No-Regret Learning in Convex Games

Geoff Gordon, Amy Greenwald, Casey Marks
Background

- No-regret algorithms can learn well, even in adversarial environments.
- Seems useful for learning in a repeated game.
- Well-known result: no-regret learners reach minimax equilibrium in zero-sum normal-form games.
- More recent: no-internal-regret learners reach correlated equilibrium in general-sum normal-form games.
Normal-form picture

Regret

no external ↔ no internal

coarse correlated ← correlated

Equilibrium
This talk

- What about games with more structure?
- E.g., extensive-form games
- E.g., classification or regression
- In general, convex games
Convex games (known)

Regret

no external $\iff$

$\upharpoonleft\upharpoonright$

coarse cor. $\iff$

$\iff$ no swap

$\upharpoonleft\upharpoonright$

$\iff$ correlated

Equilibrium
Convex games (contrib)

Regret

no external $\iff$ no EF $\iff$ no linear $\iff$ no FE $\iff$ no swap

coarse cor. $\iff$ EFCE $\iff$ “linear” $\iff$ “FE” $\iff$ correlated

Equilibrium
Convex games (contrib)

Regret

no external ⇐ no EF ⇐ no linear ⇐ no FE ⇐ no swap

EFCE ⇐ “linear” ⇐ “FE” ⇐ correlated

Equilibrium

all equivalent in matrix games
Convex games (contrib)

Regret

no external $\iff$ no EF $\iff$ no linear $\iff$ no FE $\iff$ no swap

$\updownarrow$  $\updownarrow$  $\updownarrow$  $\updownarrow$  $\updownarrow$  $\updownarrow$

coarse cor. $\iff$ EFCE $\iff$ “linear” $\iff$ “FE” $\iff$ correlated

efficient algorithms

Equilibrium
Convex games (contrib)

Regret

- no external \iff no EF \iff no linear \iff no FE \iff no swap
- coarse cor. \iff EFCE \iff “linear” \iff “FE” \iff correlated

Equilibrium

efficient algorithms
Outline

• Convex games and OCPs
• Regret and $\Phi$-regret
• Algorithm
• Making it fast
• Summary & related work
Convex games

- Generalization of normal-form games, extensive-form games, ...
- $N$ players, convex action sets $A_i$
- $\text{Loss}(i) = c(a_{-i}) \cdot a_i$
- E.g., $A_i = \text{simplex}: \text{normal-form game}$
Online convex programs

- Single agent’s view of repeated convex game
- Convex feasible region $A$
- Alternately choose action, see cost vector
  - $a_1 \rightarrow c_1 \rightarrow a_2 \rightarrow c_2 \rightarrow \ldots$
- Total cost = $a_1 \cdot c_1 + a_2 \cdot c_2 + \ldots$
- If $A =$ simplex: “expert advice” problem
External regret

$$\rho_t = \sum c_t \cdot a_t - \min_{a \in A} \sum c_t \cdot a$$

= (observed cost) – (cost of best action post-hoc)
Action transformations

- Function \( \varphi: A \mapsto A \)
- maps OCP’s feasible region to itself
- E.g., linear fn mapping unit square into itself

\[
\begin{pmatrix}
\pm p & \pm (1-p) \\
\pm q & \pm (1-q)
\end{pmatrix}
\]

\( p, q \in [0,1] \)
Action transformations

- Function $\varphi: A \mapsto A$
- maps OCP’s feasible region to itself
- E.g., linear fn mapping unit square into itself

$\begin{pmatrix} \pm p & \pm (1-p) \\ \pm q & \pm (1-q) \end{pmatrix}$

$p, q \in [0,1]$
$\Phi$-regret

$$\rho_t = \sum c_t \cdot a_t - \min \sum c_t \cdot \varphi(a_t)$$

$$\varphi \in \Phi$$

= (observed cost) − (cost of best transformation post-hoc)
Regret examples

- $\Phi = \text{constant transformations (}= \text{external})$
- $\Phi = \text{all measurable transformations (}= \text{swap})$
- $\Phi = \text{linear transformations}$
$\Phi = \text{finite element}$
\( \Phi = \text{EF transformations} \)

- Extensive-form correlated equilibrium
- No time for these—come to the poster!
Algorithm idea
Algorithm idea

- Reduce the no-$\Phi$-regret problem for $A$ to a no-external-regret problem on a more-complicated feasible region
Algorithm idea

- Reduce the no-$\Phi$-regret problem for $A$ to a no-external-regret problem on a more-complicated feasible region
- Namely, $\Phi$, considered as a subset of a vector space
Algorithm

\[ a_1 \rightarrow c_1 \rightarrow a_2 \rightarrow c_2 \rightarrow a_3 \rightarrow c_3 \rightarrow \ldots \]

- Get first play \( \varphi_1 \) from NER subroutine
- For \( t = 1, 2, \ldots \)
  - Find fixed point \( a_t \) of \( \varphi_t \), play \( a_t \)
  - Observe \( c_t \)
  - Construct \( m_t(\varphi) = c_t \cdot \varphi(a_t) \)
  - Give \( m_t, \varphi_t \) to NER subroutine, get \( \varphi_{t+1} \)
Algorithm

\[ a_1 \rightarrow c_1 \rightarrow a_2 \rightarrow c_2 \rightarrow a_3 \rightarrow c_3 \ldots \]

\[ \varphi_1 \rightarrow m_1 \rightarrow \varphi_2 \rightarrow m_2 \rightarrow \varphi_3 \rightarrow m_3 \ldots \]

• Get first play \( \varphi_1 \) from NER subroutine
• For \( t = 1, 2, \ldots \)
  • Find fixed point \( a_t \) of \( \varphi_t \), play \( a_t \)
  • Observe \( c_t \)
  • Construct \( m_t(\varphi) = c_t \cdot \varphi(a_t) \)
  • Give \( m_t, \varphi_t \) to NER subroutine, get \( \varphi_{t+1} \)
Algorithm

1. Get first play $\varphi_1$ from NER subroutine
2. For $t = 1, 2, \ldots$
   - Find fixed point $a_t$ of $\varphi_t$, play $a_t$
   - Observe $c_t$
   - Construct $m_t(\varphi) = c_t \cdot \varphi(a_t)$
   - Give $m_t, \varphi_t$ to NER subroutine, get $\varphi_{t+1}$
Algorithm

- $a_1 \rightarrow c_1$, $a_2 \rightarrow c_2$, $a_3 \rightarrow c_3$, ...
- $\uparrow$, $\downarrow$, $\uparrow$, $\downarrow$, $\uparrow$, $\downarrow$, ...
- $\varphi_1$, $m_1 \rightarrow \varphi_2$, $m_2 \rightarrow \varphi_3$, $m_3$, ...

- Get first play $\varphi_1$ from NER subroutine
- For $t = 1, 2, ...$
  - Find fixed point $a_t$ of $\varphi_t$, play $a_t$
  - Observe $c_t$
  - Construct $m_t(\varphi) = c_t \cdot \varphi(a_t)$
  - Give $m_t$, $\varphi_t$ to NER subroutine, get $\varphi_{t+1}$

*Geoff Gordon, Amy Greenwald, Casey Marks—No Regret in Convex Games*
Theorem

- The algorithm achieves no $\Phi$-regret
- It runs in poly time if its subroutines do
Proof of no $\Phi$-regret

\[
\forall \varphi \in \Phi, \sum m_t(\varphi_t) \leq \sum m_t(\varphi) + o(T)
\]

\[
\sum c_t \cdot \varphi_t(a_t) \leq \sum c_t \cdot \varphi(a_t) + o(T)
\]

\[
\sum c_t \cdot a_t \leq \sum c_t \cdot \varphi(a_t) + o(T)
\]
Making it fast

- $\Phi$ may be a complex set
- Expensive to achieve no external regret
- Solution: use (half of) the kernel trick to get efficient linear representation
Half of kernel trick

- Nonlinear function $\varphi(a)$
- Represent as $\varphi(a) = M_\varphi \circ K(a)$

  adjustable linear fn

  fixed nonlinearity
Kernelized $\Phi \Rightarrow$ fast

- For any $\varphi \in \Phi$
  - $m_t(\varphi) = c_t \cdot M_\varphi \ K(a_t) = \text{tr}((K(a_t) \ c_t^T) \ M_\varphi)$
  - I.e., $m_t(\varphi)$ is a linear function of $M_\varphi$
  - And, $\mathcal{M} = \{ \text{feasible } M_\varphi \}$ is convex
  - Standard OCP $\Rightarrow$ standard OCP algorithms
Theorem

• If we play only vertices of mesh, and achieve no FE-regret, we also have no swap regret

• We can extend our algorithm by “warping” its plays to nearby vertices, while preserving regret guarantees

⇒ efficient algorithm to learn CE
Summary

Regret

no external $\iff$ no EF $\iff$ no linear $\iff$ no FE $\iff$ no swap

coarse cor. $\iff$ EFCE $\iff$ “linear” $\iff$ “FE” $\iff$ correlated

efficient algorithms

Equilibrium

FE $\Rightarrow$ swap if “careful”
Summary

Regret

- no external $\iff$ no EF $\iff$ no linear $\iff$ no FE $\iff$ no swap

- coarse cor. $\iff$ EFCE $\iff$ “linear” $\iff$ “FE” $\iff$ correlated

Efficient algorithms:

1st efficient no-EF-regret learner
Summary

Regret

no external ⇛ no EF ⇛ no linear ⇛ no FE ⇛ no swap

coarse cor. ⇛ EFCE ⇛ “linear” ⇛ “FE” ⇛ correlated

Equilibrium

1st efficient no-EF-regret learner

efficient algorithm

exponentially-more-efficient CE
Related work

- Blum & Mansour [2005]
  - fixed-point trick for normal-form learners
- Stoltz & Lugosi [2007]
  - nice analysis of $\Phi$-regret in OCPs
  - first algorithm for $\Phi = \text{swap}$
Related work

- Hazan & Kale [simultaneous w/ us]
  - proposed non-kernelized version of algorithm
  - nice reduction between finding fixed points and achieving no Phi-regret
  - lack of kernel trick slows implementation by arbitrarily large factor
Thanks!