** $D_\psi(x, y) = D_\psi(z, y) + D_\psi(x, z) + \langle \nabla\psi(z) - \nabla\psi(y),\, x - z \rangle$.
- **Combinations:** $D_{a\psi+b\phi}(x, y) = aD_\psi(x, y) + bD_\phi(x, y)$.
- The Bregman divergence of $D_\psi(\cdot, z)$, that is, $D_{D_\psi(\cdot,z)}(:, z)$ is itself. More generally: $D_{D_\psi(\cdot,y)}(x, z) = D_\psi(x, z)$.
- If ψ is σ-strongly convex, $D_\psi(x, y) \geq \frac{\sigma}{2}\|x - y\|^2$.

Online Learning algorithms: Follow-The-Leader (FTL)

$$x_t = \arg\min_{x \in \mathcal{X}} \sum_{i=1}^{t-1} \ell_i(x).$$

- Works when minimizers move slowly (e.g. squared loss).
- **Fails** badly for adversarial linear losses.

Online Learning algorithms: From FTL to FTRL

- Since FTL can be *unstable*...
- ...stabilize it by adding a *regularizer* ψ_t:

$$x_t \in \operatorname*{arg\,min}_{x \in \mathcal{X}} \Big\{ \sum_{i=1}^{t-1} \ell_i(x) + \psi_t(x) \Big\}.$$

- E.g., choose ψ_t so that $\sum_{i=1}^{t-1} \ell_i + \psi_t$ is *strongly convex*.

Online Learning algorithms: From FTL to FTRL

- Since FTL can be *unstable*...
- ...stabilize it by adding a *regularizer* ψ_t:

$$x_t \in \operatorname*{arg\,min}_{x \in \mathcal{X}} \Big\{ \sum_{i=1}^{t-1} \ell_i(x) + \psi_t(x) \Big\}.$$

- E.g., choose ψ_t so that $\sum_{i=1}^{t-1} \ell_i + \psi_t$ is *strongly convex*.

Theorem (FTRL Regret Guarantee)

Suppose each $\ell_t : \mathcal{X} \to \mathbb{R}$ is convex, subdifferentiable, and we use increasing regularizers $\psi_t \stackrel{\text{def}}{=} \frac{1}{\eta_t}\psi(x)$ for ψ being Legendre and 1-strongly convex in $\mathcal{X}$, and $\min_{x \in \mathcal{X}} \psi(x) = 0$. Then, for any $u \in \mathcal{X}$ and $g_t \in \partial \ell_t(x_t)$,

$$\text{Regret}_T(u) \leq \frac{\psi(u)}{\eta_{T+1}} + \sum_{t=1}^{T} \frac{\eta_t \|g_t\|_*^2}{2}.$$

FTRL for Lipschitz Losses

- Suppose ℓ_t is G-Lipschitz w.r.t. $\|\cdot\|$, and $\mathcal{X}$ is compact.
- Use $\psi_t(x) = \frac{1}{\eta_t} D_\psi(x, x_1)$ with ψ 1–strongly convex, $D^2 = \max_{u \in \mathcal{X}} D_\psi(u, x_1)$, and $\eta_t = \dfrac{D}{G}\sqrt{\dfrac{2}{t}}$.

Then, from the theorem, one obtains

$$\text{Regret}_T \leq 3G\sqrt{D^2 T}.$$

Hence $\text{Regret}_T / T = O(1/\sqrt{T})$.
Online-to-batch conversion, for convex Lipschitz optimization over $\mathcal{X}$, gives a point $\bar{x}$ such that

$$f(\bar{x}) - f(x^*) \leq \frac{R_T}{T} \leq \frac{3GD}{\sqrt{T}}.$$

The rate above can be shown to be worse-case optimal up to constant, as long as $T <$ dimension.

Example: Online Subgradient Descent

Take linearized losses $\mathcal{L}_t(x) = \ell_t(x_t) + \langle g_t, x - x_t \rangle$, $g_t \in \partial \ell_t(x_t)$, and

$$\psi_t(x) = \tfrac{\sigma}{2}\|x - x_1\|_2^2.$$

Looser losses lack losing less:

$$\mathrm{Regret}_T^{\ell} \overset{\text{def}}{=} \sum_{t=1}^{T} \ell_t(x_t) - \min_{u \in \mathcal{X}} \sum_{t=1}^{T} \ell_t(u) \le \sum_{t=1}^{T} \mathcal{L}_t(x_t) - \min_{u \in \mathcal{X}} \sum_{t=1}^{T} \mathcal{L}_t(u) \overset{\text{def}}{=} \mathrm{Regret}_T^{\mathcal{L}},$$

Then the FTRL update

$$x_t = \arg\min_x \left\{ \sum_{i=1}^{t-1} \langle g_i, x \rangle + \tfrac{\sigma}{2}\|x - x_1\|^2 \right\} = x_{t-1} - \tfrac{1}{\sigma}\, g_{t-1},$$

recovers subgradient descent.

E.g.: Neg. Entropy-regularized OMD for Experts aka Multiplicative Weights

- Action set $\mathcal{X} = \Delta^{d-1}$, loss $\ell_t(x) = \langle g_t, x \rangle$, $\|g_t\|_\infty \leq G$.
- Use $\psi(x) = \sum_i x_i \ln x_i$ on Δ^{d-1}.
- OMD update in closed form, known as Multiplicative Weights:

$$x_{t+1,i} = \frac{x_{t,i} \exp(-\eta_{t+1} g_{t,i})}{\sum_j x_{t,j} \exp(-\eta_{t+1} g_{t,j})}.$$

- Regret $R_T \leq 3G\sqrt{\ln(d)\, T}$, optimal when $T \to \infty$, up to constants.

Dual View of FTRL

Let $\Psi_t(x) = \psi_t(x) + \mathcal{I}_{\mathcal{X}}(x)$. Then one can show

$$z_1 = 0, \quad z_t = z_{t-1} - g_{t-1}, \quad x_t = \nabla \Psi_t^*(z_t).$$

Equivalently:

$$\hat{x}_t = \nabla \psi_t^*\Big(-\sum_{i=1}^{t-1} g_i\Big), \quad x_t = \operatorname*{arg\,min}_{x \in \mathcal{X}} D_{\psi_t}(x, \hat{x}_t).$$

This is a *lazy* update. The greedy one, aka Online Mirror Descent, often comes w/ similar guarantees:

$$z_1 = 0, \quad z_t = \nabla \psi_t(x_{t-1}) - g_{t-1}, \quad x_t = \nabla \Psi_t^*(z_t).$$

Common regularizers

	Squared ℓ_2	$p \in (0,1)$	$p \in (1,\infty)$	Gen. neg. entropy	$p \in (1,2)$
$\psi(x)$	$\frac{1}{2}\|x\|_2^2$	$-\frac{1}{p}\sum_{i=1}^d x_i^p$	$\frac{1}{p}\|x\|_p^p$	$\sum_{i=1}^d x_i \log x_i - x_i$	$\frac{1}{2}\|x\|_p^2$
$\operatorname{int}(\operatorname{dom}\psi)$	$\mathbb{R}^d$	$\mathbb{R}^d_{>0}$	$\mathbb{R}^d$	$\mathbb{R}^d_{>0}$	$\mathbb{R}^d$
$\nabla_i \psi(x)$	x_i	$-x_i^{p-1}$	$\operatorname{sign}(x_i)\,\lvert x_i\rvert^{p-1}$	$\log x_i$	$\operatorname{sign}(x_i)\,\|x\|_p^{2-p}\lvert x_i\rvert^{p-1}$
$\nabla^2 \psi(x)$	I_d	$(1-p)\operatorname{diag}(1/x_i^{2-p})$	$(p-1)\operatorname{diag}(x_i^{p-2})$	$\operatorname{diag}(1/x_i)$	*(omitted)*
$\psi^*(u)$	$\frac{1}{2}\|u\|_2^2$	$-\frac{1}{p^*}\sum_i(-u_i)^{p^*} + 1_{\mathbb{R}^d_{\le 0}}(u)$	$\frac{1}{p^*}\|u\|_{p^*}^{p^*}$	$\sum_{i=1}^d e^{u_i}$	$\frac{1}{2}\|u\|_{p^*}^2$
$\operatorname{int}(\operatorname{dom}\psi^*)$	$\mathbb{R}^d$	$\mathbb{R}^d_{<0}$	$\mathbb{R}^d$	$\mathbb{R}^d$	$\mathbb{R}^d$
$\nabla_i \psi^*(u)$	u_i	$\lvert u_i\rvert^{p^*-1}$ (p^* s.t. $\frac{1}{p}+\frac{1}{p^*}=1$)	$\operatorname{sign}(u_i)\,\lvert u_i\rvert^{p^*-1}$	e^{u_i}	$\operatorname{sign}(u_i)\,\|u\|_{p^*}^{2-p^*}\lvert u_i\rvert^{p^*-1}$
$D_\psi(x,y)$	$\frac{1}{2}\|x-y\|_2^2$	$\sum_i\left(-\frac{1}{p}x_i^p + \frac{1-p}{p}y_i^p + y_i^{p-1}x_i\right)$	$\leftarrow$similar(omitted)	$\sum_i\left(x_i\log\frac{x_i}{y_i} - x_i + y_i\right)$	*(omitted)*
str. cvx.?	1-wrt $\|\cdot\|_2$	–	$(p-1)$-wrt $\|\cdot\|_p$ on $\|x\|_p \le 1$, $p\in(1,2]$	1-wrt $\|\cdot\|_1$ on $\|x\|_1 \le 1$, $x>0$	$(p-1)$-wrt $\|\cdot\|_p$

Implicit Bias

Theorem (Implicit bias of OMD)

Suppose we have a generalized linear model of the form: $\min_x \mathcal{L}(x) := \sum_{n=1}^N \ell(\langle x, z_n \rangle, y_n)$. and the data is realizable, that is, the minimum loss is 0. Under mild assumptions on the loss (unique finite root property), the iterates x_t of OMD with step sizes so that $x_\infty \overset{\text{def}}{=} \lim_{t\to\infty} x_t$ is a minimizer $\mathcal{L}(x_\infty) = 0$, yield a minimizer satisfying

$$x_\infty = \operatorname*{arg\,min}_{x:\forall n, \langle x, z_n \rangle = y_n} D_\psi(x, x_0).$$

See (Gunasekar et al., 2018).

- If $x_0 = \arg\min_x \psi(x)$, then x_∞ is the solution $\min_{x\in\text{sol}} \psi(x)$ (maximum entropy, minimum Euclidean norm, etc). This implicit bias can benefit generalization.
- Holds because OMD is GD in the dual space & the gradients are in $\text{span}\{z_1, \ldots, z_n\}$.
- Holds even if for stochastic methods using gradients of minibatches.

Problem Adaptivity

$$\text{Regret}_T(u) \leq \frac{\psi(u)}{\eta_{T+1}} + \sum_{t=1}^{T} \frac{\eta_t \|g_t\|_*^2}{2}.$$

Problem Adaptivity

$$\text{Regret}_T(u) \leq \frac{\psi(u)}{\eta} + \sum_{t=1}^{T} \frac{\eta G^2}{2}.$$

Problem Adaptivity

- A small variant with $\eta_{t+1} = \sqrt{\frac{2D^2}{\sum_{i=1}^t \|g_i\|_*^2}}$, where recall $D^2 \stackrel{\text{def}}{=} \max_{u \in \mathcal{X}} D_\psi(u, x_1)$, allows to show

$$R_T(u) \;\leq\; 3\sqrt{D^2 \sum_{t=1}^T \|g_t\|_*^2}\,.$$

- No need to know the future or a uniform bound G.
- Immediately yields $O(1/\sqrt{T})$ optimization rates in G-Lipschitz convex optimization via online-to-batch.
- Other techniques (comparator-adaptive, a.k.a. parameter-free) yield:

$$\text{Regret}_T(u) = \widetilde{O}\left(\sqrt{D_\psi(u, x_1) \sum_{i=1}^t \|g_i\|_*^2}\right),$$

making use of the value of G.

Key Takeaways

- FTRL or Online Mirror Descent generalize subgradient descent to arbitrary geometries via a strongly-convex mirror map ψ.
- Achieves $O(\sqrt{T})$ regret, which leads to $O(1/\sqrt{T})$ Lipschitz convex optimization convergence and adapts easily to $\sum \|g_t\|_*^2$.
- Tight for many problems (e.g. experts, convex Lipschitz minimization,...).
- Forms the basis for accelerated and stochastic algorithmic variants.

Lower Bound for Online Learning: Lipschitz Linear Losses in Compact Sets

Setting. Feasible set $\mathcal{X} \subseteq \mathbb{R}^d$, of diameter D. Losses $\ell_t(x) = \langle g_t, x \rangle$, $\|g_t\| \le G$.

Theorem (OL Lower Bound)

For any algorithm and any $T \ge 1$, there exists a sequence of G-Lipschitz linear losses with

$$\text{Regret}_T \ge \frac{GD}{2\sqrt{2}}\sqrt{T}.$$

Matches the $O(GD\sqrt{T})$ upper bound of FTRL/OMD up to constants.

Proof Sketch: Rademacher Adversary

1. Pick two points $v, w \in \mathcal{X}$ with $\|v - w\| = D$ and direction $z = (v - w)/D$.
2. Let $\varepsilon_t \in \{\pm 1\}$ i.i.d. Rademacher, set $g_t = G\,\varepsilon_t\,z$.
3. Then

$$\sup_{g_1,\dots,g_T} \mathrm{Regret}_T \geq \mathbb{E}\left[\sum_t \langle g_t, x_t\rangle - \min_u \sum_t \langle g_t, u\rangle\right] \;\geq\; \frac{GD}{2}\,\mathbb{E}\left[\left|\sum_t \varepsilon_t\right|\right] \;\geq\; \frac{GD}{2\sqrt{2}}\sqrt{T}.$$

Linearity and Khintchine's inequality give the $\sqrt{T}$ lower bound.

Overview

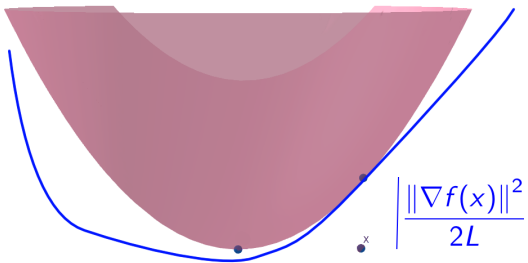
I am looking for PhD students and Postdocs. Tell your friends and/or contact me: David Martínez-Rubio (Madrid, Spain)

Steepest Descent

Smoothness inequality: $f(y) - f(x_t) - \langle \nabla f(x_t), y - x_t \rangle = (y - x_t)^T \nabla^2 f(z)(y - x_t) \leq \frac{L}{2} \|y - x_t\|^2$.
Minimize the upper bound:

$$f(x_{t+1}) - f(x_t) \leq \min_{x \in \mathbb{R}^d} \left\{ \langle \nabla f(x_t), x - x_t \rangle + \frac{L}{2} \|x - x_t\|^2 \right\} = -\frac{1}{2L} \|\nabla f(x_t)\|_*^2.$$

In the Euclidean norm case: $x_{t+1} \leftarrow x_t - \frac{1}{L} \nabla f(x_t)$.



Estimate sequences / Approximate Duality Gap Technique

General strategy to show convergence in convex optimization:

- Define a lower bound on a minimizer: $f(x^*) \geq L_t$. In fact, we could compare to an arbitrary point.
- Upper bound $f(y)$ for y a point you compute: $f(y) \leq U_t$.
- The primal-dual gap is $G_t = U_t - L_t$.
- For increasing weights $A_t > 0$, $A_0 = 0$, bound $A_t G_t - A_{t-1} G_{t-1} \leq E_t$.
- We get

$$f(y) - f(x^*) \leq \frac{A_t G_t}{A_t} \leq \frac{A_0 G_0 + \sum_{i=0}^{t} E_t}{A_t} = \frac{\sum_{i=0}^{t} E_t}{A_t}.$$

- If $E_t \leq 0$, we say that $A_t G_t$ is Lyapunov.

Steepest Descent: Convergence Rate

Theorem (Steepest Descent Convergence)

Let f be convex and L-smooth w.r.t. $\|\cdot\|$ with a minimizer x^*. Steepest Descent satisfies

$$f(x_t) - f(x^*) \leq \frac{2LD_t^2}{t+1},$$

where D_t is the maximum distance from iterates to x^*. They are in the sublevel set.

For the Euclidean norm (Gradient Descent), updates are non-expansive.
For $a_t > 0$ to be chosen later and for $A_t = \sum_{i=1}^{t} a_i$:

$$A_t f(x^*) \geq \sum_{i=0}^{t} a_i f(x_i) + \sum_{i=0}^{t} a_i \langle \nabla f(x_i), x^* - x_i \rangle =: A_t L_t \qquad A_t f(x_{t+1}) =: A_t U_t$$

Steepest Descent: Convergence Rate

$$
\begin{aligned}
A_t G_t - A_{t-1} G_{t-1} &= A_t (f(x_{t+1}) - f(x_t)) + \cancel{a_t f(x_t)} \\
&\quad - \left(\cancel{a_t f(x_t)} + \sum_{i=0}^{t-1} a_i f(x_i) + a_t \langle \nabla f(x_t), x^* - x_t \rangle + \cancel{\sum_{i=0}^{t-1} a_i (\langle \nabla f(x_i), x^* - x_i \rangle)} \right) \\
&\quad + \left(\cancel{\sum_{i=0}^{t-1} a_i f(x_i)} + \cancel{\sum_{i=0}^{t-1} a_i (\langle \nabla f(x_i), x^* - x_i \rangle)} \right) \\
&\overset{①}{\leq} -\frac{A_t}{2L} \|\nabla f(x_t)\|_*^2 + a_t \|\nabla f(x_t)\|_* D_t \overset{②}{\leq} \cancel{-\frac{A_t}{2L} \|\nabla f(x_t)\|_*^2} + \cancel{\frac{A_t}{2L} \|\nabla f(x_t)\|_*^2} + \frac{L a_t^2}{2 A_t} D_t^2 \overset{\text{def}}{=} E_t,
\end{aligned}
$$

Steepest Descent: Convergence Rate

$$
\begin{aligned}
A_t G_t - A_{t-1} G_{t-1} &= A_t (f(x_{t+1}) - f(x_t)) + \cancel{a_t f(x_t)} \\
&\quad - \left(\cancel{a_t f(x_t)} + \sum_{i=0}^{t-1} a_i f(x_i) + a_t \langle \nabla f(x_t), x^* - x_t \rangle + \cancel{\sum_{i=0}^{t-1} a_i (\langle \nabla f(x_i), x^* - x_i \rangle)} \right) \\
&\quad + \left(\cancel{\sum_{i=0}^{t-1} a_i f(x_i)} + \cancel{\sum_{i=0}^{t-1} a_i (\langle \nabla f(x_i), x^* - x_i \rangle)} \right) \\
&\overset{①}{\leq} -\frac{A_t}{2L} \|\nabla f(x_t)\|_*^2 + a_t \|\nabla f(x_t)\|_* D_t \overset{②}{\leq} \cancel{-\frac{A_t}{2L} \|\nabla f(x_t)\|_*^2} + \cancel{\frac{A_t}{2L} \|\nabla f(x_t)\|_*^2} + \frac{L a_t^2}{2A_t} D_t^2 \overset{\text{def}}{=} E_t,
\end{aligned}
$$

So

$$
f(x_{t+1}) - f(x^*) \leq \frac{\sum_{i=0}^t E_i}{A_t} = \frac{L D_t^2}{2A_t} \sum_{i=0}^t \frac{a_i^2}{A_i} = O\left(\frac{L D_t^2}{t} \right).
$$

Composite Steepest Descent

Goal:

$$\min_{x \in \mathbb{R}^d} f(x) + \psi(x).$$

Algorithm:

$$x_{t+1} \in \arg\min_{x \in \mathbb{R}^d} \left\{ f(x_t) + \langle \nabla f(x_t), x - x_t \rangle + \tfrac{L}{2} \|x - x_t\|^2 + \psi(x) \right\}.$$

Encode constraints or regularization with ψ. E.g.: $\mathsf{Ind}_{\mathcal{X}}(x)$, $\|x\|_2^2$, $\|x\|_1$, $\mathsf{KL}(x, x_0)$, etc.

Composite Steepest Descent

Goal:

$$\min_{x \in \mathbb{R}^d} f(x) + \psi(x).$$

Algorithm:

$$x_{t+1} \in \arg\min_{x \in \mathbb{R}^d} \left\{ f(x_t) + \langle \nabla f(x_t), x - x_t \rangle + \tfrac{L}{2} \|x - x_t\|^2 + \psi(x) \right\}.$$

Encode constraints or regularization with ψ. E.g.: $\mathrm{Ind}_{\mathcal{X}}(x)$, $\|x\|_2^2$, $\|x\|_1$, $\mathrm{KL}(x, x_0)$, etc.

Theorem (Composite Steepest Descent Convergence)

Let f be convex and L-smooth w.r.t. $\|\cdot\|$ such that $f + \psi$ has a minimizer at x^*. Composite Steepest Descent satisfies

$$(f + \psi)(x_t) - (f + \psi)(x^*) \leq \frac{2LD_t^2}{t+1},$$

where D_t is the maximum distance from iterates to x^*. They are in the sublevel set of $f + \psi$.

Composite Steepest Descent

Goal:

$$\min_{x \in \mathbb{R}^d} f(x) + \psi(x).$$

Algorithm:

$$x_{t+1} \in \arg\min_{x \in \mathbb{R}^d} \left\{ f(x_t) + \langle \nabla f(x_t), x - x_t \rangle + \tfrac{L}{2}\|x - x_t\|^2 + \psi(x) \right\}.$$

Encode constraints or regularization with ψ. E.g.: $\mathrm{Ind}_{\mathcal{X}}(x)$, $\|x\|_2^2$, $\|x\|_1$, $\mathrm{KL}(x, x_0)$, etc.

Theorem (Composite Steepest Descent Convergence)

Let f be convex and L-smooth w.r.t. $\|\cdot\|$ such that $f + \psi$ has a minimizer at x^*. Composite Steepest Descent satisfies

$$(f + \psi)(x_t) - (f + \psi)(x^*) \leq \frac{2LD_t^2}{t+1},$$

where D_t is the maximum distance from iterates to x^*. They are in the sublevel set of $f + \psi$.

Message: If you can optimize a model of f plus ψ exactly, then ψ generally does not affect convergence.

Composite Steepest Descent

Let $\hat{x}_{t+1} \stackrel{\text{def}}{=} \frac{a_t}{A_t} u + \frac{A_{t-1}}{A_t} x_t$ so that $a_t(u - \hat{x}_{t+1}) = A_{t-1}(\hat{x}_{t+1} - x_t)$. Then

$$A_t G_t - A_{t-1} G_{t-1} = A_t(f(x_{t+1}) - f(x_t)) + a_t f(x_t) + A_t \psi(x_{t+1}) - A_{t-1}\psi(x_t)$$

$$- \left(a_t f(x_t) + \sum_{i=0}^{t-1} a_i f(x_i) + a_t \langle \nabla f(x_t), u - x_t \rangle + \sum_{i=0}^{t-1} a_i (\langle \nabla f(x_i), u - x_i \rangle) + A_{t-1}\psi(u) + a_t \psi(u) \right)$$

$$+ \left(\sum_{i=0}^{t-1} a_i f(x_i) + \sum_{i=0}^{t-1} a_i (\langle \nabla f(x_i), u - x_i \rangle) + A_{t-1}\psi(u) \right)$$

$$\stackrel{①}{\leq} \left(A_t \langle \nabla f(x_t), \hat{x}_{t+1} - x_t \rangle + \frac{L a_t^2}{2A_{t-1}} \|u - x_t\|^2 \right) + a_t \langle \nabla f(x_t), u - x_t \rangle + A_t \psi(\hat{x}_{t+1}) - A_{t-1}\psi(x_t) - a_t \psi(u)$$

$$\stackrel{②}{\leq} \frac{L a_t^2}{2A_{t-1}} D_t^2 \stackrel{\text{def}}{=} E_t.$$

Weight Decay as Regularized Optimization

If we have a function f and we want to solve the regularized optimization problem

$$\min_x \left\{ f(x) + \frac{\lambda}{2}\|x\|_2^2 \right\} \quad \text{for some } \lambda > 0,$$

and we sequentially optimize some linear (or quadratic) model of f, like Composite Steepest Descent does, we get

$$x_{t+1} \in \arg\min_{x \in \mathbb{R}^d} \left\{ f(x_t) + \langle \nabla f(x_t), x - x_t \rangle + \frac{L}{2}\|x - x_t\|^2 + \frac{\lambda}{2}\psi(x) \right\} = x_t - \frac{1}{L}\nabla f(x_t) - \lambda x_t.$$

This is weight decay!

Weight Decay as Regularized Optimization

If we have a function f and we want to solve the regularized optimization problem

$$\min_x \left\{ f(x) + \frac{\lambda}{2}\|x\|_2^2 \right\} \quad \text{for some } \lambda > 0,$$

and we sequentially optimize some linear (or quadratic) model of f, like Composite Steepest Descent does, we get

$$x_{t+1} \in \arg\min_{x \in \mathbb{R}^d} \left\{ f(x_t) + \langle \nabla f(x_t), x - x_t \rangle + \frac{L}{2}\|x - x_t\|^2 + \frac{\lambda}{2}\psi(x) \right\} = x_t - \frac{1}{L}\nabla f(x_t) - \lambda x_t.$$

This is weight decay!

However, if the model of f is more complex, Composite Optimization $\neq$ adding the weight decay vector to the gradient.

Adam vs AdamW

Algorithm 2 Adam with L_2 regularization and Adam with decoupled weight decay (AdamW)

given $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}, \lambda \in \mathbb{R}$
initialize time step $t \leftarrow 0$, parameter vector $\boldsymbol{\theta}_{t=0} \in \mathbb{R}^n$, first moment vector $\boldsymbol{m}_{t=0} \leftarrow \boldsymbol{0}$, second moment vector $\boldsymbol{v}_{t=0} \leftarrow \boldsymbol{0}$, schedule multiplier $\eta_{t=0} \in \mathbb{R}$
repeat
  $t \leftarrow t + 1$
  $\nabla f_t(\boldsymbol{\theta}_{t-1}) \leftarrow \text{SelectBatch}(\boldsymbol{\theta}_{t-1})$ ▷ select batch and return the corresponding gradient
  $\boldsymbol{g}_t \leftarrow \nabla f_t(\boldsymbol{\theta}_{t-1}) + \lambda \boldsymbol{\theta}_{t-1}$
  $\boldsymbol{m}_t \leftarrow \beta_1 \boldsymbol{m}_{t-1} + (1 - \beta_1)\boldsymbol{g}_t$ ▷ here and below all operations are element-wise
  $\boldsymbol{v}_t \leftarrow \beta_2 \boldsymbol{v}_{t-1} + (1 - \beta_2)\boldsymbol{g}_t^2$
  $\hat{\boldsymbol{m}}_t \leftarrow \boldsymbol{m}_t / (1 - \beta_1^t)$ ▷ β_1 is taken to the power of t
  $\hat{\boldsymbol{v}}_t \leftarrow \boldsymbol{v}_t / (1 - \beta_2^t)$ ▷ β_2 is taken to the power of t
  $\eta_t \leftarrow \text{SetScheduleMultiplier}(t)$ ▷ can be fixed, decay, or also be used for warm restarts
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \left(\alpha \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon) + \lambda \boldsymbol{\theta}_{t-1} \right)$
until *stopping criterion is met*
return optimized parameters $\boldsymbol{\theta}_t$

Figure: Treating the regularized function as a composite objective (Loshchilov and Hutter, 2017)

Overview

I am looking for PhD students and Postdocs. Tell your friends and/or contact me: David Martínez-Rubio (Madrid, Spain)

Convex Theory for the Non-Convex World I: Convex Until Proven Guilty

Let f, possibly non-convex, satisfy gradient and Hessian Lipschitzness, that is:

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \text{and} \quad \|\nabla^2 f(x) - \nabla^2 f(y)\| \leq M\|x - y\| \quad \text{for all } x, y.$$

- The fastest provable algorithm known to date for $\|\nabla f(x_T)\| \leq \varepsilon$ runs in $T = O(\varepsilon^{-7/4})$.
- It is a $\varepsilon^{-1/28}$ away from the lower bound (Carmon et al., 2020).
- The algorithms is Nesterov's accelerated gradient descent, with its momentum restarted after some "**guilty of non-convex**" criteria are satisfied.

Convex Theory for the Non-Convex World I: Convex Until Proven Guilty

Let f, possibly non-convex, satisfy gradient and Hessian Lipschitzness, that is:

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|, \quad \text{and} \quad \|\nabla^2 f(x) - \nabla^2 f(y)\| \le M\|x - y\| \quad \text{for all } x, y.$$

- The fastest provable algorithm known to date for $\|\nabla f(x_T)\| \le \varepsilon$ runs in $T = O(\varepsilon^{-7/4})$.
- It is a $\varepsilon^{-1/28}$ away from the lower bound (Carmon et al., 2020).
- The algorithms is Nesterov's accelerated gradient descent, with its momentum restarted after some "**guilty of non-convex**" criteria are satisfied.
- **Convex algorithm design becomes useful for provably-fast non-convex algorithms!**
- (Carmon et al., 2017, Convex until proven guilty), (Li and Lin, 2023, Removing polylog factors), (Marumo and Takeda, 2024, Parameter-free version).

Convex Theory for the Non-Convex World II: Weakening convexity

Let x^* be a minimizer of f.

- Polyak Lojasiewicz (PL): $(f(x) - f(x^*)) \geq \mu \|\nabla f(x)\|$.
  - If NNs are very wide, they $\approx$ Neural Tangent Kernel, and are PL-like. (Du et al., 2019; Allen-Zhu et al., 2019).
- For $\gamma \in (0, 1]$, γ-quasar strongly convex: $f(x^*) \geq f(x) + \frac{1}{\gamma}\langle \nabla f(x), x^* - x \rangle + \frac{\mu}{2}\|x - x^*\|^2$.
  - DNNs landscapes along wrt its trajectory (Tran et al., 2024).
- Weak-convexity $\nabla^2 f \succeq -\sigma I$.
- Many more: Quadratic growth, restricted secant condition, one-point strong convexity, variational coherence, quasi-convexity, pseudoconvexity, invexity... Cf. (Hinder et al., 2020).

- Metric of optimality?
- Convex minimization: $f(x_t) - f(x^*)$ or $\|\nabla f(x_t)\|$.
- Non convex smooth optimization: $\mathbb{E}\left[\nabla f(x_i)_2^2\right]$.
- Non-convex non-smooth optimization? I.e., Deep NNs with ReLUs.

Definition. Point x is a (δ, ε)-stationary point of a.e. differentiable f if $\exists y_1, \ldots, y_n \in B(x, \delta)$ s.t.:

$$\frac{1}{n}\sum_{i=1}^{n} y_i = x, \quad \text{and} \quad \left\|\frac{1}{n}\sum_{i=1}^{n} \nabla F(y_i)\right\| \leq \varepsilon,$$

cf. (Zhang, Lin, et al., 2020).

Under G-Lipschitzness and mild function regularity:

$$f(y) - f(x) = \int_0^1 \langle \nabla f(x + t(y - x)), y - x \rangle \mathrm{d}t.$$

- (Cutkosky et al., 2023; Zhang and Cutkosky, 2024) designed reductions where an OCO or a restarted OCO is fed with linear losses $\ell_t(x) = \langle \nabla f(x_t), x \rangle$ and decides the updates of the algorithm.
- Step sizes that decide the next point are random variables.
- Achieves worse-case optimality $O(\delta/\varepsilon^3)$ when $\varepsilon = O(\delta)$, with a very general reduction.
- Promising direction for understanding current methods.

Convex Theory for the Non-Convex World IV: Empirical similar behaviors

The Surprising Agreement between Convex Optimization Theory and Learning-Rate Scheduling (Schaipp et al., 2025).

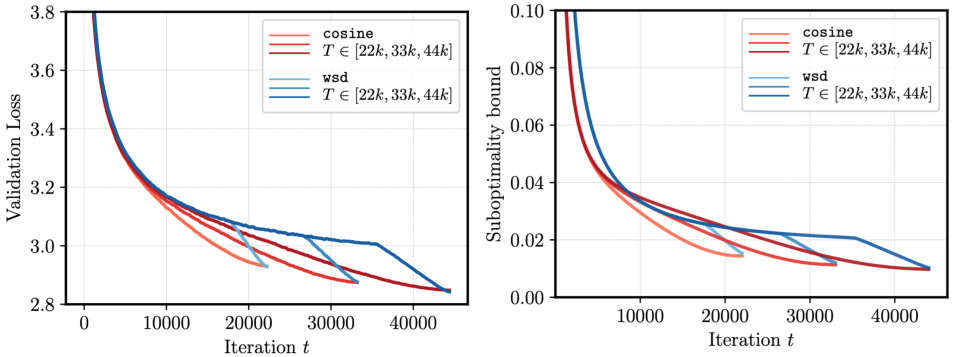


Figure 1. Real loss curves **(left)** and theoretical bound **(right)**

Convex Theory for the Non-Convex World IV: Empirical similar behaviors

Prediction of loss decay by (Defazio et al., 2023), depending on learning rates. One of several works in this direction.

$$\mathbb{E}[f(x_T) - f(x_\star)] \leq \frac{1}{2\gamma \sum_{t=1}^{T} \eta_t} \left[D^2 + \gamma^2 G^2 \sum_{t=1}^{T} \eta_t^2 \right] + \frac{\gamma G^2}{2} \sum_{k=1}^{T-1} \left(\frac{\eta_k}{\left(\sum_{t=k+1}^{T} \eta_t \right) \left(\sum_{t=k}^{T} \eta_t \right)} \sum_{t=k}^{T} \eta_t^2 \right).$$

Convex Theory for the Non-Convex World IV: Empirical similar behaviors

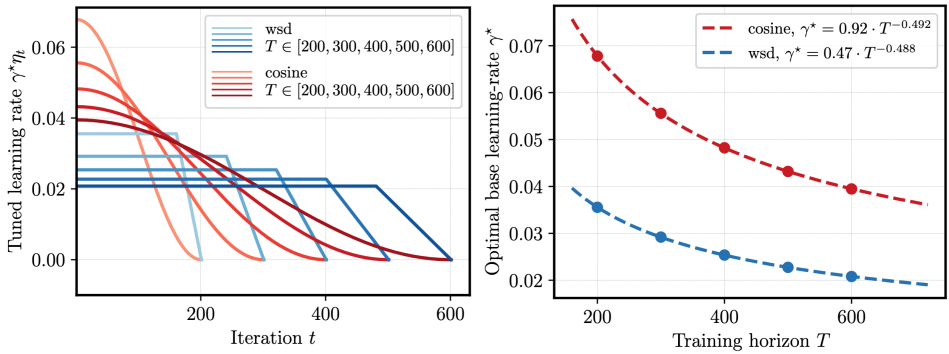


Figure: Left: Cosine & ctant + cooldown schedules. Right: Optimal initial learning rate. Theory predicts one should be $\approx$ double than the other one.

Overview

I am looking for PhD students and Postdocs. Tell your friends and/or contact me: David Martínez-Rubio (Madrid, Spain)

Frank–Wolfe (a.k.a. Conditional Gradients)

- **Goal:**

$$\min_{x \in \mathcal{X}} f(x),$$

where f is L-smooth, $\mathcal{X}$ is compact and convex. (We can also phrase the composite problem).

Frank–Wolfe (a.k.a. Conditional Gradients)

- **Goal:**

$$\min_{x \in \mathcal{X}} f(x),$$

where f is L-smooth, $\mathcal{X}$ is compact and convex. (We can also phrase the composite problem).

- Projection $\arg\min_{x \in \mathcal{X}} \|x - y\|^2$ may be expensive; linear minimization $\arg\min\{\langle \nabla f(x), v \rangle + \psi(v)\}$ can be much cheaper.

Set	**Linear minimization**	**Projection**
ℓ_p-ball, $p \notin \{1, 2, \infty\}$	$O(d)$	$\widetilde{O}(d\|y - x^*\|^2/\hat{\varepsilon}^2)$
Nuclear norm ball aka Spectrahedron	$O(\mathrm{nnz}(Y)\log(m+n)\sqrt{\sigma_1/\hat{\varepsilon}})$	$O(mn\min\{m,n\})$
Flow polytope	$O(m+n)$	$\widetilde{O}(m^3 n + n^2)$
Birkhoff polytope	$O(d^3)$	$O(d^2 d_z/\hat{\varepsilon}^2)$

Frank–Wolfe (a.k.a. Conditional Gradients)

$$
\begin{aligned}
v_t &\in \arg\min_v \{\langle \nabla f(x_t), v\rangle + \psi(v)\},\\
x_{t+1} &= \tfrac{t}{t+2} x_t + \tfrac{2}{t+2} v_t.
\end{aligned}
$$

Convergence of Frank–Wolfe

Theorem (Frank-Wolfe Convergence)

If f is L-smooth w.r.t. $\|\cdot\|$ on $\operatorname{dom}\psi$, and x^* is a minimizer, then after t steps of the FW algorithm:

$$f(x_t) + \psi(x_t) - (f(x^*) + \psi(x^*)) \leq \frac{2L\,D^2}{t+1},$$

where $D = \max_{0 \leq i \leq t} \|v_i - x_i\|$ is the "step-size diameter" (e.g. diameter of the domain).

- No need to know t in advance.
- Each iteration calls one Linear Minimization Oracle.

Proof Sketch of FW Rate

1. Maintain a similar "dual" lower bound but substituting x^* by the minimizer v_t:

$$(f+\psi)(x^*) \geq \frac{1}{A_t}\sum_{i=0}^{t} a_i\big(f(x_i) + \langle \nabla f(x_i), x^* - x_i\rangle + \psi(x^*)\big).$$
$$\geq \frac{1}{A_t}\sum_{i=0}^{t} a_i\big(f(x_i) + \langle \nabla f(x_i), v_i - x_i\rangle + \psi(v_i)\big) =: L_t.$$

2. For $G_t \stackrel{\text{def}}{=} (f(x_{t+1}) + \psi(x_{t+1})) - L_t$, show

$$A_t G_t - A_{t-1} G_{t-1} \leq \frac{L\, a_t^2}{2A_t}\|v_t - x_t\|^2.$$

3. Choose $a_t = 2t+2$, $A_t = (t+1)(t+2)$.
4. Similarly as before, it telescopes to $O(1/t)$.

Remarks and Variants

- **Short-steps**: line-search on the upper bound given by smoothness.
- **Affine-invariant**: best norm and scaling automatically.
- **Lower bound**: any LMO-based method on $L\|x\|^2$ over the unit simplex needs $\Omega(LD^2/\varepsilon)$ queries when $t \leq d$.
- **Better rates for better sets**: Polytopes, strongly convex sets...
- **Applications**: Approximate Carathéodory, matrix completion (nuclear ball regularization), flow polytope, etc.

Example: Approximate Carathéodory

- Given $\hat{x} \in \mathcal{P} \stackrel{\text{def}}{=} \operatorname{conv}\{V\}$, find $\tilde{x}$ that is a convex combination of $\leq k$ vertices with $\|\hat{x} - \tilde{x}\| \leq \varepsilon$.
- Set $f(x) = \|x - \hat{x}\|^2$, $\psi = \mathcal{I}_{\mathcal{P}}$.
- Frank–Wolfe finds $\tilde{x} \in \mathcal{P}$ with $f(\tilde{x}) \leq \varepsilon^2$ in $O(D^2/\varepsilon^2)$ iterations $\implies \|\hat{x} - \tilde{x}\| \leq \varepsilon$. Each uses one linear minimization over V.
- Improves the classical $d + 1$ bound when $d \gg D^2/\varepsilon^2$.

Nesterov's Accelerated Gradient Descent (AGD) Methods

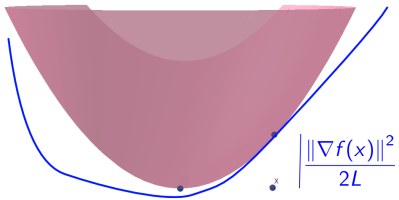
- Optimal first-order method for the minimization of Euclidean convex (resp. μ-strongly convex) and L-smooth functions.

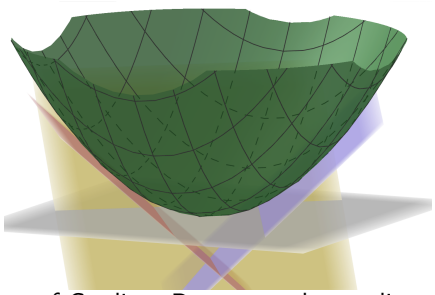
	$\mu > 0$ $[\kappa \stackrel{\text{def}}{=} L/\mu]$	$\mu = 0$
Accelerated Gradient Descent	$O(\sqrt{\kappa} \log 1/\varepsilon)$	$O(\sqrt{L/\varepsilon})$
Gradient Descent	$O(\kappa \log 1/\varepsilon)$	$O(L/\varepsilon)$

Nesterov's Accelerated Gradient Descent (AGD) Methods

- Optimal first-order method for the minimization of Euclidean convex (resp. μ-strongly convex) and L-smooth functions.

	$\mu > 0$ $[\kappa \stackrel{\text{def}}{=} L/\mu]$	$\mu = 0$
Accelerated Gradient Descent	$O(\sqrt{\kappa} \log 1/\varepsilon)$	$O(\sqrt{L/\varepsilon})$
Gradient Descent	$O(\kappa \log 1/\varepsilon)$	$O(L/\varepsilon)$





Accelerated Gradient Descent can be seen as a combination of Gradient Descent and an online learning algorithm that have, respectively, progress and instantaneous regret that are proportional to each other (proportional to $\|\nabla f(x)\|^2$ in the unconstrained case).

AGD: Algorithm (Sketch)

1. Maintain three sequences $x_t, z_t, y_t \in \mathcal{X}$ for optimizing $f + \phi$.
2. Weights $a_t = \frac{t}{2L}$, cumulative $A_t = \sum_{i=1}^{t} a_i = \frac{t(t+1)}{4L}$.
3. *Coupling step:*
$$x_t = \frac{A_{t-1}}{A_t} y_{t-1} + \frac{a_t}{A_t} z_{t-1}.$$
4. *Online subproblem (FTRL):*
$$z_t = \arg\min_{z \in \mathcal{X}} \Big\{ \sum_{i=1}^{t} a_i \langle \nabla f(x_i), z \rangle + A_t \phi(z) + \psi(z) \Big\}.$$
5. *Descent step:*
$$y_t = \arg\min_{y \in \mathcal{X}} \Big\{ Q_{f,x_t}(y) + \phi(y) \Big\},$$
where $Q_{f,x}(y) = f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|^2$.

Pseudocode in the Euclidean case

First version. (Nesterov, 1983)

$$x_{k+1} \leftarrow y_k + (a_k - 1)(y_k - y_{k-1})/a_{k+1}$$
$$y_{k+1} \leftarrow \arg\min_{y \in \mathcal{X}} \left\{ f(x_{k+1}) + \langle \nabla f(x_{k+1}), y - x_{k+1} \rangle + \frac{L}{2} \|y - x_{k+1}\|^2 \right\}$$

We have that $y_{k+1} = x_{k+1} - \frac{1}{L} \nabla f(x_{k+1})$ if the problem is unconstrained. If it's constrained gradients are queried possible outside of $\mathcal{X}$. Undesirable.

Improved version. (Nesterov, 1998)

$$a_{k+1} = (k+2)/L$$
$$\tau_k = 1/(k+2)$$
$$x_{k+1} \leftarrow \tau_k z_k + (1 - \tau_k) y_k$$
$$z_{k+1} \leftarrow \arg\min_{z \in \mathcal{X}} \left\{ \sum_{i=0}^{k} a_{i+1} \langle \nabla f(x_{i+1}), z \rangle + D_\psi(z, x_0) \right\}$$
$$y_{k+1} \leftarrow \tau_k z_{k+1} + (1 - \tau_k) y_k$$

Optimism and Acceleration

- Combine two ideas:

  1. *Anytime online-to-batch*: Last-iterate guarantees for optimization.

  2. *Optimistic FTRL*: use "hints" to reduce regret when losses vary smoothly.

- Leads to a simple accelerated method in the smooth convex case.

Optimistic FTRL (OFTRL)

Suppose $\psi_t = \frac{1}{\eta_t}\psi$ for a 1-strongly convex regularizer ψ with minimizer at x_1. *Given hints* $\tilde{\ell}_t$, *play*

$$x_t = \operatorname*{arg\,min}_{x \in \mathcal{X}} \Big\{ \sum_{i=1}^{t-1} \ell_i(x) + \tilde{\ell}_t(x) + \psi_t(x) \Big\}.$$

If $\ell_t - \tilde{\ell}_t$ is convex and $g_t - \tilde{g}_t \in \partial(\ell_t - \tilde{\ell}_t)(x_t)$, then

$$R_T(u) \;\le\; \frac{\psi(u)}{\eta_T} + \sum_{t=1}^{T} \left(\langle g_t - \tilde{g}_t, x_t - x_{t+1} \rangle - \frac{1}{2\eta_t} \|x_t - x_{t+1}\|^2 \right) \le \frac{\psi(u)}{\eta_T} + \sum_{t=1}^{T} \frac{\eta_t}{2} \|g_t - \tilde{g}_t\|_*^2.$$

Hints that predict the next gradient well, sharply reduce regret. But in the Lipschitz case, if predictions are quite off, we have a similar worse-case performance.
For acceleration, choose $\tilde{\ell}_t \equiv \langle \nabla f(x_{t-1}), \cdot \rangle$ given by the previous gradient.

Optimistic Acceleration Guarantee

Theorem

If $f : \mathbb{R}^n \to \mathbb{R}$ is L-smooth and convex with a minimizer x^*, the output of Anytime OFTRL with hint $\langle \nabla f(x_{t-1}), \cdot \rangle$ satisfies

$$f(\bar{x}_T) - f(x^*) = O\left(\frac{L D_\psi(u, x_0)}{T^2}\right).$$

Key steps:

- Apply anytime online-to-batch to relate $f(\bar{x}_T) - f(x^*)$ to regret and negative Bregman terms.
- Bound regret of OFTRL with hint-errors $\|\nabla f(\bar{x}_t) - \nabla f(\bar{x}_{t-1})\|_*^2$.
- Use smoothness to absorb these into the negative Bregman terms.
- Choose $a_t^2 \leq L\, A_{t-1}$ so all error terms cancel.

Generalizations

- Similar ideas can be used in the case of having an unbiased gradient oracle $x \mapsto g_x$ such that $\mathbb{E}[g_x] = \nabla f(x)$.
- Example. For $f(x) \stackrel{\text{def}}{=} \sum_{i=1}^n f_i(x)$, we can take a minibatch:

$$g_x \stackrel{\text{def}}{=} \sum_{i \in \mathcal{B}} \nabla f_i(x),$$

  where $\mathcal{B}$ is a subset of fixed size chosen uniformly at random among all such subsets.
- If we have bounded variance, one can achieve:

$$\mathbb{E}[f(x_T) - f(x^*)] = O\left(\frac{LR^2}{T^2} + \frac{D\sigma}{\sqrt{T}}\right).$$

- In the finite-sum case one has:

$$O\left(\sqrt{\frac{nLR^2}{\varepsilon}} + n \log\log(n)\right) \text{ for stochastic algs vs } O\left(n\sqrt{\frac{LR^2}{\varepsilon}}\right) \text{ for deterministic algs.}$$

Generalizations

- Accelerated Sparse PageRank (Martínez-Rubio, Wirth, et al., 2023).
- Accelerated Packing Proportional Fair Allocations (Criado et al., 2022).
- Accelerated Riemannian Optimization (Martínez-Rubio, 2022; Martínez-Rubio and Pokutta, 2023).
- Accelerated Non-Euclidean High-Order Smooth Convex Optimization (Contreras et al., 2024).
- Black-box Accelerated Stable Algorithms (Vary et al., 2024).

Overview

I am looking for PhD students and Postdocs. Tell your friends and/or contact me: David Martínez-Rubio (Madrid, Spain)

Scale Invariance of Optimization Methods

- **Scale Invariance:** An algorithm $\mathcal{A}$ is *scale invariant* if solving

$$\min_x^{\mathcal{A}} f(x) \quad \text{is equivalent to} \quad \min_{\hat{x}}^{\mathcal{A}} af(b\hat{x} + c) + d,$$

for any $a, b > 0$, $c \in \mathbb{R}^d$, $d \in \mathbb{R}$. That is, the iterates of the algorithm are equivalent.

Scale Invariance of Optimization Methods

- **Scale Invariance:** An algorithm $\mathcal{A}$ is *scale invariant* if solving

$$\min_x^{\mathcal{A}} f(x) \quad \text{is equivalent to} \quad \min_{\hat{x}}^{\mathcal{A}} af(b\hat{x} + c) + d,$$

  for any $a, b > 0$, $c \in \mathbb{R}^d$, $d \in \mathbb{R}$. That is, the iterates of the algorithm are equivalent.

- **Affine invariance:** For some methods $\mathcal{A}$, such as Frank-Wolfe, $\min^{\mathcal{A}} f(x) \leftrightarrow \min^{\mathcal{A}} f(Ax + b)$.

Scale Invariance of Optimization Methods

- **Scale Invariance:** An algorithm $\mathcal{A}$ is *scale invariant* if solving

$$\min_x^{\mathcal{A}} f(x) \quad \text{is equivalent to} \quad \min_{\hat{x}}^{\mathcal{A}} af(b\hat{x} + c) + d,$$

  for any $a, b > 0$, $c \in \mathbb{R}^d$, $d \in \mathbb{R}$. That is, the iterates of the algorithm are equivalent.

- **Affine invariance:** For some methods $\mathcal{A}$, such as Frank-Wolfe, $\min^{\mathcal{A}} f(x) \leftrightarrow \min^{\mathcal{A}} f(Ax + b)$.

- **Feature scale invariance:** For any vector λ, this is satisfied for algorithms such that $\min^{\mathcal{A}} f(x) \leftrightarrow \min^{\mathcal{A}} f(\text{diag}(\lambda)x)$.

  - Essentially satisfied by RMSProp, Adagrad, Adam, Lion...

Scale Invariance of Optimization Methods

(Kingma and Ba, 2014, Adam) is feature scale invariant.

$$g_t \leftarrow \nabla f_t(x_{t-1})$$

$$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t; \quad \hat{m}_t \leftarrow m_t/(1-\beta_1^t)$$

$$v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2; \quad \hat{v}_t \leftarrow v_t/(1-\beta_2^t)$$

$$x_t \leftarrow x_{t-1} - \eta \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \varepsilon)$$

Assigning Units for Scale Invariant Algorithms

Distance $\|x-y\|$: $[\|x-y\|] = \mathrm{u}_x$.

Function gap $f(x)-f(x^*)$: $[\cdot] = \mathrm{u}_f$.

Gradient $\nabla f(x)$: $[\nabla f] = \mathrm{u}_f/\mathrm{u}_x$.

Hessian $\nabla^2 f(x)$: $[\nabla f] = \mathrm{u}_f/{\mathrm{u}_x}^2$.

Smoothness $\nabla^2 f \preceq L\, I$: $[L] = \mathrm{u}_f/\mathrm{u}_x^2$.

Lipschitzness $|f(x)-f(y)| \leq G\|x-y\|$: $[G] = \mathrm{u}_f/\mathrm{u}_x$.

- **Check both sides of your bounds have the same units!** Nice for spotting errors!
  - $f(x_t)-f(x^*) = O(\frac{LR^2}{T})$, $x_{t+1} = x_t - \eta(\nabla^2 f(x_t))^{-1}\nabla f(x_t)$,
  - $\|x_t - x^*\| \leq 3R + 4R^2$, for $R = \|x_0 - x^*\|$ the initial distance. Possibly wrong!
  - Claim: An algorithm on a smooth convex problem satisfies $\|\nabla f(x_t)\| \leq \frac{LR^2}{\varepsilon}$. Wrong! If you scale you get as fast as you want, breaking lower bounds!
- **Sometimes in papers people assume wlog some constants are 1. Hard to keep track of units, try not to do that.**

A Cautionary Example

Suppose one proves

$$T = O\left(\max\{\frac{1}{L}, \frac{L}{\mu}\} \log\frac{LR^2}{\varepsilon}\right).$$

- Inside the log: $\frac{LR^2}{\varepsilon}$ is unitless, that is, $[\frac{LR^2}{\varepsilon}] = \mathrm{u}_x^0\mathrm{u}_f^0$.
- But $\frac{1}{L}$ has units $\mathrm{u}_x^2/\mathrm{u}_f$, while $\frac{L}{\mu}$ is unitless.
- Rate can be arbitrary slow if we optimize the rescaled $af(x)$, when a is small! Because $L \to 0$.
- Scaling with large a results in $T = O(\frac{L}{\mu} \log \frac{LR^2}{\varepsilon})$. Probably step sizes were wrong to begin with.

Overview

I am looking for PhD students and Postdocs. Tell your friends and/or contact me: David Martínez-Rubio (Madrid, Spain)

Reduction from Strongly Convex $\to$ Convex: Restarts

Suppose f is μ–strongly convex and an algorithm satisfies ① below

$$\tfrac{\mu}{2}\|x_t - x^\star\|^2 \;\le\; f(x_t) - f(x^\star) \overset{①}{\le} c_t\,\|x_0 - x^\star\|^2, \quad c_t \to_{t\to\infty} 0.$$

Then for some t with $c_t \le \frac{\mu}{8}$, $\|x_t - x^\star\| \le \frac{1}{2}\|x_0 - x^\star\|$. Restart to halve the distance each stage. Restarts work well in several applications (Roulet and d'Aspremont, 2017).

Reduction from Strongly Convex $\to$ Convex: Restarts

Suppose f is μ–strongly convex and an algorithm satisfies ① below

$$\tfrac{\mu}{2}\|x_t - x^\star\|^2 \;\le\; f(x_t) - f(x^\star) \overset{①}{\le} c_t\,\|x_0 - x^\star\|^2, \quad c_t \to_{t\to\infty} 0.$$

Then for some t with $c_t \le \frac{\mu}{8}$, $\|x_t - x^\star\| \le \frac{1}{2}\|x_0 - x^\star\|$. Restart to halve the distance each stage. Restarts work well in several applications (Roulet and d'Aspremont, 2017).

Rates of some algorithms for L-convex functions:

- **Gradient descent** converges as $O(\frac{1}{t})$ so it needs $t = O(L/\mu)$ to halve the distance to x^*. $\implies$ overall $O\big(\frac{L}{\mu}\log\frac{1}{\varepsilon}\big)$ steps.
- **Accelerated GD** needs $t = O(\sqrt{L/\mu})$ per halving. $\implies$ $O\big(\sqrt{\frac{L}{\mu}}\log\frac{1}{\varepsilon}\big)$.

These *linear convergence* rates.

Reduction from Strongly Convex $\to$ Convex: Restarts

Suppose f is μ–strongly convex and an algorithm satisfies ① below

$$\tfrac{\mu}{2}\|x_t - x^\star\|^2 \;\le\; f(x_t) - f(x^\star) \overset{①}{\le} c_t\,\|x_0 - x^\star\|^2, \quad c_t \to_{t\to\infty} 0.$$

Then for some t with $c_t \le \frac{\mu}{8}$, $\|x_t - x^\star\| \le \frac{1}{2}\|x_0 - x^\star\|$. Restart to halve the distance each stage. Restarts work well in several applications (Roulet and d'Aspremont, 2017).

- **Subgradient Descent (G-Lipschitz)** converges as $O(G\|x_0^{(t)} - x^*\|/\sqrt{T})$, so we restart every $O(G^2/(\|x_0^{(t)} - x^*\|^2\mu^2))$ we can halve the squared distance to x^* until it's $\le \frac{\mu}{\varepsilon}$ computing the following number of gradients.

$$O\left(\frac{G^2}{\mu^2}\left(\frac{1}{D^2} + \frac{4}{D^2} + \frac{16}{D^2} + \cdots + \frac{\mu}{\varepsilon}\right)\right) = O\left(\frac{G^2}{\mu^2}\cdot\frac{\mu}{\varepsilon}\sum_{i=0}^{\infty} 2^{-2i}\right) = O\left(\frac{G^2}{\mu\varepsilon}\right),$$

  which is optimal up to constants.

Last iteration as expensive as the rest. **Often the case:** with $\frac{1}{\varepsilon}$ convergence rates: to 2ε takes $\frac{1}{2\varepsilon}$, but from 2ε to ε takes the same $\frac{1}{\varepsilon} - \frac{1}{2\varepsilon}$.

Reductions from Convex $\to$ Strongly Convex: Regularization

- To solve $\min f(x)$ with minimizer $x^\star$, add $r(x) \geq 0$ s.t. $r(x^\star) \leq \frac{\varepsilon}{2}$, and approximately solve $\min_x \{f(x) + r(x)\}$.
- If $\hat{x}$ satisfies $(f + r)(\hat{x}) \leq \min(f + r) + \frac{\varepsilon}{2}$, then $f(\hat{x}) \leq f(x^\star) + \varepsilon$.
- Choose e.g. $r(x) = \frac{\varepsilon}{2D^2}\|x - x_0\|^2$.
- **Smooth case:** $f + r$ is $(L + \frac{\varepsilon}{D^2})$–smooth, $(\frac{\varepsilon}{D^2})$–strongly convex. Accelerated methods yield $\tilde{O}(\sqrt{LD^2/\varepsilon})$ oracle calls.
- **Lipschitz case:** $f + r$ is $(\frac{\varepsilon}{D^2})$-strongly convex, and we obtain rates of $\tilde{O}(G^2D^2/\varepsilon^2)$ from an optimal algorithm that runs in $O(G^2/(\mu\varepsilon))$.

Overview

I am looking for PhD students and Postdocs. Tell your friends and/or contact me: David Martínez-Rubio (Madrid, Spain)

Lower Bound Techniques: Hard functions

- The simplest hard function for the Euclidean Lipschitz convex class for $x_0 = 0$:

$$x \mapsto \max_{i \in [d]} \left\{ x_i - \frac{i}{d} \right\} \text{ for } x \in B(0,1).$$

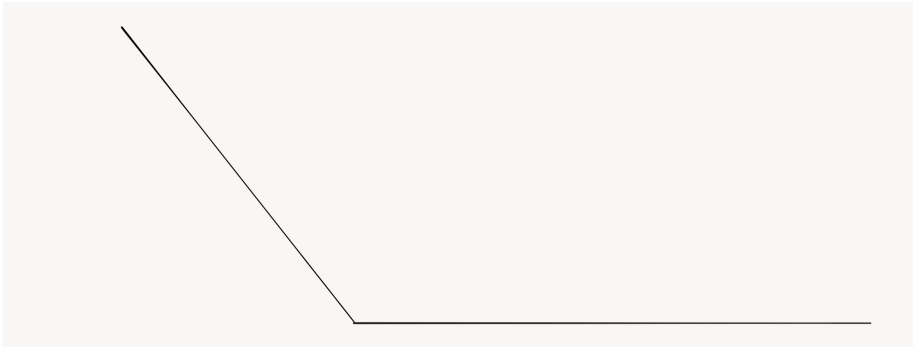
- If we have a point $x = (x_1, \ldots, x_k, 0, \ldots, 0)$ we only observe k of the linear functions!
- We need to observe them all to find the minimizer. We need many to approximate it.
- Leads to $T = \Omega(\frac{G^2 D^2}{\varepsilon^2})$ as long as this is $< d$.

Moreau Envelope: Smoothing the Hard Instance

$$M_{\lambda,f}(x) \stackrel{\text{def}}{=} \min_y \left\{ f(y) + \frac{1}{2\lambda}\|y-x\|_2^2 \right\}; \quad \text{prox}_{\lambda,f}(x) \stackrel{\text{def}}{=} \arg\min_y \left\{ f(y) + \frac{1}{2\lambda}\|y-x\|_2^2 \right\}.$$

By optimality $\nabla_y \left(f(y) + \frac{1}{2\lambda}\|y-x\|_2^2 \right) (\text{prox}(x)) = 0$, so

$\text{prox}(x) = x - \lambda \nabla f(\text{prox}(x))$, i.e., implicit Gradient Descent. And minimizers are preserved.

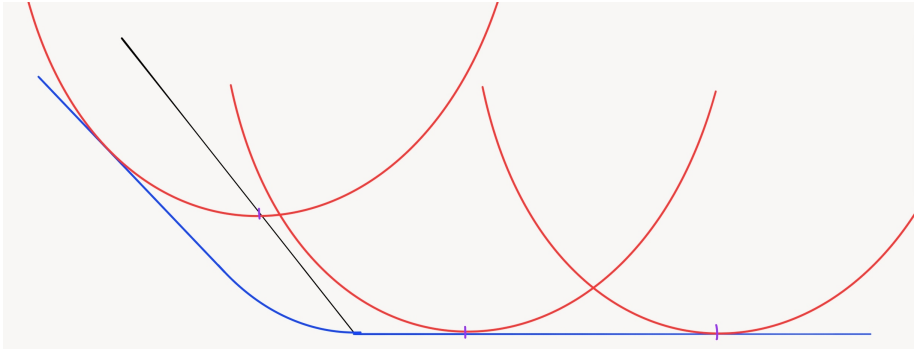


Moreau Envelope: Smoothing the Hard Instance

$$M_{\lambda,f}(x) \stackrel{\text{def}}{=} \min_y \left\{ f(y) + \frac{1}{2\lambda}\|y-x\|_2^2 \right\}; \quad \text{prox}_{\lambda,f}(x) \stackrel{\text{def}}{=} \arg\min_y \left\{ f(y) + \frac{1}{2\lambda}\|y-x\|_2^2 \right\}.$$

By optimality $\nabla_y \left(f(y) + \frac{1}{2\lambda}\|y-x\|_2^2 \right) (\text{prox}(x)) = 0$, so

$\text{prox}(x) = x - \lambda \nabla f(\text{prox}(x))$, i.e., implicit Gradient Descent. And minimizers are preserved.



Moreau Envelope: Smoothing the Hard Instance

Goal. Lower bound $T = \Omega(\sqrt{L}R/\sqrt{\varepsilon})$ for L-smooth f on $\|x\| \le R$.
Properties:

1. Well-defined & *jointly* strongly convex $\implies$ unique prox.
2. $M_{f,\lambda}$ is convex (marginalization of a jointly convex function).
3. *Surrogate:*
$$M_{f,\lambda}(x) \le f(x), \quad M_{f,\lambda}(x^\star) = f(x^\star) \implies \mathrm{prox}(x^\star) = x^\star.$$
4. $\nabla M_{f,\lambda}(x) = \frac{1}{\lambda}\left(x - \mathrm{prox}_{f,\lambda}(x)\right)$.
5. $M_{f,\lambda}(x)$ is $\frac{1}{\lambda}$-smooth.
6. Local if f is G-Lipschitz: depends only on f near x (radius $O(\lambda G)$).
7. Close to f if f is G-Lipschitz: $|M_{f,\lambda}(x) - f(x)| \le G^2\lambda$.

Apply it to the hard Lipschitz instance and obtain a hard smooth instance.

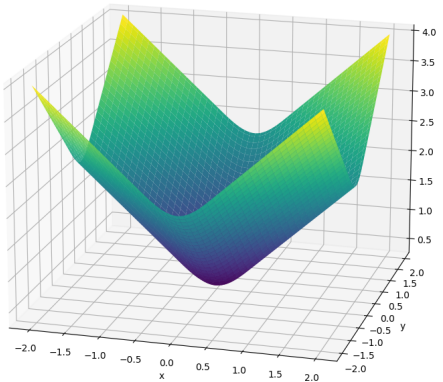
Lower Bound Techniques: Randomized Smoothing

- Randomized Smoothing of G-Lipschitz f wrt $\|\cdot\|$:

$$S_\beta[f](x) \stackrel{\text{def}}{=} \mathbb{E}_{v\sim\nu_{B_{\|\cdot\|}(\mathbf{o},\beta)}}[f(x+v)]$$

- $S_\beta[f](x)$ is also G-Lipschitz and its gradient is $\frac{dG}{\beta}$-Lipschitz wrt $\|\cdot\|$.
- $|S_\beta[f](x) - f(x)| \leq \beta G$.
- Since $S_\beta[f](x)$ or $M_{\lambda,f}$ depends on f locally it will preserve local hardness.

Figure: A smoothing of $x \mapsto \|x\|_1$ wrt $\|\cdot\|_1$



Lower Bound Techniques: Softmax and our Final Hard Function

- Let $A(x) = (\langle a^{(1)}, x\rangle, \ldots, \langle a^{(d)}, x\rangle)$, with $\|a^{(i)}\|_* \leq 1$.
- Define the softmax function as $\mathrm{smax}_\mu(x) \stackrel{\mathrm{def}}{=} \mu \ln\left(\sum_{j=1}^{d} \exp(x_i/\mu)\right)$.

Lower Bound Techniques: Softmax and our Final Hard Function

- Let $A(x) = (\langle a^{(1)}, x\rangle, \ldots, \langle a^{(d)}, x\rangle)$, with $\|a^{(i)}\|_* \leq 1$.
- Define the softmax function as $\mathrm{smax}_\mu(x) \stackrel{\text{def}}{=} \mu \ln\left(\sum_{j=1}^d \exp(x_i/\mu)\right)$.
- $\mathrm{smax}_\mu(Ax)$ is 1-Lipschitz wrt $\|\cdot\|$.
- $\nabla^q \mathrm{smax}_\mu$ is $\widetilde{O}_q(\frac{1}{\mu^q})$-Lipschitz wrt $\|\cdot\|$.

Lower Bound Techniques: Softmax and our Final Hard Function

- Let $A(x) = (\langle a^{(1)}, x\rangle, \ldots, \langle a^{(d)}, x\rangle)$, with $\|a^{(i)}\|_* \leq 1$.
- Define the softmax function as $\mathrm{smax}_\mu(x) \stackrel{\text{def}}{=} \mu \ln\left(\sum_{j=1}^d \exp(x_i/\mu)\right)$.
- $\mathrm{smax}_\mu(Ax)$ is 1-Lipschitz wrt $\|\cdot\|$.
- $\nabla^q \mathrm{smax}_\mu$ is $\widetilde{O}_q(\frac{1}{\mu^q})$-Lipschitz wrt $\|\cdot\|$.
- $f_i \equiv$ softmax of $(Ax)_1 - \gamma, \ldots, (Ax)_i - i\gamma$, for some $\gamma > 0$, up to some shifts.
- $h(x) \stackrel{\text{def}}{=} \max_{i\in[T]} f_i(x)$.
- Hard function $g(x) = (S_{\beta/2^q} \circ S_{\beta/2^{q-1}} \circ \cdots \circ S_{\beta/2})(h)$.

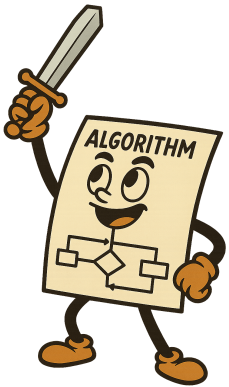
Convergence Results

Suppose convexity and $\|\nabla^q f(x) - \nabla^q f(y)\|_* \leq L_q \|x - y\|$.
Initial distance: $R_p \overset{\text{def}}{=} \|x_0 - x^*\|_p$; Accuracy: ε;
$m \overset{\text{def}}{=} \max\{2, p\}$. $\widetilde{O}_p(\cdot)$: big-O notation up to log factors and constants on p.
(Contreras et al., 2024).

Algorithm	$p \in [1, \infty)$	$p = \infty$
Accelerated ($q < \infty$)	$\widetilde{O}_{q,p}\left(\left(\frac{LR_p^{q+\mathbf{1}}}{\varepsilon}\right)^{\frac{m}{(m+\mathbf{1})(q+\mathbf{1})-m}}\right)$	–
Unaccelerated ($q < \infty$)	$\widetilde{O}_q\left(\left(\frac{LR_p^{q+\mathbf{1}}}{\varepsilon}\right)^{\frac{\mathbf{1}}{q}}\right)$	

Corresponding Lower Bounds

If f convex and $\|\nabla^q f(x) - \nabla^q f(y)\|_* \leq L_q \|x - y\|$ (Contreras et al., 2024):



Algorithms	Lower Bounds
q-th order oracle	Local oracle
Deterministic	Possibly random algs
Single-call per round	$\text{poly}(d)$ parallel queries
Any norm	Any norm
$\|\cdot\|_p$-setting: they match up to log factors	

Overview

I am looking for PhD students and Postdocs. Tell your friends and/or contact me: David Martínez-Rubio (Madrid, Spain)

Optimizers in Deep Learning 0

- **The following is an incomplete overview.** There are many more important methods.
- We will use $f_t \stackrel{\text{def}}{=} \frac{1}{m} \sum_{i=1}^{m} f^{(j_i)}$ to represent a minibatch.
- We will use capital letters $G_t \leftarrow \nabla f_t$ to represent the gradient of f_t grouped as a collection of matrices, one per-layer.

Optimizers in Deep Learning 1

(Robbins and Monro, 1951, SGD)

$$g_t \leftarrow x_{t-1} - \eta_t \nabla f_t(x_{t-1})$$

Simple, effective.

(McMahan and Streeter, 2010; Duchi et al., 2011, AdaGrad). For a minibatch f_t:

$$
\begin{aligned}
g_t &\leftarrow \nabla f_t(x_{t-1}) \\
v_t &\leftarrow v_{t-1} + g_t^2 \\
x_t &\leftarrow x_{t-1} - \eta \frac{g_t}{\sqrt{v_t} + \varepsilon}
\end{aligned}
$$

Theorem (AdaGrad's Guarantee)

$$
\text{Regret}_T \leq \max_{t \leq T} \|x_t - x^*\|_\infty d^{1/2} \sqrt{\inf \left\{ \sum_{t=1}^{T} \langle g_t, \text{diag}(s)^{-1} g_t \rangle : s \succcurlyeq 0, \langle 1, s \rangle \leq d \right\}},
$$

where x^* is the best fixed action defining the regret.

Optimizers in Deep Learning 3

(Tieleman and Hinton, 2012, RMSProp)

- Version of Adagrad but with v_t accumulating with exponential average (forget distant past).
- β is chosen to be close to 1.

$$
\begin{aligned}
g_t &\leftarrow \nabla_\theta f_t(\theta_{t-1}) \\
v_t &\leftarrow \beta v_{t-1} + (1-\beta) g_t^2 \\
\theta_t &\leftarrow \theta_{t-1} - \eta \frac{g_t}{\sqrt{v_t} + \varepsilon}
\end{aligned}
$$

Optimizers in Deep Learning 4

(Kingma and Ba, 2014, Adam)

- Mutation of Adagrad and RMSProp to incorporate momentum in the numerator.
- It also contains a bias correction factor.
- Only cited 221,892 times (by night August 7, 2025). Only 222,029 in the morning of August 8.

$$
\begin{aligned}
g_t &\leftarrow \nabla f_t(x_{t-1}) \\
m_t &\leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t; \quad \hat{m}_t \leftarrow m_t/(1-\beta_1^t) \\
v_t &\leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2; \quad \hat{v}_t \leftarrow v_t/(1-\beta_2^t) \\
x_t &\leftarrow x_{t-1} - \eta \cdot \hat{m}_t/(\sqrt{\hat{v}_t} + \varepsilon)
\end{aligned}
$$

- Coordinatewise scale invariance.

Optimizers in Deep Learning 5

(Loshchilov and Hutter, 2017, AdamW) decoupling momentum from weight decay (this is like proximal gradient). Leads to better regularization.

Algorithm 2 Adam with L_2 regularization and Adam with decoupled weight decay (AdamW)

given $\alpha = 0.001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}, \lambda \in \mathbb{R}$
initialize time step $t \leftarrow 0$, parameter vector $\boldsymbol{\theta}_{t=0} \in \mathbb{R}^n$, first moment vector $\boldsymbol{m}_{t=0} \leftarrow \boldsymbol{0}$, second moment vector $\boldsymbol{v}_{t=0} \leftarrow \boldsymbol{0}$, schedule multiplier $\eta_{t=0} \in \mathbb{R}$
repeat
  $t \leftarrow t + 1$
  $\nabla f_t(\boldsymbol{\theta}_{t-1}) \leftarrow \text{SelectBatch}(\boldsymbol{\theta}_{t-1})$ ▷ select batch and return the corresponding gradient
  $\boldsymbol{g}_t \leftarrow \nabla f_t(\boldsymbol{\theta}_{t-1}) + \lambda \boldsymbol{\theta}_{t-1}$
  $\boldsymbol{m}_t \leftarrow \beta_1 \boldsymbol{m}_{t-1} + (1 - \beta_1)\boldsymbol{g}_t$ ▷ here and below all operations are element-wise
  $\boldsymbol{v}_t \leftarrow \beta_2 \boldsymbol{v}_{t-1} + (1 - \beta_2)\boldsymbol{g}_t^2$
  $\hat{\boldsymbol{m}}_t \leftarrow \boldsymbol{m}_t / (1 - \beta_1^t)$ ▷ β_1 is taken to the power of t
  $\hat{\boldsymbol{v}}_t \leftarrow \boldsymbol{v}_t / (1 - \beta_2^t)$ ▷ β_2 is taken to the power of t
  $\eta_t \leftarrow \text{SetScheduleMultiplier}(t)$ ▷ can be fixed, decay, or also be used for warm restarts
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \eta_t \left(\alpha \hat{\boldsymbol{m}}_t / (\sqrt{\hat{\boldsymbol{v}}_t} + \epsilon) + \lambda \boldsymbol{\theta}_{t-1} \right)$
until *stopping criterion is met*
return optimized parameters $\boldsymbol{\theta}_t$

(Gupta et al., 2018, SHAMPOO). This is a of normalizing the per-layer gradient matrices. Normalized subgradient descent $x_t \leftarrow x_t - \frac{g_t}{\|g_t\|}$ was used as early as in (Nesterov, 1998, Theorem 3.2.2) to obtain rates for convex non-smooth methods without knowledge of the (local) Lipschitz constant.

$$
\begin{aligned}
G_t &\leftarrow \nabla f_t(W_t) \in \mathbb{R}^{m \times n} \\
L_t &\leftarrow L_{t-1} + G_t \, G_t^\top \\
R_t &\leftarrow R_{t-1} + G_t^\top \, G_t \\
W_{t+1} &\leftarrow W_t - \eta \, L_t^{-\frac{1}{4}} \, G_t \, R_t^{-\frac{1}{4}}
\end{aligned}
$$

Lion (Chen et al., 2023)
More direct normalized gradient descent, with momentum.

$$
\begin{aligned}
g_t &\leftarrow \nabla f_t(x_{t-1}) \\
x_t &\leftarrow x_{t-1} - \eta \cdot \operatorname{sign}(\beta_2 m_{t-1} + (1-\beta_2) g_t) \\
m_t &\leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t
\end{aligned}
$$

Optimizers in Deep Learning 8

(Jordan, 2025, Muon)

- Muon is an optimizer by layers, where G_t represents a collection of 2D gradients. And starts with a momentum parameter $B_0 \leftarrow 0$.
- The update is approximately orthogonalized before being used.

$$
\begin{aligned}
G_t &\leftarrow \nabla f_t(x_{t-1}) \\
B_t &\leftarrow \mu B_{t-1} + G_t \\
O_t &\leftarrow \text{NewtonSchultz5}(B_t) \\
\theta_t &\leftarrow \theta_{t-1} - \eta O_t
\end{aligned}
$$

- Layer orthogonalization explored before by, e.g. (Arjovsky et al., 2016; Mhammedi et al., 2017; Maduranga et al., 2019; Lezcano-Casado and Martínez-Rubio, 2019), in the context of RNNs for preventing vanishing or exploding gradients for long sequences.
- Current models are very deep, and layer orthogonalization also becomes a very useful tool. It also acts as a normalization step. The algorithm is scale invariant.

Thank you!

Questions?

References I

Allen-Zhu, Zeyuan, Yuanzhi Li, and Zhao Song (2019). "A convergence theory for deep learning via over-parameterization". In: *International conference on machine learning*. PMLR, pp. 242–252.

Arjovsky, Martin, Amar Shah, and Yoshua Bengio (2016). "Unitary evolution recurrent neural networks". In: *International conference on machine learning*. PMLR, pp. 1120–1128.

Carmon, Yair, John C Duchi, Oliver Hinder, and Aaron Sidford (2020). "Lower bounds for finding stationary points I". In: *Mathematical Programming* 184.1, pp. 71–120.

— (June 2017). ""Convex Until Proven Guilty": Dimension-Free Acceleration of Gradient Descent on Non-Convex Functions". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, pp. 654–663.

Chen, Xiangning, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, et al. (2023). "Symbolic discovery of optimization algorithms". In: *Advances in neural information processing systems* 36, pp. 49205–49233.

Contreras, Juan Pablo, Cristóbal Guzmán, and David Martínez-Rubio (2024). "Non-Euclidean High-Order Smooth Convex Optimization". In: *arXiv preprint arXiv:2411.08987*.

References II

Criado, Francisco, David Martínez-Rubio, and Sebastian Pokutta (2022). "Fast algorithms for packing proportional fairness and its dual". In: *Advances in Neural Information Processing Systems* 35, pp. 25754–25766.

Cutkosky, Ashok (2019). "Anytime Online-to-Batch, Optimism and Acceleration". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, pp. 1446–1454.

Cutkosky, Ashok, Harsh Mehta, and Francesco Orabona (2023). "Optimal stochastic non-smooth non-convex optimization through online-to-non-convex conversion". In: *International Conference on Machine Learning*. PMLR, pp. 6643–6670.

Defazio, Aaron, Ashok Cutkosky, Harsh Mehta, and Konstantin Mishchenko (2023). "Optimal linear decay learning rate schedules and further refinements". In: *arXiv preprint arXiv:2310.07831*.

Du, Simon, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai (2019). "Gradient descent finds global minima of deep neural networks". In: *International conference on machine learning*. PMLR, pp. 1675–1685.

References III

Duchi, John, Elad Hazan, and Yoram Singer (2011). "Adaptive subgradient methods for online learning and stochastic optimization." In: *Journal of machine learning research* 12.7.

Gunasekar, Suriya, Jason Lee, Daniel Soudry, and Nathan Srebro (2018). "Characterizing implicit bias in terms of optimization geometry". In: *International Conference on Machine Learning*. PMLR, pp. 1832–1841.

Gupta, Vineet, Tomer Koren, and Yoram Singer (2018). "Shampoo: Preconditioned stochastic tensor optimization". In: *International Conference on Machine Learning*. PMLR, pp. 1842–1850.

Hinder, Oliver, Aaron Sidford, and Nimit Sohoni (2020). "Near-optimal methods for minimizing star-convex functions and beyond". In: *Conference on learning theory*. PMLR, pp. 1894–1938.

Jordan, Keller (2025). [Online; accessed July-28-2025], https://kellerjordan.github.io/posts/muon/.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Lezcano-Casado, Mario and David Martínez-Rubio (2019). "Cheap orthogonal constraints in neural networks: A simple parametrization of the orthogonal and unitary group". In: *International Conference on Machine Learning*. PMLR, pp. 3794–3803.

References IV

Li, Huan and Zhouchen Lin (2023). "Restarted Nonconvex Accelerated Gradient Descent: No More Polylogarithmic Factor in the in the O (epsilonˆ(-7/4)) Complexity". In: *Journal of Machine Learning Research* 24.157, pp. 1–37.

Loshchilov, Ilya and Frank Hutter (2017). "Decoupled weight decay regularization". In: *arXiv preprint arXiv:1711.05101*.

Maduranga, Kehelwala DG, Kyle E Helfrich, and Qiang Ye (2019). "Complex unitary recurrent neural networks using scaled cayley transform". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01, pp. 4528–4535.

Martínez-Rubio, David (2022). "Global Riemannian acceleration in hyperbolic and spherical spaces". In: *International conference on algorithmic learning theory*. PMLR, pp. 768–826.

Martínez-Rubio, David and Sebastian Pokutta (2023). "Accelerated riemannian optimization: Handling constraints with a prox to bound geometric penalties". In: *The Thirty Sixth Annual Conference on Learning Theory*. PMLR, pp. 359–393.

Martínez-Rubio, David, Elias Wirth, and Sebastian Pokutta (2023). "Accelerated and sparse algorithms for approximate personalized PageRank and beyond". In: *The Thirty Sixth Annual Conference on Learning Theory*. PMLR, pp. 2852–2876.

References V

Marumo, Naoki and Akiko Takeda (2024). "Parameter-free accelerated gradient descent for nonconvex minimization". In: *SIAM Journal on Optimization* 34.2, pp. 2093–2120.

McMahan, H Brendan and Matthew Streeter (2010). "Adaptive bound optimization for online convex optimization". In: *arXiv preprint arXiv:1002.4908*.

Mhammedi, Zakaria, Andrew Hellicar, Ashfaqur Rahman, and James Bailey (2017). "Efficient orthogonal parametrisation of recurrent neural networks using householder reflections". In: *International Conference on Machine Learning*. PMLR, pp. 2401–2409.

Nesterov, Yu and Vladimir Shikhman (2015). "Quasi-monotone subgradient methods for nonsmooth convex minimization". In: *Journal of Optimization Theory and Applications* 165.3, pp. 917–940.

Nesterov, Yurii (1983). "A method for solving the convex programming problem with convergence rate $O(1/k^2)$". In: *Dokl. akad. nauk Sssr*. Vol. 269. 3, pp. 543–547.

— (1998). "Introductory lectures on convex programming volume I: Basic course". In: *Lecture notes* 3.4, p. 5.

Robbins, Herbert and Sutton Monro (1951). "A stochastic approximation method". In: *The annals of mathematical statistics*, pp. 400–407.

References VI

Roulet, Vincent and Alexandre d'Aspremont (2017). "Sharpness, restart and acceleration". In: *Advances in Neural Information Processing Systems* 30.

Schaipp, Fabian, Alexander Hägele, Adrien Taylor, Umut Simsekli, and Francis Bach (2025). "The surprising agreement between convex optimization theory and learning-rate scheduling for large model training". In: *arXiv preprint arXiv:2501.18965*.

Tieleman, Tijmen and Geoffrey Hinton (2012). "Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning". In: *COURSERA Neural Networks Mach. Learn* 17, p. 6.

Tran, Hoang, Qinzi Zhang, and Ashok Cutkosky (2024). "Empirical tests of optimization assumptions in deep learning". In: *arXiv preprint arXiv:2407.01825*.

Vary, Simon, David Martínez-Rubio, and Patrick Rebeschini (2024). "Black-Box Uniform Stability for Non-Euclidean Empirical Risk Minimization". In: *arXiv preprint arXiv:2412.15956*.

Zhang, Jingzhao, Hongzhou Lin, Stefanie Jegelka, Suvrit Sra, and Ali Jadbabaie (2020). "Complexity of finding stationary points of nonconvex nonsmooth functions". In: *International Conference on Machine Learning*. PMLR, pp. 11173–11182.

References VII

Zhang, Qinzi and Ashok Cutkosky (2024). "Random scaling and momentum for non-smooth non-convex optimization". In: *arXiv preprint arXiv:2405.09742*.