Lecture 5. Online Convex Optimization

Advanced Optimization (Fall 2026)

Peng Zhao

zhaop@lamda.nju.edu.cn

Nanjing University

Outline

- Online Optimization
- Online Convex Optimization
- Online-to-Batch Conversion

Part 1. Online Optimization

- Fixed Optimization
- Interactive Optimization
- Online Optimization: Problem and Measure

Optimization

- Originally, we focus on the optimization problem: $\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$

Iterative algorithm (like GD) generates a sequence of iterates $\mathbf{x}_1 \ldots, \mathbf{x}_T$, aiming to ensure that $F(\mathbf{x}_T) - F(\mathbf{x}^\star)$ is small with respect to T

for simplicity, consider last-iterate convergence

For each round $t = 1, \ldots, T$:

- Algorithm outputs a decision $\mathbf{x}_t \in \mathcal{X}$.
- Algorithm has information about $F(\mathbf{x}_t)$ and $\nabla F(\mathbf{x}_t)$.

⇨ Note that the objective function remains ***fixed*** throughout the entire optimization process as the iterative algorithm progresses.

Interactive Optimization

- However, in many ML applications, the optimization objective ***does not stay fixed***.

> For each round $t = 1, \ldots, T$:
>
> - Algorithm outputs a decision $\mathbf{x}_t \in \mathcal{X}$.
> - A new loss function $f_t : \mathcal{X} \to \mathbb{R}$ is revealed.
> - Algorithm suffers the loss $f_t(\mathbf{x}_t)$.

Optimization is no longer performed against a single, fixed target;

instead, it involves interactions between an algorithm and ***an evolving objective***.

Example 1: Large-Scale ERM

- Consider the model training. Our goal is to minimize ERM

$$F(\mathbf{x}) = \frac{1}{M} \sum_{i=1}^{M} \ell(\mathbf{x}; z_i)$$

  - Big data: facing millions of samples (M is very large).
  - Computing full gradient is almost impossible (like due to limited memory of GPU facilities).

Stochastic Optimization as Interaction

Stochastic optimization using ***mini-batch*** (by SGD/Adam or others)

$$f_t(\mathbf{x}) = \frac{1}{|S_t|} \sum_{z_i \in S_t} \ell(\mathbf{x}; z_i)$$

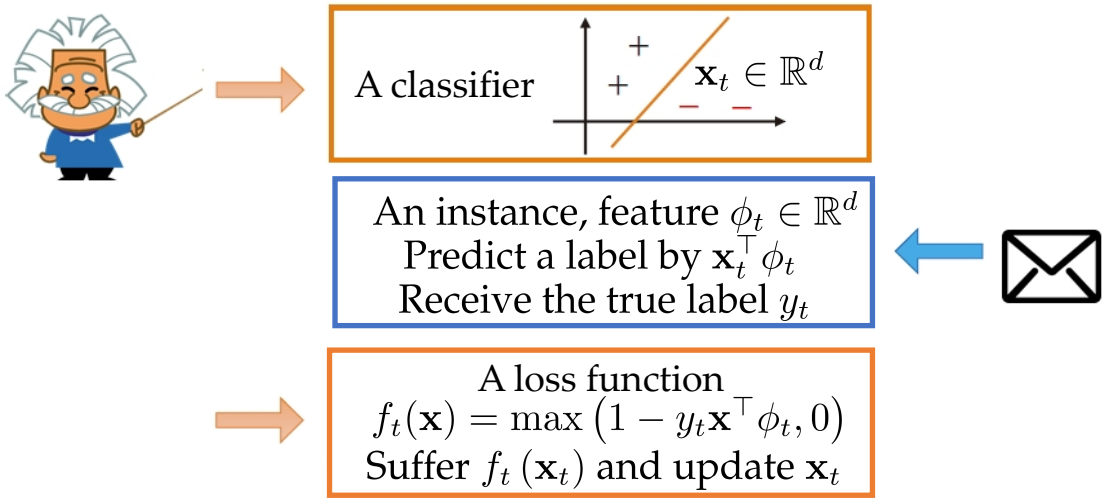
Note that S_t is still sampled from a fixed distribution (over all M samples)

- The optimization objective f_t varies across iterations (due to changing mini-batches).
- This renders the optimization process *interactive* — the algorithm continuously adapts in response to a new batch of samples.

Example 2: Learning with Data Streams

- In applications, data may arrive in a form of stream, which requires the model to make ***sequential predictions or even decisions***.

- For example: spam filtering



Sequential Predictions as Interaction

- We can formulate the sequential prediction problem as follows:

At each step t:

- The learner outputs a prediction $\mathbf{x}_t$.
- Then a new sample z_t is revealed.
- The instantaneous loss is $f_t(\mathbf{x}_t) = \ell(\mathbf{x}_t; z_t)$.

Note that the loss function f_t is sampled from (unknown) distribution $\mathcal{D}_\star$ or $\mathcal{D}_t$

- If data are stationary, we can view f_t as sampled from a fixed distribution $\mathcal{D}_\star$.
- Alternatively, if having non-stationarity, distribution $\mathcal{D}_t$ may vary with t.

Online Optimization: distribution-free

- Online optimization: an even general framework without distributional assumption.
- The function f_t may be arbitrary, or even chosen adversarially

- The general interactive procedures of online optimization:

For each round $t = 1, \ldots, T$:

- Learner gives a decision $\mathbf{x}_t \in \mathcal{X}$.
- Learner observes $f_t : \mathcal{X} \to \mathbb{R}$ and suffers the loss $f_t(\mathbf{x}_t)$.

A Game-theoretic Language

- Online (Interactive) optimization as a ***repeated game*** between
  - **Player**: essentially the learner, or you can think as the "learning model"
  - **Environments**: an abstraction of all factors evaluating the model.

At each round $t = 1, 2, \cdots$

- Player first picks a model $\mathbf{x}_t \in \mathcal{X}$.
- Simultaneously environments pick an online function $f_t : \mathcal{X} \to \mathbb{R}$.
- Player suffers loss $f_t(\mathbf{x}_t)$, observes some information about f_t and updates the model.

Fixed vs Interactive Optimization

- Fixed Optimization (Closed-world Learning)
  - The training data are all available on hand
  - The objective is fixed with no changes.

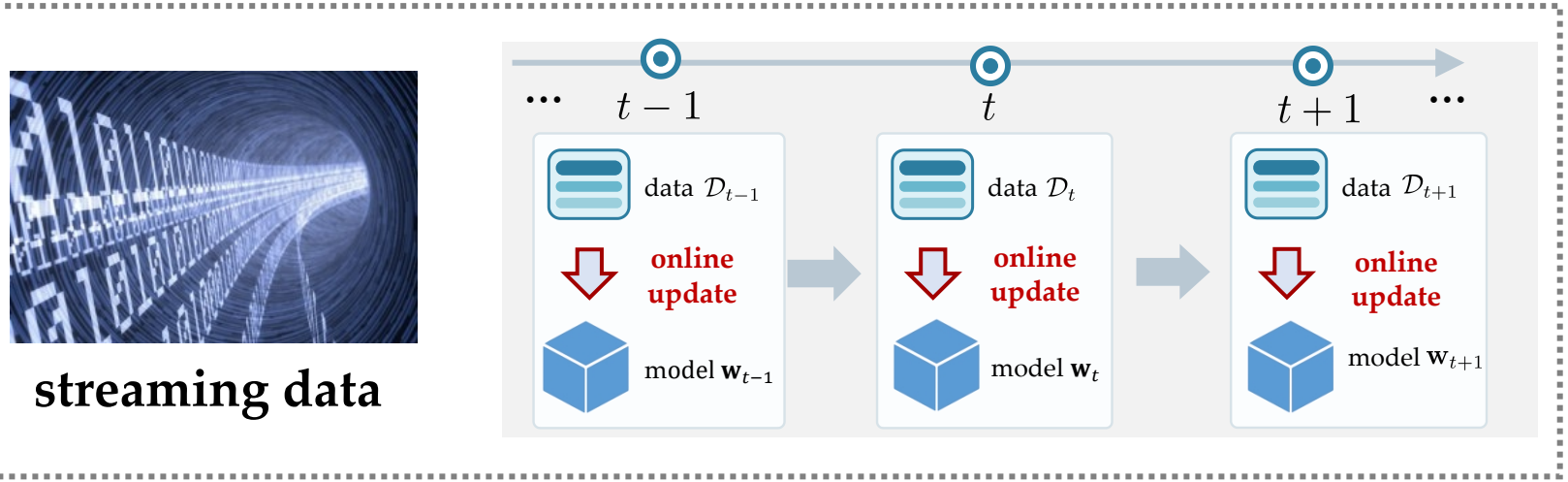


- Online Optimization (Open-ended Learning)
  - This may be due to sampling issue, so connected to stochastic optimization
  - Or since data are in the form of stream, so it is crucial for continual learning
  - Even more complicated: decision this round may also influent the environment, decision-theoretic RL/control

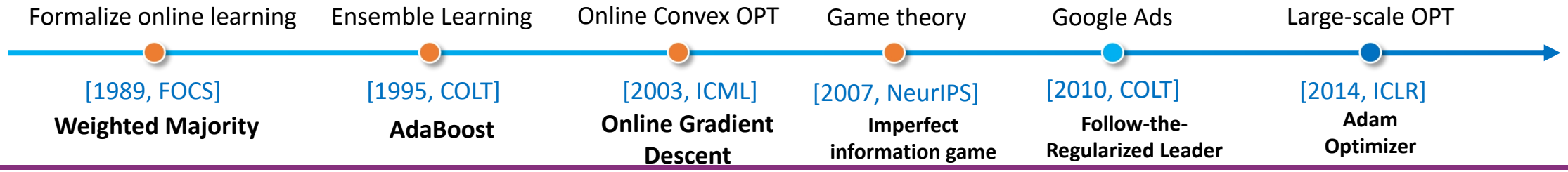


Online Learning/Optimization

- Online learning: for scenarios where data accumulates continuously



- Fundamental role in ML and its connections to many disciplines



Performance Measure

- The best in hindsight

  After T rounds, suppose we look back and imagine that we *did* know all $\{f_t\}_{t=1}^T$.

  Then the best single decision in hindsight would be:

$$\mathbf{x}^\star \in \arg\min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^T f_t(\mathbf{x})$$

  which we obviously could ***not*** know in advance.

Regret: $$\mathrm{REG}_T = \sum_{t=1}^T f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^T f_t(\mathbf{x})$$

benchmark performance with the offline model (optimal in hindsight)

"If I had known all the outcomes ahead of time, I would have chosen differently — and I regret the extra loss I've accumulated."

Performance Measure

- In online learning, we define ***regret*** as measure for $\{\mathbf{x}_t\}_{t=1}^T$:

$$\text{REG}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\mathbf{x})$$

benchmark performance with the offline model (optimal in hindsight)

We hope the regret be sub-linear dependence with T

$$\frac{\text{REG}_T}{T} \to 0 \text{ as } T \to \infty$$

Hannan Consistency

ALT'16

Hannan Consistency in On-Line Learning in Case of Unbounded Losses Under Partial Monitoring⋆,⋆⋆

Chamy Allenberg[1], Peter Auer[2], László Györfi[3], and György Ottucsák[3]

worst-case-dynamic regret $\text{REG}_T(\{\mathbf{x}_t^*\}) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}_t^*)$ *suffer from overfitting issue*

general dynamic regret $\text{REG}_T(\{\mathbf{u}_t\}) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{u}_t)$ *"online ensemble" framework*

Part 2. Online Convex Optimization

- Online Convex Optimization
- Problem Space
- Online Gradient Descent
- Lower Bound

Is Online Optimization (provably) solvable?

- In general, the online optimization is ***too hard to solve***.

At each round $t = 1, 2, \cdots$

- Player first picks a model $\mathbf{x}_t \in \mathcal{X}$.
- Simultaneously environments pick an online function $f_t : \mathcal{X} \to \mathbb{R}$.
- Player suffers loss $f_t(\mathbf{x}_t)$, observes some information about f_t and updates the model.

⇒ A Trackable Case: ***Online Convex Optimization***

requiring feasible domain and online functions to be convex

Online Convex Optimization

- Requirements:
  (1) feasible domain is a convex set
  (2) online functions are convex

At each round $t = 1, 2, \cdots$

- Player first picks a model $\mathbf{x}_t$ from a convex set $\mathcal{X} \subseteq \mathbb{R}^d$.
- Environments pick an online convex function $f_t : \mathcal{X} \to \mathbb{R}$.
- Player suffers loss $f_t(\mathbf{x}_t)$, observes some information about f_t and updates the model.

Online Convex Optimization: Hardness

Clearly, curvature information influences:

- Convex vs Strongly convex?
- Lipschitz vs Smooth?

There are other issues due to the ***interaction*** nature:

- **Feedback**: How much information can the learner access from environments?
- **Environments**: How powerful is the environment?

  Given that it can choose the loss function!

Hardness: Different Feedback

At each round $t = 1, 2, \cdots$

- Player first picks a model $\mathbf{x}_t$ from a convex set $\mathcal{X} \subseteq \mathbb{R}^d$.
- Environments pick an online convex function $f_t : \mathcal{X} \to \mathbb{R}$.
- Player suffers loss $f_t(\mathbf{x}_t)$, observes some information about f_t and updates the model.

on the feedback information:

- ***full information***: observe entire f_t (or at least gradient $\nabla f_t(\mathbf{x}_t)$)
- ***partial information (bandits)***: observe function value $f_t(\mathbf{x}_t)$ only

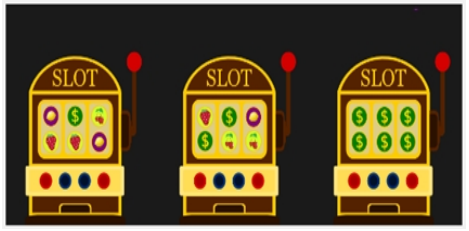
less information

full information



horse racing

partial information



multi-armed bandits

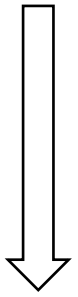
Hardness: Different Environments

At each round $t = 1, 2, \cdots$

- Player first picks a model $\mathbf{x}_t$ from a convex set $\mathcal{X} \subseteq \mathbb{R}^d$.
- Environments pick an online convex function $f_t : \mathcal{X} \to \mathbb{R}$.
- Player suffers loss $f_t(\mathbf{x}_t)$, observes some information about f_t and updates the model.

on the difficulty of environments:

- stochastic setting
- adversarial setting { oblivious; adaptive (non-oblivious) }



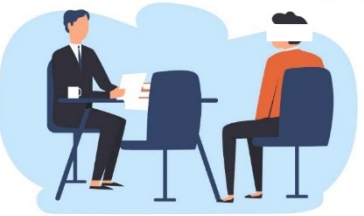
less restricted but harder

oblivious adversary



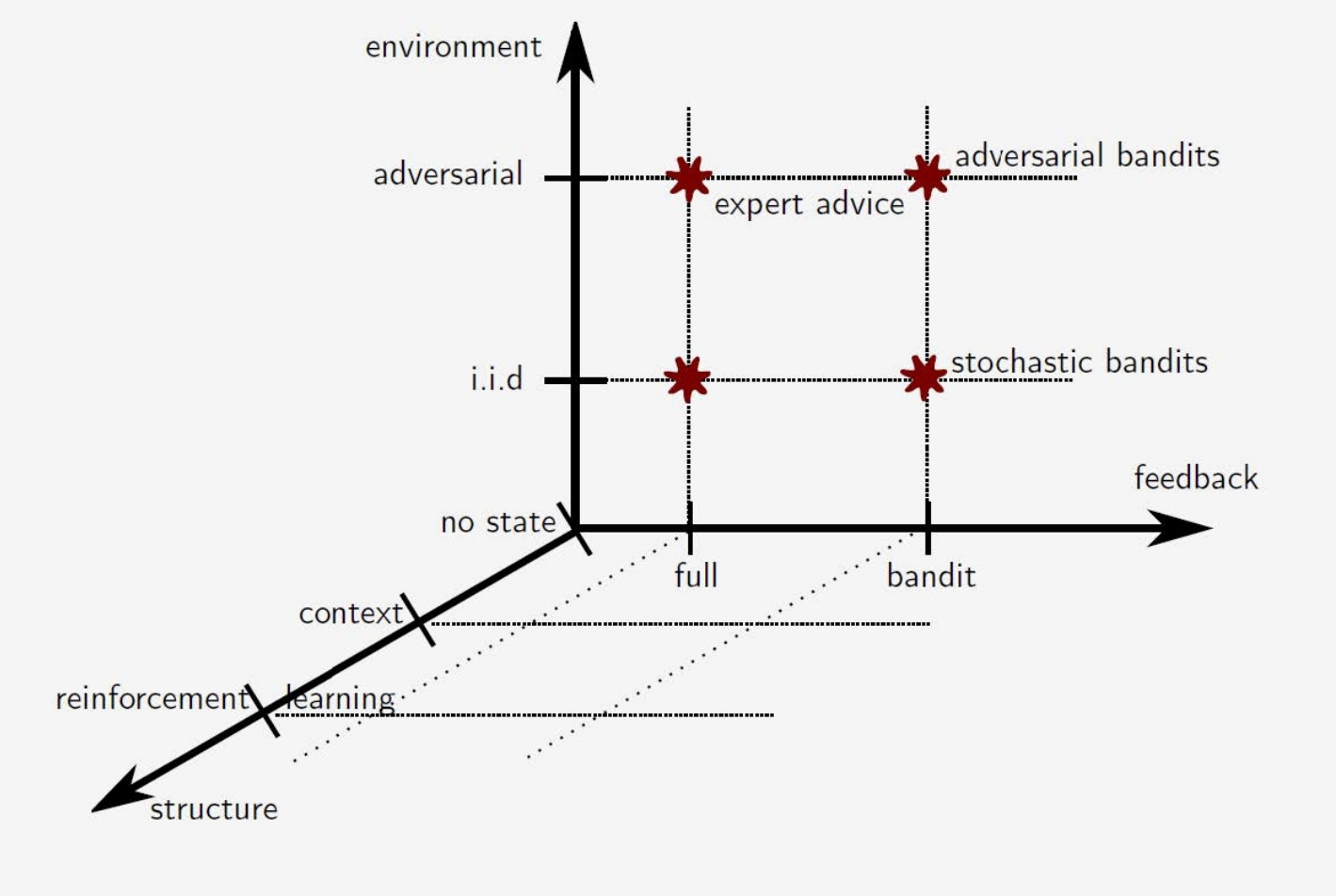
examination

adaptive adversary



interview

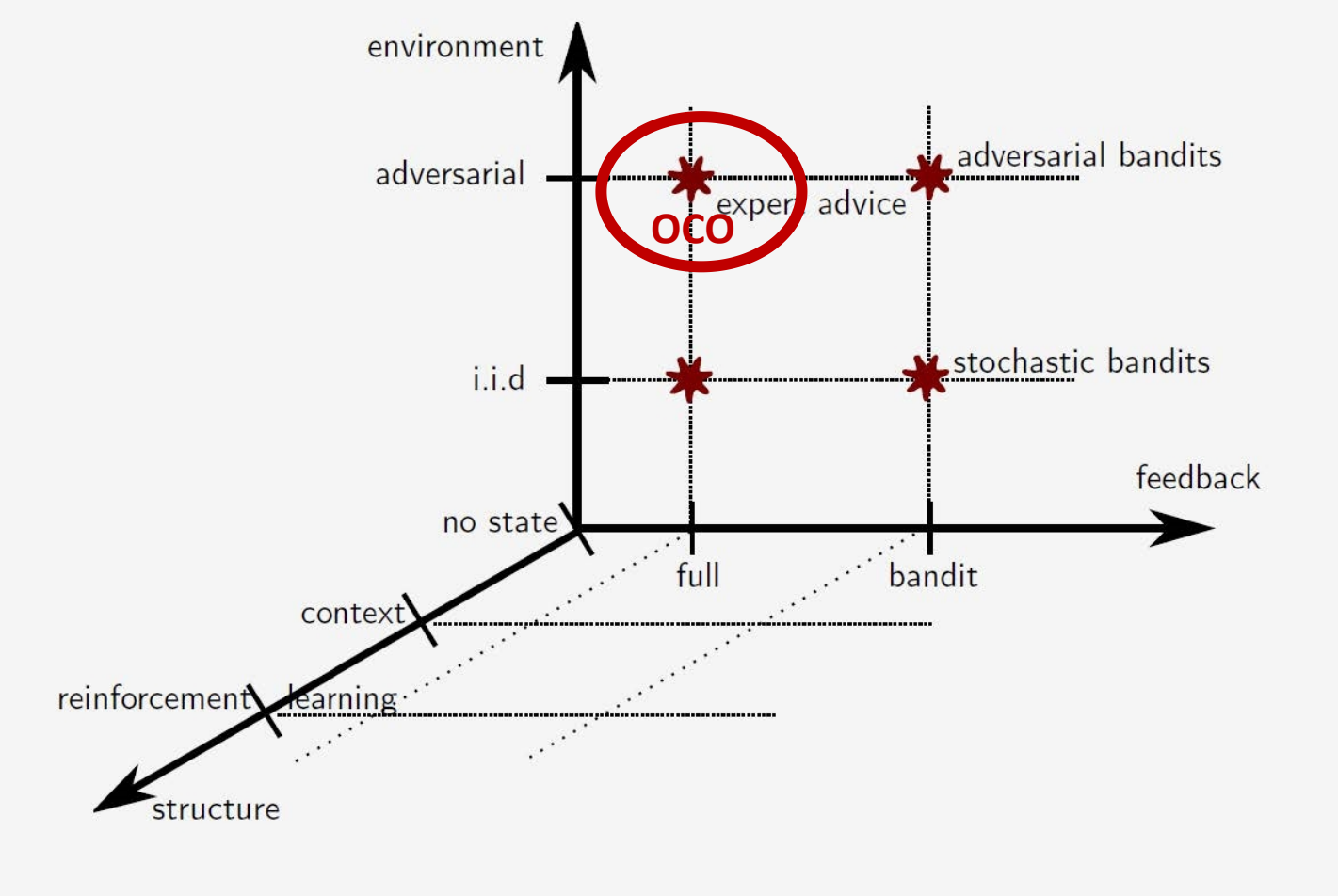
Online Learning: Problem Space



- Full-information setting:
  - Online Convex Optimization
  - Prediction with Expert Advice
  - ...
- Partial-information setting:
  - Multi-Armed Bandits
  - Linear Bandits
  - Parametric Bandits
  - Bandit Convex Optimization
  - ...

Yevgeny Seldin. The Space of Online Learning Problems, ECML-PKDD, Porto, Portugal, 2015.

Online Learning: Problem Space



- Full-information setting:
  - Online Convex Optimization
  - Prediction with Expert Advice
  - ...
- Partial-information setting:
  - Multi-Armed Bandits
  - Linear Bandits
  - Parametric Bandits
  - Bandit Convex Optimization
  - ...

Yevgeny Seldin. The Space of Online Learning Problems, ECML-PKDD, Porto, Portugal, 2015.

OCO: Convex Functions

Definition 2 (Convex Function). A function $f : \mathcal{X} \mapsto \mathbb{R}$ is convex if for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, it holds $\forall \alpha \in [0, 1], f((1-\alpha)\mathbf{x} + \alpha\mathbf{y}) \leq (1-\alpha)f(\mathbf{x}) + \alpha f(\mathbf{y})$.

Equivalently, if f is differentiable, we have that $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$,

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}).$$

The feasible set $\mathcal{X}$ is closed and convex in Euclidean space, and $f_1, \ldots, f_T$ are convex functions.

Regret Minimization in OCO

- We focus on the G-Lipschitz functions

> **Assumption 1** (Bounded Gradient). The norm of the subgradients is upper bounded by G, i.e., $\|\nabla f_t(\mathbf{x})\| \leq G$ for all $\mathbf{x} \in \mathcal{X}$ and $t \in [T]$.

- The following is ***domain boundedness***

> **Assumption 2** (Bounded Domain). The diameter of the feasible domain $\mathcal{X}$ is upper bounded by D, i.e., $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}, \|\mathbf{x} - \mathbf{y}\| \leq D$.

Essentially working on Lipschitz (online) optimization with a bounded feasible domain.

On the domain boundedness

Theorem 5.1. *Let $\mathcal{V} \subset \mathbb{R}^d$ be any non-empty bounded closed convex subset. Let $D = \sup_{\boldsymbol{v},\boldsymbol{w}\in\mathcal{V}} \|\boldsymbol{v} - \boldsymbol{w}\|_2$ be the diameter of $\mathcal{V}$. Let $\mathcal{A}$ be any (possibly randomized) algorithm for OLO on $\mathcal{V}$. Let T be any non-negative integer. Then, there exists a sequence of vectors $\boldsymbol{g}_1, \dots, \boldsymbol{g}_T$ with $\|\boldsymbol{g}_t\|_2 \le L$ and $\boldsymbol{u} \in \mathcal{V}$ such that the regret of algorithm $\mathcal{A}$ satisfies*

$$\text{Regret}_T(\boldsymbol{u}) = \sum_{t=1}^{T} \langle \boldsymbol{g}_t, \boldsymbol{x}_t \rangle - \sum_{t=1}^{T} \langle \boldsymbol{g}_t, \boldsymbol{u} \rangle \ge \frac{\sqrt{2}LD\sqrt{T}}{4} .$$

Theorem 5.6. *For $T \in \mathbb{N}$ suppose that there is an online convex optimization algorithm that guarantees regret at most ϵ_T against the null competitor on any sequence of T linear and L-Lipschitz losses $\ell_t : \mathbb{R} \to \mathbb{R}$ for $t = 1, \dots, T$. Let $U > 0$ be such that $1 \le W(\frac{\sqrt{T}UL}{5\epsilon_T}) \le \frac{\sqrt{T}}{2}$, then there exists a sequence of $\boldsymbol{g}_t$ with $\|\boldsymbol{g}_t\|_2 \le L$ and a competitor $\boldsymbol{u} \in \mathbb{R}^d$ with $\|\boldsymbol{u}\|_2 = U$, such that*

$$\sum_{t=1}^{T} \langle \boldsymbol{g}_t, \boldsymbol{x}_t - \boldsymbol{u} \rangle \ge R_T(U) := UL\sqrt{T}\left(\sqrt{2W\left(\frac{\sqrt{T}UL}{5\epsilon_T}\right) - 1}\right) - 2UL + \epsilon_T,$$

where $W : \mathbb{R}_{\ge 0} \to \mathbb{R}_{\ge 0}$ is the Lambert function.

Francesco Orabona. A Modern Introduction to Online Learning. 2025 (Version 7) Chapter 5.

Recall Lipschitz (offline) optimization

For G-Lipschitz optiminization with $\min_{\mathbf{x}} f(\mathbf{x})$, we use GD to optimize it and obtain $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$ optimal rate.

Optimal Result with Known T

Theorem 5. *Under the same assumptions with Theorem 1, assume the feasible domain $\mathcal{X}$ is bounded and convex with a diameter $D > 0$, that is, $\|\mathbf{x} - \mathbf{y}\|_2 \leq D$ holds for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$. Let $\{\mathbf{x}_t\}_{t=1}^T$ be the sequence generated by GD with step size*

$$\eta_t = \frac{D}{G\sqrt{T}}.$$

Then

$$f(\bar{\mathbf{x}}_T) - f^\star \leq \frac{DG}{\sqrt{T}} = \mathcal{O}\left(\frac{1}{\sqrt{T}}\right),$$

where $\bar{\mathbf{x}}_T \triangleq \arg\min_{\{\mathbf{x}_t\}_{t=1}^T} f(\mathbf{x}_t)$ or $\bar{\mathbf{x}}_T \triangleq \frac{1}{T}\sum_{t=1}^T \mathbf{x}_t$.

Advanced Optimization (Fall 2026) Lecture 3. Gradient Descent Method 29

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t)\right]$$

Can we extend the GD idea to online optimization?

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)\right]$$

OCO: OGD Algorithm

Online Gradient Descent (OGD)

At each round $t = 1, 2, \cdots$

1. the player first picks a model $\mathbf{x}_t \in \mathcal{X}$;
2. and simultaneously environments pick a ***convex*** online function $f_t : \mathcal{X} \to \mathbb{R}$;
3. the player suffers loss $f_t(\mathbf{x}_t)$, observes the information of f_t and update the model according to $\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)\right]$.

This belongs to the full-information setting, so player can access the gradient $\nabla f_t(\mathbf{x}_t)$.

Actually, only gradient is required, so it's also called ***gradient-feedback*** OCO model.

Regret Analysis of OGD

Theorem 3 (Regret bound for OGD)**.** *Under Assumption 1 (G-Lipschitz) and Assumption 2 (D-bounded domain), online gradient descent (OGD) with step sizes* $\eta_t = \frac{D}{G\sqrt{t}}$ *for* $t \in [T]$ *guarantees:*

$$\textsc{Reg}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\mathbf{x}) \leq \frac{3}{2} GD\sqrt{T} = \mathcal{O}(\sqrt{T}).$$

The First Gradient Descent Lemma

Lemma 1. *Suppose that f_t is proper, closed and convex; the feasible domain $\mathcal{X}$ is nonempty, closed and convex. Let $\{\mathbf{x}_t\}_{t=1}^T$ be the sequence generated by the gradient descent method. Then for any $\mathbf{u} \in \mathcal{X}$ and $t \geq 0$,*

$$\|\mathbf{x}_{t+1} - \mathbf{u}\|^2 \leq \|\mathbf{x}_t - \mathbf{u}\|^2 - 2\eta_t(f_t(\mathbf{x}_t) - f_t(\mathbf{u})) + \eta_t^2 \|\nabla f_t(\mathbf{x}_t)\|^2.$$

Proof:

$$\begin{aligned}
\|\mathbf{x}_{t+1} - \mathbf{u}\|^2 &= \|\Pi_{\mathcal{X}}[\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)] - \mathbf{u}\|^2 && \text{(GD)} \\
&\leq \|\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t) - \mathbf{u}\|^2 && \text{(Pythagoras Theorem)} \\
&= \|\mathbf{x}_t - \mathbf{u}\|^2 - 2\eta_t \langle \nabla f_t(\mathbf{x}_t), \mathbf{x}_t - \mathbf{u} \rangle + \eta_t^2 \|\nabla f_t(\mathbf{x}_t)\|^2 \\
&\leq \|\mathbf{x}_t - \mathbf{u}\|^2 - 2\eta_t (f_t(\mathbf{x}_t) - f_t(\mathbf{u})) + \eta_t^2 \|\nabla f_t(\mathbf{x}_t)\|^2 \\
&\qquad \text{(convexity: } f_t(\mathbf{x}_t) - f_t(\mathbf{u}) = f_t(\mathbf{x}_t) - f_t(\mathbf{u}) \leq \langle \nabla f_t(\mathbf{x}_t), \mathbf{x}_t - \mathbf{u} \rangle) \qquad \square
\end{aligned}$$

Proof for OGD Regret Bound

Proof: We use the first gradient descent lemma to analyze online gradient descent.

Lemma 1. *Suppose that f_t is proper, closed and convex; the feasible domain $\mathcal{X}$ is nonempty, closed and convex. Let $\{\mathbf{x}_t\}_{t=1}^T$ be the sequence generated by the gradient descent method. Then for any $\mathbf{u} \in \mathcal{X}$ and $t \geq 0$,*

$$\|\mathbf{x}_{t+1} - \mathbf{u}\|^2 \leq \|\mathbf{x}_t - \mathbf{u}\|^2 - 2\eta_t(f_t(\mathbf{x}_t) - f_t(\mathbf{u})) + \eta_t^2 \|\nabla f_t(\mathbf{x}_t)\|^2.$$

By Lemma 1 and the gradient boundedness, we have

$$2(f_t(\mathbf{x}_t) - f_t(\mathbf{u})) \leq \frac{\|\mathbf{x}_t - \mathbf{u}\|^2 - \|\mathbf{x}_{t+1} - \mathbf{u}\|^2}{\eta_t} + \eta_t G^2$$

Proof for OGD Regret Bound

Proof: By setting $\eta_t = \frac{D}{G\sqrt{t}}$ $\left(\text{with } \frac{1}{\eta_0} := 0\right)$, summing over T:

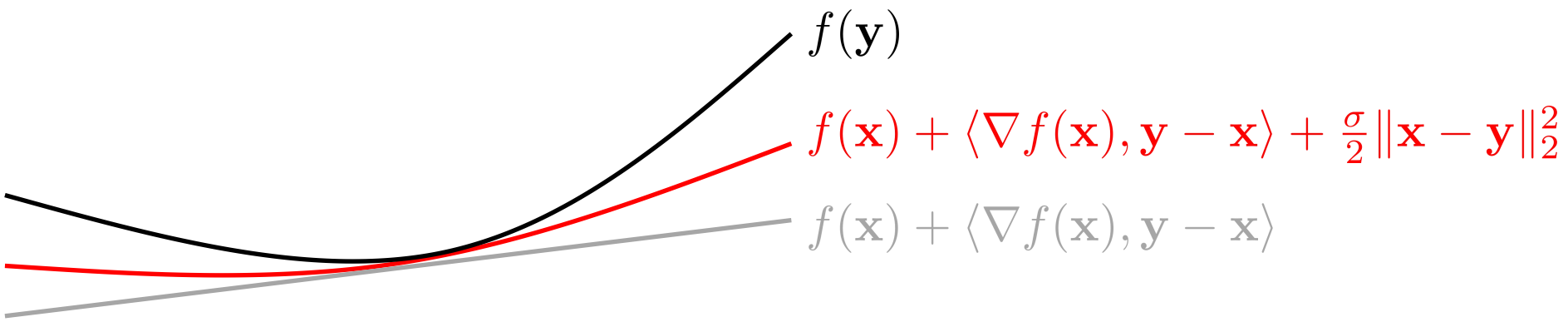
$$
\begin{aligned}
2\left(\sum_{t=1}^{T} f_t(\mathbf{x}_t) - f_t(\mathbf{u})\right) &\le \sum_{t=1}^{T} \frac{\|\mathbf{x}_t - \mathbf{u}\|^2 - \|\mathbf{x}_{t+1} - \mathbf{u}\|^2}{\eta_t} + G^2 \sum_{t=1}^{T} \eta_t && \text{(GD lemma)} \\
&\le \sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{u}\|^2 \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right) + G^2 \sum_{t=1}^{T} \eta_t && (\|\mathbf{x}_{T+1} - \mathbf{u}\|^2 \ge 0) \\
&\le D^2 \sum_{t=1}^{T} \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right) + G^2 \sum_{t=1}^{T} \eta_t \\
&\le D^2 \frac{1}{\eta_T} + G^2 \sum_{t=1}^{T} \eta_t && (\eta_t = \tfrac{D}{G\sqrt{t}} \text{ and } \textstyle\sum_{t=1}^{T} \tfrac{1}{\sqrt{t}} \le 2\sqrt{T}) \\
&\le 3DG\sqrt{T}. && \square
\end{aligned}
$$

OCO: Strongly Convex Functions

Definition 3 (Strong Convexity). A function f is σ-strongly convex if, for any $\mathbf{x}, \mathbf{y} \in \operatorname{dom} f$,

$$f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) + \frac{\sigma}{2} \|\mathbf{y} - \mathbf{x}\|^2,$$

or equivalently, $\nabla^2 f(\mathbf{x}) \succeq \alpha I$.



Recall Lipschitz (offline) Optimization

For G-Lipschitz optiminization with $\min_{\mathbf{x}} f(\mathbf{x})$, we use GD to optimize it and obtain $\mathcal{O}\left(\frac{1}{T}\right)$ optimal rate.

Strongly Convex and Lipschitz

Theorem 7. *Under the same assumptions with Theorem 1, except that f is σ-strongly-convex. Let $\{\mathbf{x}_t\}_{t=1}^T$ be the sequence generated by GD with step size*

$$\eta_t = \frac{2}{\sigma(t+1)}.$$

Then (i)

$$f(\bar{\mathbf{x}}_T) - f^\star \leq \frac{2G^2}{\sigma(T+1)} = \mathcal{O}\left(\frac{1}{T}\right),$$

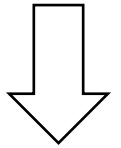
where $\bar{\mathbf{x}}_T \triangleq \arg\min_{\{\mathbf{x}_t\}_{t=1}^T} f(\mathbf{x}_t)$ or $\bar{\mathbf{x}}_T \triangleq \sum_{t=1}^T \frac{2t}{T(T+1)}\mathbf{x}_t$.

And (ii)

$$\|\bar{\mathbf{x}}_T - \mathbf{x}^\star\| \leq \frac{2G}{\sigma\sqrt{T+1}}.$$

Advanced Optimization (Fall 2026) Lecture 3. Gradient Descent Method 41

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t)\right]$$



$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)\right]$$

Again, we extend it to online optimization.

OGD for Strongly Convex Functions

Online Gradient Descent (OGD)

At each round $t = 1, 2, \cdots$

(1) the player first picks a model $\mathbf{x}_t \in \mathcal{X}$;

(2) and simultaneously environments pick a ***strongly convex function*** $f_t : \mathcal{X} \to \mathbb{R}$;

(3) the player suffers loss $f_t(\mathbf{x}_t)$, observes the information of f_t and update the model according to $\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \nabla f_t(\mathbf{x}_t)\right]$.

OGD for Strongly Convex Functions

Theorem 4 (Regret bound for strongly-convex functions). *Under Assumption 1 (G-Lipschitz), for σ-strongly convex loss functions, online gradient descent with step sizes $\eta_t = \frac{1}{\sigma t}$ achieves the following guarantee*

$$\textsc{Reg}_T \leq \frac{G^2}{2\sigma}(1 + \log T) = \mathcal{O}(\log T).$$

- Strongly convex case compared with convex case: $\mathcal{O}(\log T)$ vs. $\mathcal{O}(\sqrt{T})$
- A caveat is that we now don't need Assumption 2 (bounded domain).

OGD for Strongly Convex Functions

Proof: we start by extending ***the first GD lemma*** to strongly convex case.

Strongly convex case:

$$\begin{aligned}
\|\mathbf{x}_{t+1}-\mathbf{u}\|^2 &\le \|\mathbf{x}_t-\mathbf{u}\|^2 - 2\eta_t\langle\nabla f_t(\mathbf{x}_t),\mathbf{x}_t-\mathbf{u}\rangle + \eta_t^2\|\nabla f_t(\mathbf{x}_t)\|^2 \\
&\le \|\mathbf{x}_t-\mathbf{u}\|^2 - 2\eta_t\left(f_t(\mathbf{x}_t)-f_t(\mathbf{u})+\frac{\sigma}{2}\|\mathbf{x}_t-\mathbf{u}\|^2\right) + \eta_t^2\|\nabla f_t(\mathbf{x}_t)\|^2 \\
&\qquad \text{(strong convexity: } f_t(\mathbf{x}_t)-f_t(\mathbf{u})+\tfrac{\sigma}{2}\|\mathbf{x}_t-\mathbf{u}\|^2 \le \langle\nabla f_t(\mathbf{x}_t),\mathbf{x}_t-\mathbf{u}\rangle\text{)} \\
&\le (1-\sigma\eta_t)\|\mathbf{x}_t-\mathbf{u}\|^2 - 2\eta_t\left(f_t(\mathbf{x}_t)-f_t(\mathbf{u})\right) + \eta_t^2\|\nabla f_t(\mathbf{x}_t)\|^2
\end{aligned}$$

$$\Longrightarrow f_t(\mathbf{x}_t)-f_t(\mathbf{u}) \le \frac{\eta_t^{-1}-\sigma}{2}\|\mathbf{x}_t-\mathbf{u}\|^2 - \frac{\eta_t^{-1}}{2}\|\mathbf{x}_{t+1}-\mathbf{u}\|^2 + \frac{\eta_t G^2}{2} \quad \text{(rearranging)}$$

OGD for Strongly Convex Functions

Proof: $f_t(\mathbf{x}_t) - f_t(\mathbf{u}) \leq \frac{\eta_t^{-1} - \sigma}{2}\|\mathbf{x}_t - \mathbf{u}\|^2 - \frac{\eta_t^{-1}}{2}\|\mathbf{x}_{t+1} - \mathbf{u}\|^2 + \frac{\eta_t G^2}{2}$

Summing from $t = 1$ to T, setting $\eta_t = \frac{1}{\sigma t}$ (define $\frac{1}{\eta_0} := 0$):

$$
\begin{aligned}
2\sum_{t=1}^{T}\left(f_t(\mathbf{x}_t) - f_t(\mathbf{u})\right) \leq & \sum_{t=1}^{T}\|\mathbf{x}_t - \mathbf{u}\|^2\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} - \sigma\right) + G^2\sum_{t=1}^{T}\eta_t \quad \left(\tfrac{1}{\eta_0} := 0\right) \\
= & 0 + G^2\sum_{t=1}^{T}\frac{1}{\sigma t} \quad \left(\tfrac{1}{\eta_0} \triangleq 0, \|\mathbf{x}_{T+1} - \mathbf{u}\|^2 \geq 0, \tfrac{1}{\eta_t} - \tfrac{1}{\eta_{t-1}} - \sigma = 0\right) \\
\leq & \frac{G^2}{\sigma}(1 + \log T).
\end{aligned}
$$

□

Comparisons

Convex

Property: $f_t(\mathbf{x}) \geq f_t(\mathbf{y}) + \nabla f_t(\mathbf{y})^\top (\mathbf{x} - \mathbf{y})$

OGD: $\mathbf{x}_{t+1} = \Pi_{\mathcal{X}} \left[\mathbf{x}_t - \frac{1}{\sqrt{t}} \nabla f_t(\mathbf{x}_t) \right]$

$$\text{REG}_T \leq \frac{3}{2} GD\sqrt{T}$$

Strongly Convex

Property: $f_t(\mathbf{y}) \geq f_t(\mathbf{x}) + \nabla f_t(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) + \frac{\sigma}{2} \|\mathbf{y} - \mathbf{x}\|^2$

OGD: $\mathbf{x}_{t+1} = \Pi_{\mathcal{X}} \left[\mathbf{x}_t - \frac{1}{\sigma t} \nabla f_t(\mathbf{x}_t) \right]$

$$\text{REG}_T \leq \frac{G^2}{2\sigma} (1 + \log T)$$

Lower Bounds

- A natural question: whether previous regret can be improved?

- Lower bound argument:

  minimax bound: smallest possible worst-case regret of any algorithm

$$\min_{\mathcal{A}} \max_{\ell_1,\ldots,\ell_T} \textsc{Reg}_T$$

Theorem 5 (Lower Bound for OCO). *Any algorithm for online convex optimization incurs $\Omega(DG\sqrt{T})$ regret in the worst case. This is true even if the cost functions are generated from a fixed stationary distribution.*

Lower Bounds

Theorem 5 (Lower Bound for OCO). *Any algorithm for online convex optimization incurs $\Omega(DG\sqrt{T})$ regret in the worst case. This is true even if the cost functions are generated from a fixed stationary distribution.*

Proof Sketch.

Construct a ***hard*** environment:

- Binary classification, loss functions in each iteration are chosen at random
- The hardness comes from the compression of a sequence of T-round random bits

Comparison

	Algo.	Step size	Upper Bound	Lower Bound
Convex	OGD	$\frac{D}{G\sqrt{t}}$	$\mathcal{O}(\sqrt{T})$	$\Omega(\sqrt{T})$
σ-Strongly Convex	OGD	$\frac{1}{\sigma t}$	$\mathcal{O}\left(\frac{\log T}{\sigma}\right)$	$\Omega\left(\frac{\log T}{\sigma}\right)$

Lower bound for strongly convex functions is more non-trivial.



Optimal Strategies and Minimax Lower Bounds for Online Convex Games

Jacob Abernethy[*]
UC Berkeley
jake@cs.berkeley.edu

Peter L. Bartlett[†]
UC Berkeley
bartlett@cs.berkeley.edu

Alexander Rakhlin[*]
UC Berkeley
rakhlin@cs.berkeley.edu

Ambuj Tewari
TTI Chicago
ambuj@cs.berkeley.edu

Abstract

A number of learning problems can be cast as an Online Convex Game: on each round, a learner makes a prediction x from a convex set, the environment plays a loss function f, and the learner's long-term goal is to minimize regret. Algorithms have been proposed by Zinkevich, when f is assumed to be convex, and Hazan et al., when f is assumed to be strongly convex, that have provably low regret. We consider these two settings and analyze such games from a minimax perspective, proving minimax strategies and lower bounds in each case. These results prove that the existing algorithms are essentially optimal.

1 Introduction

The decision maker's greatest fear is *regret*: knowing, with the benefit of hindsight, that a better alternative existed. Yet, given only hindsight and not the gift of foresight, imperfect decisions can not be avoided. It is thus the decision maker's ultimate goal to suffer as little regret as possible.

In the present paper, we consider the notion of "regret minimization" for a particular class of decision problems. Assume we are given a set X and some set of functions $\mathcal{F}$ on X. On each round $t = 1, \ldots, T$, we must choose some $\mathbf{x}_t$ from a set X. After we have made this choice, the *environment* chooses a function $f_t \in \mathcal{F}$. We incur a cost (loss) $f_t(\mathbf{x}_t)$, and the game proceeds to the next round. Of course, had we the fortune of perfect foresight and had access to the sum $f_1 + \ldots + f_T$, we would know the optimal choice $\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{t=1}^{T} f_t(\mathbf{x})$. Instead, at time t, we will have only seen $f_1, \ldots, f_{t-1}$, and we must make the decision $\mathbf{x}_t$ with only historical knowledge. Thus, a natural long-term goal is to minimize the *regret*, which here we define as

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \inf_{\mathbf{x}\in X} \sum_{t=1}^{T} f_t(\mathbf{x}).$$

A special case of this setting is when the decision space X is a convex set and $\mathcal{F}$ is some set of convex functions on X. In the literature, this framework has been referred to as Online Convex Optimization (OCO), since our goal is to minimize a global function, i.e. $f_1 + f_2 + \cdots + f_T$, while this objective is revealed to us but one function at a time. Online Convex Optimization has attracted much interest in recent years [4, 9, 6, 1], as it provides a general analysis for a number of standard online learning problems including, among others, online classification and regression, prediction with expert advice, the portfolio selection problem, and online density estimation.

While instances of OCO have been studied over the past two decades, the general problem was first analyzed by Zinkevich [9], who showed that a very simple and natural algorithm, online gradient descent, elicits a bound on the regret that is on the order of $\sqrt{T}$. Online gradient descent can be described simply by the update $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \nabla f_t(\mathbf{x}_t)$, where η is some parameter of the algorithm. This regret bound only required that f_t be smooth, convex, and with bounded derivative.

A regret bound of order $O(\sqrt{T})$ is not surprising: a number of online learning problems give rise to similar bounds. More recently, however, Hazan et al. [4] showed that when $\mathcal{F}$ consists of *curved* functions, i.e. f_t is strongly convex, then we get a bound of the form $O(\log T)$. It is quite surprising that curvature gives such a great advantage to the player. Curved loss functions, such as square loss or logarithmic loss, are very natural in a number of settings.

Finding algorithms that can guarantee low regret is, however, only half of the story; indeed, it is natural to ask "can we obtain even lower regret?" or "do better algorithms exist?" The goal of the present paper is to address these questions, in some detail, for several classes of such online optimization problems. We answer both in the negative: the algorithms of Zinkevich and Hazan et al. are tight even up to their multiplicative constants.

This is achieved by a game-theoretic analysis: if we pose the above online optimization problem as a game between a Player who chooses $\mathbf{x}_t$ and an Adversary who chooses f_t, we may consider the regret achieved when each player is playing optimally. This is typically referred to as the *value* V_T of the game. In general, computing the value of zero-sum games is difficult, as we may have to consider exponentially many, or even uncountably many, strategies of the Player and the Adversary. Ultimately we will show that this value, as well as the optimal strategies of both the player and the adversary,

[*]Division of Computer Science
[†]Division of Computer Science, Department of Statistics

Jacob Abernethy, Peter L. Bartlett, Alexander Rakhlin, and Ambuj Tewari. Optimal strategies and minimax lower bounds for online convex games. In COLT, 2008.

Part 3. Online-to-Batch Conversion

- Convex Functions

- Strongly Convex Functions

What and Why?

- Online-to-Batch Conversion: solve ***offline optimization*** (namely, batch learning) using ***online optimization/learning*** technique.

- Why?
  - Offline optimization does not mean a fixed objective! In particular, ***stochastic*** (offline) optimization actually involves mini-batch sampling, which involves interactions and naturally requires online opt.
  - While online optimization adds a bit more complexity, it can render certain sophisticated tricks ineffective, thereby exposing the essential challenges and avoiding reliance on "hacked" solutions.

Offline Optimization

- Consider ***offline optimization*** $\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$ with stochastic opt method

> **Computational oracle**: only access ***noisy*** gradient oracle, namely, $\mathbf{g}(\mathbf{x})$, such that
>
> $$\mathbb{E}[\mathbf{g}(\mathbf{x})] = \nabla F(\mathbf{x}), \text{ and } \mathbb{E}[\|\mathbf{g}(\mathbf{x})\|^2] \leq G^2$$
>
> for some $G > 0$.

Example (large-scale opt.). Given dataset $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)\}$, ERM optimizes

$$\min_{h\in\mathcal{H}} \sum_{i=1}^{m} \ell\big(h(\mathbf{x}_i), y_i\big) \quad \Longrightarrow$$

full gradient computation requires a pass of ***all data***

stochastic method only uses a ***mini batch*** at each round

SGD: A Natural Solution

- Consider ***offline optimization*** $\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$ with stochastic opt method

Computational oracle: only access ***noisy*** gradient oracle, namely, $\mathbf{g}(\mathbf{x})$, such that

$$\mathbb{E}[\mathbf{g}(\mathbf{x})] = \nabla F(\mathbf{x}), \text{ and } \mathbb{E}[\|\mathbf{g}(\mathbf{x})\|^2] \leq G^2$$

for some $G > 0$.

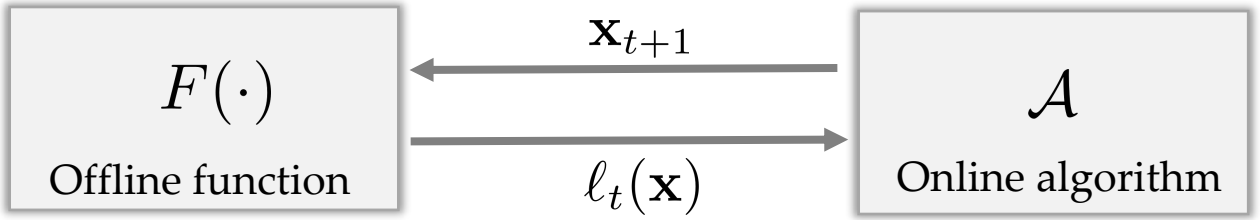
- Stochastic Gradient Descent (SGD)

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}} \left[\mathbf{x}_t - \eta_t \mathbf{g}(\mathbf{x}_t)\right]$$

using stochastic gradient instead of full gradient

Online-to-Batch Conversion

- Reducing ***offline optimization*** as an ***online optimization***.



Algorithm 2 Online-to-Batch Conversion for Convex Functions

Input: noisy gradient oracle $\mathbf{g}(\cdot)$, online learning algorithm $\mathcal{A}$

1: **for** $t = 1, \ldots, T$ **do**
2: Obtain noisy gradient $\mathbf{g}(\mathbf{x}_t)$
3: Pass loss function $\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$ to online learning algorithm $\mathcal{A}$
4: Receive next point $\mathbf{x}_{t+1}$ from $\mathcal{A}$
5: **end for**
6: **return** $\bar{\mathbf{x}}_T = \frac{1}{T} \sum_{t=1}^{T} \mathbf{x}_t$

Online-to-Batch Conversion

Theorem 6 (O2B Conversion for Convex Functions). *Assume $F(\mathbf{x})$ is convex. Let $\mathbf{x}^\star \in \arg\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$. Define the online loss function $\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x}\rangle$, then Algorithm 2 achieves the following guarantee:*

$$\mathbb{E}\left[F\left(\overline{\mathbf{x}}_T\right)\right] - F\left(\mathbf{x}^\star\right) \le \frac{\mathbb{E}\left[\mathrm{REG}_T^{\mathcal{A}}\left(\mathbf{x}^\star\right)\right]}{T}$$

where $\mathrm{REG}_T^{\mathcal{A}}(\mathbf{u}) = \sum_{t=1}^T \left(\ell_t(\mathbf{x}_t) - \ell_t(\mathbf{u})\right)$ be the regret of OL algorithm $\mathcal{A}$ over loss $\{\ell_t(\mathbf{x})\}_t^T$.

Proof: Using Jensen's inequality, we have

$$\mathbb{E}[F(\bar{\mathbf{x}}_T)] - F(\mathbf{u}) \le \frac{1}{T}\sum_{t=1}^T \mathbb{E}[F(\mathbf{x}_t)] - F(\mathbf{u}) \le \frac{1}{T}\sum_{t=1}^T \mathbb{E}\left[\langle \nabla F(\mathbf{x}_t), \mathbf{x}_t - \mathbf{u}\rangle\right] \quad \text{(Convexity)}$$

(Jensen)

$$= \frac{1}{T}\sum_{t=1}^T \mathbb{E}\left[\langle \mathbf{g}(\mathbf{x}_t), \mathbf{x}_t - \mathbf{u}\rangle\right] = \frac{\mathbb{E}\left[\mathrm{REG}_T^{\mathcal{A}}(\mathbf{u})\right]}{T}. \quad \square$$

Stochastic Optimization

- Optimization Goal

$$\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$$

Computational oracle: only access ***noisy*** gradient oracle, namely, $\mathbf{g}(\mathbf{x})$, such that

$$\mathbb{E}[\mathbf{g}(\mathbf{x})] = \nabla F(\mathbf{x}), \text{ and } \mathbb{E}[\|\mathbf{g}(\mathbf{x})\|^2] \leq G^2.$$

By deploying OGD as $\mathcal{A}$ in Algorithm 2, we obtain a practical algorithm for convex functions, which updates as follows:

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \mathbf{g}\left(\mathbf{x}_t\right)\right]$$

Algorithm 2 Online-to-Batch Conversion for Convex Functions

Input: noisy gradient oracle $\mathbf{g}(\cdot)$, online learning algorithm $\mathcal{A}$

1: **for** $t = 1, \ldots, T$ **do**
2: Obtain noisy gradient $\mathbf{g}(\mathbf{x}_t)$
3: Pass loss function $\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$ to online learning algorithm $\mathcal{A}$
4: Receive next point $\mathbf{x}_{t+1}$ from $\mathcal{A}$
5: **end for**
6: **return** $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t$

Plugging the $\mathcal{O}(\sqrt{T})$ regret bound of OGD into Theorem 6, we can achieve $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$ rate.

Understand SGD from Online Learning

Algorithm 3 Stochastic Gradient Descent

Input: noisy gradient oracle $\mathbf{g}(\cdot)$, step sizes $\{\eta_t\}$

1: **for** $t = 1, \ldots, T$ **do**
2: Obtain noisy gradient $\mathbf{g}(\mathbf{x}_t)$
3: Update the model $\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \mathbf{g}(\mathbf{x}_t)\right]$
4: **end for**
5: **return** $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t$

OGD: $\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}[\mathbf{x}_t - \eta_t \nabla \ell_t(\mathbf{x}_t)]$
$= \Pi_{\mathcal{X}}[\mathbf{x}_t - \eta_t \mathbf{g}(\mathbf{x}_t)]$

SGD is equivalent to deploying OGD in Algorithm 2 over functions $\{\ell_t(\mathbf{x})\}_{t=1}^{T}$.

Theorem 7 (Convergence of SGD). *Assume $F(\mathbf{x})$ is convex, and let $\mathbf{x}^\star \in \arg\min_{\mathbf{x}} F(\mathbf{x})$. Algorithm 3 achieves the following guarantee:*

$$\mathbb{E}[F(\bar{\mathbf{x}}_T)] - F(\mathbf{x}^\star) \leq \mathcal{O}\left(\frac{1}{\sqrt{T}}\right).$$

Stochastic Optimization

- Optimization Goal

$$\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$$

Computational oracle: only access ***noisy*** gradient oracle, namely, $\mathbf{g}(\mathbf{x})$, such that

$$\mathbb{E}[\mathbf{g}(\mathbf{x})] = \nabla F(\mathbf{x}), \text{ and } \mathbb{E}[\|\mathbf{g}(\mathbf{x})\|^2] \leq G^2.$$

By deploying OGD as $\mathcal{A}$ in Algorithm 2, we obtain a practical algorithm for convex functions, which updates as follows:

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}\left[\mathbf{x}_t - \eta_t \mathbf{g}\left(\mathbf{x}_t\right)\right]$$

Algorithm 2 Online-to-Batch Conversion for Convex Functions

Input: noisy gradient oracle $\mathbf{g}(\cdot)$, online learning algorithm $\mathcal{A}$

1: **for** $t = 1, \ldots, T$ **do**
2: Obtain noisy gradient $\mathbf{g}(\mathbf{x}_t)$
3: Pass loss function $\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$ to online learning algorithm $\mathcal{A}$
4: Receive next point $\mathbf{x}_{t+1}$ from $\mathcal{A}$
5: **end for**
6: **return** $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t$

Plugging the $\mathcal{O}(\sqrt{T})$ regret bound of OGD into Theorem 6, we can achieve $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$ rate.

⇨ Note that function $\ell_t(\mathbf{x}) \triangleq \mathbf{g}(\mathbf{x}_t)^\top \mathbf{x}$ actually ***depends*** on the decision $\mathbf{x}_t$, which reveals that OGD regret can hold even against ***adaptive adversary***.

Stochastic Optimization as Interaction

- As mentioned, stochastic opt can be deemed as an ***interaction*** between learning algo with the entire dataset (due to sampling)

Algorithm 2 Online-to-Batch Conversion for Convex Functions

Input: noisy gradient oracle $\mathbf{g}(\cdot)$, online learning algorithm $\mathcal{A}$

1: **for** $t = 1, \ldots, T$ **do**
2: Obtain noisy gradient $\mathbf{g}(\mathbf{x}_t)$
3: Pass loss function $\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$ to online learning algorithm $\mathcal{A}$
4: Receive next point $\mathbf{x}_{t+1}$ from $\mathcal{A}$
5: **end for**
6: **return** $\bar{\mathbf{x}}_T = \frac{1}{T} \sum_{t=1}^{T} \mathbf{x}_t$

So, this method: essentially an online optimization method working on the ***interactive loss*** $\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$ based on this round sampling.

O2B Conversion for Strongly Convex

- For σ-strongly convex functions, we can revise the O2B conversion and select OGD as $\mathcal{A}$ to achieve an $\mathcal{O}\left(\frac{\log T}{T}\right)$ convergence rate.
- But the key challenge is how to retain strong convexity in the online optimization, note that the vanilla conversion sets $\ell_t = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$, which is a linear function.
- Surrogate loss technique: for σ-strongly convex function $F(\cdot)$

$$F(\mathbf{x}_t) - F(\mathbf{u}) \leq \langle \nabla F(\mathbf{x}_t), \mathbf{x}_t - \mathbf{u} \rangle - \frac{\sigma}{2} \|\mathbf{x}_t - \mathbf{u}\|_2^2$$
$$\overset{\mathbb{E}}{=} \ell_t(\mathbf{x}_t) - \ell_t(\mathbf{u})$$

$$\boxed{\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle + \frac{\sigma}{2} \|\mathbf{x}_t - \mathbf{x}\|_2^2}$$

O2B Conversion for Strongly Convex

- Strong convexity seems incompatible with strongly convex optimization, as the interactive loss $\ell_t = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$ is a linear function.

- To achieve O2B for strongly convex optimization, we need to retain strong convexity in the online optimization, rather than the vanilla conversion using linear function $\ell_t = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle$.

- Surrogate loss technique: for σ-strongly convex function $F(\cdot)$

$$\begin{aligned} F(\mathbf{x}_t) - F(\mathbf{u}) &\leq \langle \nabla F(\mathbf{x}_t), \mathbf{x}_t - \mathbf{u} \rangle - \frac{\sigma}{2} \|\mathbf{x}_t - \mathbf{u}\|_2^2 \\ &\overset{\mathbb{E}}{=} \ell_t(\mathbf{x}_t) - \ell_t(\mathbf{u}) \end{aligned}$$

$$\boxed{\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle + \frac{\sigma}{2} \|\mathbf{x}_t - \mathbf{x}\|_2^2}$$

O2B Conversion for Strongly Convex

- For σ-strongly convex functions, we can revise the O2B conversion and select OGD as $\mathcal{A}$ to achieve an $\mathcal{O}\left(\frac{\log T}{T}\right)$ convergence rate.

Algorithm 4 Online-to-Batch Conversion for Convex Functions

Require: noisy gradient oracle $\mathbf{g}(\cdot)$, online learning algorithm $\mathcal{A}$

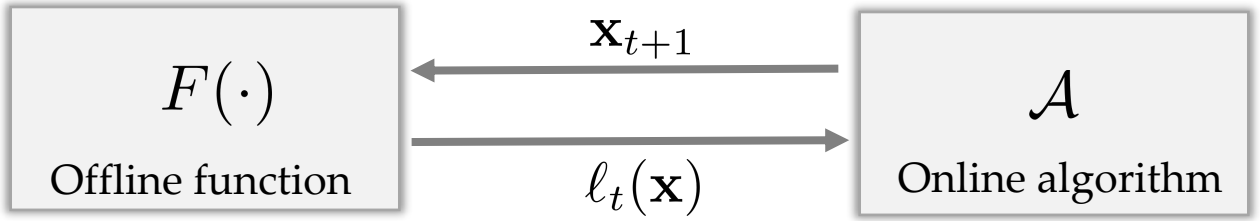
1: **for** $t = 1, \ldots, T$ **do**
2: Obtain noisy gradient $\mathbf{g}(\mathbf{x}_t)$
3: Pass loss function $\ell_t(\mathbf{x}) = \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle + \frac{\sigma}{2}\|\mathbf{x}_t - \mathbf{x}\|^2$ to online learning algorithm $\mathcal{A}$
4: Receive next point $\mathbf{x}_{t+1}$ from $\mathcal{A}$
5: **end for**
6: **return** $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t$

retain strong convexity

However, this rate exhibits a gap to the $\Omega(1/T)$ lower bound for stochastic optimization over strongly convex functions.

Weighted O2B Conversion

- To achieve the optimal rate for strongly convex optimization, we introduce the ***weighted*** Online-to-Batch Conversion.



Algorithm 5 Weighted O2B Conversion for Strongly Convex Functions

Input: noisy gradient oracle $\mathbf{g}(\cdot)$, online learning algorithm $\mathcal{A}$, and weights $\{\alpha_t\}_{t=1}^T$

1: **for** $t = 1, \ldots, T$ **do**
2: Obtain noisy gradient $\mathbf{g}(\mathbf{x}_t)$
3: Pass loss function $\ell_t(\mathbf{x}) = \langle \alpha_t \mathbf{g}(\mathbf{x}_t), \mathbf{x} \rangle + \frac{\alpha_t \sigma}{2} \|\mathbf{x}_t - \mathbf{x}\|^2$ to online learning algorithm $\mathcal{A}$
4: Receive next point $\mathbf{x}_{t+1}$ from $\mathcal{A}$
5: **end for**
6: **return** $\bar{\mathbf{x}}_T = \frac{1}{\sum_{t=1}^T \alpha_t} \sum_{t=1}^T \alpha_t \mathbf{x}_t$

Weighted O2B Conversion

Theorem 8 (Weighted O2B Conversion for Strongly Convex Functions). *Assume $F(\mathbf{x})$ is σ-strongly convex. Let $\mathbf{x}^\star = \arg\min_{\mathbf{x}} F(\mathbf{x})$. Define the surrogate loss function $\ell_t(\mathbf{x}) = \langle \alpha_t \mathbf{g}_t, \mathbf{x} \rangle + \frac{\alpha_t \sigma}{2} \|\mathbf{x}_t - \mathbf{x}\|^2$, and Algorithm 4 achieves the following guarantee:*

$$\mathbb{E}[F(\bar{\mathbf{x}}_T)] - F(\mathbf{x}^\star) \leq \frac{\mathbb{E}\left[\textsc{Reg}_T^{\mathcal{A}}(\mathbf{u})\right]}{\sum_{t=1}^T \alpha_t},$$

where $\textsc{Reg}_T^{\mathcal{A}}(\mathbf{u}) = \sum_{t=1}^T \left(\ell_t(\mathbf{x}_t) - \ell_t(\mathbf{u})\right)$ be the regret of OL algorithm $\mathcal{A}$ over loss $\{\ell_t(\mathbf{x})\}_t^T$.

Proof: $$\mathbb{E}[F(\bar{\mathbf{x}}_T)] - F(\mathbf{u}) \leq \frac{1}{\sum_{t=1}^T \alpha_t} \sum_{t=1}^T \alpha_t (\mathbb{E}[F(\mathbf{x}_t)] - F(\mathbf{u}))$$

$$\leq \frac{1}{\sum_{t=1}^T \alpha_t} \sum_{t=1}^T \mathbb{E}\left[\alpha_t \langle \mathbf{g}(\mathbf{x}_t), \mathbf{x}_t - \mathbf{u} \rangle - \frac{\alpha_t \sigma}{2} \|\mathbf{x}_t - \mathbf{u}\|^2\right] = \frac{\mathbb{E}\left[\textsc{Reg}_T^{\mathcal{A}}(\mathbf{u})\right]}{\sum_{t=1}^T \alpha_t}. \qquad \square$$

O2B for Strongly Convex Functions

Theorem 9 (O2B for Strongly Convex Functions). *Assume $F(\mathbf{x})$ is σ-strongly-convex, and let $\mathbf{x}^\star = \arg\min_{\mathbf{x}} F(\mathbf{x})$. By setting $\mathcal{A}$ as OGD with learning rate $\eta_t = \frac{1}{\sigma \sum_{i=1}^t \alpha_i}$ and $\alpha_t = t$, Algorithm 4 achieves the following guarantee:*

$$\mathbb{E}[F(\bar{\mathbf{x}}_T)] - F(\mathbf{x}^\star) \leq \mathcal{O}\left(\frac{1}{T}\right).$$

Recall that in Theorem 8, we have

$$\mathbb{E}[F(\bar{\mathbf{x}}_T)] - F(\mathbf{x}^\star) \leq \frac{\mathbb{E}\left[\textsc{Reg}_T^{\mathcal{A}}(\mathbf{u})\right]}{\sum_{t=1}^T \alpha_t}.$$

First, as $\alpha_t = t$, the denominator becomes $\frac{T(T+1)}{2}$.

Next, we focus on $\mathbb{E}\left[\textsc{Reg}_T^{\mathcal{A}}(\mathbf{u})\right]$, where $\ell_t(\mathbf{x})$ is defined by $\ell_t(\mathbf{x}) = \langle \alpha_t \mathbf{g}_t, \mathbf{x} \rangle + \frac{\alpha_t \sigma}{2} \|\mathbf{x}_t - \mathbf{x}\|^2$.

O2B for Strongly Convex Functions

$G_t = \alpha_t G$

Proof:

$$\mathbb{E}\left[\text{REG}_T^{\mathcal{A}}\right] = \mathbb{E}\left[\sum_{t=1}^{T}\left(f_t(\mathbf{x}_t) - f_t(\mathbf{u})\right)\right] \le \mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{x}_t - \mathbf{u}\|^2\left(\frac{1}{2\eta_t} - \frac{1}{2\eta_{t-1}} - \frac{\sigma}{2}\right)\right] + \mathbb{E}\left[\sum_{t=1}^{T}\frac{G_t^2\eta_t}{2}\right]$$

Choosing $\eta_t = \frac{1}{\sigma\sum_{s=1}^{t}\alpha_s}$, we have

$$\mathbb{E}\left[\text{REG}_T^{\mathcal{A}}(\mathbf{x}^\star)\right] \le \sum_{t=1}^{T}\frac{G_t^2}{2\sigma\sum_{i=1}^{t}\alpha_i}$$

$$= \sum_{t=1}^{T}\frac{G^2t^2}{2\sigma\sum_{i=1}^{t}t} = \frac{G^2}{2\sigma}\sum_{t=1}^{T}\frac{t}{t-1} = \mathcal{O}(T).$$

Plugging it back to Theorem 8, we obtain

$$\mathbb{E}[F(\bar{\mathbf{x}}_T)] - F(\mathbf{x}^\star) \le \mathcal{O}\left(\frac{1}{T}\right).$$

□

OGD for Strongly Convex Functions

Proof: $f_t(\mathbf{x}_t) - f_t(\mathbf{u}) \le \frac{\eta_t^{-1} - \sigma}{2}\|\mathbf{x}_t - \mathbf{u}\|^2 - \frac{\eta_t^{-1}}{2}\|\mathbf{x}_{t+1} - \mathbf{u}\|^2 + \frac{\eta_t G^2}{2}$

Summing from $t = 1$ to T, setting $\eta_t = \frac{1}{\sigma t}$ (define $\frac{1}{\eta_0} := 0$):

$$2\sum_{t=1}^{T}\left(f_t(\mathbf{x}_t) - f_t(\mathbf{u})\right) \le \sum_{t=1}^{T}\|\mathbf{x}_t - \mathbf{u}\|^2\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} - \sigma\right) + G^2\sum_{t=1}^{T}\eta_t \quad \left(\tfrac{1}{\eta_0} := 0\right)$$

$$= 0 + G^2\sum_{t=1}^{T}\frac{1}{\sigma t} \quad \left(\tfrac{1}{\eta_0} \triangleq 0, \|\mathbf{x}_{T+1} - \mathbf{u}\|^2 \ge 0, \tfrac{1}{\eta_t} - \tfrac{1}{\eta_{t-1}} - \sigma = 0\right)$$

$$\le \frac{G^2}{\sigma}(1 + \log T).$$

□

Summary of G-Lipschitz functions

- Offline Optimization: $\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$ with gradient oracle $\nabla F(\cdot)$

  convergence rate Convex: $\mathcal{O}(\frac{1}{\sqrt{T}})$; Strongly Convex $\mathcal{O}(\frac{1}{T})$.

- Online Optimization: $\{f_t(\mathbf{x})\}_{t=1}^T$

  regret bound Convex: $\mathcal{O}(\sqrt{T})$; Strongly Convex $\mathcal{O}(\log T)$.

- Stochastic Optimization: $\min_{\mathbf{x}\in\mathcal{X}} F(\mathbf{x})$ with noisy gradient oracle $\mathbf{g}(\cdot)$

  convergence rate Convex: $\mathcal{O}(\frac{1}{\sqrt{T}})$; Strongly Convex $\mathcal{O}(\frac{1}{T})$.

by (weighted) online-to-batch conversion

More bits of OGD

- Note that function $\ell_t(\mathbf{x}) \triangleq \mathbf{g}(\mathbf{x}_t)^\top \mathbf{x}$ ***depends*** on the decision $\mathbf{x}_t$, which actually reveals that OGD regret can hold even against ***adaptive adversary***.

At each round $t = 1, 2, \cdots$

(1) the player first picks a model $\mathbf{x}_t \in \mathcal{X}$;

(2) and simultaneously environments pick an online function $f_t : \mathcal{X} \to \mathbb{R}$;

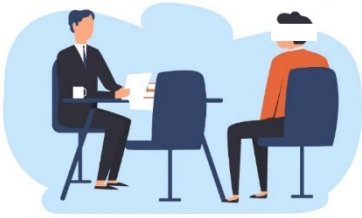
(3) the player suffers loss $f_t(\mathbf{x}_t)$, observes some information about f_t and updates the model.

oblivious adversary



examination

adaptive adversary



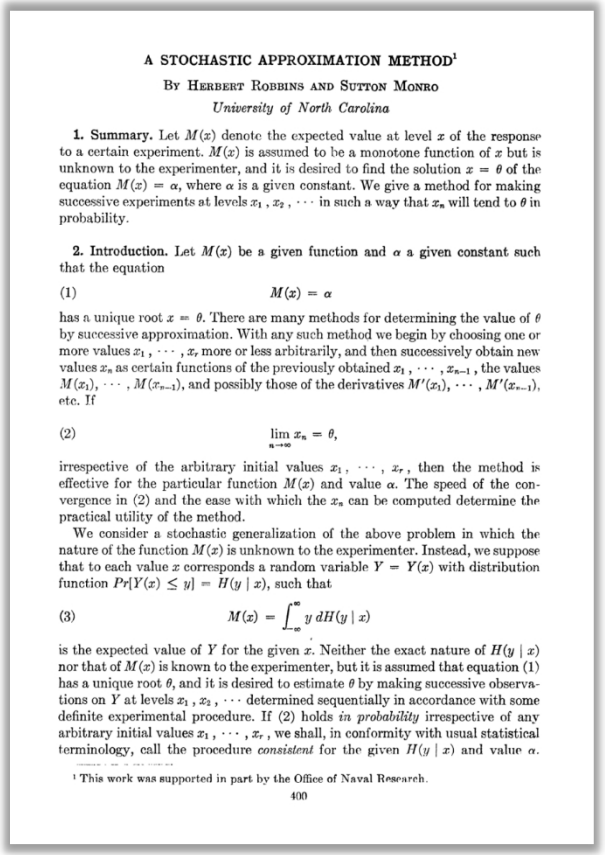
interview

The "simultaneously" requirement can be sometimes not necessary!

OGD for full-info OCO can handle the case when online functions depend on $\mathbf{x}_t$!

History: SGD

Robbins-Monro Method



A STOCHASTIC APPROXIMATION METHOD[1]

By Herbert Robbins and Sutton Monro

University of North Carolina

1. Summary. Let $M(x)$ denote the expected value at level x of the response to a certain experiment. $M(x)$ is assumed to be a monotone function of x but is unknown to the experimenter, and it is desired to find the solution $x = \theta$ of the equation $M(x) = \alpha$, where α is a given constant. We give a method for making successive experiments at levels $x_1, x_2, \cdots$ in such a way that x_n will tend to θ in probability.

2. Introduction. Let $M(x)$ be a given function and α a given constant such that the equation

$$M(x) = \alpha \tag{1}$$

has a unique root $x = \theta$. There are many methods for determining the value of θ by successive approximation. With any such method we begin by choosing one or more values $x_1, \cdots, x_r$ more or less arbitrarily, and then successively obtain new values x_n as certain functions of the previously obtained $x_1, \cdots, x_{n-1}$, the values $M(x_1), \cdots, M(x_{n-1})$, and possibly those of the derivatives $M'(x_1), \cdots, M'(x_{n-1})$, etc. If

$$\lim_{n\to\infty} x_n = \theta, \tag{2}$$

irrespective of the arbitrary initial values $x_1, \cdots, x_r$, then the method is effective for the particular function $M(x)$ and value α. The speed of the convergence in (2) and the ease with which the x_n can be computed determine the practical utility of the method.

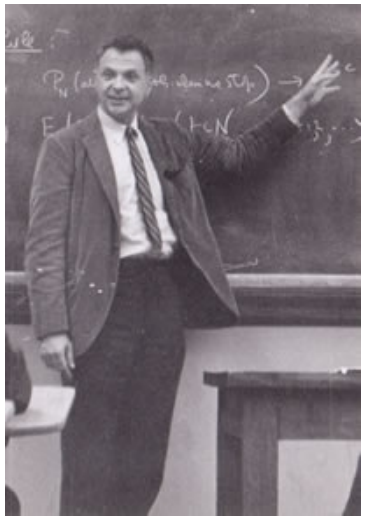
We consider a stochastic generalization of the above problem in which the nature of the function $M(x)$ is unknown to the experimenter. Instead, we suppose that to each value x corresponds a random variable $Y = Y(x)$ with distribution function $Pr[Y(x) \leq y] = H(y \mid x)$, such that

$$M(x) = \int_{-\infty}^{\infty} y\, dH(y \mid x) \tag{3}$$

is the expected value of Y for the given x. Neither the exact nature of $H(y \mid x)$ nor that of $M(x)$ is known to the experimenter, but it is assumed that equation (1) has a unique root θ, and it is desired to estimate θ by making successive observations on Y at levels $x_1, x_2, \cdots$ determined sequentially in accordance with some definite experimental procedure. If (2) holds *in probability* irrespective of any arbitrary initial values $x_1, \cdots, x_r$, we shall, in conformity with usual statistical terminology, call the procedure *consistent* for the given $H(y \mid x)$ and value α.

[1] This work was supported in part by the Office of Naval Research.

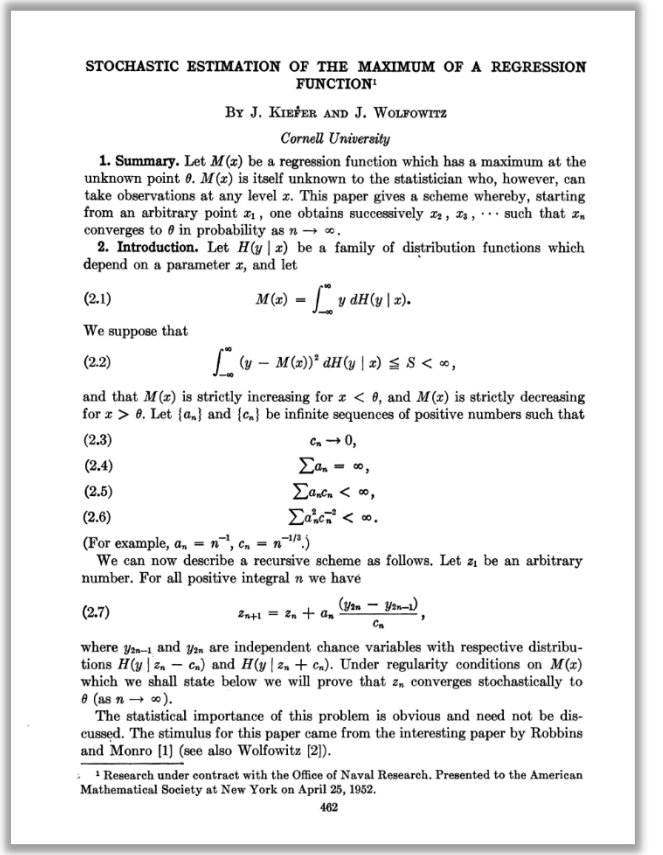
400



Herbert Ellis Robbins
(1915 - 2001)

A Stochastic Approximation Method.
Herbert Robbins, Sutton Monro
Ann. Math. Statist. 22(3): 400-407 (September, 1951).

Kiefer-Wolfowitz Method



STOCHASTIC ESTIMATION OF THE MAXIMUM OF A REGRESSION FUNCTION[1]

By J. Kiefer and J. Wolfowitz

Cornell University

1. Summary. Let $M(x)$ be a regression function which has a maximum at the unknown point θ. $M(x)$ is itself unknown to the statistician who, however, can take observations at any level x. This paper gives a scheme whereby, starting from an arbitrary point x_1, one obtains successively $x_2, x_3, \cdots$ such that x_n converges to θ in probability as $n \to \infty$.

2. Introduction. Let $H(y \mid x)$ be a family of distribution functions which depend on a parameter x, and let

$$M(x) = \int_{-\infty}^{\infty} y\, dH(y \mid x). \tag{2.1}$$

We suppose that

$$\int_{-\infty}^{\infty} (y - M(x))^2\, dH(y \mid x) \leqq S < \infty, \tag{2.2}$$

and that $M(x)$ is strictly increasing for $x < \theta$, and $M(x)$ is strictly decreasing for $x > \theta$. Let $\{a_n\}$ and $\{c_n\}$ be infinite sequences of positive numbers such that

$$c_n \to 0, \tag{2.3}$$

$$\sum a_n = \infty, \tag{2.4}$$

$$\sum a_n c_n < \infty, \tag{2.5}$$

$$\sum a_n^2 c_n^{-2} < \infty. \tag{2.6}$$

(For example, $a_n = n^{-1}$, $c_n = n^{-1/3}$.)

We can now describe a recursive scheme as follows. Let z_1 be an arbitrary number. For all positive integral n we have

$$z_{n+1} = z_n + a_n \frac{(y_{2n} - y_{2n-1})}{c_n}, \tag{2.7}$$

where y_{2n-1} and y_{2n} are independent chance variables with respective distributions $H(y \mid z_n - c_n)$ and $H(y \mid z_n + c_n)$. Under regularity conditions on $M(x)$ which we shall state below we will prove that z_n converges stochastically to θ (as $n \to \infty$).

The statistical importance of this problem is obvious and need not be discussed. The stimulus for this paper came from the interesting paper by Robbins and Monro [1] (see also Wolfowitz [2]).

[1] Research under contract with the Office of Naval Research. Presented to the American Mathematical Society at New York on April 25, 1952.

462



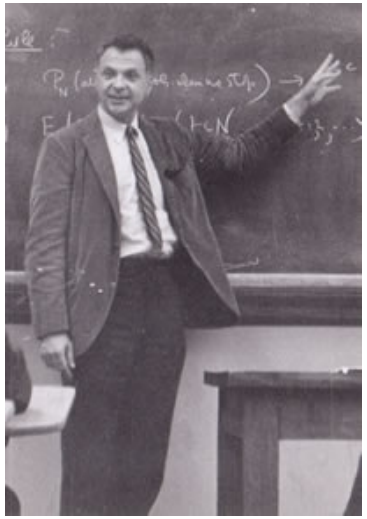
Jack Kiefer
(1924 - 1981)



Jacob Wolfowitz
(1910 - 1981)

Stochastic Estimation of the Maximum of a Regression Function
Jack Kiefer, Jacob Wolfowitz
Ann. Math. Statist. 23(3): 462-466 (September, 1952)

History: SGD



Herbert Ellis Robbins
(1915 - 2001)

Statistical Science
1986, Vol. 1, No. 2, 276–284

The Contributions of Herbert Robbins to Mathematical Statistics

Tze Leung Lai and David Siegmund

Herbert Robbins was born on January 12, 1915, in New Castle, Pennsylvania. In 1931 he entered Harvard College at the age of 16. Although his interests until then had been predominantly literary, he found himself increasingly attracted to mathematics under the influence of Marston Morse, who during many long conversations conveyed a vivid sense of the intellectual challenge of creative work in that field (cf. Page, 1984, p. 7). He received the A.B. summa cum laude in 1935, and the Ph.D. in 1938, also from Harvard. His thesis, in the field of combinatorial topology and written under the supervision of Hassler Whitney, was published in 1941 [3]. (Numbers in brackets refer to Robbins' bibliography at the end of this article.)

After graduation, Robbins worked for a year at the Institute for Advanced Study at Princeton as Marston Morse's assistant. He then spent the next three years at New York University as instructor in mathematics. He became nationally known in 1941 as the coauthor, North Carolina at Chapel Hill. Having read [7] and [10], and greatly impressed by Robbins' mathematical skills, Hotelling offered him the position of associate professor to teach measure theory and probability to the graduate students in the new department. Robbins accepted the position and spent the next six years at Chapel Hill. During this relatively short period Robbins not only studied and developed an increasingly deep interest in statistics, but he also made a number of profound contributions to his new field: complete convergence [12], compound decision theory [25], stochastic approximation [26], and the sequential design of experiments [28], to name a few.

After a Guggenheim Fellowship at the Institute for Advanced Study during 1952–1953, Robbins moved from Chapel Hill to Columbia University as professor and chairman of the Department of Mathematical Statistics. Since 1953, with the exception of the three years 1965–1968 spent at Minnesota, Purdue, Berkeley, and Michigan, he has been at Columbia, where he

History: Two-Player Zero-Sum Games

Theory of repeated games



James Hannan
(1922–2010)



David Blackwell
(1919–2010)

Learning to play a game (1956)

Play a game repeatedly against a possibly suboptimal opponent

Zero-sum 2-person games played more than once

	1	2	...	M
1	$\ell(1,1)$	$\ell(1,2)$	...	
2	$\ell(2,1)$	$\ell(2,2)$	...	
⋮	⋮	⋮	⋱	
N				

$N \times M$ known loss matrix

- Row player (player) has N actions
- Column player (opponent) has M actions

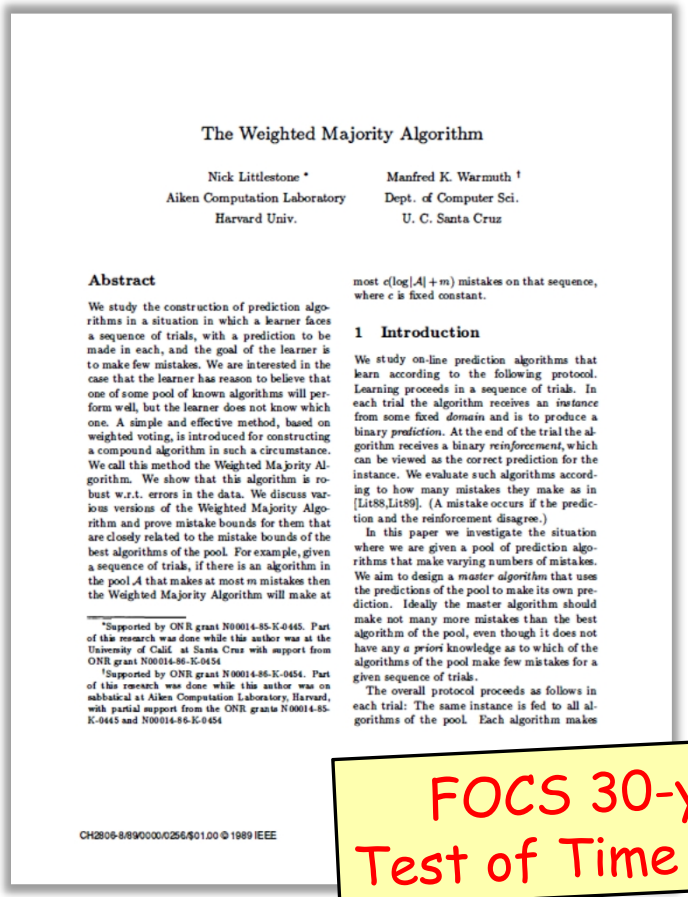
For each game round $t = 1, 2, \ldots$

- Player chooses action i_t and opponent chooses action y_t
- The player suffers loss $\ell(i_t, y_t)$ (= gain of opponent)

Player can learn from opponent's history of past choices $y_1, \ldots, y_{t-1}$

Nicolo Cesa-Bianchi, Online Learning and Online Convex Optimization. Tutorial at the Simons Institute. 2017.

History: Prediction with Expert Advice



The Weighted Majority Algorithm

Nick Littlestone * — Aiken Computation Laboratory, Harvard Univ.

Manfred K. Warmuth † — Dept. of Computer Sci., U. C. Santa Cruz

Abstract

We study the construction of prediction algorithms in a situation in which a learner faces a sequence of trials, with a prediction to be made in each, and the goal of the learner is to make few mistakes. We are interested in the case that the learner has reason to believe that one of some pool of known algorithms will perform well, but the learner does not know which one. A simple and effective method, based on weighted voting, is introduced for constructing a compound algorithm in such a circumstance. We call this method the Weighted Majority Algorithm. We show that this algorithm is robust w.r.t. errors in the data. We discuss various versions of the Weighted Majority Algorithm and prove mistake bounds for them that are closely related to the mistake bounds of the best algorithms of the pool. For example, given a sequence of trials, if there is an algorithm in the pool A that makes at most m mistakes then the Weighted Majority Algorithm will make at most $c(\log|A|+m)$ mistakes on that sequence, where c is fixed constant.

1 Introduction

We study on-line prediction algorithms that learn according to the following protocol. Learning proceeds in a sequence of trials. In each trial the algorithm receives an *instance* from some fixed *domain* and is to produce a binary *prediction*. At the end of the trial the algorithm receives a binary *reinforcement*, which can be viewed as the correct prediction for the instance. We evaluate such algorithms according to how many mistakes they make as in [Lit88,Lit89]. (A mistake occurs if the prediction and the reinforcement disagree.)

In this paper we investigate the situation where we are given a pool of prediction algorithms that make varying numbers of mistakes. We aim to design a *master algorithm* that uses the predictions of the pool to make its own prediction. Ideally the master algorithm should make not many more mistakes than the best algorithm of the pool, even though it does not have any *a priori* knowledge as to which of the algorithms of the pool make few mistakes for a given sequence of trials.

The overall protocol proceeds as follows in each trial: The same instance is fed to all algorithms of the pool. Each algorithm makes

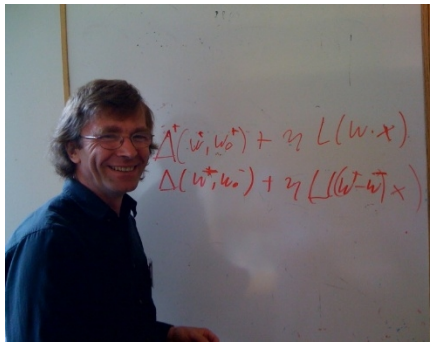
*Supported by ONR grant N00014-85-K-0445. Part of this research was done while this author was at the University of Calif. at Santa Cruz with support from ONR grant N00014-86-K-0454

†Supported by ONR grant N00014-86-K-0454. Part of this research was done while this author was on sabbatical at Aiken Computation Laboratory, Harvard, with partial support from the ONR grants N00014-85-K-0445 and N00014-86-K-0454

CH2806-8/89/0000/0256/$01.00 © 1989 IEEE

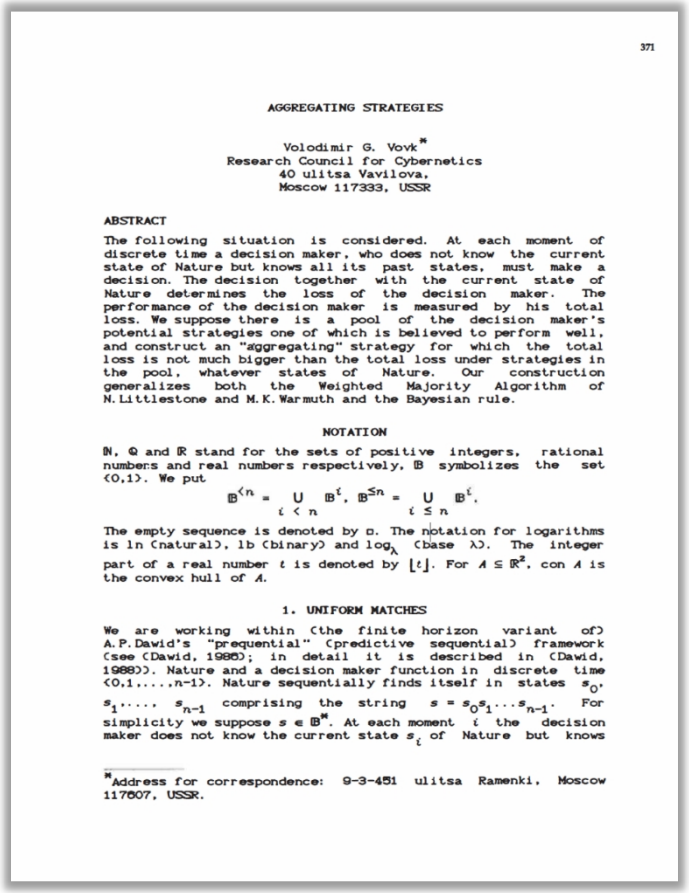
Nick Littlestone and Manfred K. Warmuth.
"The Weighted Majority Algorithm." FOCS 1989: 256-261.

FOCS 30-year Test of Time Award!



Manfred Warmuth

UC Santa Cruz



371

AGGREGATING STRATEGIES

Volodimir G. Vovk*
Research Council for Cybernetics
40 ulitsa Vavilova,
Moscow 117333, USSR

ABSTRACT

The following situation is considered. At each moment of discrete time a decision maker, who does not know the current state of Nature but knows all its past states, must make a decision. The decision together with the current state of Nature determines the loss of the decision maker. The performance of the decision maker is measured by his total loss. We suppose there is a pool of the decision maker's potential strategies one of which is believed to perform well, and construct an "aggregating" strategy for which the total loss is not much bigger than the total loss under strategies in the pool, whatever states of Nature. Our construction generalizes both the Weighted Majority Algorithm of N. Littlestone and M. K. Warmuth and the Bayesian rule.

NOTATION

$\mathbb{N}$, $\mathbb{Q}$ and $\mathbb{R}$ stand for the sets of positive integers, rational numbers and real numbers respectively, $\mathbb{B}$ symbolizes the set $\langle 0,1\rangle$. We put

$$\mathbb{B}^{<n} = \bigcup_{i<n} \mathbb{B}^i, \quad \mathbb{B}^{\le n} = \bigcup_{i\le n} \mathbb{B}^i.$$

The empty sequence is denoted by $\Box$. The notation for logarithms is ln (natural), lb (binary) and $\log_\lambda$ (base λ). The integer part of a real number t is denoted by $\lfloor t \rfloor$. For $A \subseteq \mathbb{R}^2$, con A is the convex hull of A.

1. UNIFORM MATCHES

We are working within (the finite horizon variant of) A. P. Dawid's "prequential" (predictive sequential) framework (see (Dawid, 1985); in detail it is described in (Dawid, 1988)). Nature and a decision maker function in discrete time $\langle 0,1,\dots,n-1\rangle$. Nature sequentially finds itself in states $s_0, s_1, \dots, s_{n-1}$ comprising the string $s = s_0 s_1 \dots s_{n-1}$. For simplicity we suppose $s \in \mathbb{B}^n$. At each moment i the decision maker does not know the current state s_i of Nature but knows

*Address for correspondence: 9-3-451 ulitsa Ramenki, Moscow 117607, USSR.

Volodimir G. Vovk. "Aggregating Strategies." COLT 1990: 371-383.



Volodimir G. Vovk
Royal Holloway,
University of London

Summary

ONLINE OPTIMIZATION
- Interaction optimization
- Game-theoretic language

ONLINE-TO-BATCH CONVERSION
- Online-to-batch conversion
- Weighted O2B conversion
- SGD, stochastic optimization

ONLINE CONVEX OPTIMIZATION
- Problem formulation
- Regret measure
- Convex functions: OGD
- Strongly convex functions: OGD

Q & A

Thanks!