# DEALING WITH NON-TRANSITIVITY IN TWO-PLAYER ZERO-SUM GAMES

Dr. Yaodong Yang
www.yangyaodong.com
07/2021

# Contents

- **What is Non-Transitivity in Games**
- **How to Measure Non-Transitivity**
- **Solutions: Double Oracle / PSRO Methods**
- **Recent advances: Diverse-PSRO**
- **Recent advances: Online-PSRO**
- **Recent advances: Auto-PSRO**

# Multi-Agent Reinforcement Learning in Games

## Great advantages have been made in 2019!

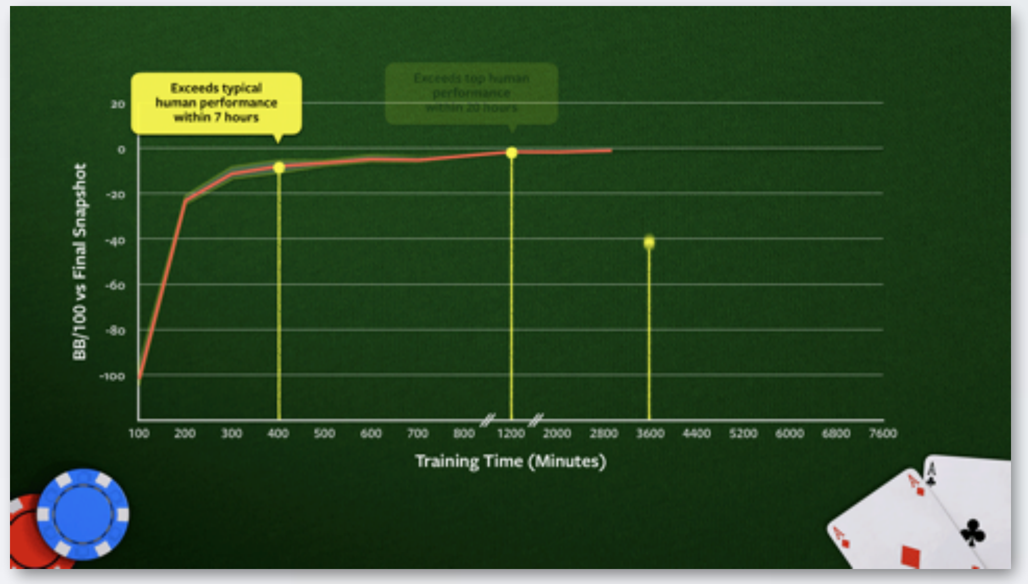
Jan 2016 | Dec 2017 | July 2018 | Jan 2019 | Apr 2019 | July 2019 | Sep 2019

AlphaGO Series

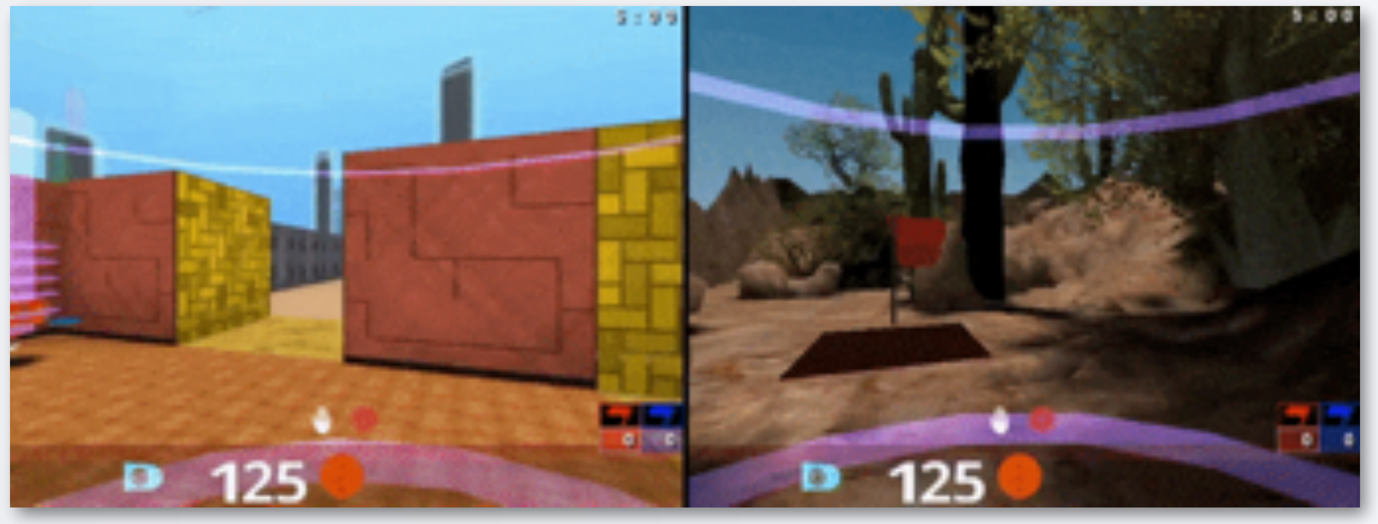

technique of single-agent decision-making is mature

AlphaStar (DeepMind)

Pluribus Poker (FAIR)

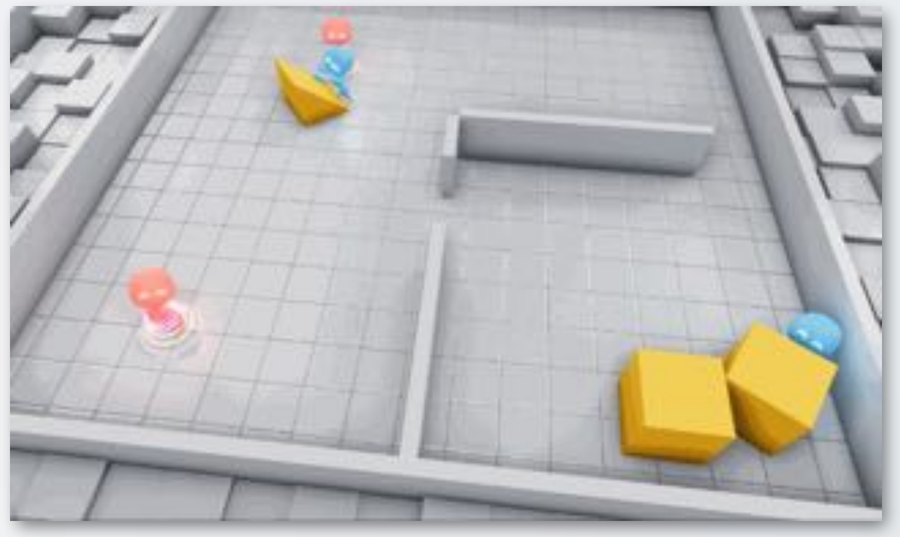


Capture-the-flag (DeepMind)




Dota2 (OpenAI)

Hide and Seek (OpenAI)



**techniques of multi-agent decision-making is getting mature !**

# A General Solver to Two-Player Zero-Sum Games

**Output:** the reward $(R^1, \ldots, R^N)$

**Black-box multi-agent game engine**

**Input:** a joint strategy $(\pi^1, \ldots, \pi^N)$

**input**

**Our algorithm:**

**output**

Low-exploitability strategy $(\pi^{1,*} \ldots, \pi^{N,*})$

$$\mathbf{Br}^i(\pi^{-i}) = \arg\max_{\pi^i} \mathbf{E}_{a^i \sim \pi^i, a^{-i} \sim \pi^{-i}} \left[ R^i\left(a^i, a^{-i}\right) \right]$$

$$\mathbf{Exploitability}\left(\boldsymbol{\pi}\right) = \sum_{i=1}^{2} R^i\left(\mathbf{Br}^i(\boldsymbol{\pi}^{-i}), \boldsymbol{\pi}^{-i}\right) - R^i\left(\boldsymbol{\pi}\right)$$

# Computing Nash Equilibrium via Linear Programming

- In two-player zero-sum discrete case, it can be solved in polynomial time. The matrix $\mathbf{A}_{\mathfrak{P}}$ is anti-symmetrical, i.e., $\mathbf{A}_{\mathfrak{P}} = -\mathbf{A}_{\mathfrak{P}}^{\top}$.

$$\mathbf{A}_{\mathfrak{P}} := \left\{ \phi\left(\mathbf{w}_i, \mathbf{w}_j\right) : \left(\mathbf{w}_i, \mathbf{w}_j\right) \in \mathfrak{P} \times \mathfrak{P} \right\} =: \phi(\mathfrak{P} \otimes \mathfrak{P})$$

- The minimax theorem is a natural outcome of the duality theorem in LP.

**Prime problem**

$$\max_{v \in \mathbb{R}} v$$
$$\text{s.t. } \mathbf{p}^{\top}\mathbf{A}_{\mathfrak{P}} \succeq v \cdot \mathbf{1}$$
$$\mathbf{p} \succeq \mathbf{0} \text{ and } \mathbf{p}^{\top}\mathbf{1} = 1$$

/

**Dual problem**

$$\min_{v \in \mathbb{R}} v$$
$$\text{s.t. } \mathbf{q}^{\top}\mathbf{A}_{\mathfrak{P}}^{\top} \preceq v \cdot \mathbf{1}$$
$$\mathbf{q} \succeq \mathbf{0} \text{ and } \mathbf{q}^{\top}\mathbf{1} = 1$$

⟺

**Minimax theorem**

$$\max_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^{\top}\mathbf{A}_{\mathfrak{P}}\mathbf{q}$$
$$= \min_{\mathbf{q}} \max_{\mathbf{p}} \mathbf{p}^{\top}\mathbf{A}_{\mathfrak{P}}\mathbf{q}$$

- However, real-world games are open-ended, since there are infinitely many strategies.

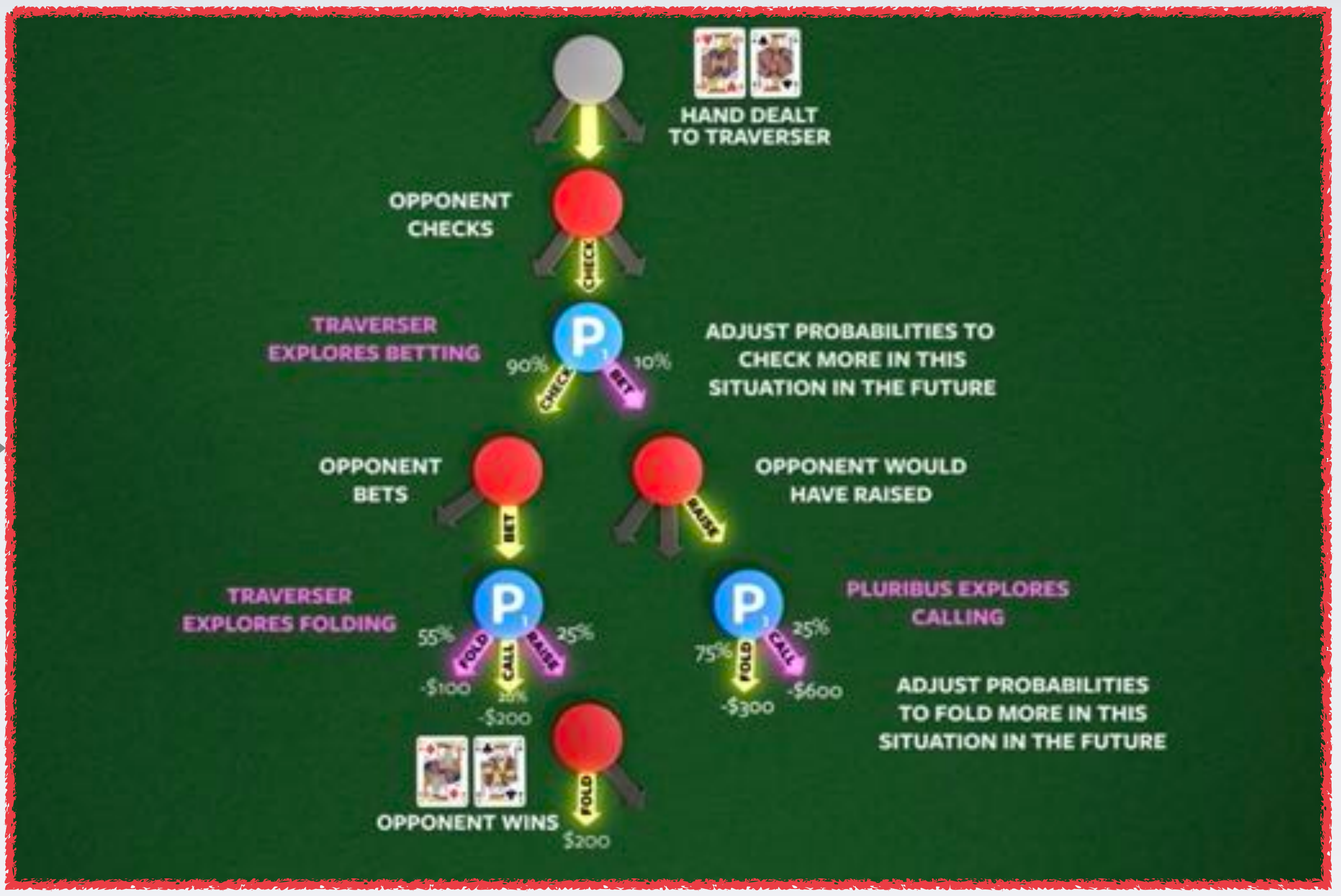
- We have to look at the game from at the policy space (meta-games).

# Two Main-Streams of Solutions: Regret based vs. Best Response based

**Output:** the reward $(R^1, \ldots, R^N)$

**Black-box multi-agent game engine**

**Input:** a joint strategy $(\pi^1, \ldots, \pi^N)$




Regret based methods: Poker Type

When planning is feasible (game tree is easily accessible), existing techniques can solve the games really well.

Perfect-information games:
MCTS, alpha-beta search, AlphaGO series (AlphaZero, MuZero, etc)

Imperfect-information:
CFR series (DeepCFR, Libratus/Pluribus, Deepstack), XFP/NFSP series

Best response based methods: StarCraft type

Planning is not always feasible. StarCraft has $10^{26}$ choices per step (vs. the game tree size of chess $10^{50}$, Texas holdem $10^{80}$, GO $10^{170}$)

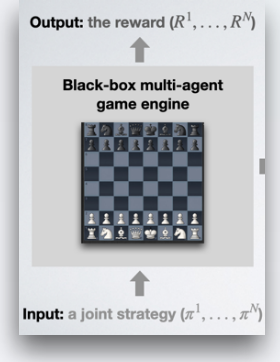
Enumerating all policies' actions at each state and then playing a randomise best response is infeasible (i.e. RPS can not apply)

Solution: design a game of game — meta-game, the problem problem, auto-curricula.

# Problem Formulation of Two-Player Zero-Sum Games

- Let's formulate the self-play process.
  - Suppose two agents, agent 1 adopts policy parameterised by $\boldsymbol{v} \in \mathbb{R}^d$, and agent 2 adopts policy $\boldsymbol{w} \in \mathbb{R}^d$. They can be considered as two neural networks.
  - Define a **functional-form game (FFG)** [Balduzzi 2019] to be represented by a function

$$\phi : \underset{\text{RL model}}{V} \times \underset{\text{RL model}}{W} \rightarrow \mathbb{R}$$




  - $\phi$ represents the game rule, it is anti-symmetrical.
  - $\phi > 0$ means agent 1 wins over agent 2, the higher $\phi(\boldsymbol{v}, \boldsymbol{w})$ the better for agent 1.
  - with $\phi_{\mathbf{w}}(\bullet) := \phi(\bullet, \mathbf{w})$, we can have the best response defined by:

$$\boldsymbol{v}' := \textbf{Br}(\boldsymbol{w}) = \textbf{Oracle}\big(\boldsymbol{v}, \phi_{\boldsymbol{w}}(\cdot)\big) \quad \textbf{s.t.} \quad \phi_{\mathbf{w}}(\mathbf{v}') > \phi_{\mathbf{w}}(\mathbf{v}) + \epsilon$$

  - **Oracle**: a god tells us how to beat the enemy, it can be implemented by a RL algorithm, for example PPO + PBT as we have mentioned early, or other optimiser such as evolutionary algorithm.

# Naive Self-play Will Not Work

Question: Can we use it as a general framework to solve any games?

self-plays

**PPO + PBT + Self-play = Panacea ?**

**Algorithm 2** Self-play

> **input:** agent $\mathbf{v}_1$
> **for** $t = 1, \ldots, T$ **do**
>     $\mathbf{v}_{t+1} \leftarrow \text{oracle}\left(\mathbf{v}_t, \phi_{\mathbf{v}_t}(\bullet)\right)$
> **end for**
> **output:** $\mathbf{v}_{T+1}$

$$\left(\pi^1, \pi^2\right) \rightarrow \left(\pi^1, \pi^{2,*} = \mathbf{Br}(\pi^1)\right) \rightarrow \left(\pi^{1,*} = \mathbf{Br}(\pi^{2,*}), \pi^{2,*}\right)$$

**It depends. In most of the games, it does not work.**

# Naive Self-play Will Not Work

- It is because of Non-Transitivity

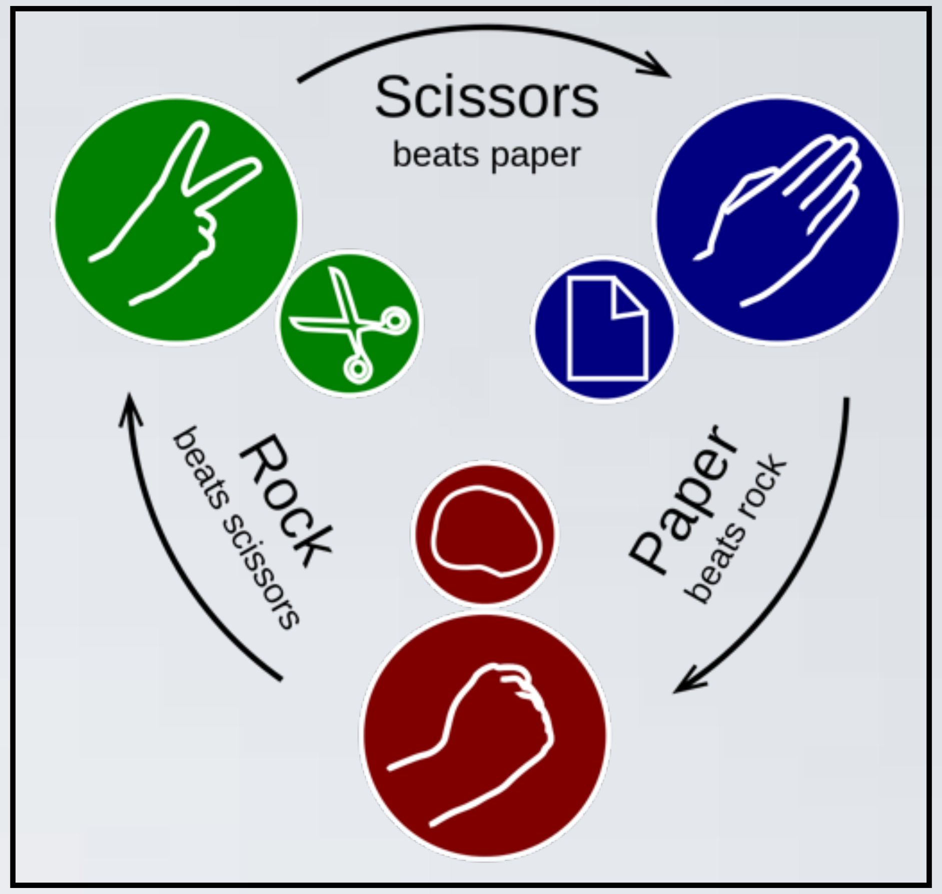
$$\int_W \phi(\mathbf{v}, \mathbf{w}) \cdot d\mathbf{w} = 0, \quad \forall \mathbf{v} \in W$$

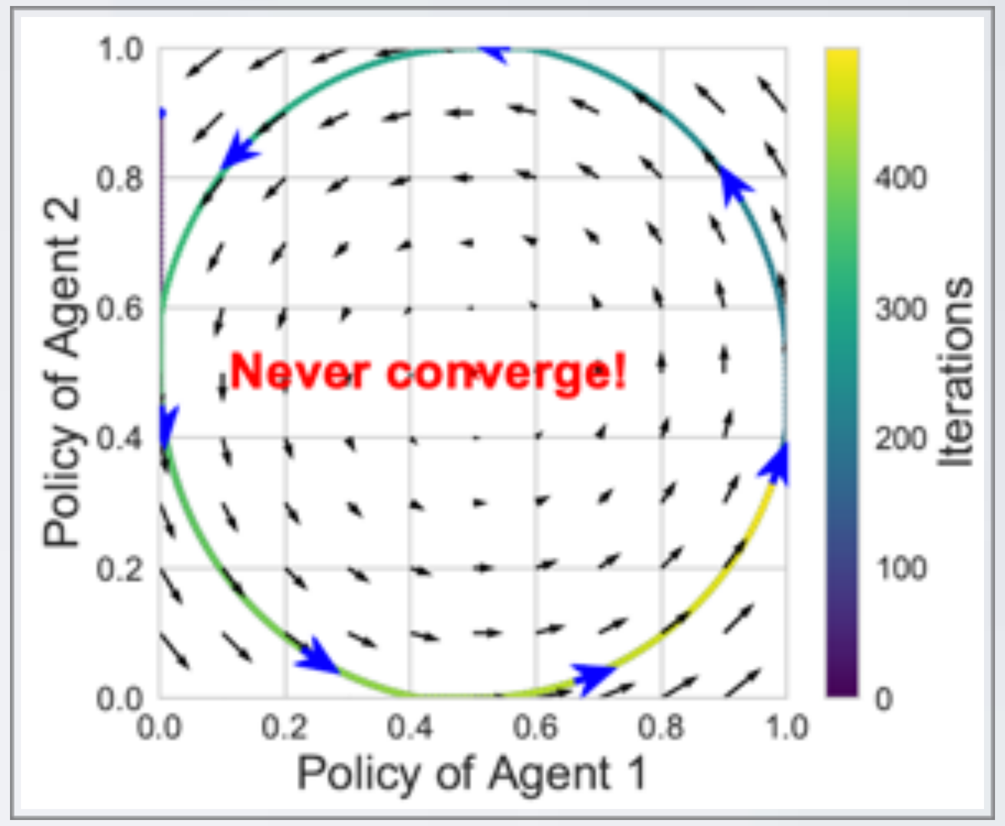
  - Rock-Paper-Scissor game:

$$\begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & 1 \\ 1 & -1 & 0 \end{bmatrix}$$

  - Disc game:

$$\phi(\mathbf{v}, \mathbf{w}) = \mathbf{v}^\top \cdot \begin{pmatrix} 0, & -1 \\ 1, & 0 \end{pmatrix} \cdot \mathbf{w} = v_1 w_2 - v_2 w_1$$




# Game Decomposition

- Every FFG can be decomposed into two parts [Balduzzi 2019]

**FFG = Transitive game ⊕ Non-transitivegame**

- Let $v, w \in W$ be a compact set and $\phi(v, w)$ prescribe the flow from $v$ to $w$, then this is a natural result after applying *combinatorial hodge theory* [Jiang 2011].
- We can write any games $\phi$ as summation of two **orthogonal** components

$\text{grad}(f)(\mathbf{v}, \mathbf{w}) := f(\mathbf{v}) - f(\mathbf{w})$ $\quad$ $\text{div}(\phi)(\mathbf{v}) := \int_W \phi(\mathbf{v}, \mathbf{w}) \cdot d\mathbf{w}$ $\quad$ $\text{curl}(\phi)(\mathbf{u}, \mathbf{v}, \mathbf{w}) := \phi(\mathbf{u}, \mathbf{v}) + \phi(\mathbf{v}, \mathbf{w}) - \phi(\mathbf{u}, \mathbf{w})$

$$\phi = \underbrace{\text{grad} \circ \text{div}(\phi)}_{\text{curl}(\cdot)=0} + \underbrace{(\phi - \text{grad} \circ \text{div}(\phi))}_{\text{div}(\cdot)=0}$$

**Transitive game** $\quad$ **Non-transitive game**

- Example on Rock-Paper-Scissor

| | R | P | S |
|---|---|---|---|
| R | $0, 0$ | $-3x, 3x$ | $3y, -3y$ |
| P | $3x, -3x$ | $0, 0$ | $-3z, 3z$ |
| S | $-3y, 3y$ | $3z, -3z$ | $0, 0$ |

(a) Generalized RPS Game

=

| | R | P | S |
|---|---|---|---|
| R | $(y-x), (y-x)$ | $(y-x), (x-z)$ | $(y-x), (z-y)$ |
| P | $(x-z), (y-x)$ | $(x-z), (x-z)$ | $(x-z), (z-y)$ |
| S | $(z-y), (y-x)$ | $(z-y), (x-z)$ | $(z-y), (z-y)$ |

(c) Potential Component

**Transitive game**

+

| | R | P | S |
|---|---|---|---|
| R | $0, 0$ | $-(x+y+z), (x+y+z)$ | $(x+y+z), -(x+y+z)$ |
| P | $(x+y+z), -(x+y+z)$ | $0, 0$ | $-(x+y+z), (x+y+z)$ |
| S | $-(x+y+z), (x+y+z)$ | $(x+y+z), -(x+y+z)$ | $0, 0$ |

(d) Harmonic Component

**Non-transitive game**

+

| | R | P | S |
|---|---|---|---|
| R | $(x-y), (x-y)$ | $(z-x), (x-y)$ | $(y-z), (x-y)$ |
| P | $(x-y), (z-x)$ | $(z-x), (z-x)$ | $(y-z), (z-x)$ |
| S | $(x-y), (y-z)$ | $(z-x), (y-z)$ | $(y-z), (y-z)$ |

(b) Nonstrategic Component

# What is Transitivity ?

- Every FFG can be decomposed into two parts

$$\text{FFG} = \text{Transitive game} \oplus \text{Non-transitivegame}$$

- **Transitive Game**: the rules of winning are transitive across different players.

$$\boldsymbol{v}_t \text{ beats } \boldsymbol{v}_{t-1}, \quad \boldsymbol{v}_{t+1} \text{ beats } \boldsymbol{v}_t \quad \rightarrow \quad \boldsymbol{v}_{t+1} \text{ beats } \boldsymbol{v}_{t-1}$$

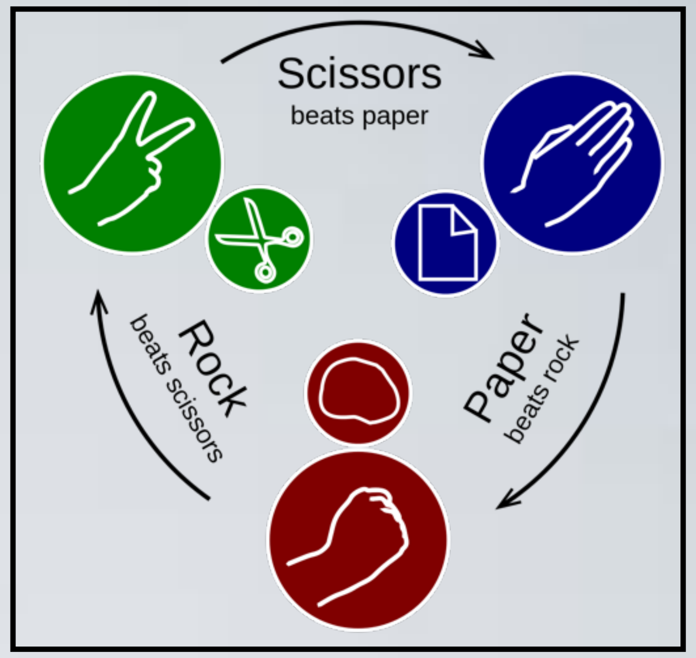
  - Example: Elo rating (段位) offers rating scores $f(\cdot)$ that assume transitivity.

$$\phi(\mathbf{v}, \mathbf{w}) = \text{softmax}\big(f(\mathbf{v}) - f(\mathbf{w})\big)$$

  - Larger score means you are likely to win over players with lower scores.
  - Elo score is widely used in GO and Chess.
  - This explains why you don't want to play with rookies, when $f(v_t) \gg f(w)$ ,

$$\nabla_{\mathbf{v}}\phi\left(\mathbf{v}_t, \mathbf{w}\right) \approx 0$$

# What is Non-Transitivity ?




- Every FFG can be decomposed into two parts

$$\text{FFG} = \text{Transitive game} \oplus \text{Non-transitivegame}$$

- **Non-transitive Game**: the rules of winning are not-transitive across players.

$$\boldsymbol{v}_t \text{ beats } \boldsymbol{v}_{t-1}, \quad \boldsymbol{v}_{t+1} \text{ beats } \boldsymbol{v}_t \nrightarrow \boldsymbol{v}_{t+1} \text{ beats } \boldsymbol{v}_{t-1}$$

  - Mutual dominance across different types of modules in a game. This is commonly observed in modern MOBA games.

  - For this types of game, self-play is not helpful at all because transitivity assumption does not hold. Self-play will lead to cyclic loops forever.

# Visualisation of Transitive and Non-Transitive Games

- Let us define the evaluation matrix for a population of $N$ agents to be

$$\mathbf{A}_{\mathfrak{P}} := \left\{ \phi\left(\mathbf{w}_i, \mathbf{w}_j\right) : \left(\mathbf{w}_i, \mathbf{w}_j\right) \in \mathfrak{P} \times \mathfrak{P} \right\} =: \phi\left(\mathfrak{P} \otimes \mathfrak{P}\right)$$

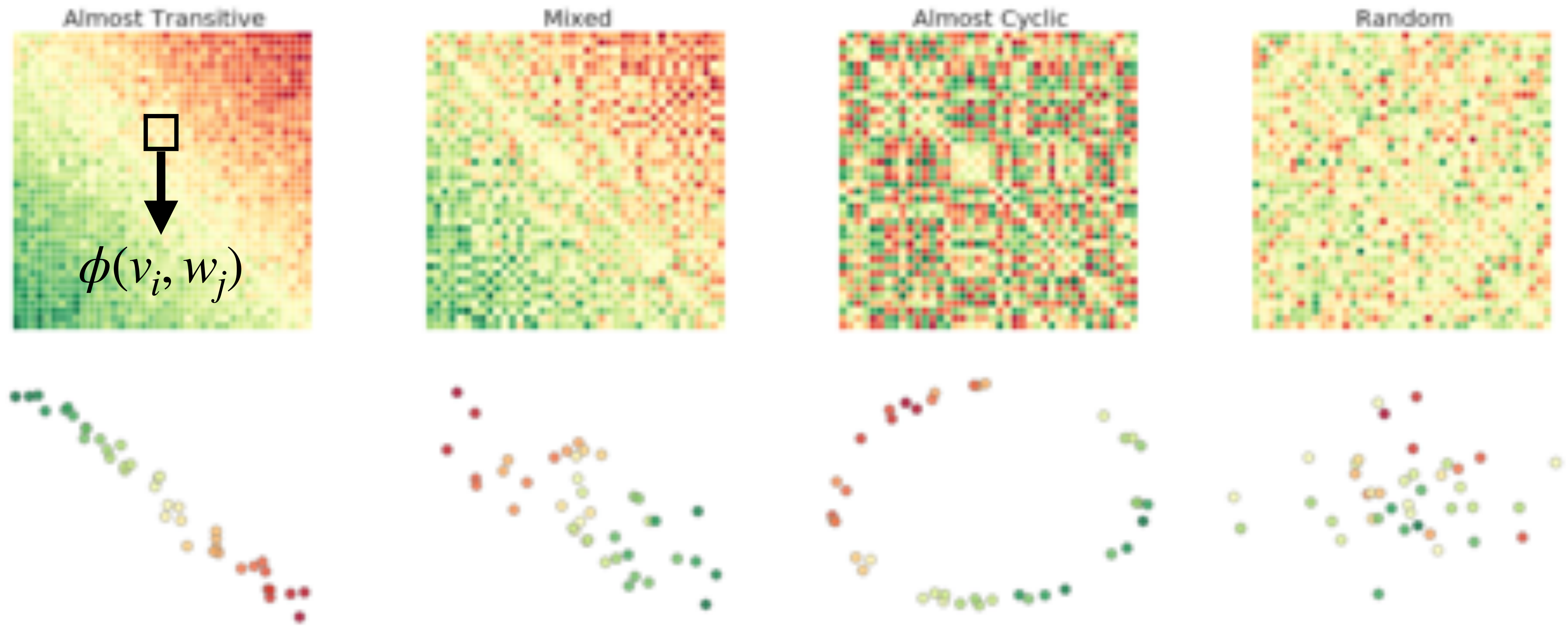
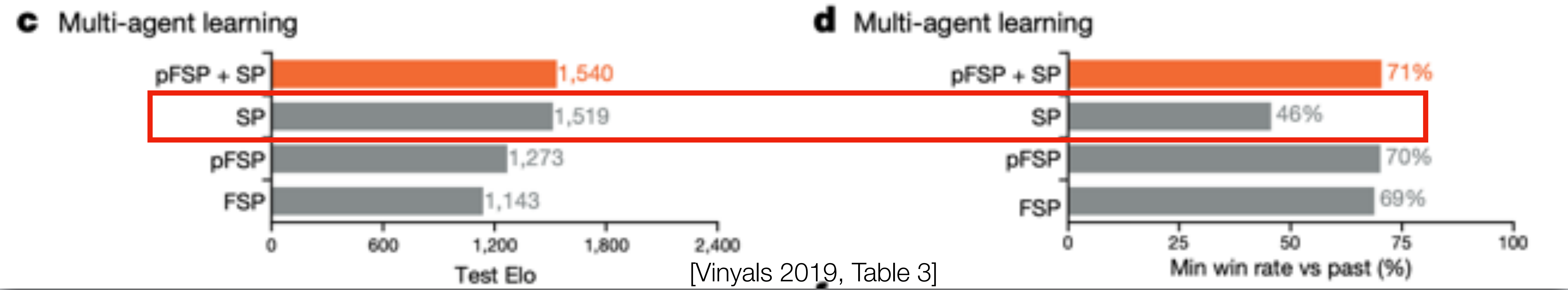



*Figure 1. Low-dim gamescapes of various basic game structures.* **Top row:** Evaluation matrices of populations of 40 agents each; colors vary from red to green as $\phi$ ranges over $[-1, 1]$. **Bottom row:** 2-dim embedding obtained by using first 2 dimensions of Schur decomposition of the payoff matrix; Color corresponds to average payoff of an agent against entire population; EGS of the transitive game is a line; EGS of the cyclic game is two-dim near-circular polytope given by convex hull of points. For extended version see Figure 6 in the Appendix.

[Balduzzi 2019]

# Non-Transitivity Harms Training !

## Example on training AlphaStar:




[Vinyals 2019, Table 3]

## Example on training Soccer AI:

Table 2: Average goal difference ± one standard deviation across 5 repetitions of the experiment.

| | |
|---|---|
| $A$ vs built-in AI | $4.25 \pm 1.72$ |
| $B$ vs $A$ | $11.93 \pm 2.19$ |
| $B$ vs built-in AI | $-0.27 \pm 0.33$ |

[Karol 2020, table 2]

## Example on training AlphaGO:

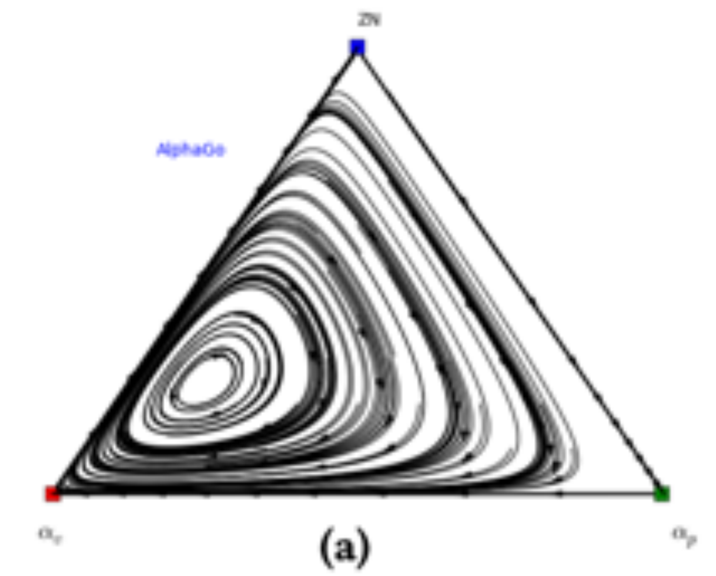
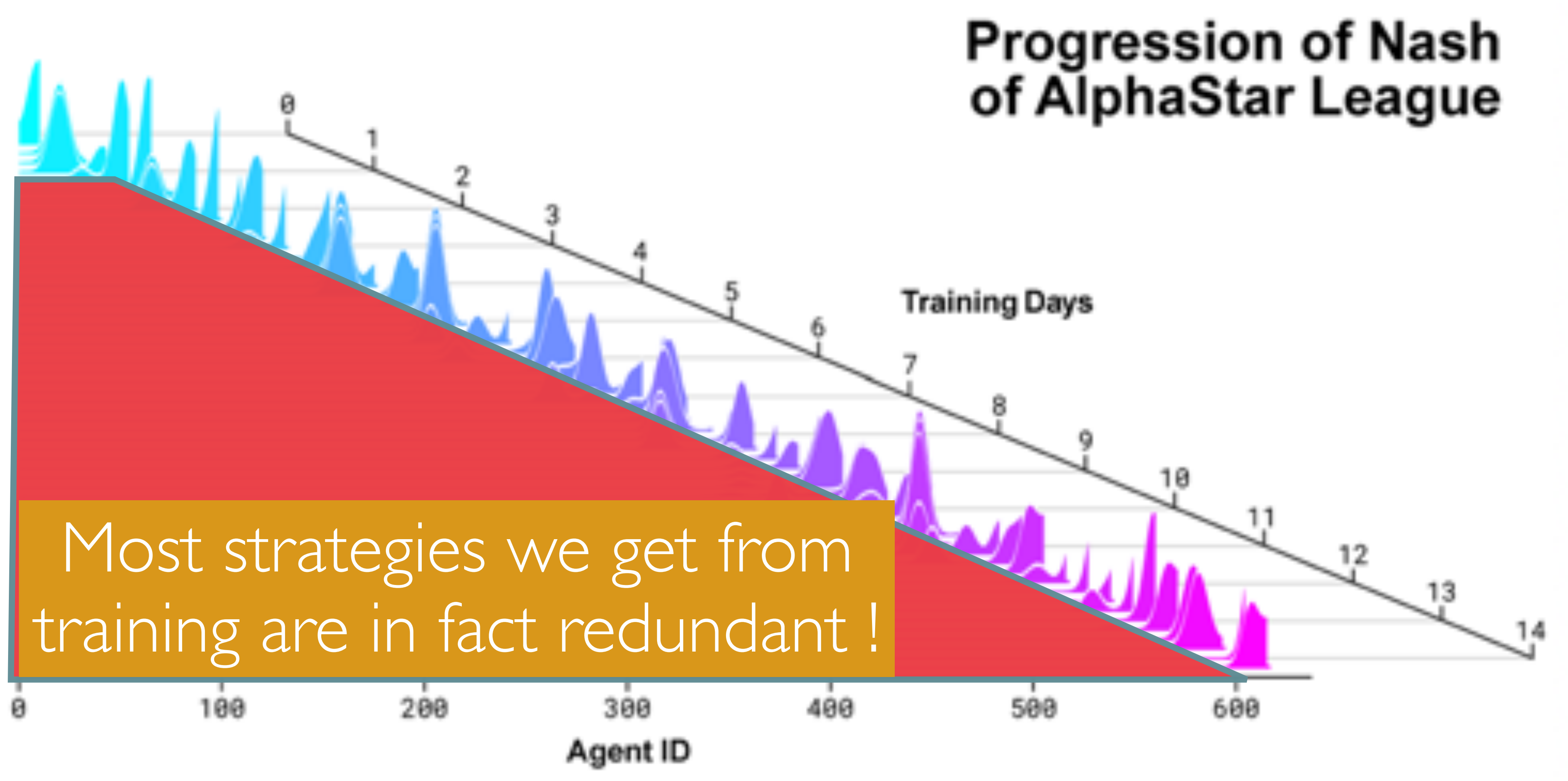



**Figure 5:** Intransitive behaviour for $\alpha_v$, $\alpha_p$, and *Zen*.

[Silver 2016, table 9]

# Dealing With Non-Transitivity Helps Save Training Time




THE NASH DISTRIBUTION OVER COMPETITORS AS THE ALPHASTAR LEAGUE PROGRESSED AND NEW COMPETITORS WERE CREATED. THE NASH DISTRIBUTION, WHICH IS THE LEAST EXPLOITABLE SET OF COMPLEMENTARY COMPETITORS, WEIGHTS THE NEWEST COMPETITORS MOST HIGHLY, DEMONSTRATING CONTINUAL PROGRESS AGAINST ALL PREVIOUS COMPETITORS.

[AlphaStar Blog]

Table 2: Size of the Nash Support of Games

| Game | Total Strategies | Size of Nash support |
|---|---|---|
| 3-Move Parity Game 2 | 160 | 1 |
| 5,4-Blotto | 56 | 6 |
| AlphaStar | 888 | 3 |
| Connect Four | 1470 | 23 |
| Disc Game | 1000 | 27 |
| Elo game + noise=0.1 | 1000 | 6 |
| Elo game | 1000 | 1 |
| Go (boardsize=3,komi=6.5) | 1933 | 13 |
| Misere (game=tic tac toe) | 926 | 1 |
| Normal Bernoulli game | 1000 | 5 |
| Quoridor (boardsize=3) | 1404 | 1 |
| Random game of skill | 1000 | 5 |
| Tic Tac Toe | 880 | 1 |
| Transitive game | 1000 | 1 |
| Triangular game | 1000 | 1 |

[online double oracle]

# Contents

# The Spinning Top Hypothesis

- Real-world games are mixtures of both transitive and in-transitive components, e.g., Go, DOTA, StarCraft II.
- Though winning is often harder than losing a game, finding a strategy that always loses is also challenging.
- Players who regularly practice start to beat less skilled players, this corresponds to the transitive dynamics.
- At certain level (the red part), players will start to find many different strategy styles. Despite not providing a universal advantage against all opponents, players will counter each other within the same transitive group. This provide direct information of improvement.
- As players get stronger to the highest level, seeing many strategy styles, the outcome relies mostly on skill and less on one particular game styles (以不变应万变).

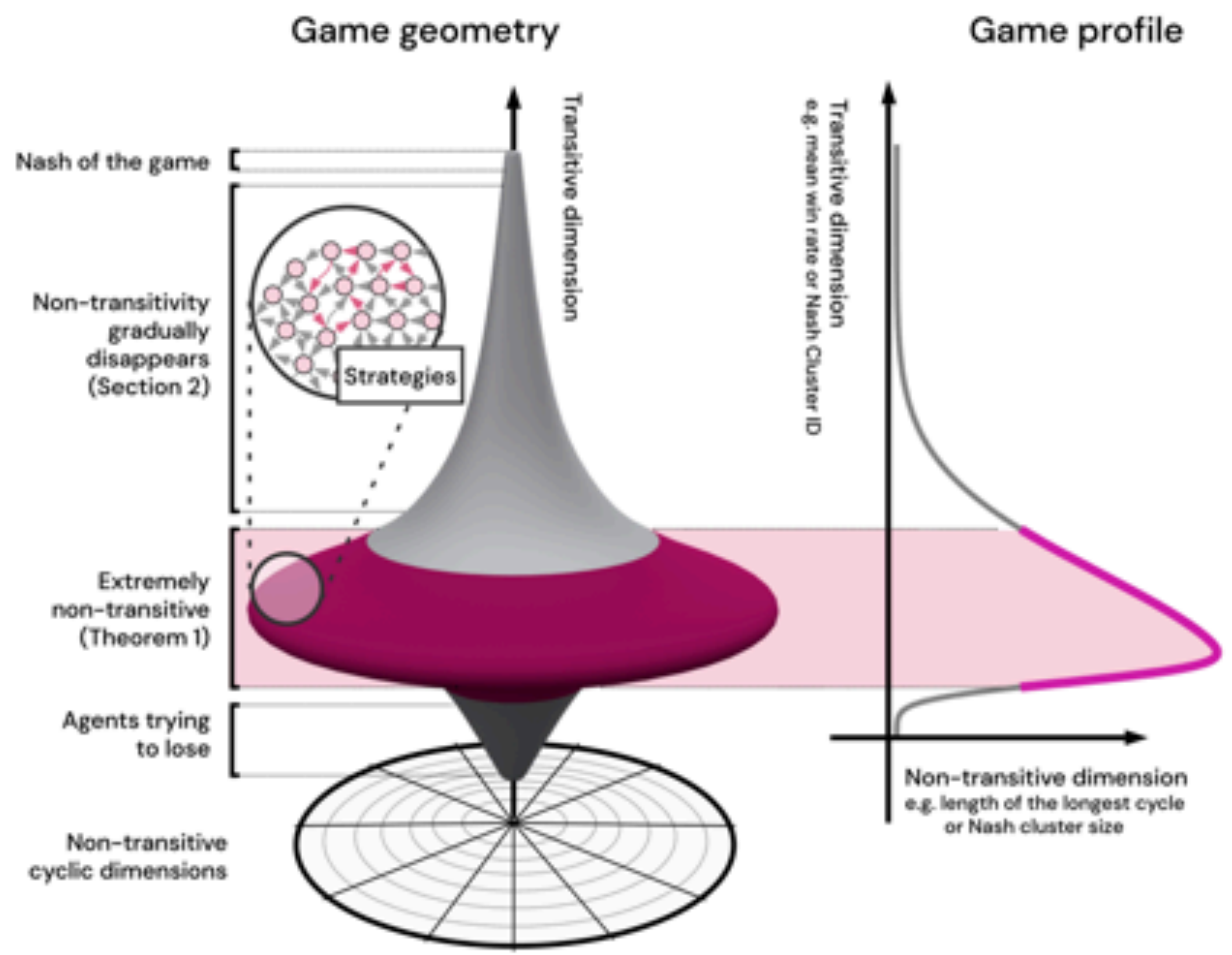



Figure 1: High-level visualisation of the geometry of Games of Skill. It shows a strong transitive dimension, that is accompanied by the highly cyclic dimensions, which gradually diminishes as skill grows towards the Nash Equilibrium (upward), and diminishes as skill evolves towards the worst possible strategies (downward). The simplest example of non-transitive behaviour is a cycle of length 3 that one finds e.g. in the Rock Paper Scissors game.

[Czarnecki 2020]

# Measuring the Non-Transitivity

- A theoretical lower bound of the size of non-transitivity [Czarnecki 2020]
  - n-bit communicative game

**Definition 1.** *Consider the extensive form view of the win-draw-loss version of any underlying game; the underlying game is called n-bit communicative if each player can transmit $n \in \mathbb{R}_+$ bits of information to the other player before reaching the node whereafter at least one of the outcomes 'win' or 'loss' is not attainable.*

*bit: how many action one can take before the outcome of the game is predetermined*

**Theorem 1.** *For every game that is at least n-bit communicative, and every antisymmetric win-loss payoff matrix $\mathbf{P} \in \{-1, 0, 1\}^{\lfloor 2^n \rfloor \times \lfloor 2^n \rfloor}$, there exists a set of $\lfloor 2^n \rfloor$ pure strategies $\{\pi_1, ..., \pi_{\lfloor 2^n \rfloor}\} \subset \Pi$ such that $\mathbf{P}_{ij} = \mathbf{f}^\dagger(\pi_i, \pi_j)$, and $\lfloor x \rfloor = \max_{a \in \mathbb{N}} a \leq x$.*

*n-bit game = there exists at least a non-transitive circle of size $2^n$*

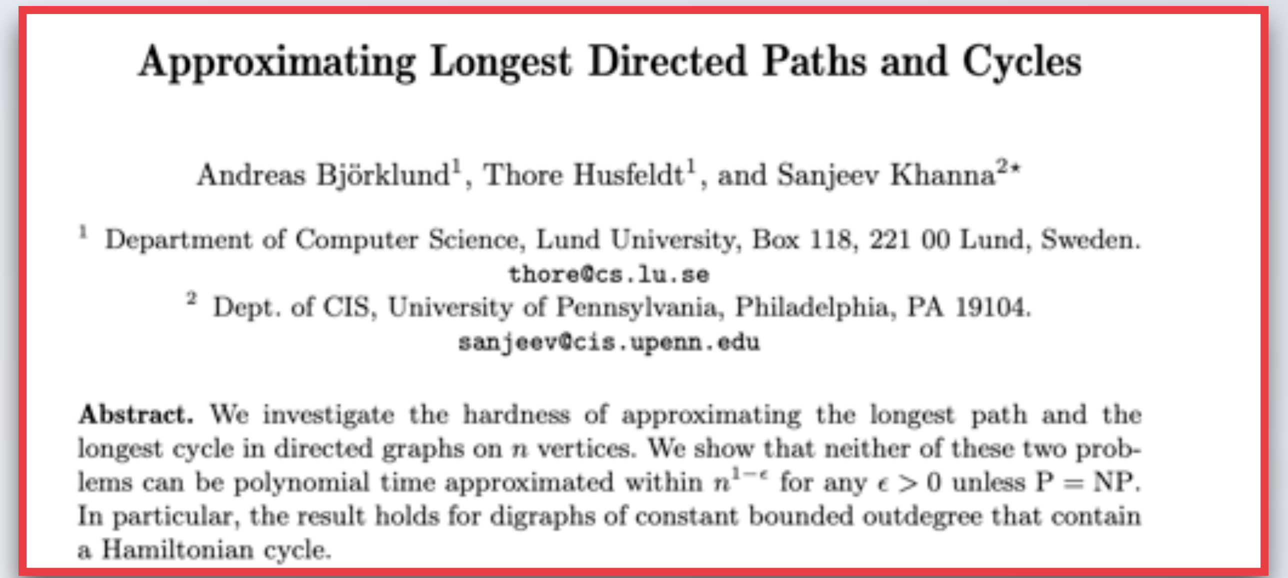
  - Results on GO and MOBA games:

**Proposition 1.** *The game of Go is at least 1000-bit communicative and contains a cycle of length at least $2^{1000}$.*

**Proposition 2.** *Modern games, such as StarCraft, DOTA or Quake, when limited to 10 minutes play, are at least 36000-bit communicative.*

# Measuring the Non-Transitivity

- A practical way of measurement through meta-game analysis
  - computing n-bit communicative game needs full tree traversing, thus intractable
  - Deciding a graph has a path of length higher than k is NP-hard



**Approximating Longest Directed Paths and Cycles**

Andreas Björklund[1], Thore Husfeldt[1], and Sanjeev Khanna[2*]

[1] Department of Computer Science, Lund University, Box 118, 221 00 Lund, Sweden.
thore@cs.lu.se
[2] Dept. of CIS, University of Pennsylvania, Philadelphia, PA 19104.
sanjeev@cis.upenn.edu

**Abstract.** We investigate the hardness of approximating the longest path and the longest cycle in directed graphs on $n$ vertices. We show that neither of these two problems can be polynomial time approximated within $n^{1-\epsilon}$ for any $\epsilon > 0$ unless P = NP. In particular, the result holds for digraphs of constant bounded outdegree that contain a Hamiltonian cycle.

  - **Method I**, count the *number of RPS cycles*.
    - when k=3, we can compute by constructing $A_{i,j} = 1 \iff \phi_{i,j} > 0$, then

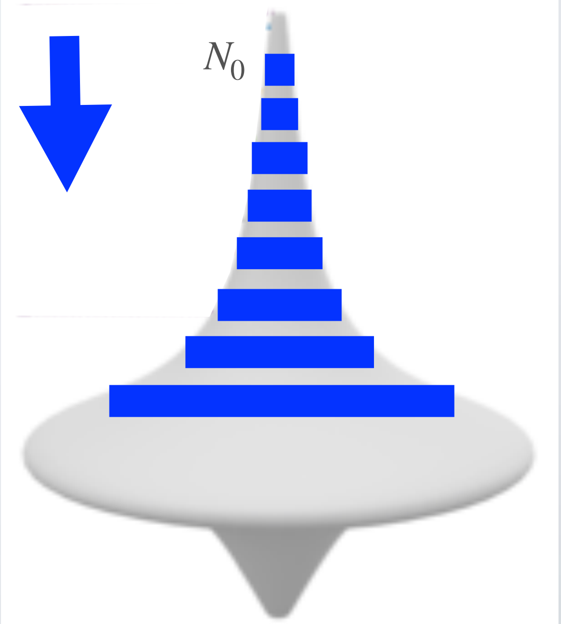
$$\mathrm{diag}(A^3)$$

  - **Method II**, at each transitivity level, we can measure the *Nash Clustering*

**Definition 3.** *Nash clustering* C *of the finite zero-sum symmetric game strategy* $\Pi$ *set by setting for each* $i \geq 1$*:* $N_{i+1} = \mathrm{supp}(\mathrm{Nash}(\mathbf{P}|\Pi \setminus \bigcup_{j \leq i} N_j))$ *for* $N_0 = \emptyset$ *and* $\mathrm{C} = (N_j : j \in \mathbb{N} \wedge N_j \neq \emptyset)$.

$$N_{i+1} = \mathrm{supp}\left(\mathrm{Nash}(\mathbf{P} \mid \Pi \setminus \cup_{j \leq i} N_j)\right)$$

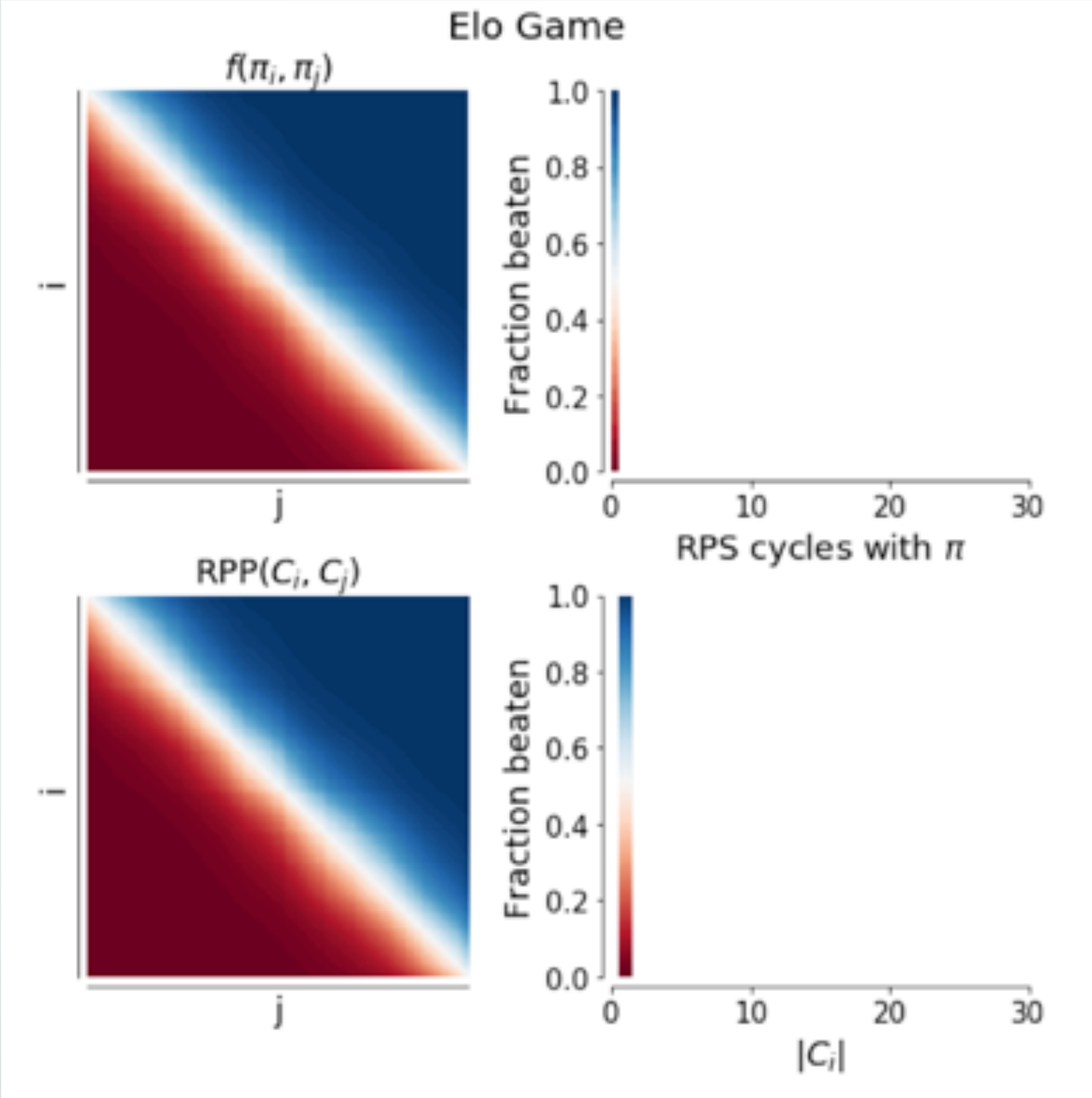
strategies that at the higher level of transitivity



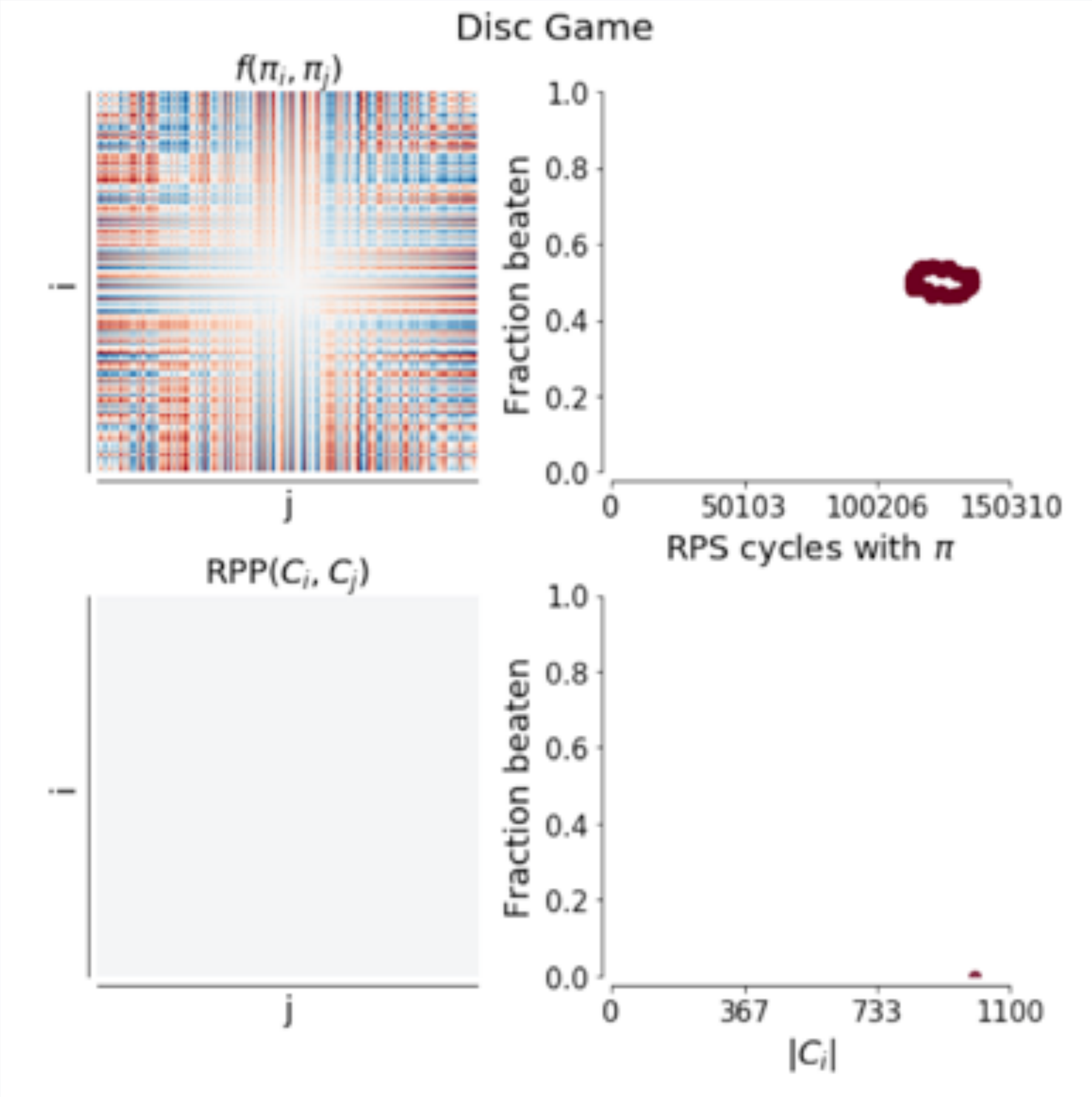
# Measuring the Non-Transitivity

- Some meta-game examples
  - each $\pi_i$ is an RL/DNN model, each $C_i$ is a Nash Cluster.
  - $\mathrm{RPP}\left(\Pi_A, \Pi_B\right) = \mathrm{Nash}\left(\mathbf{P}_{AB} \mid (A, B)\right)$

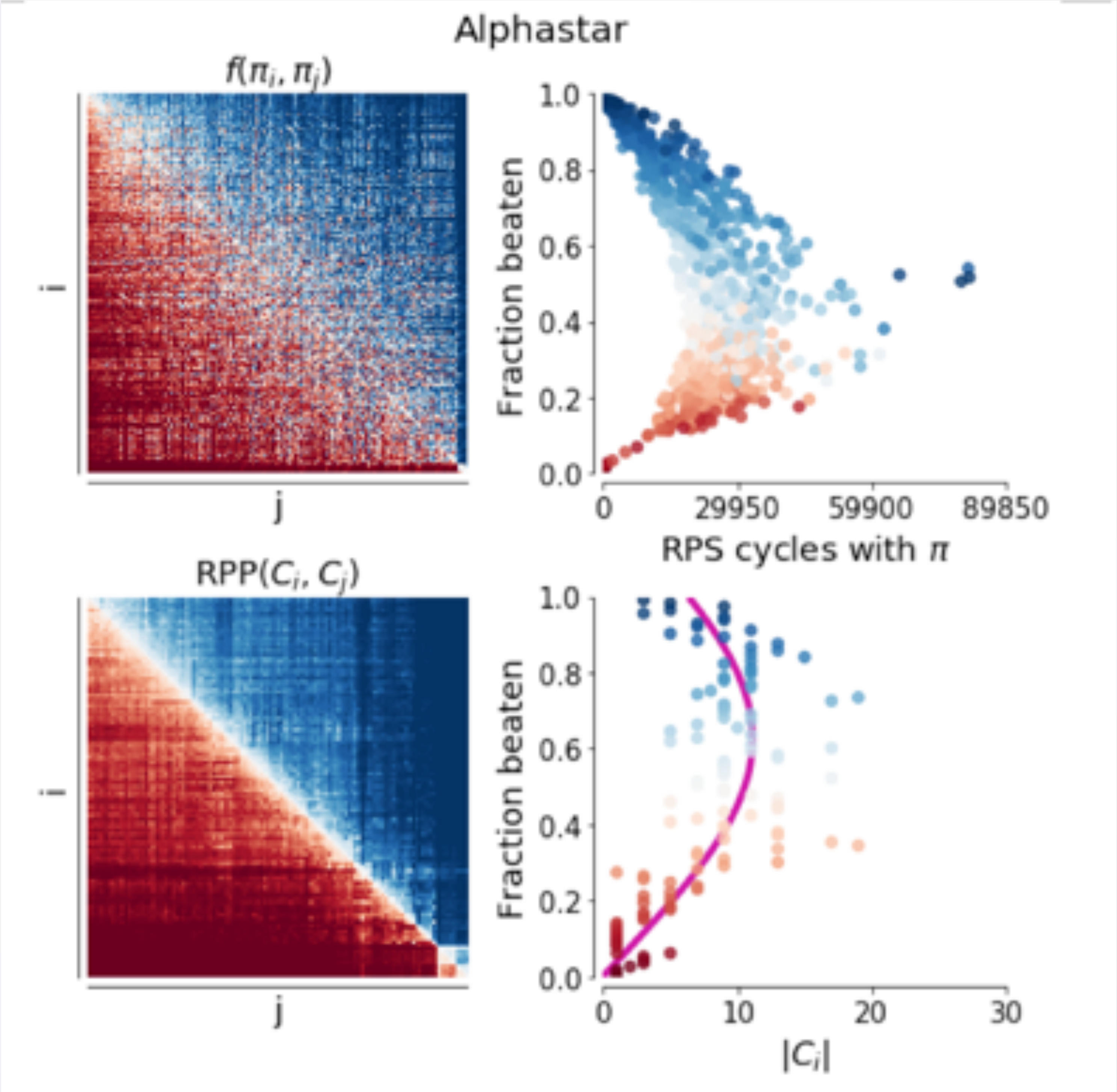
transitive games




non-transitive games

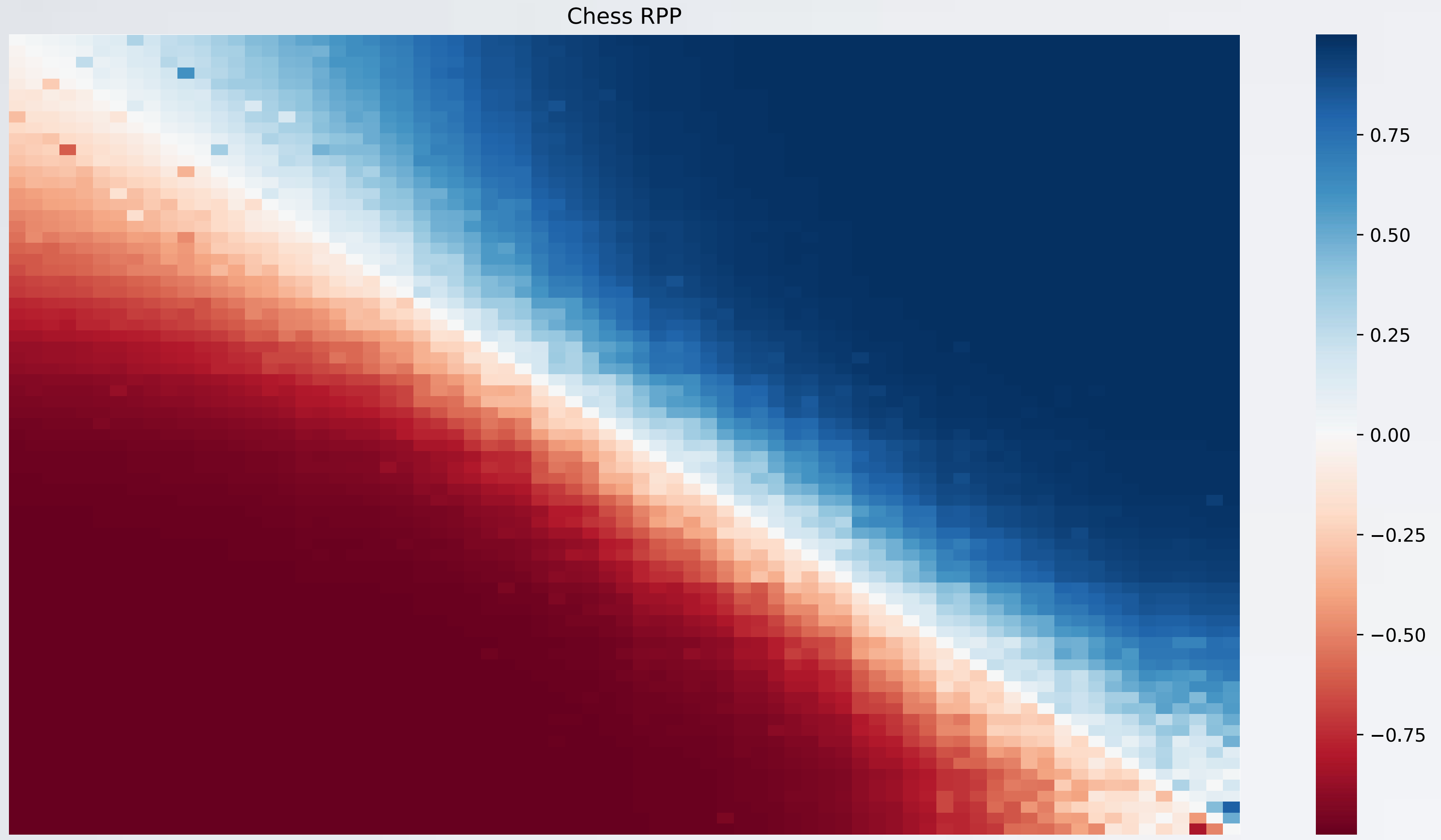


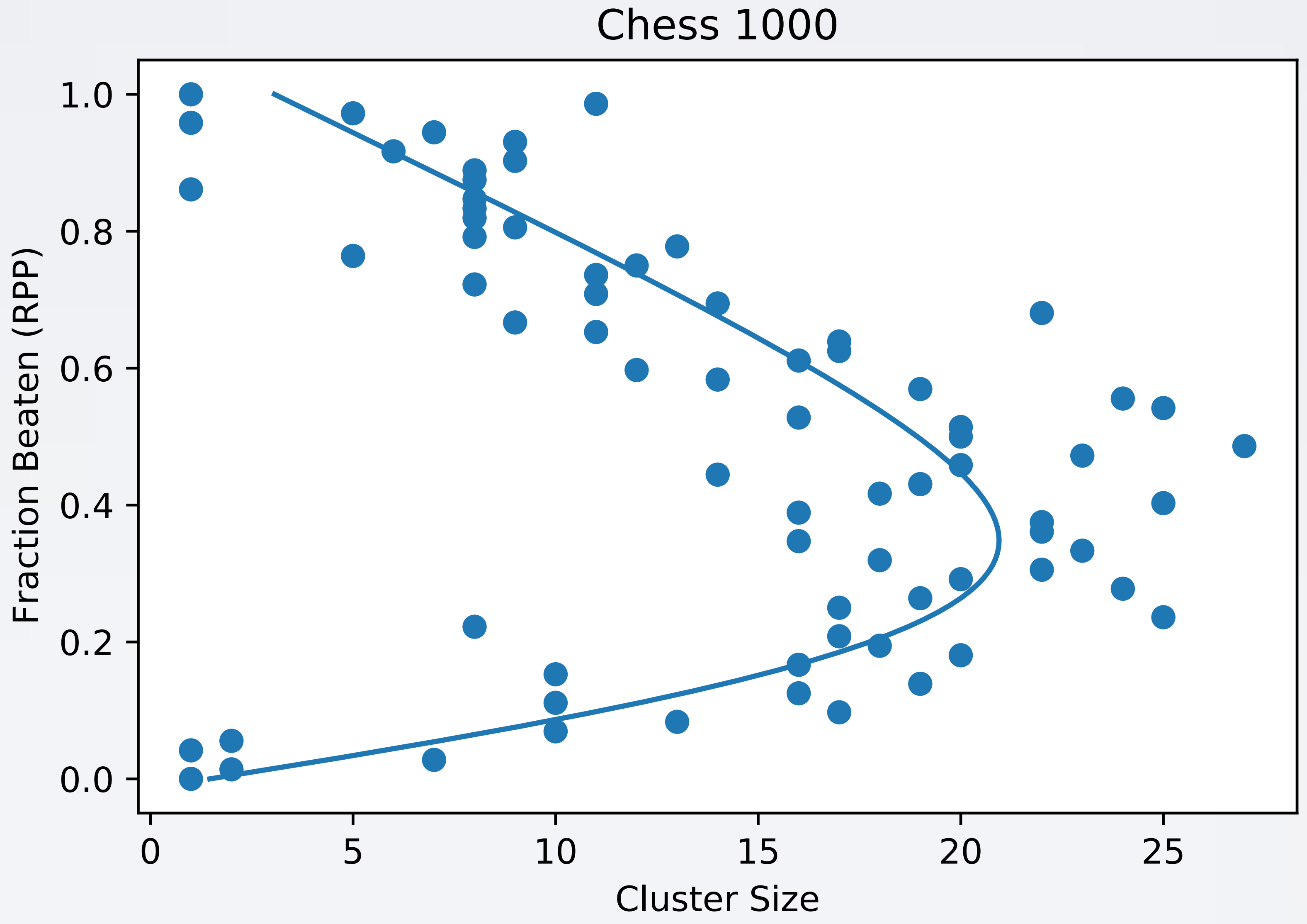

real-world games

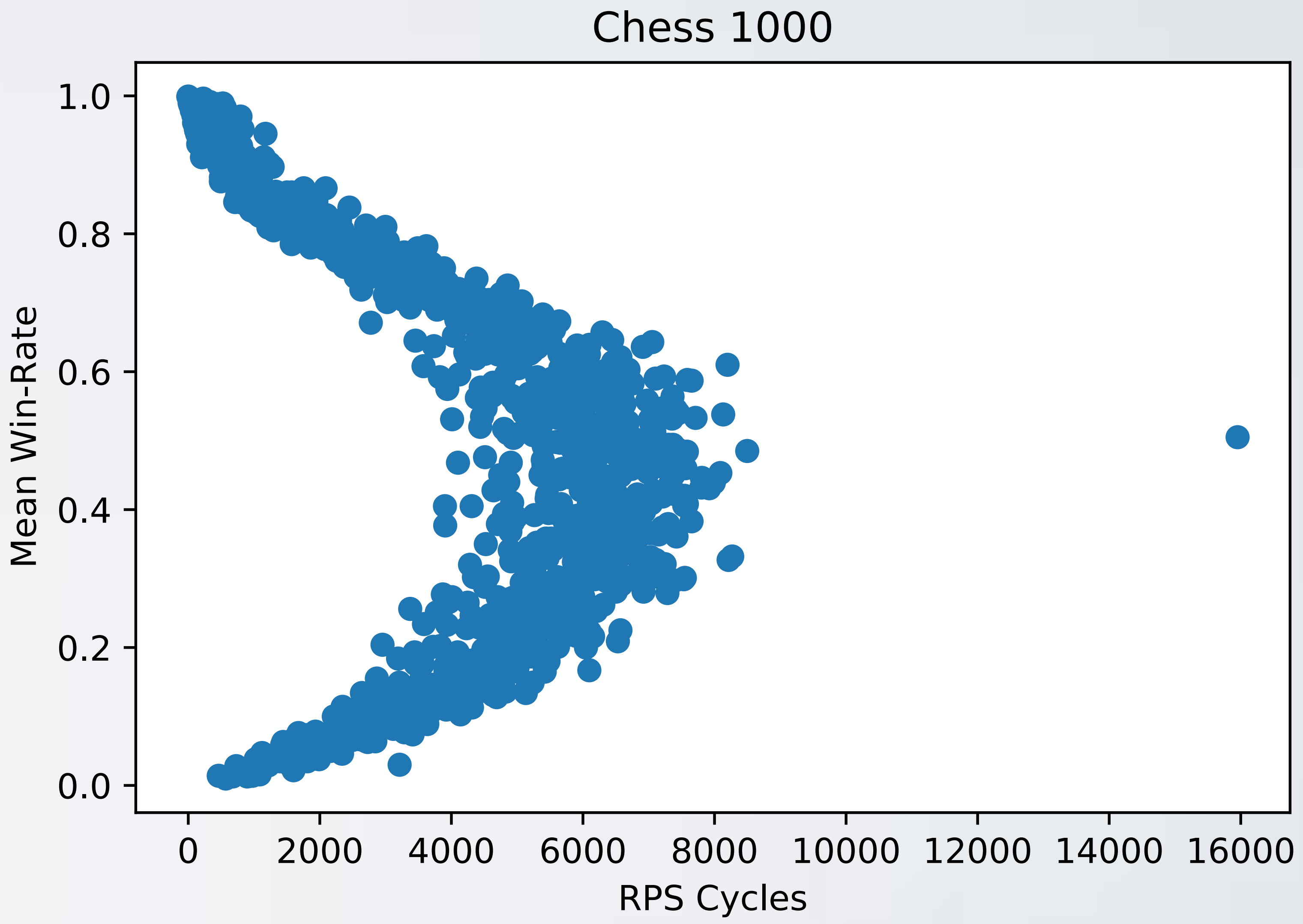


# Measuring the Non-Transitivity

- Real-world data set from human players on Chess
  - previous results are based on AI, now we study 1000 human players from Lichess
  - Chess presents the same spinning top pattern, which verifies the hypothesis



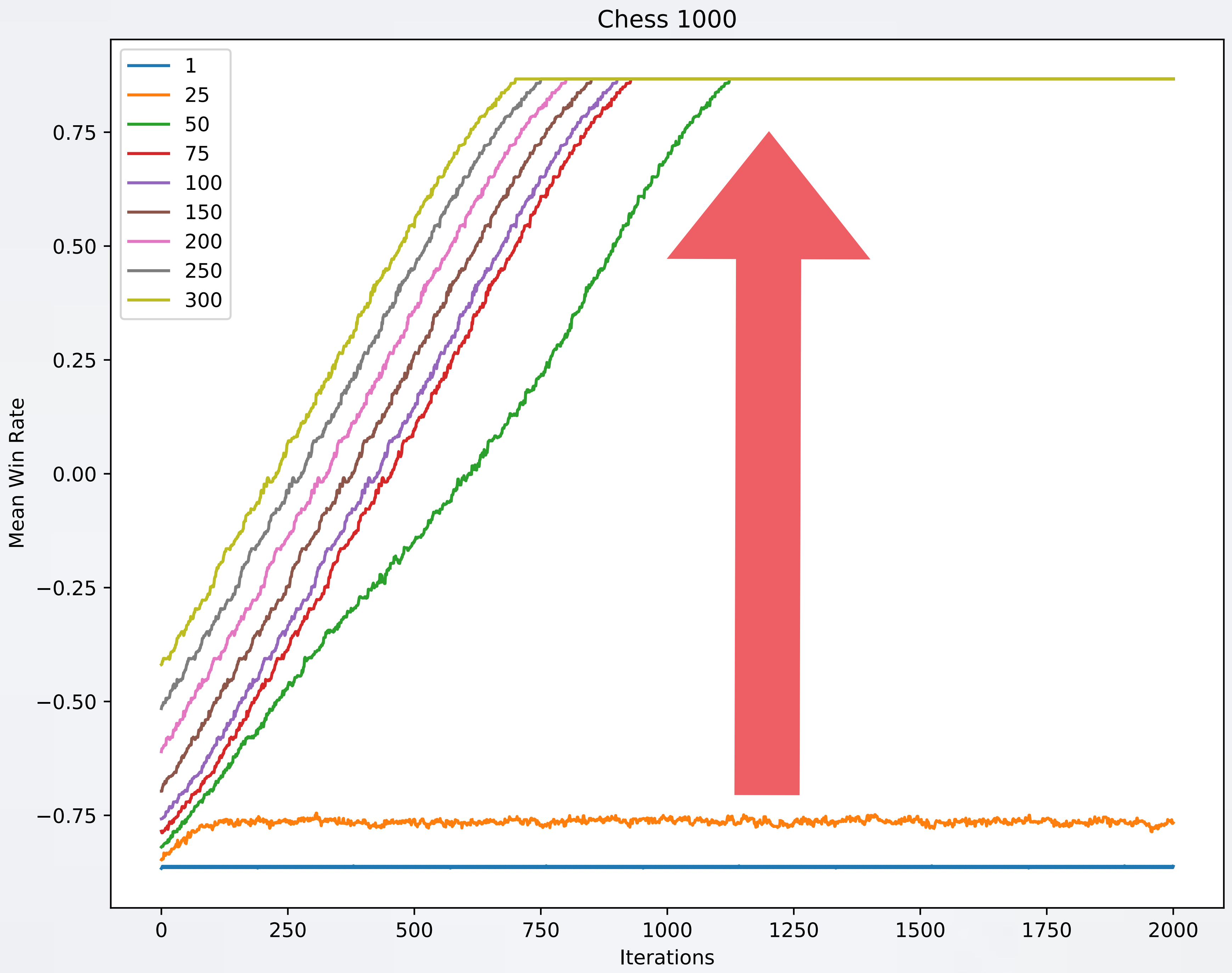

[Ricky Sanjaya]

# Understanding Non-Transitivity Helps Develop Algorithms !

- Topological structure at the policy space affects the efficiency of training algorithm.
  - for example, there is a reason why we need diversity in the policy space.

**Theorem 3.** *If at any point in time, the training population $\mathcal{P}^t$ includes any full Nash cluster $\mathrm{C}_i \subset \mathcal{P}^t$, then training against $\mathcal{P}^t$ by finding $\pi$ such that $\forall_{\pi_j \in \mathcal{P}^t} \mathbf{f}(\pi, \pi_j) > 0$ guarantees transitive improvement in terms of the Nash clustering $\exists_{k<i}\ \pi \in \mathrm{C}_k$.*

  - on chess, large population size (thus more diversity) will have a phase change in the strength !

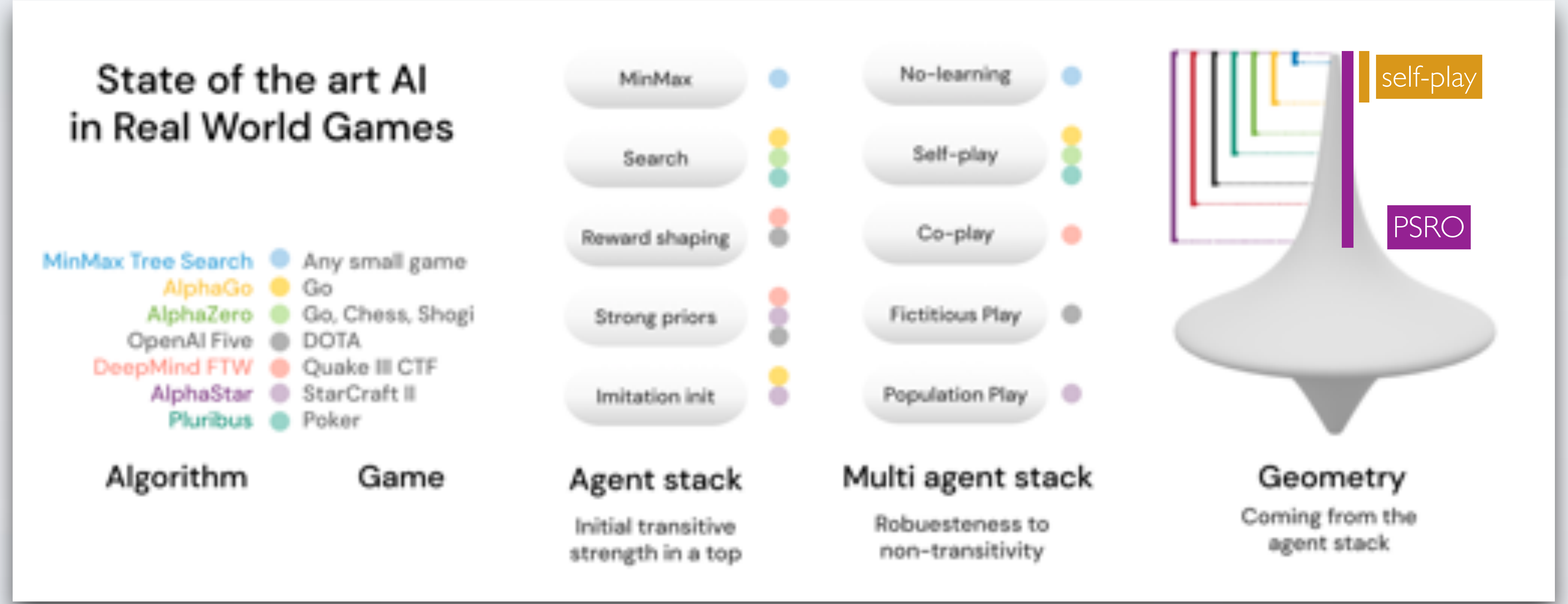


# Understanding Non-Transitivity Helps Develop Algorithms !

- Topological structure at the policy space affects the efficiency of training algorithm.
  - for example, there is a reason why we need diversity in the policy space.

**Theorem 3.** *If at any point in time, the training population $\mathcal{P}^t$ includes any full Nash cluster $C_i \subset \mathcal{P}^t$, then training against $\mathcal{P}^t$ by finding $\pi$ such that $\forall_{\pi_j \in \mathcal{P}^t} \mathbf{f}(\pi, \pi_j) > 0$ guarantees transitive improvement in terms of the Nash clustering $\exists_{k<i}\ \pi \in C_k$.*

  - similarly, for other techniques in the stack, there is an effective domain where they can be applied.




[Czarnecki 2020]

# Contents

- **What is Non-Transitivity in Games**
- **How to Measure Non-Transitivity**
- **Solutions: Double Oracle / PSRO Methods**
- **Recent advances: Diverse-PSRO**
- **Recent advances: Online-PSRO**
- **Recent advances: Auto-PSRO**

# Fictitious Play [Brown 1951]

- Maintain a belief over the historical actions that the opponent has played, and the learning agent then takes the best response to this empirical distribution.

$$a_i^{t,*} \in \mathbf{BR}_i\left(p_{-i}^t = \frac{1}{t}\sum_{\tau=0}^{t-1} \mathcal{I}\left\{a_{-i}^{\tau} = a, a \in \mathbb{A}\right\}\right)$$

$$p_i^{t+1} = \left(1 - \frac{1}{t}\right)p_i^t + \frac{1}{t}a_i^{t,*}, \text{ for all } i$$

- It guarantees to converge, in terms of the Nash value, in two-player zero-sum games, and, potential games which include fully-cooperative games.

- Examples:

| | | Player 2 | |
|---|---|---|---|
| | | $a$ | $b$ |
| Player 1 | $A$ | (1,1) | (0,0) |
| | $B$ | (0,0) | (1,1) |

| $t$ | $p_1^t$ | $p_2^t$ | $a_1^t$ | $a_2^t$ |
|---|---|---|---|---|
| 0 | (3/4, 1/4) | (1/4, 3/4) | B | a |
| 1 | (3/4, 5/4) | (5/4, 3/4) | A | b |
| 2 | (7/4, 5/4) | (5/4, 7/4) | B | a |
| 3 | (7/4, 9/4) | (9/4, 7/4) | A | b |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| ∞ | (1/2, 1/2) | (1/2, 1/2) | | |

# Generalised Weakened Fictitious Play [Leslie 2006]

- It releases the FP by allowing approximate best response and perturbed average strategy updates, while maintaining the same convergence guarantee if conditions met.

$$\mathbf{Br}_i^{\epsilon}(p_{-i}) = \left\{ p_i : R_i\big(p_i, p_{-i}\big) \geq R_i\Big(\mathbf{Br}_i(p_{-i}), p_{-i}\Big) - \epsilon \right\}$$

$$p_i^{t+1} = \left(1 - \alpha^{t+1}\right) p_i^t + \alpha^{t+1}\left(\mathbf{Br}_i^{\epsilon}(p_{-i}) + M_i^{t+1}\right), \text{ for all } i$$

$t \to \infty, \alpha_t \to 0, \epsilon^t \to 0, \sum_{t=1} \alpha^t = \infty, \{M^t\}$ meets $\lim_{t\to\infty} \sup_k \left\{ \left\| \sum_{i=t}^{k-1} \alpha^{i+1} M^{i+1} \right\| \text{ s.t. } \sum_{i=t}^{k-1} \alpha^{i+1} \leq T \right\} = 0$

- Recovers normal Fictitious Play when $\alpha^t = 1/t, \epsilon_t = 0, M_t = 0.$
- **Why important:** it allows us to use a broad class of best responses such as RL algorithms, and also, the policy exploration, e.g., the entropy term in soft-Q learning, can now be considered through the $M$ term.

# Double Oracle [McMahan 2003]

- Double Oracle best responds to the opponent's Nash equilibrium at each iteration.
- To solve the game before seeing all pure strategies (not all of them are in Nash), much faster than LP, but In the worst-case scenario, it recovers to solve the original game.

**Algorithm 1** Double Oracle (McMahan et al., 2003)

```
Input: A set Π, C strategy set of players
Π_0, C_0: initial set of strategies
for t = 1 to ∞ do
    if Π_t ≠ Π_{t−1} or C_t ≠ C_{t−1} then
        Solve the NE of the subgame G_t:
        (π*_t, c*_t) = arg min_{π∈Δ_{Π_t}} arg max_{c∈Δ_{C_t}} π^⊤ A c
        Find the best response a_{t+1} and c_{t+1} to (π*_t, c*_t):
                a_{t+1} = arg min_{a∈Π} a^⊤ A c*_t
                c_{t+1} = arg max_{c∈C} π*_t^⊤ A c
        Update Π_{t+1} = Π_t ∪ {a_{t+1}}, C_{t+1} = C_t ∪ {c_{t+1}}
    else if Π_t = Π_{t−1} and C_t = C_{t−1} then
        Terminate
    end if
end for
```

- **iteration 0**: restricted game R vs R
- **iteration 1**:
  - solve Nash of restricted game (1, 0, 0) , (1, 0, 0)
  - unrestricted $\mathbf{Br}^1, \mathbf{Br}^2$ = P, P
- **iteration 2**:
  - solve Nash of restricted games (0, 1, 0) , (0, 1, 0)
  - unrestricted $\mathbf{Br}^1, \mathbf{Br}^2$ = S, S
- **iteration 3**:
  - solve Nash of restricted game (1/3, 1/3, 1/3) , (1/3, 1/3, 1/3)
- **iteration 4**: no new response, END
  - output (1/3, 1/3, 1/3)

| | R | P | S |
|---|---|---|---|
| R | 0 | -1 | 1 |
| P | 1 | 0 | -1 |
| S | -1 | 1 | 0 |

# Double Oracle [McMahan 2003]

- It guarantees to converge to Nash equilibrium in two-player zero-sum games, and coarse correlated equilibrium in multi-player general-sum games.
- **Convergence proof:**
  - DO finally recovers to solve the whole game
- **Correctness proof:**
  - suppose DO stops at the j-th sub-game (i.e., no new best responses are added)
  - $\forall p, V(p, q_j) \geq v \Rightarrow \forall p, \max_q V(p, q) \geq v$

    $\forall q, V(p_j, q) \leq v \Rightarrow \max_q V(p_j, q) \leq v$

    $\Rightarrow \forall p, \max_q V(p_j, q) \leq max_q(p, q)$

    $p_j$ must be the minimax optimal,

    $q_j$ vice versa

# Policy Space Response Oracle = DO + RL Oracle

- A generalisation of double oracle methods on meta-games, with the best responser is implemented through deep RL algorithms.
- A meta-game is $(\Pi, U, n)$ where $\Pi = (\Pi_1, \ldots, \Pi_n)$ is the set of policies for each agent and $U : \Pi \rightarrow \mathbb{R}^n$ is the reward values for each agent given a joint strategy profile.
- $\sigma_{-i}$ is distribution over $(\Pi_1^0, \ldots, \Pi_1^T)$, a.k.a meta-solver
- PSRO generalises all previous methods by varying $\sigma_{-i}$.
  - independent learning: $\sigma_{-i} = (0,...,0,0,1)$
  - self-play: $\sigma_{-i} = (0,...,0,1,0)$
  - fictitious play: $\sigma_{-i} = (1/T, 1/T, \ldots, 1/T, 0)$
  - PSRO: $\sigma_{-i} = \textbf{Nash}\left(\Pi^{T-1}, U\right)$ or $\textbf{RD}\left(\Pi^{T-1}, U\right)$

**Algorithm 1:** Policy-Space Response Oracles

**input** :initial policy sets for all players $\Pi$
Compute exp. utilities $U^{\Pi}$ for each joint $\pi \in \Pi$
Initialize meta-strategies $\sigma_i = \text{UNIFORM}(\Pi_i)$
**while** *epoch e in* $\{1, 2, \cdots\}$ **do**
  **for** *player* $i \in [[n]]$ **do**
    **for** *many episodes* **do**
      select opponent policies — Sample $\pi_{-i} \sim \sigma_{-i}$
      compute the best response — Train oracle $\pi'_i$ over $\rho \sim (\pi'_i, \pi_{-i})$
    augment strategy pool — $\Pi_i = \Pi_i \cup \{\pi'_i\}$
  expand the payoff matrix — Compute missing entries in $U^{\Pi}$ from $\Pi$
  solve the new meta game — Compute a meta-strategy $\sigma$ from $U^{\Pi}$
Output current solution strategy $\sigma_i$ for player $i$

# PSRO-rN [Balduzzi 2019]

**key changes:** only selecting opponents that I have already won over (i.e. rectifying the Nash)

$$\mathbf{v}_{t+1} \leftarrow \text{oracle}\left(\mathbf{v}_t\,,\ \sum_{\mathbf{w}_i \in \mathfrak{P}_t} \mathbf{p}_t[i] \cdot \lfloor \phi_{\mathbf{w}_i}(\bullet) \rfloor_+ \right)$$

**Proposition 6.** *If* $\mathbf{p}$ *is a Nash equilibrium on* $\mathbf{A}_{\mathfrak{P}}$ *and* $\sum_i p_i \phi_{\mathbf{w}_i}(\mathbf{v}) > 0$, *then adding* $\mathbf{v}$ *to* $\mathfrak{P}$ *strictly enlarges the empirical gamescape:* $\mathcal{G}_{\mathfrak{P}} \subsetneq \mathcal{G}_{\mathfrak{P}\cup\{\mathbf{v}\}}$.

**Algorithm 4** Response to rectified Nash (PSRO$_{\text{rN}}$)

**input:** population $\mathfrak{P}_1$
**for** $t = 1, \ldots, T$ **do**
  $\mathbf{p}_t \leftarrow$ Nash on $\mathbf{A}_{\mathfrak{P}_t}$
  **for** agent $\mathbf{v}_t$ with positive mass in $\mathbf{p}_t$ **do**
    $\mathbf{v}_{t+1} \leftarrow \text{oracle}\left(\mathbf{v}_t, \sum_{\mathbf{w}_i \in \mathfrak{P}_t} \mathbf{p}_t[i] \cdot \lfloor \phi_{\mathbf{w}_i}(\bullet) \rfloor_+\right)$
  **end for**
  $\mathfrak{P}_{t+1} \leftarrow \mathfrak{P}_t \cup \{\mathbf{v}_{t+1} : \text{updated above}\}$
**end for**
**output:** $\mathfrak{P}_{T+1}$

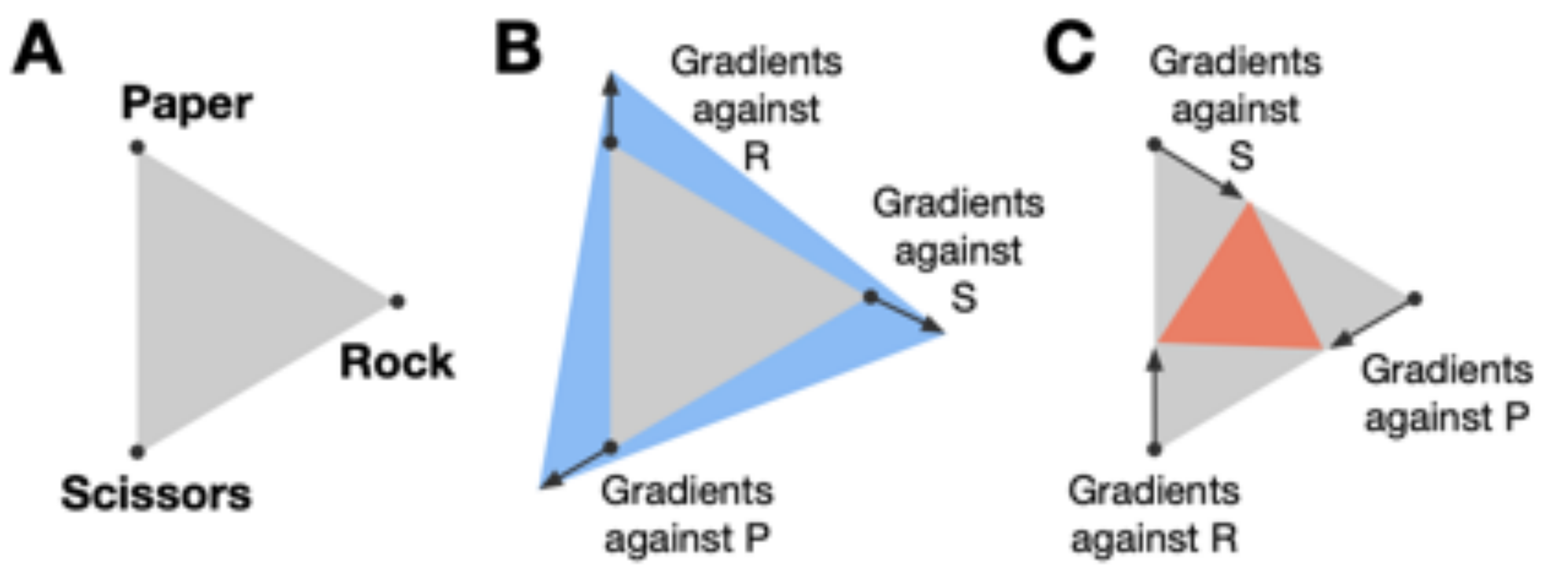
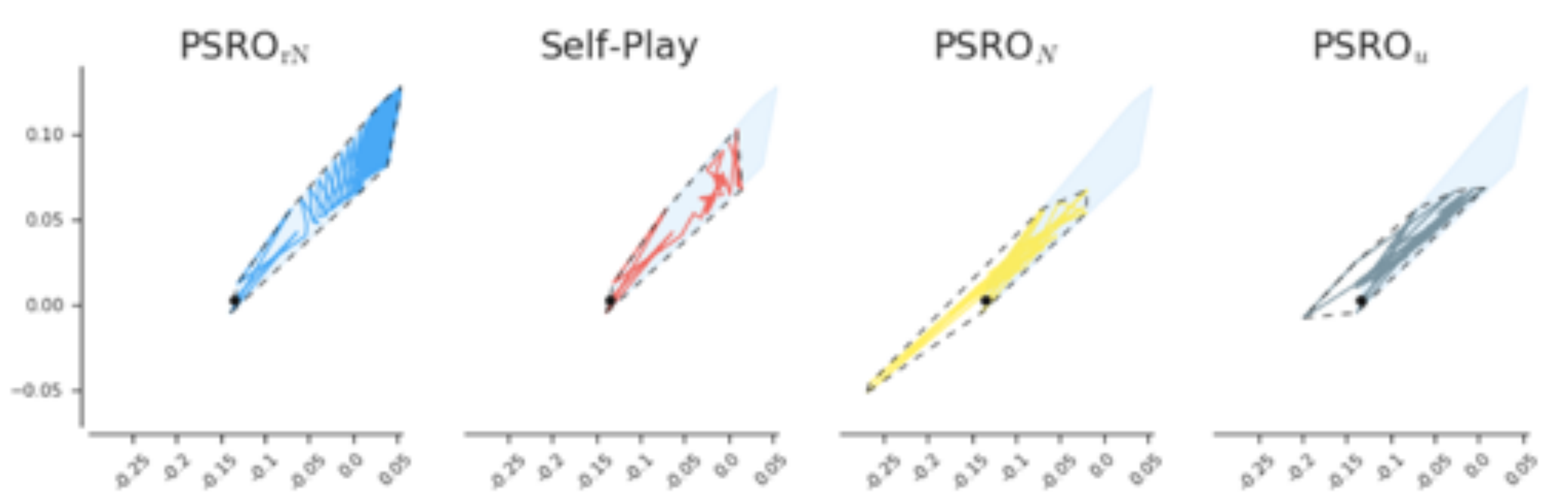



*Figure 3.* **A:** Rock-paper-scissors. **B:** Gradient updates obtained from PSRO$_{\text{rN}}$, amplifying strengths, grow gamescape (gray to blue). **C:** Gradients obtained by optimizing agents to reduces their losses shrink gamescape (gray to red).

**Intuition: maintaining strength can keep exploring larger and large strategy space (强者恒强/马太效应)**



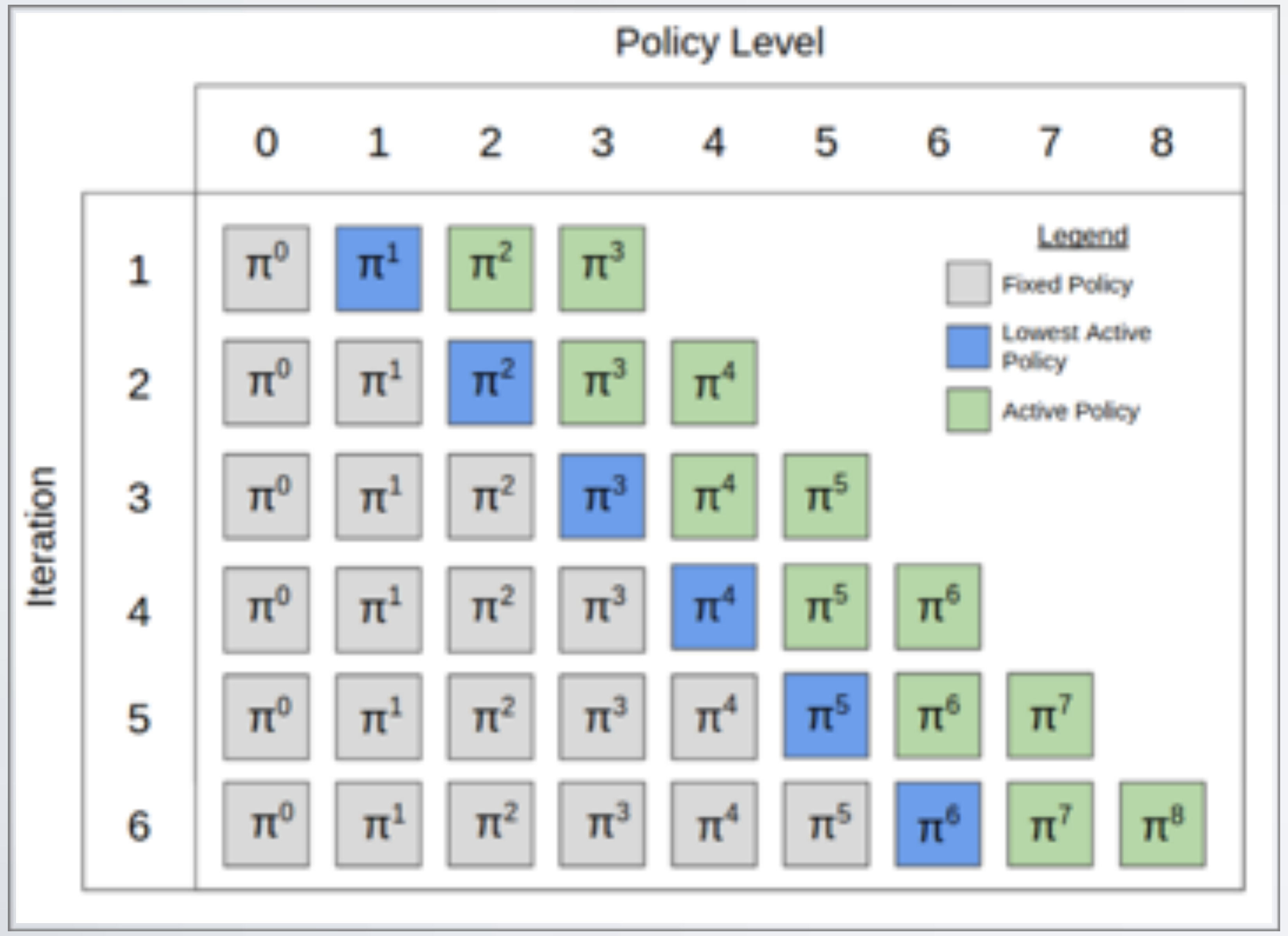

**diversity can also help explore the strategy space more efficiently and effectively**

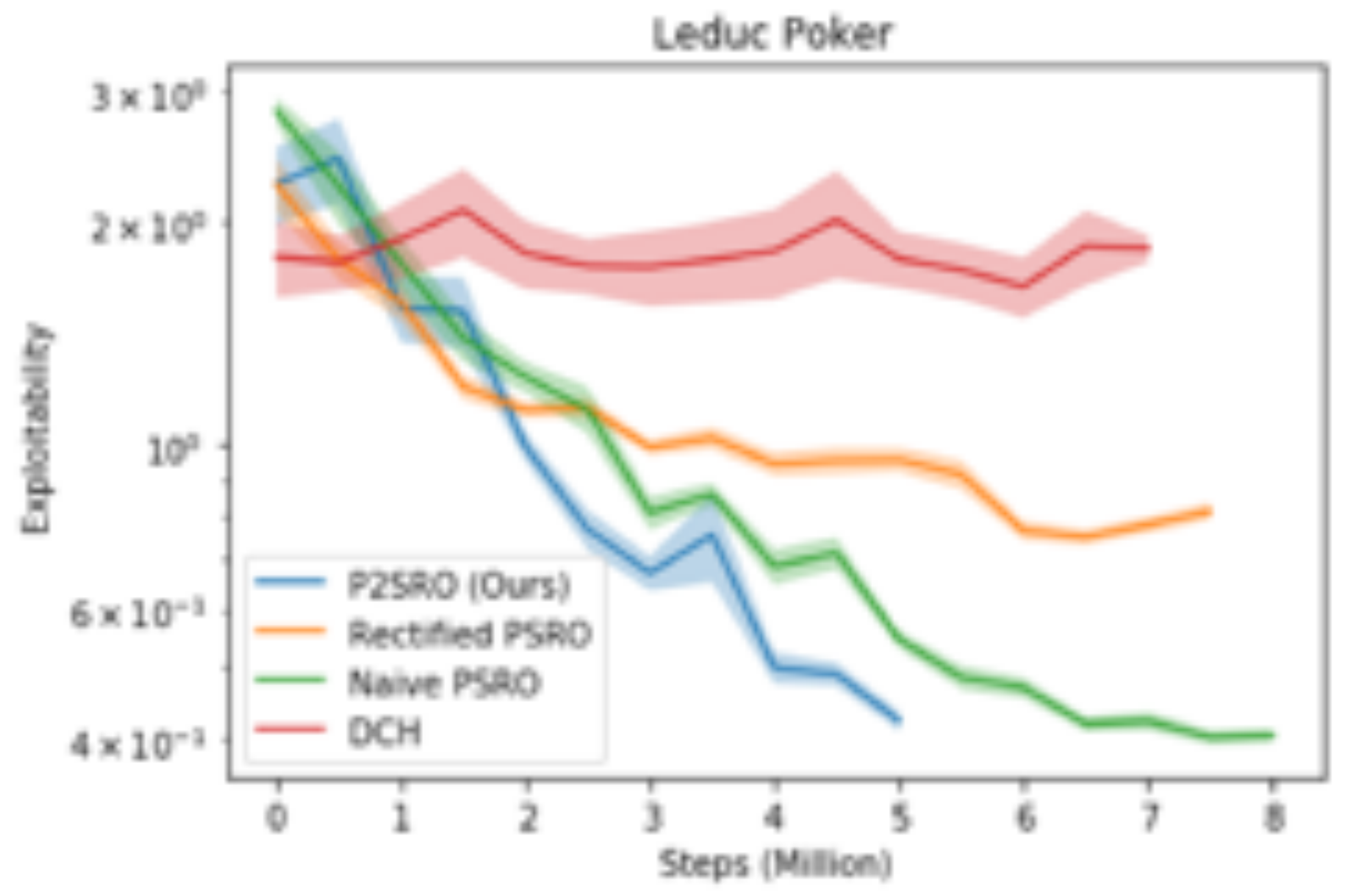
# Pipeline PSRO [McAleer 2020]

1.A counter-example that PSRO-Rectified-Nash could fail (there is really no one diversity metric that works).

$$\begin{bmatrix} 0 & -1 & 1 & -\frac{2}{5} \\ 1 & 0 & -1 & -\frac{2}{5} \\ -1 & 1 & 0 & -\frac{2}{5} \\ \frac{2}{5} & \frac{2}{5} & \frac{2}{5} & 0 \end{bmatrix}$$

2. Diversity can came from training more best-response policies!



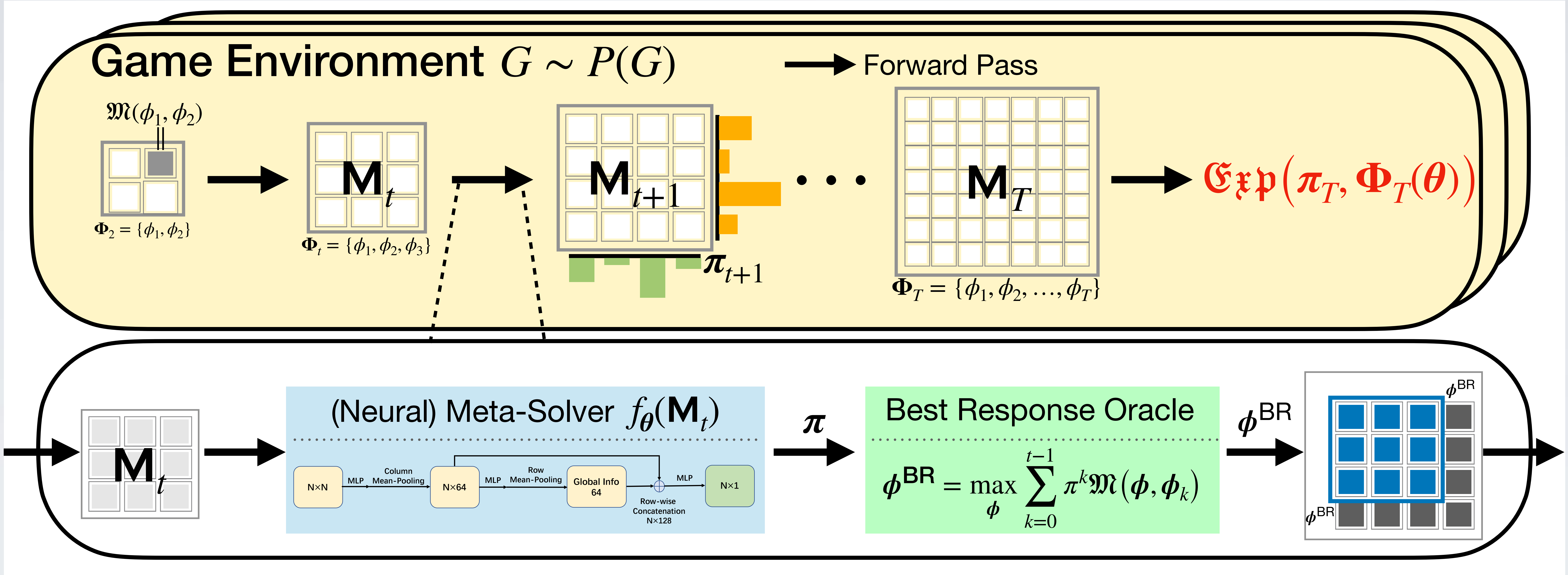
(a) Leduc poker

| Name | P2SRO Win Rate vs. Bot |
|---|---|
| Asmodeus | 81% |
| Celsius | 70% |
| Vixen | 69% |
| Celsius1.1 | 65% |
| **All Bots Average** | **71%** |

Table 1: Barrage P2SRO Results vs. Existing Bots

Game size: $10^{50}$

# PSRO Incorporate Many Variants




Elo rating
Nash equilibrium
Replicator dynamics
$\alpha$-Rank/$\alpha^\alpha$-Rank

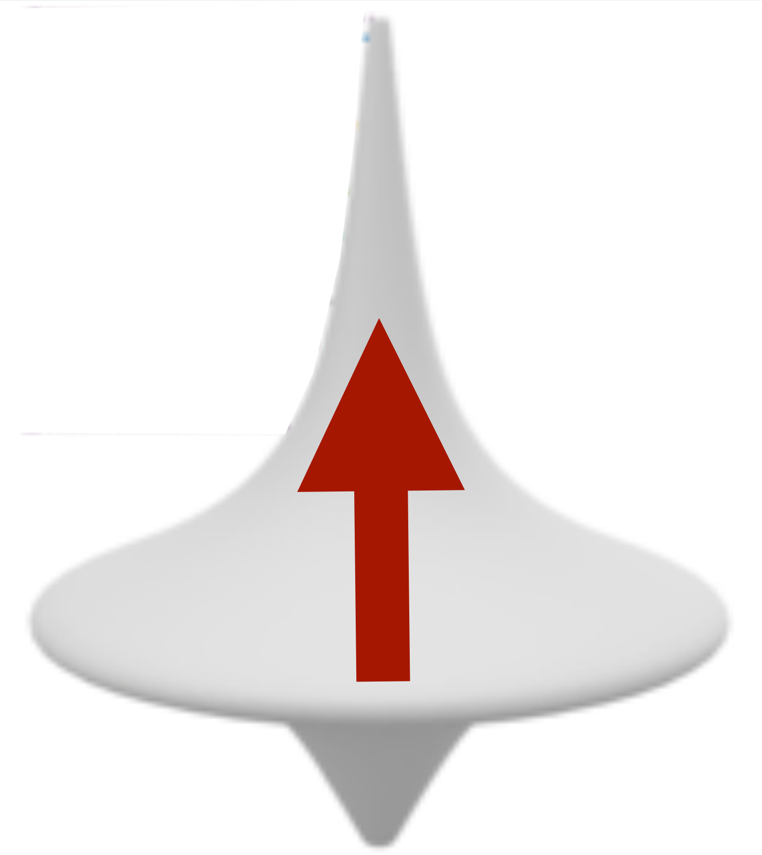
iterated best response
fictitious play
double oracle
PSRO
PSRO-Nash
PSRO-Rectified-Nash

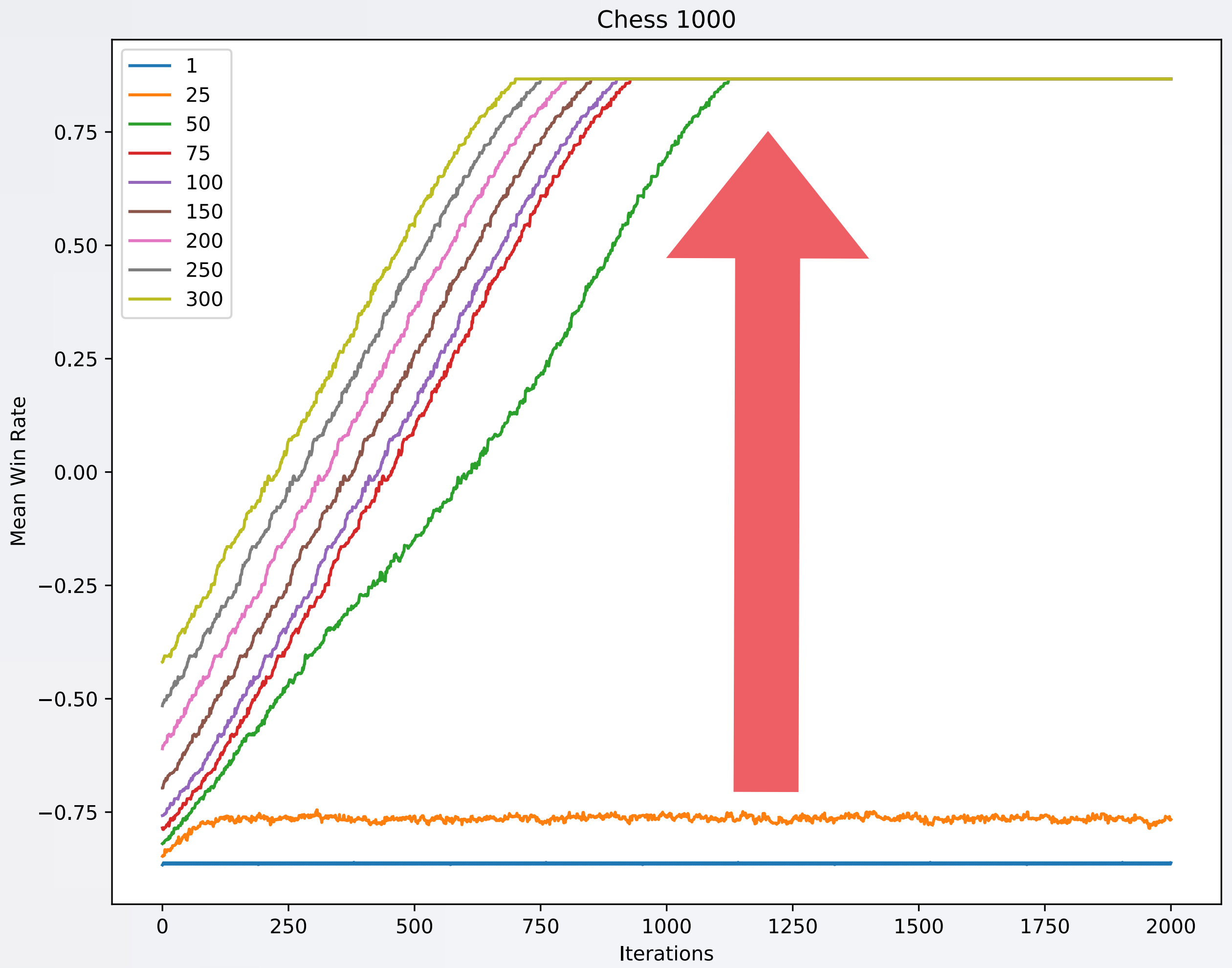
# Contents

# Why Modelling Diversity is Critical ?

- Diversity matters because the more diverse the strategy pool, the less un-exploitable. Promoting diversity can help you walk out of the in-transitive region faster.

**Theorem 3.** *If at any point in time, the training population* $\mathcal{P}^t$ *includes any full Nash cluster* $\mathrm{C}_i \subset \mathcal{P}^t$, *then training against* $\mathcal{P}^t$ *by finding* $\pi$ *such that* $\forall_{\pi_j \in \mathcal{P}^t} \mathbf{f}(\pi, \pi_j) > 0$ *guarantees transitive improvement in terms of the Nash clustering* $\exists_{k<i}\ \pi \in \mathrm{C}_k$.







**Diverse Auto-Curriculum is Critical for Successful Real-World Multiagent Learning Systems***

Blue Sky Ideas Track


Yaodong Yang†
University College London
Huawei R&D U.K.

Jun Luo
Huawei Canada

Ying Wen
Shanghai Jiao Tong University

Oliver Slumbers
University College London

Daniel Graves
Huawei Canada

Haitham Bou Ammar
Huawei R&D U.K.

Jun Wang
University College London
Huawei R&D U.K.

Matthew E. Taylor
University of Alberta
Alberta Machine Intelligence Institute


- In real-world applications, you want policies to be diverse enough, covering different skill levels. This is a realistic need from autonomous driving and gaming AI applications.

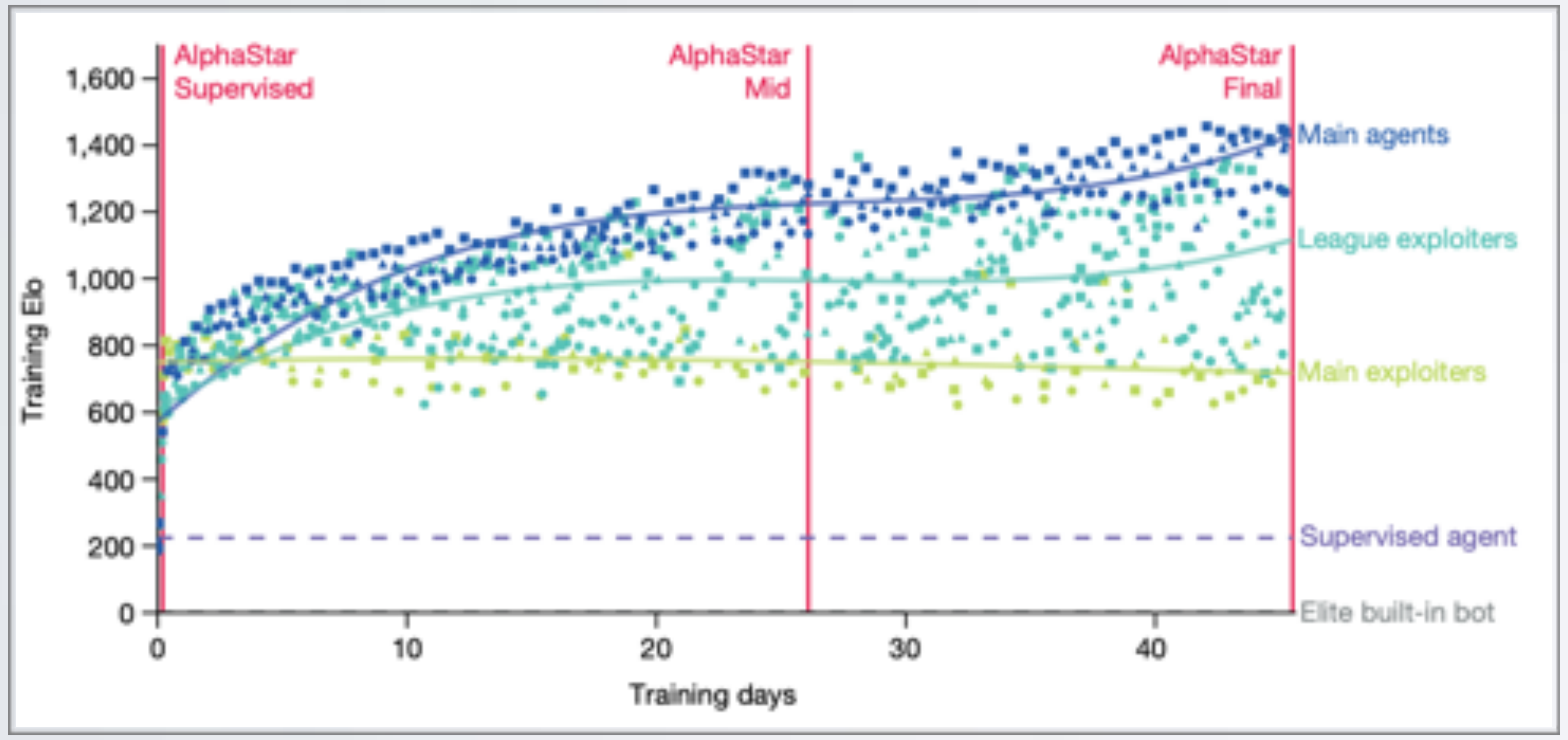
# Promoting Diversity in AlphaStar

1.Most diversity still comes from human data !

$$\pi_\theta\left(a_t \mid s_t, z\right) = \mathbb{P}\left[a_t \mid s_t, z\right]$$

The policy is also conditioned on a statistic $z$ that summarises a strategy sampled from human data

2.League Training: add different levels of exploiters (main exploiters and league exploiters) to the population.



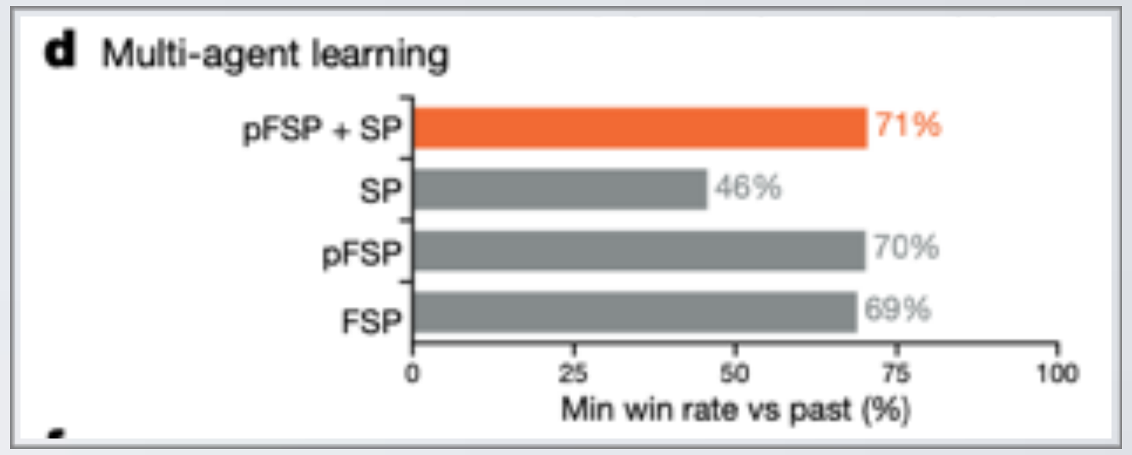

Main exploiter:
exploit main agents

League exploiter:
exploit the whole league

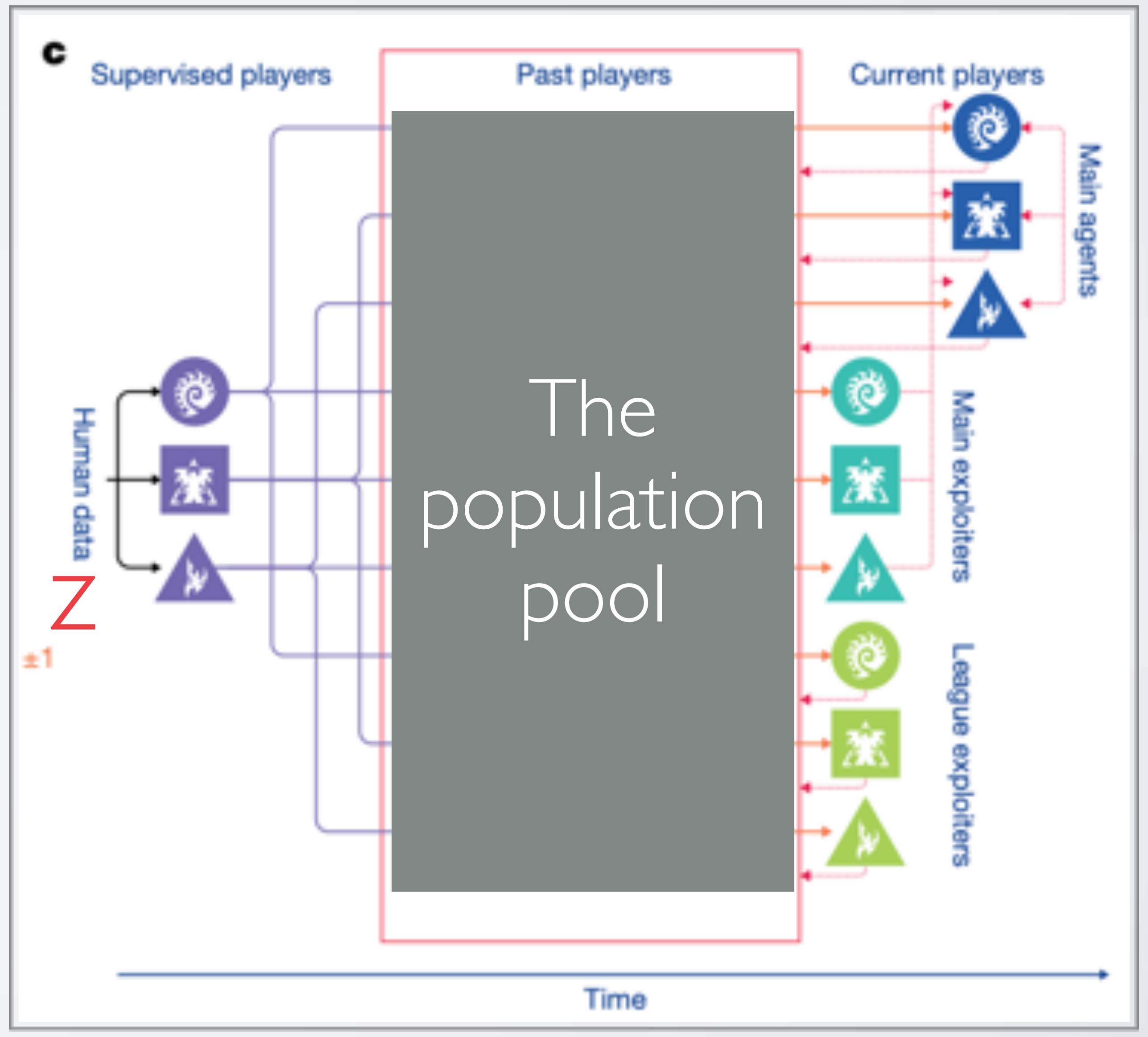
League exploiter resets every two days, and it still can improve in Elo score!
This also tells that StarCraft has strong non-transitivity in the policy space!

3.Prioritised fictitious self-play (PFSP): focus more on the unbeatable opponents. Select opponent B according to the score of

$$\frac{\mathbb{P}[B \text{ beats } A]}{\sum_{C\in\mathscr{C}} \mathbb{P}[C \text{ beats } A]}$$




Put three tricks together



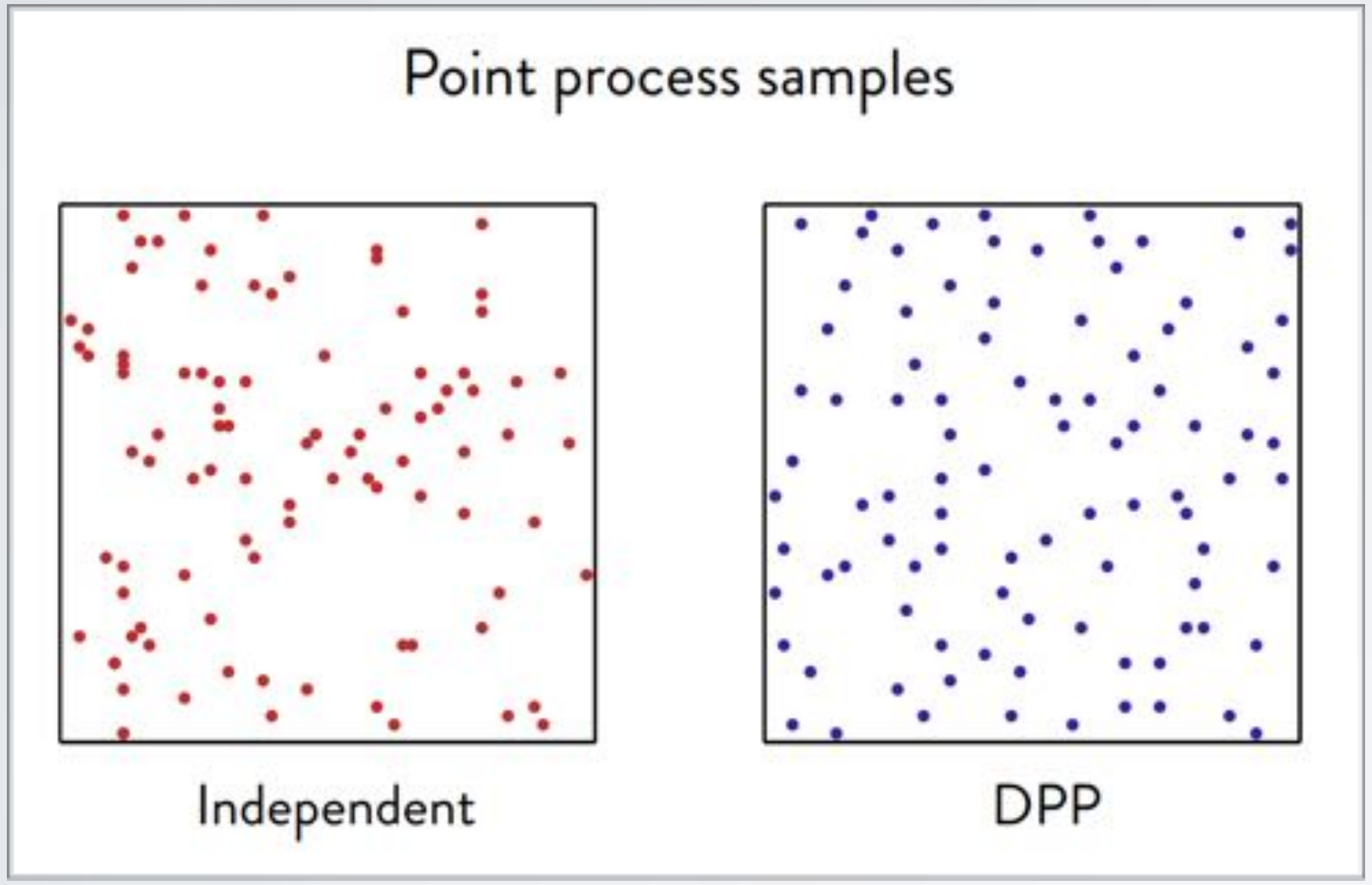
# Recent Advance (1): Diverse-PSRO

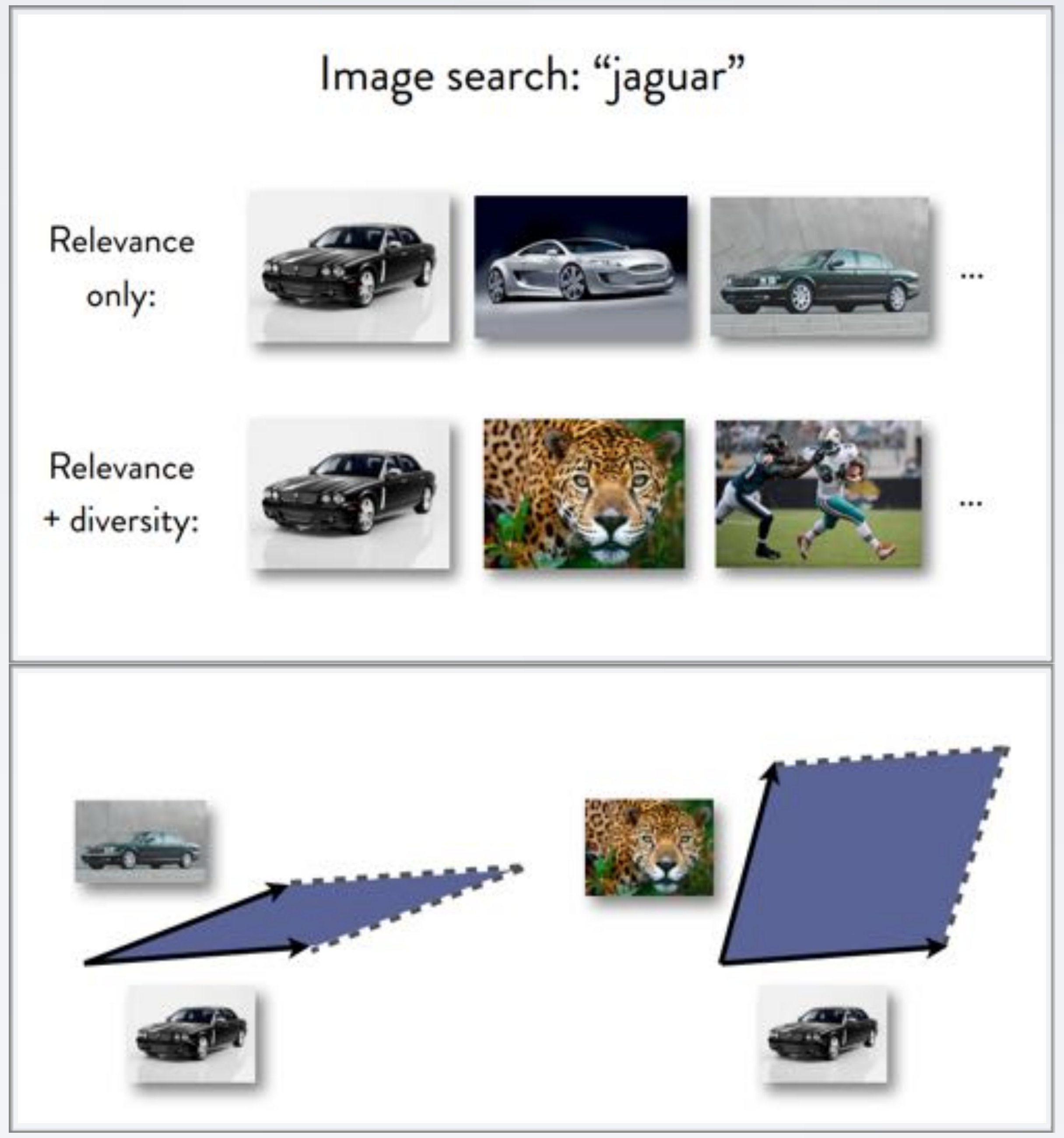
**Modelling Behavioural Diversity for Learning in Open-Ended Games**

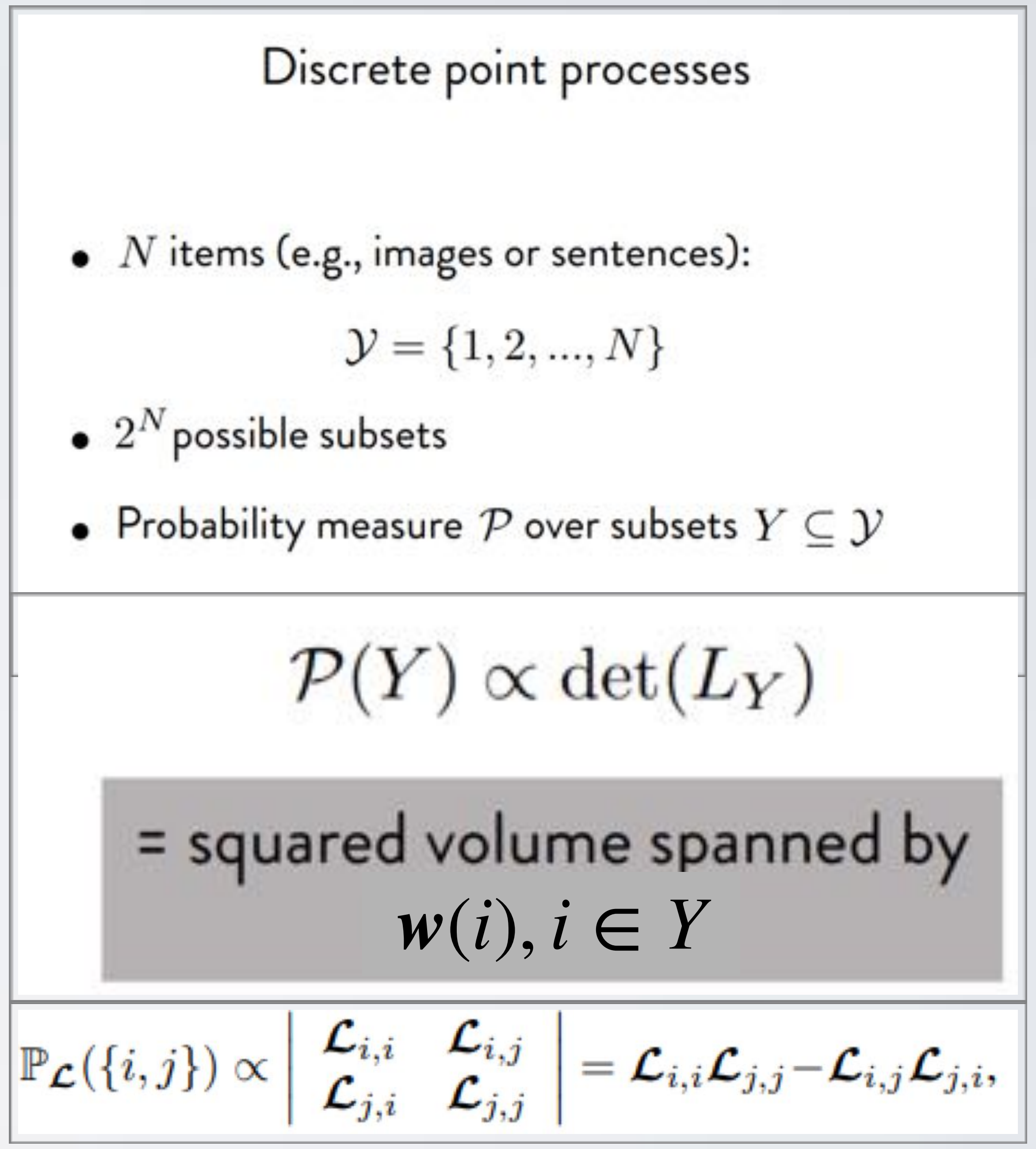
Nicolas Perez Nieves *[1,2] Yaodong Yang *[1,3] Oliver Slumbers *[3] David Henry Mguni [1] Ying Wen [3] Jun Wang [1,3]

1. Go back to the first principle: diversity should be defined on the sense of orthogonality.

- Determinantal Point Process [Alex Kulesza 2013] : a point process parameterised by a distance kernel.




$$\text{DPP}(\mathscr{L}) := \mathbb{P}_{\mathscr{L}}(\boldsymbol{Y} = Y) \propto \det(\mathscr{L}_Y) = \text{Vol}^2\big(\{\boldsymbol{w}_i\}_{i \in Y}\big)$$

# Recent Advance (1): Diverse-PSRO

1.Go back to the first principle: diversity should be defined on the sense of orthogonality.

- Policy diversity can be measured through their pay-off vectors, i.e., $\mathscr{L}_{\mathbb{S}} = \mathbf{M}\mathbf{M}^{\top}$.
- The expected cardinality of DPP is the diversity metric.

$$\text{Diversity}\,(\mathbb{S}) = \mathbb{E}_{\mathbf{Y}\sim\mathbb{P}_{\mathscr{L}}}[\,|\,\mathbf{Y}\,|\,] = \mathrm{Tr}\left(\mathbf{I} - \left(\mathscr{L}_{\mathbb{S}} + \mathbf{I}\right)^{-1}\right)$$

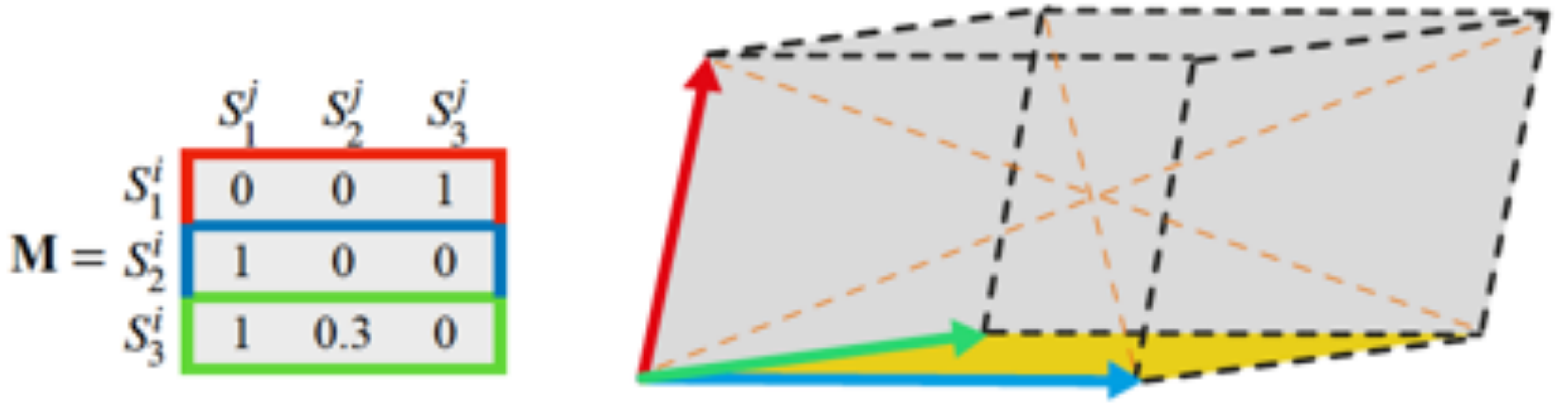


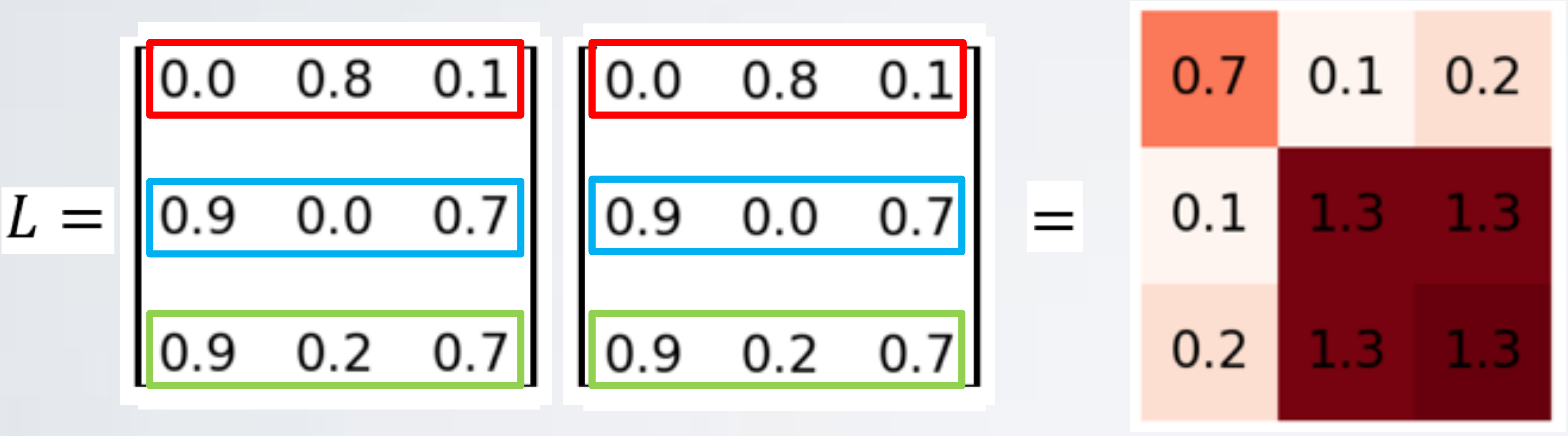

**Figure 1: Game-DPP. The squared volume of the grey cube equals to** $\det(\mathcal{L}_{\{S_1^i, S_2^i, S_3^i\}})$. **Since** $S_2^i, S_3^i$ **share similar payoff vectors, this leads to a smaller yellow area, and thus the probability of these two strategies co-occuring is low. The diversity (expected cardinality) of the population** $\{S_1^i\}, \{S_1^i, S_2^i\}, \{S_1^i, S_2^i, S_3^i\}$ **are** 0, 1, 1.21 **respectively.**

# Recent Advance (1): Diverse-PSRO

1. Go back to the first principle: diversity should be defined on the sense of orthogonality.

- Policy diversity can be measured through their pay-off vectors, i.e., $\mathscr{L}_{\mathbb{S}} = \mathbf{M}\mathbf{M}^{\top}$.
- The expected cardinality of DPP is the diversity metric.

$$\text{Diversity}\,(\mathbb{S}) = \mathbb{E}_{\mathbf{Y}\sim\mathbb{P}_{\mathscr{L}}}[\,|\,\mathbf{Y}\,|\,] = \operatorname{Tr}\left(\mathbf{I} - \left(\mathscr{L}_{\mathbb{S}} + \mathbf{I}\right)^{-1}\right)$$




$$M = \begin{bmatrix} 0.0 & 0.8 & 0.1 \\ 0.9 & 0.0 & 0.7 \\ 0.9 & 0.2 & 0.7 \end{bmatrix} \rightarrow \mathbb{E}_{Y\sim\mathbb{P}_L}[|Y|] = 1.18$$

$$M = \begin{bmatrix} 0.0 & 0.8 & 0.1 \\ 0.9 & 0.0 & 0.7 \\ 0.9 & 0.3 & 0.7 \end{bmatrix} \rightarrow \mathbb{E}_{Y\sim\mathbb{P}_L}[|Y|] = 1.25$$

# Recent Advance (1): Diverse-PSRO

- Based on diversity metric, we can design diversity-aware fictitious play and PSRO

$$\text{Diversity}\,(\mathbb{S}) = \mathbb{E}_{\mathrm{Y}\sim\mathbb{P}_{\mathscr{L}}}[\,|\,\mathrm{Y}\,|\,] = \mathrm{Tr}\left(\mathbf{I} - \left(\mathscr{L}_{\mathbb{S}} + \mathbf{I}\right)^{-1}\right)$$

  - **Diverse Fictitious Play**

$$\mathrm{BR}^{i}_{\epsilon}\left(\pi^{-i}\right) = \underset{\pi\in\Delta_{\mathbb{S}^i}}{\arg\max}\left[G^{i}\left(\pi,\pi^{-i}\right) + \tau\cdot\ \text{Diversity}\ \left(\mathbb{S}^{i}\cup\{\pi\}\right)\right]$$

  - **Diverse PSRO**

$$O^{1}\left(\pi^{2}\right) = \underset{\theta\in\mathbb{R}^{d}}{\arg\max}\sum_{S^{2}\in\mathbb{S}^{2}}\pi^{2}\left(S^{2}\right)\cdot\phi\left(S_{\theta},S^{2}\right) + \tau\cdot\ \text{Diversity}\ \left(\mathbb{S}^{1}\cup\left\{S_{\theta}\right\}\right)$$

  - **Diverse $\alpha$-PSRO ($\alpha$-Rank as meta-solver)**

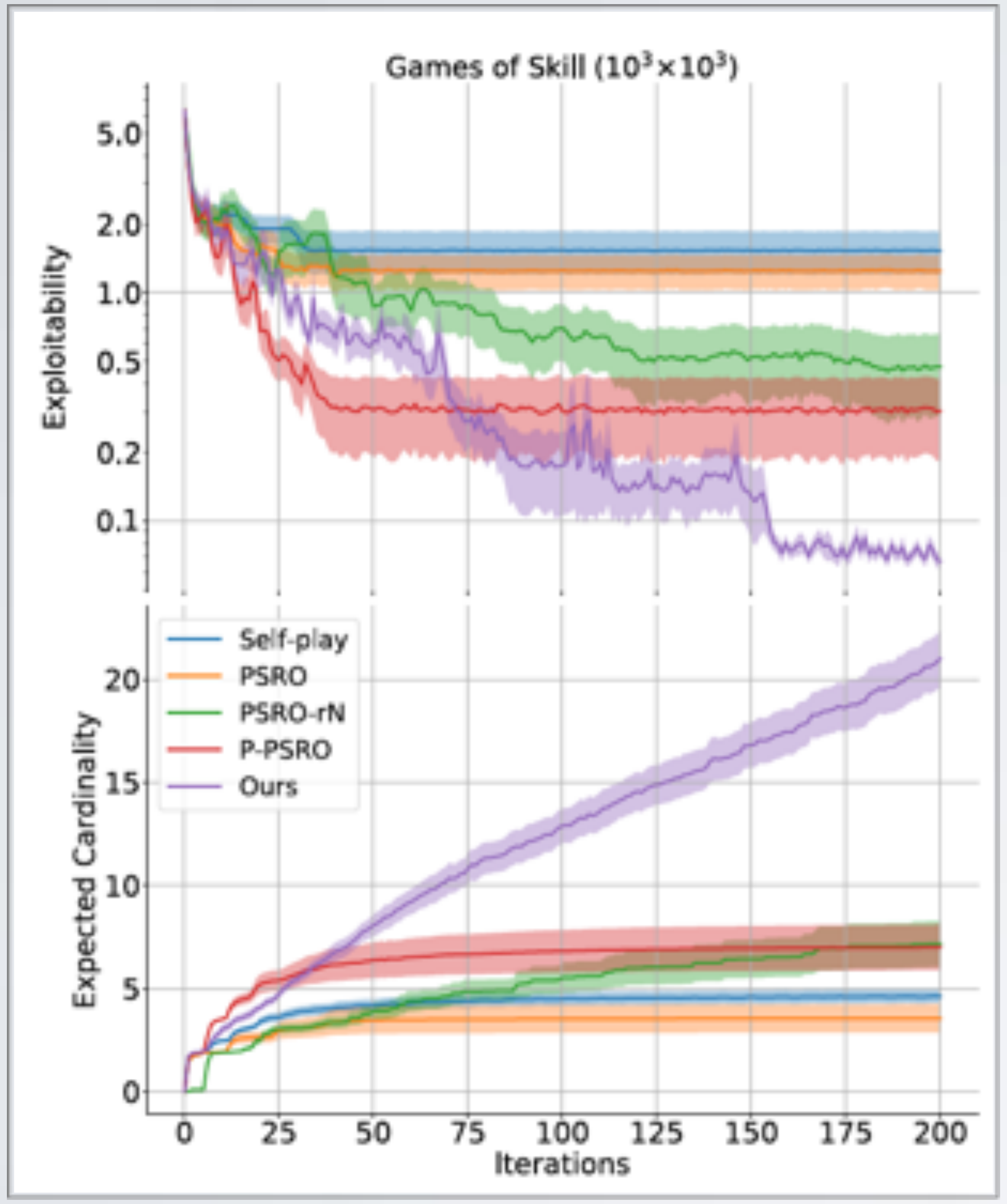
$$\mathscr{O}\left(\pi^{2}\right) = \mathrm{argmax}_{\pi\in\Delta_{S^i}}\mathrm{Tr}\left(I - \left(\mathscr{L}_{\mathbb{S}^i_t\cup\{\pi\}} + I\right)^{-1}\right)$$

  - Our diversity is strictly concave, so diverse best response is unique, and the algorithm share the same convergence guarantee as GWFP. Most importantly, we prove that

$$\text{Gamescape}\,(\mathbb{S}) \subsetneq\ \text{Gamescape}\ \left(\mathbb{S}\cup\left\{S_{\theta}\right\}\right)$$

# Recent Advance (1): Diverse-PSRO

1.Go back to the first principle: diversity should be defined on the sense of orthogonality.




the most efficient zero-sum game solver so far!

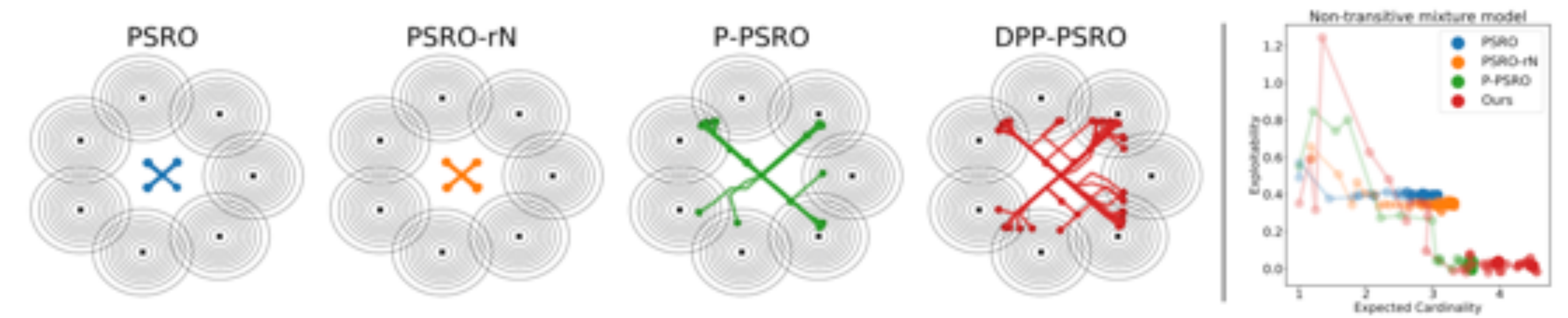



*Figure 3.* Non-transitive mixture model. Exploration trajectories during training and Performance vs. Diversity comparisons.

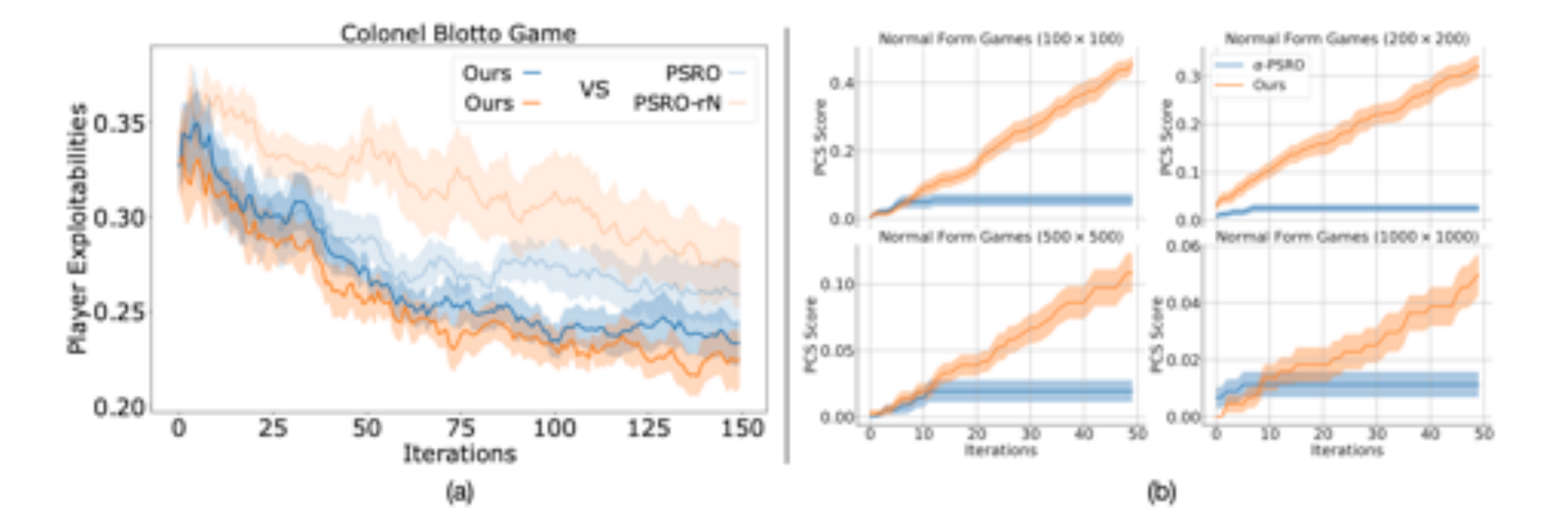



*Figure 4.* a) Performance of our diverse PSRO *vs.* PSRO, diverse PSRO *vs.* PSRO$_{rN}$ on the Blotto Game, b) PCS-Score comparison of our diverse $\alpha$-PSRO *vs.* $\alpha$-PSRO on NFGs with variable sizes.

# Recent Advance (2): Behavioural Diversity + Response Diversity

1. Diversity should include both response diversity (in terms of reward), and behavioural diversity (in terms of policy occupancy measure)

2. We want both the outcomes and the policies that lead to those outcomes to be diverse.

**Unifying Behavioral and Response Diversity for Open-ended Learning in Zero-sum Games**

**Xiangyu Liu[1], Hangtian Jia[2], Ying Wen[1]*, Yaodong Yang[3], Yujing Hu[2], Yingfeng Chen[2], Changjie Fan[2] and Zhipeng Hu[2]**
[1]Shanghai Jiao Tong University, [2]Netease Fuxi AI Lab, [3]University College London

| Method | Tool for Diversity | BD | RD | Game Type |
|---|---|---|---|---|
| DvD | Determinant | ✓ | × | Single-agent |
| $\text{PSRO}_\text{N}$ | None | × | × | n-player general-sum game |
| $\text{PSRO}_\text{rN}$ | $L_{1,1}$ norm | × | ✓ | 2-player zero-sum game |
| DPP-PSRO | Determinantal point process | × | ✓ | 2-player general-sum game |
| Our Methods | Occupancy measure & convex hull | ✓ | ✓ | n-player general-sum game |

# Recent Advance (2): Behavioural Diversity + Response Diversity

1. **behavioural diversity:** assuming existing population of policy mixed by Nash distribution is $\pi_E = (\pi_i, \pi_{E_{-i}})$, we want a new policy $\pi^{M+1}$ that has a different occupancy measure $\rho_\pi(s) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t P\left(s_t = s \mid \boldsymbol{\pi}\right)$ from $\pi_E$:

$$\mathrm{Div}_{\mathrm{occ}}\left(\pi_i^{M+1}\right) = D_f\left(\rho_{\pi_i^{M+1},\pi_{E_{-i}}} \| \rho_{\pi_i,\pi_{E_{-i}}}\right)$$

2. in practice, one can train a neural network $f_{\hat{\theta}}$ to fit $(s, \mathbf{a}) \sim \rho_{\boldsymbol{\pi}_E}$, and then assign an intrinsic reward by encouraging the new policy to visit state-action pairs with large prediction error (not covered by the existing occupancy measure).

$$\max R^{\mathrm{int}}(s, a) = \left\| f_{\hat{\theta}}(s, \mathbf{a}) - f_\theta(s, \mathbf{a}) \right\|^2$$

# Recent Advance (2): Behavioural Diversity + Response Diversity

1. response diversity: we want the new policy $\pi^{M+1}$ to expand the convex hull of the existing meta-game $A_M$ by having the new payoff vector $\mathbf{a}_{M+1} := \left[\phi_i(\pi_i^{M+1}, \pi_{-i}^{j})\right]_{j=1}^{N}$ that

$$\text{Div}_{\text{rew}}\left(\pi_i^{M+1}\right) = \min_{\substack{1^\top \beta = 1 \\ \beta \geq 0}} \left\| \mathbf{A}_M^\top \boldsymbol{\beta} - \mathbf{a}_{M+1} \right\|_2^2$$

2. the above equation has no close form, but we can optimise a lower bound

$$\text{Div}_{\text{rew}}\left(\pi_i^{M+1}\right) \geq \mathbf{F}(\pi_i^{M+1}) = \frac{\sigma_{\min}^2(\mathbf{A})\left(1 - \mathbf{1}^\top \left(\mathbf{A}^\top\right)^\dagger \mathbf{a}_{n+1}\right)^2}{M} + \left\| \left(\mathbf{I} - \mathbf{A}^\top \left(\mathbf{A}^\top\right)^\dagger\right) \mathbf{a}_{n+1} \right\|^2$$

3. chicken-egg problem: how can we know the payoff $\mathbf{a}_{M+1}$ before we train the policy ?

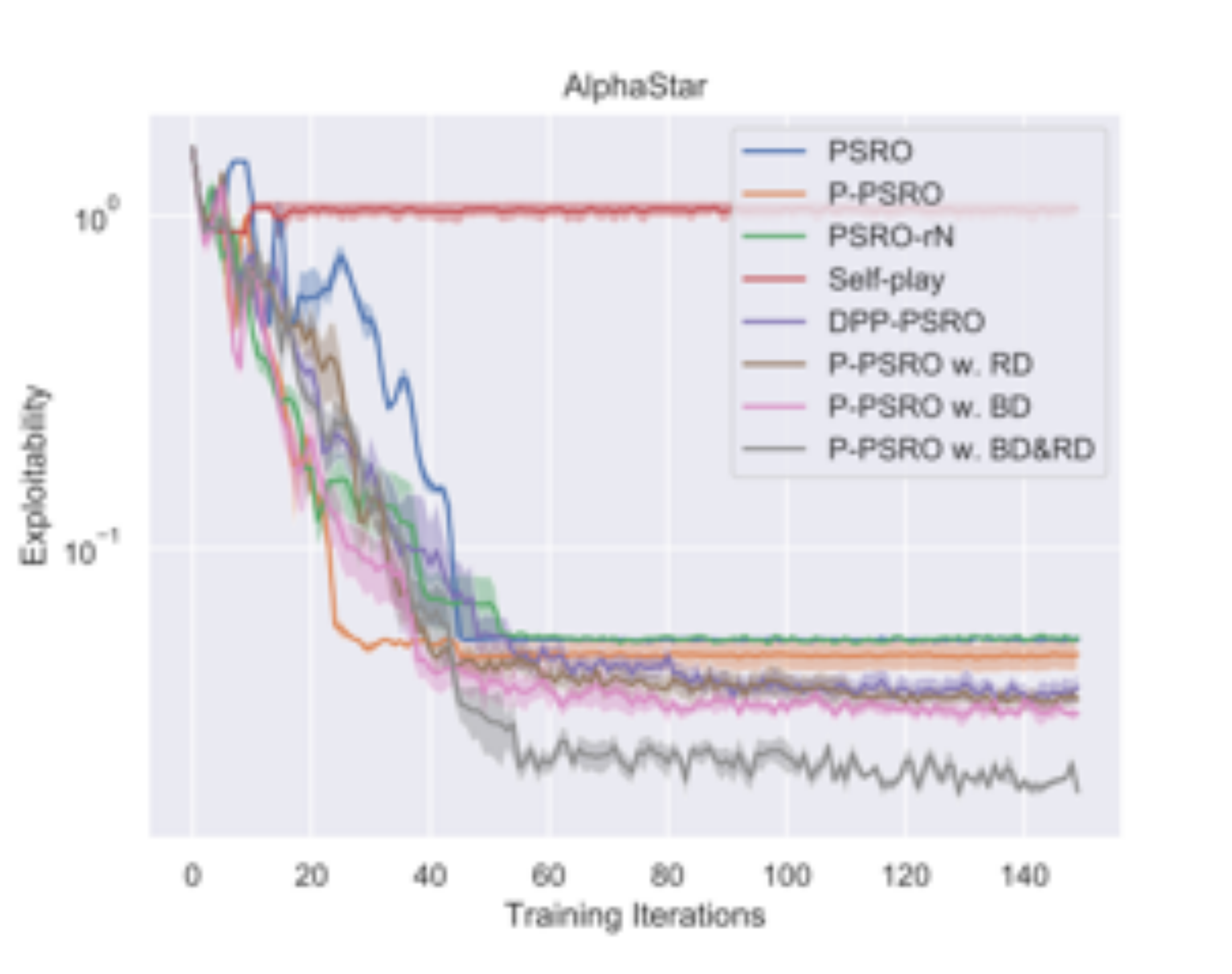
$$\frac{\partial F\left(\pi_i'(\theta)\right)}{\partial \theta} = \left(\frac{\partial \phi_i\left(\pi_i'(\theta), \pi_{-i}^1\right)}{\partial \theta}, \ldots, \frac{\partial \phi_i\left(\pi_i'(\theta), \pi_{-i}^M\right)}{\partial \theta}\right) \frac{\partial F}{\partial \mathbf{a}_{M+1}}$$

the answer: we can train against $\pi_{-i}^M$ based on the weights suggested by $\partial F / \partial \mathbf{a}_{M+1}$ !

# Recent Advance (2): Behavioural Diversity + Response Diversity

1. considering both diversity terms in the PSRO process

$$\arg\max_{\pi_i'} \mathbb{E}_{s,\mathbf{a}\sim\rho_{\pi_i',\pi_{E-i}}}[r(s,\mathbf{a})] + \lambda_1 \operatorname{Div}_{\text{occ}}\left(\pi_i'\right) + \lambda_2 \operatorname{Div}_{\text{rew}}\left(\pi_i'\right)$$



(a)

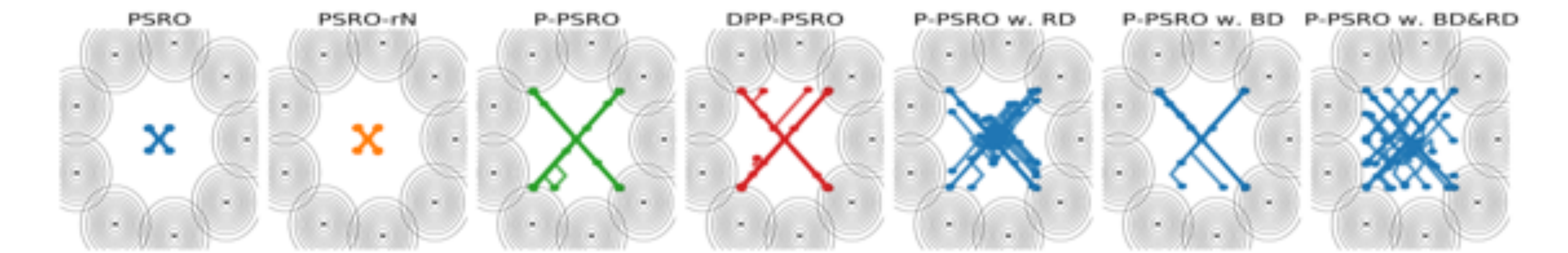


Figure 2: Exploration trajectories during training process on *Non-Transitive Mixture Games*.

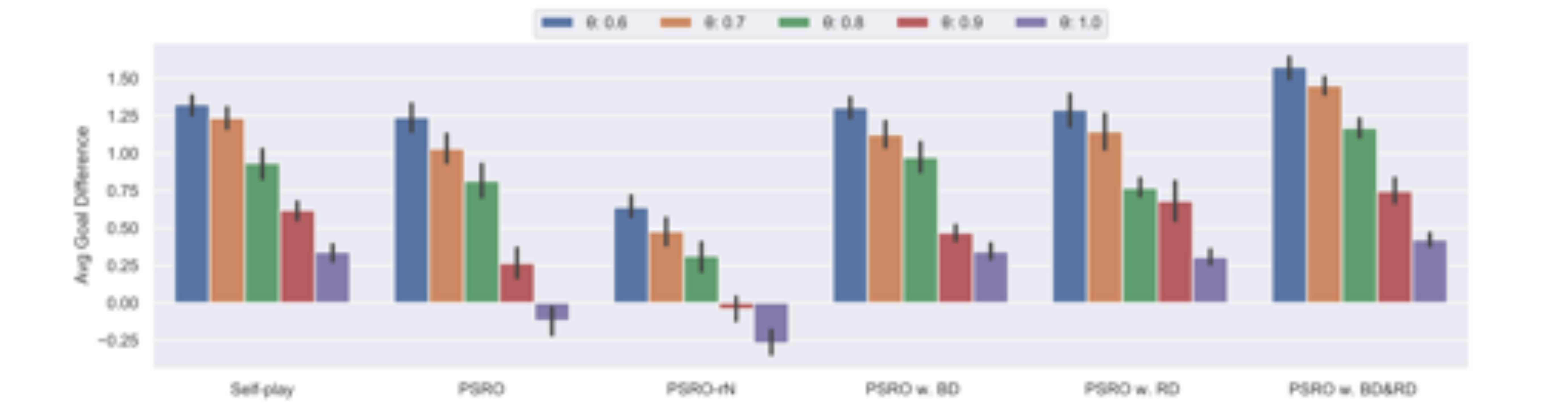


Figure 3: The average goal difference between all the methods and the built-in bots with various difficulty levels $\theta$ ($\theta \in [0,1]$ and larger $\theta$ means harder bot) on *Google Research Football*.

# Diverse Behaviours Learned on Google Football

https://sites.google.com/view/diverse-psro/


make offside


push and run

# Contents

- **What is Non-Transitivity in Games**
- **How to Measure Non-Transitivity**
- **Solutions: Double Oracle / PSRO Methods**
- **Recent advances: Diverse-PSRO**
- **Recent advances: Online-PSRO**
- **Recent advances: Auto-PSRO**

# Recent Advance (3): Online Double Oracle

**Online Double Oracle**

Le Cong Dinh[*,1,2], Yaodong Yang[*,1,4], Nicolas Perez-Nieves[3], Oliver Slumbers[4],
Zheng Tian[4], David Henry Mguni[1], Haitham Bou Ammar[1], Jun Wang[1,4]

1. Nash is unexploitbale, but when a player always plays Rock, you should play Paper rather than (1/3, 1/3, 1/3).

2. Double Oracle/PSRO assumes both players play the worst-case scenario, can be too pessimistic during training.

3. Online learning provides a framework about how to exploit opponents through minimising regret.

**Algorithm 1** Double Oracle (McMahan et al., 2003)

```
Input: A set Π, C strategy set of players
Π_0, C_0: initial set of strategies
for t = 1 to ∞ do
  if Π_t ≠ Π_{t-1} or C_t ≠ C_{t-1} then
    Solve the NE of the subgame G_t:
    (π*_t, c*_t) = arg min_{π∈Δ_{Π_t}} arg max_{c∈Δ_{C_t}} π^T A c      too pessimistic
    Find the best response a_{t+1} and c_{t+1} to (π*_t, c*_t):
        a_{t+1} = arg min_{a∈Π} a^T A c*_t
        c_{t+1} = arg max_{c∈C} π*_t^T A c
    Update Π_{t+1} = Π_t ∪ {a_{t+1}}, C_{t+1} = C_t ∪ {c_{t+1}}
  else if Π_t = Π_{t-1} and C_t = C_{t-1} then
    Terminate
  end if
end for
```

**What we want:**

if opponents play $\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_T$, we want the player to have $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_T$ s.t.

$$\lim_{T\to\infty} \frac{R_T}{T} = 0, \quad R_T = \max_{\pi\in\Delta_\Pi} \sum_{t=1}^{T} \left(\boldsymbol{\pi}_t^\top \boldsymbol{A}\boldsymbol{c}_t - \boldsymbol{\pi}^\top \boldsymbol{A}\boldsymbol{c}_t\right)$$

**What we know:**

hedge algorithm/multiplicative weight update can achieve no-regret property if one follows the below update

$$\boldsymbol{\pi}_{t+1}(i) = \boldsymbol{\pi}_t(i)\frac{\exp\left(-\mu_t \boldsymbol{a}^{i\top}\boldsymbol{A}\boldsymbol{c}_t\right)}{\sum_{i=1}^{n}\boldsymbol{\pi}_t(i)\exp\left(-\mu_t \boldsymbol{a}^{i\top}\boldsymbol{A}\boldsymbol{c}_t\right)}, \forall i \in [n]$$

the regret of MWU is $\mathcal{O}(\sqrt{T\log(n)/2})$

# Recent Advance (3): Online Double Oracle

**Algorithm 1** Double Oracle (McMahan et al., 2003)

**Input:** A set $\Pi, C$ strategy set of players
$\Pi_0, C_0$: initial set of strategies
**for** $t = 1$ to $\infty$ **do**
  **if** $\Pi_t \neq \Pi_{t-1}$ or $C_t \neq C_{t-1}$ **then**
    Solve the NE of the subgame $\boldsymbol{G}_t$:
    $(\boldsymbol{\pi}_t^*, \boldsymbol{c}_t^*) = \arg\min_{\boldsymbol{\pi} \in \Delta_{\Pi_t}} \arg\max_{\boldsymbol{c} \in \Delta_{C_t}} \boldsymbol{\pi}^\top \boldsymbol{A}\boldsymbol{c}$
    Find the best response $\boldsymbol{a}_{t+1}$ and $\boldsymbol{c}_{t+1}$ to $(\boldsymbol{\pi}_t^*, \boldsymbol{c}_t^*)$:
      $\boldsymbol{a}_{t+1} = \arg\min_{\boldsymbol{a} \in \Pi} \boldsymbol{a}^\top \boldsymbol{A}\boldsymbol{c}_t^*$
      $\boldsymbol{c}_{t+1} = \arg\max_{\boldsymbol{c} \in C} \boldsymbol{\pi}_t^{*\top} \boldsymbol{A}\boldsymbol{c}$
    Update $\Pi_{t+1} = \Pi_t \cup \{\boldsymbol{a}_{t+1}\}, C_{t+1} = C_t \cup \{\boldsymbol{c}_{t+1}\}$
  **else if** $\Pi_t = \Pi_{t-1}$ and $C_t = C_{t-1}$ **then**
    Terminate
  **end if**
**end for**

**Algorithm 2: Online Single Oracle Algorithm**

**Input:** Player's pure strategy set $\Pi$
Init. effective strategies set: $\Pi_0 = \Pi_1 = \{\boldsymbol{a}^j\}, \boldsymbol{a}^j \in \Pi$
**for** $t = 1$ to T **do**
  **if** $\Pi_t = \Pi_{t-1}$ **then**
    Compute $\boldsymbol{\pi}_t$ by the MWU in Equation (5)
  **else if** $\Pi_t \neq \Pi_{t-1}$ **then**
    Start a new time window $T_{i+1}$ and
    Reset $\boldsymbol{\pi}_t = [1/|\Pi_t|, \ldots, 1/|\Pi_t|], \ \bar{\boldsymbol{l}} = \boldsymbol{0}$
  **end if**
  Observe $\boldsymbol{l}_t$ and update the average loss in $T_i$:
  $\bar{\boldsymbol{l}} = \sum_{t \in T_i} \boldsymbol{l}_t / |T_i|$
  Calculate the best response: $\boldsymbol{a}_t = \arg\min_{\boldsymbol{\pi} \in \Pi} \langle \boldsymbol{\pi}, \bar{\boldsymbol{l}} \rangle$
  Update the set of strategies: $\Pi_{t+1} = \Pi_t \cup \{\boldsymbol{a}_t\}$
**end for**
**Output:** $\boldsymbol{\pi}_T, \Pi_T$

**Intuition:** *maintain a time window $T_i$ to track opponent's strategy, if no new best response can be found, then keep exploiting, otherwise refresh the time window to catch up with the latest change*

# Recent Advance (3): Online Double Oracle

**Algorithm 2:** Online Single Oracle Algorithm

**Input:** Player's pure strategy set $\Pi$
Init. effective strategies set: $\Pi_0 = \Pi_1 = \{\boldsymbol{a}^j\}, \boldsymbol{a}^j \in \Pi$
**for** $t = 1$ to T **do**
  **if** $\Pi_t = \Pi_{t-1}$ **then**
    Compute $\boldsymbol{\pi}_t$ by the MWU in Equation (5)
  **else if** $\Pi_t \neq \Pi_{t-1}$ **then**
    Start a new time window $T_{i+1}$ and
    Reset $\boldsymbol{\pi}_t = \left[1/|\Pi_t|, \ldots, 1/|\Pi_t|\right], \ \bar{\boldsymbol{l}} = \boldsymbol{0}$
  **end if**
  Observe $\boldsymbol{l}_t$ and update the average loss in $T_i$:
  $\bar{\boldsymbol{l}} = \sum_{t \in T_i} \boldsymbol{l}_t / |T_i|$
  Calculate the best response: $\boldsymbol{a}_t = \arg\min_{\boldsymbol{\pi} \in \Pi} \langle \boldsymbol{\pi}, \bar{\boldsymbol{l}} \rangle$
  Update the set of strategies: $\Pi_{t+1} = \Pi_t \cup \{\boldsymbol{a}_t\}$
**end for**
**Output:** $\boldsymbol{\pi}_T, \Pi_T$

1.OSO is a no-regret algorithm.

**Theorem 4** (Regret Bound of OSO). *Let $\boldsymbol{l}_1, \boldsymbol{l}_2, \ldots, \boldsymbol{l}_T$ be a sequence of loss vectors played by an adversary, and $\langle \cdot, \cdot \rangle$ be the dot product, OSO in Algorithm 2 is a no-regret algorithm with*

$$\frac{1}{T}\left(\sum_{t=1}^{T} \langle \boldsymbol{\pi}_t, \boldsymbol{l}_t \rangle - \min_{\boldsymbol{\pi} \in \Pi} \sum_{t=1}^{T} \langle \boldsymbol{\pi}, \boldsymbol{l}_t \rangle\right) \leq \frac{\sqrt{k \log(k)}}{\sqrt{2T}},$$

*where $k = |\Pi_T|$ is the size of effective strategy set in the final time window.*

2.Putting OSO into self-play settings, we get Online Double Oracle which can solve Nash.

- Recall that in two-player zero-sum game, if two no-regret methods self play, the outcome will leads to a Nash equilibrium! [Cesa-Bianchi, sec 7]

**Algorithm 3:** Online Double Oracle Algorithm

**Input:** Full pure strategy set $\Pi, C$
Init. effective strategies set: $\Pi_0 = \Pi_1, C_0 = C_1$
**for** $t = 1$ to T **do**
  Each player follows the OSO in Algorithm 2 with their respective effective strategy sets $\Pi_t, C_t$
**end for**
**Output**: $\boldsymbol{\pi}_T, \Pi_T, \boldsymbol{c}_T, C_T$

**Theorem 5.** *Suppose both players apply OSO. Let $k_1$, $k_2$ denote the size of effective strategy set for each player. Then, the average strategies of both players converge to the NE with the rate:*

$$\epsilon_T = \sqrt{\frac{k_1 \log(k_1)}{2T}} + \sqrt{\frac{k_2 \log(k_2)}{2T}}.$$

*In situation where both players follow OSO with Less-Frequent Best Response in Equation (6) and $\alpha^i_{t-|T_i|} = \sqrt{t - |T_i|}$, the convergence rate to NE will be*

$$\epsilon_T = \sqrt{\frac{k_1 \log(k_1)}{2T}} + \sqrt{\frac{k_2 \log(k_2)}{2T}} + \frac{\sqrt{k_1} + \sqrt{k_2}}{\sqrt{T}}.$$

# Recent Advance (3): Online Double Oracle

1.Summary of methods that can solve two-player zero-sum games

Table 1: Properties of existing solvers on two-player zero-sum games $A_{n\times m}$. *:DO in the worst case has to solve all sub-games till reaching the full game, so the time complexity is one order magnitude larger than LP. †: Since PSRO uses approximate best response, the total time complexity is unknown. ‡ Note that the regret bound of ODO can not be directly compared with the time complexity of DO, which are two different notions.

| Method | Rational (No-regret) | Allow $\epsilon$-Best Response | No Need to Know the Full Matrix $A$ | Time Complexity ($\tilde{\mathcal{O}}$) / Regret Bound ($\mathcal{O}$) | Large Games |
|---|---|---|---|---|---|
| Linear Programming [30] | | | | $\tilde{\mathcal{O}}(n\exp(-T/n^{2.38}))$ | |
| (Generalised) Fictitious Play [18] | | ✓ | ✓ | $\tilde{\mathcal{O}}(T^{-1/(n+m-2)})$ | |
| Multipli. Weight Update [12] | ✓ | | ✓ | $\mathcal{O}(\sqrt{\log(n)/T})$ | |
| Double Oracle [21] | | | ✓ | $\tilde{\mathcal{O}}(n\exp(-T/n^{3.38}))^*$ | ✓ |
| Policy Space Response Oracle [17] | | ✓ | ✓ | $\times^\dagger$ | ✓ |
| **Online Double Oracle** | ✓ | ✓ | ✓ | $\mathcal{O}(\sqrt{k\log(k)/T})^\ddagger$ | ✓ |

2.$k \ll n$ holds in general: for example, randomly initialised zero-sum games has only $k \approx (1/2 + \mathcal{O}(1))n$ [Johnasson 2014], also empirically, we have observed small k.

Table 2: Size of the Nash Support of Games

| Game | Total Strategies | Size of Nash support |
|---|---|---|
| 3-Move Parity Game 2 | 160 | 1 |
| 5,4-Blotto | 56 | 6 |
| AlphaStar | 888 | 3 |
| Connect Four | 1470 | 23 |
| Disc Game | 1000 | 27 |
| Elo game + noise=0.1 | 1000 | 6 |
| Elo game | 1000 | 1 |
| Go (boardsize=3,komi=6.5) | 1933 | 13 |
| Misere (game=tic tac toe) | 926 | 1 |
| Normal Bernoulli game | 1000 | 5 |
| Quoridor (boardsize=3) | 1404 | 1 |
| Random game of skill | 1000 | 5 |
| Tic Tac Toe | 880 | 1 |
| Transitive game | 1000 | 1 |
| Triangular game | 1000 | 1 |

ODO has a constant k regardless of game size

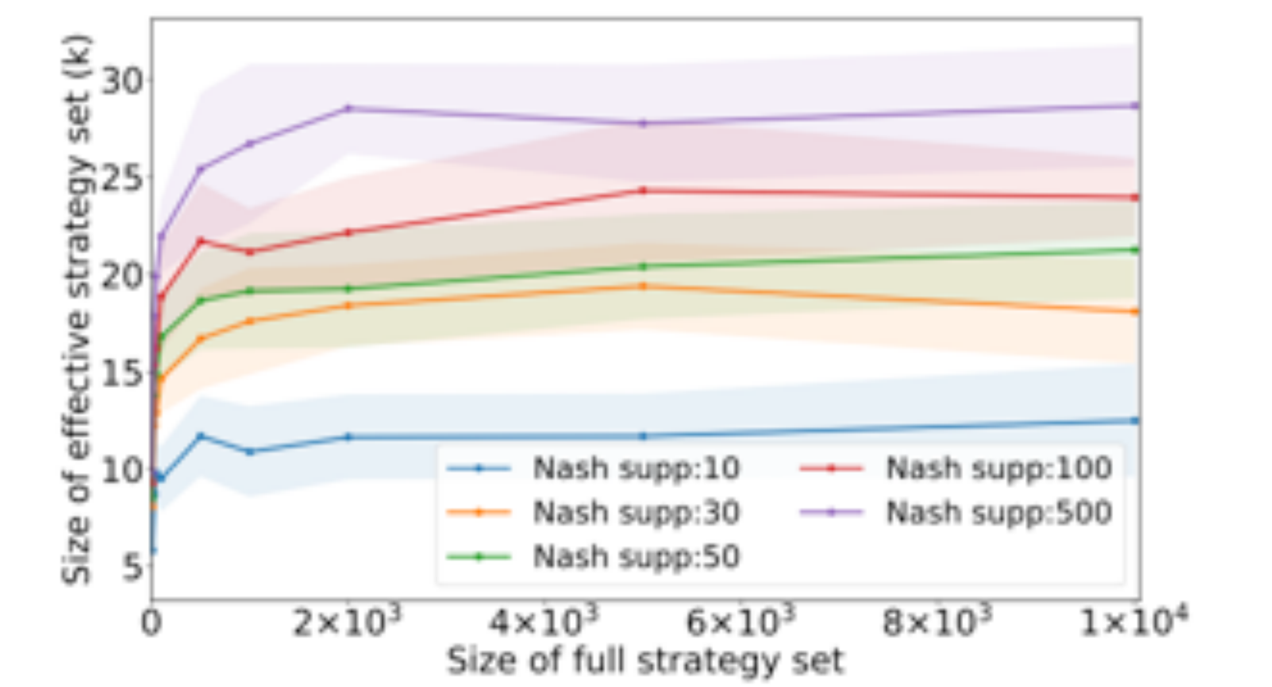



Figure 1: Sizes of effective strategy set (i.e., $k$) in cases of an OSO agent playing against an MWU opponent with different sizes of full strategy set and NE support.

# Recent Advance (3): Online Double Oracle

## Exploitability on the Spinning Top games

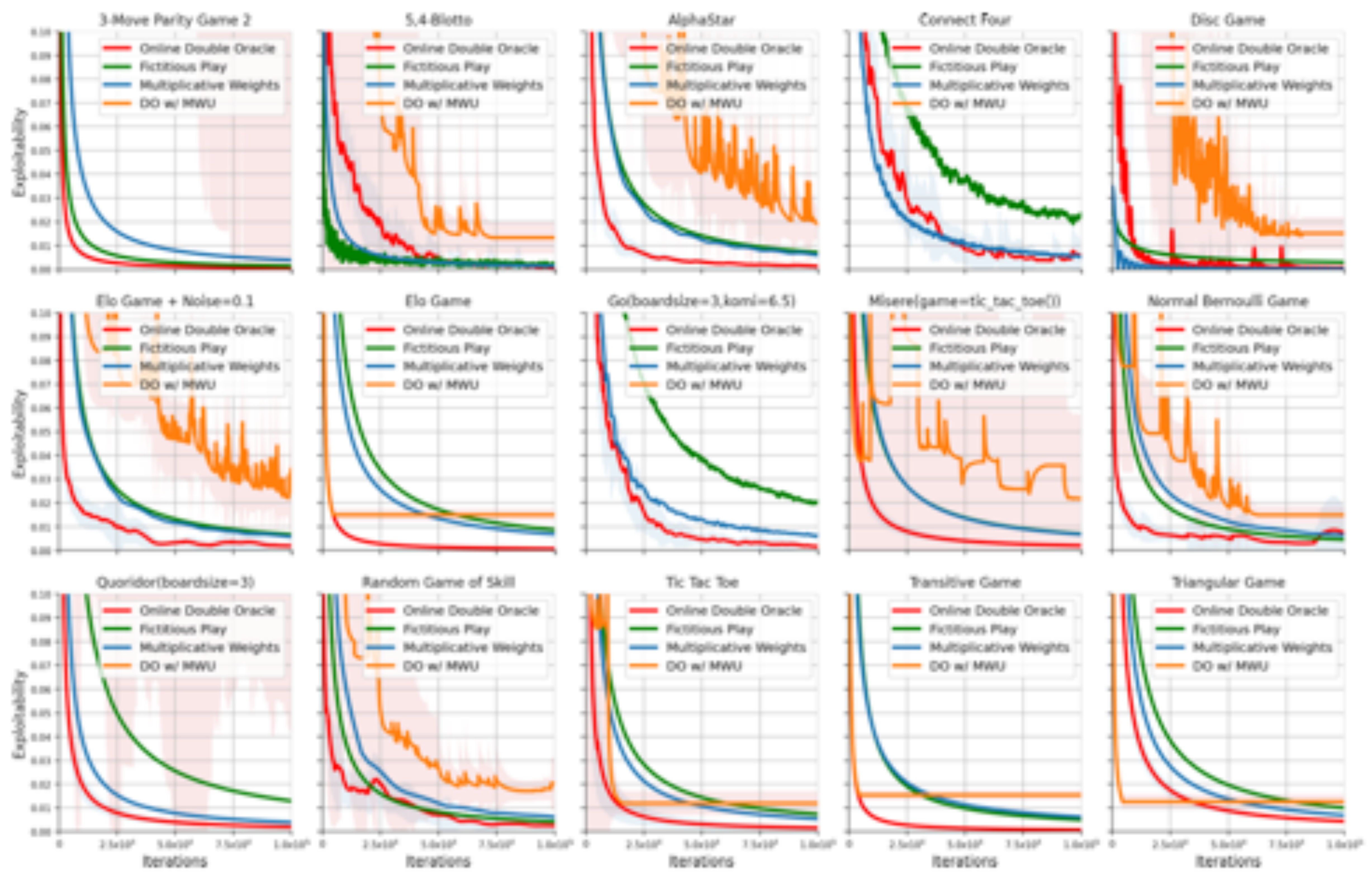
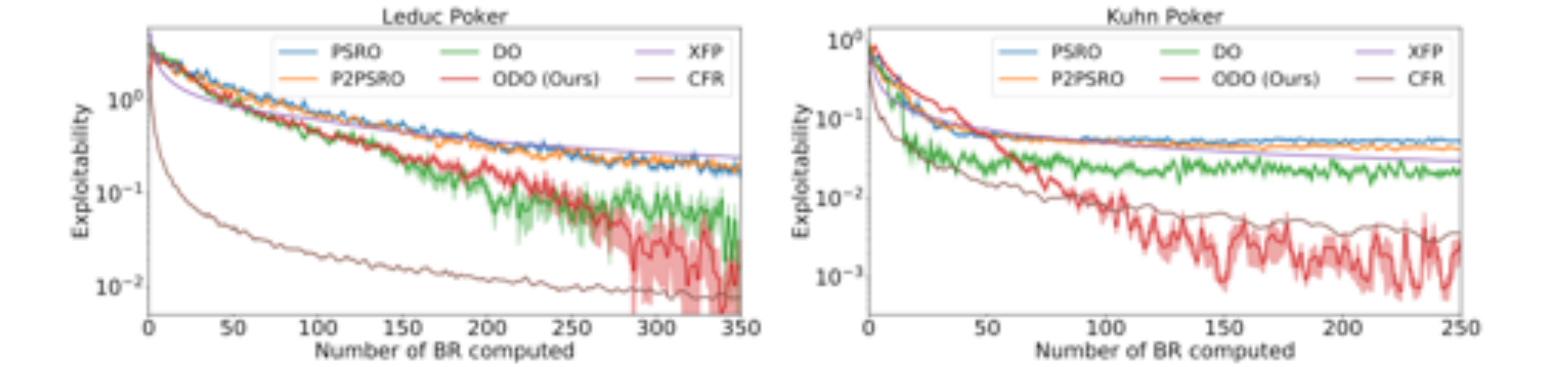



Figure 1: Performance comparisons under self-plays

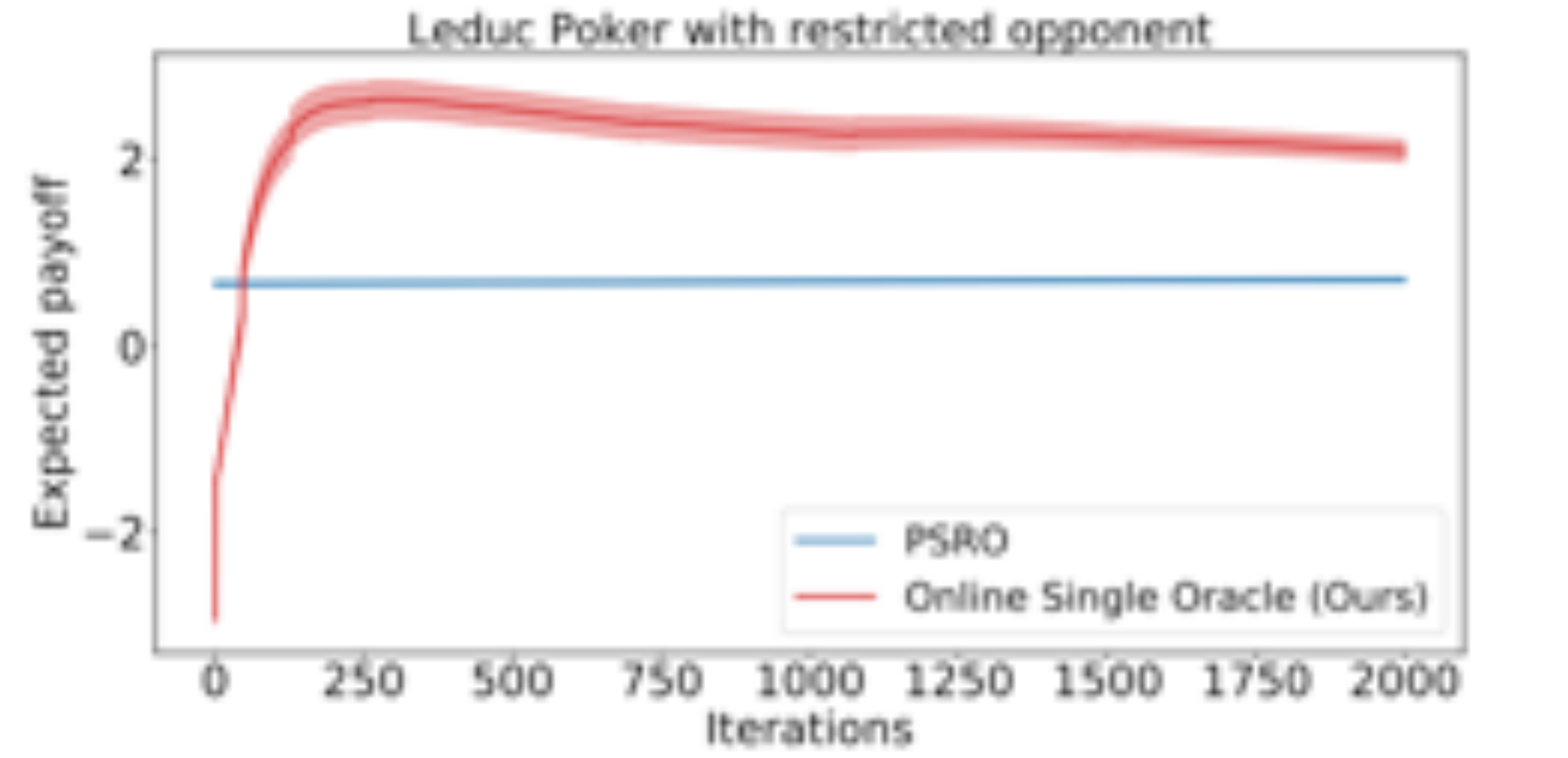
## Exploitability on Poker




(a) Exploitability on Leduc Poker (b) Exploitability on Kuhn Poker

Figure 3: Performance comparisons in exploitability on Poker games.

## Play with an imperfect opponent




(a) Leduc Poker

# Contents

- **What is Non-Transitivity in Games**
- **How to Measure Non-Transitivity**
- **Solutions: Double Oracle / PSRO Methods**
- **Recent advances: Diverse-PSRO**
- **Recent advances: Online-PSRO**
- **Recent advances: Auto-PSRO**

# Recent Advance (4): Auto-PSRO

**Discovering Multi-Agent Auto-Curricula in Two-Player Zero-Sum Games**

Xidong Feng[*,1], Oliver Slumbers[*,1], Yaodong Yang[†,1], Ziyu Wan[2], Bo Liu[3], Stephen McAleer[4], Ying Wen[2], Jun Wang[1]

1. Learning to learn: to discover multi-agent algorithms ("who to beat" and "how to beat them") from data.

2. Maybe game theoretical knowledge (transitivity/non-transitivity/Nash) are not necessarily needed, the solution algorithm can be learned purely from data.

3. The idea is to learn how to build an auto-curricula based on the type of game provided to the meta-learning algorithm, rather than what the auto-curricula should be (e.g. PSRO/DO).

4. Why it will work better than DO/PSRO: because RL oracle can only approximate best response, and using Nash, though theoretically guaranteed, may not be the best option for a solver.

5. On single-agent RL, the discovered RL methods are proved to outperform TD learning designed by humans.

**Discovering Reinforcement Learning Algorithms**

Junhyuk Oh, Matteo Hessel, Wojciech M. Czarnecki, Zhongwen Xu, Hado van Hasselt, Satinder Singh, David Silver

DeepMind

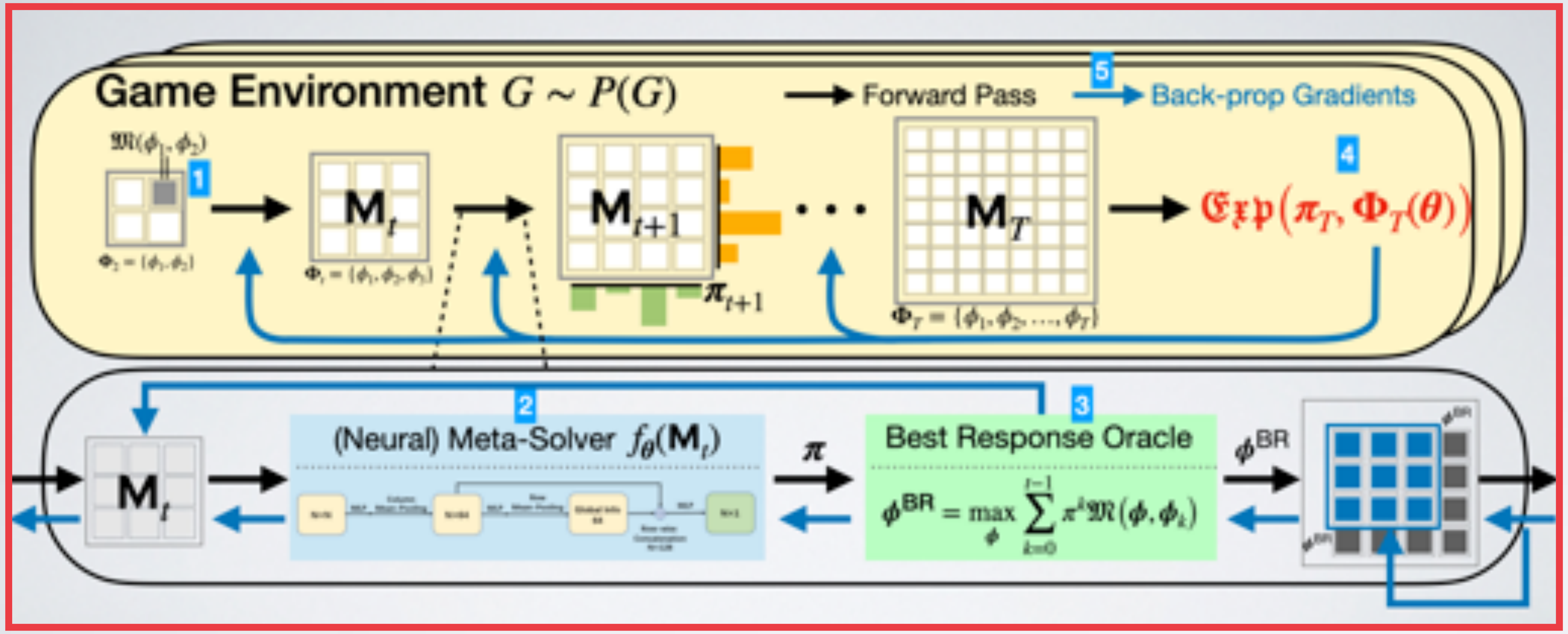
**Meta-Gradient Reinforcement Learning with an Objective Discovered Online**

Zhongwen Xu, Hado van Hasselt, Matteo Hessel
Junhyuk Oh, Satinder Singh, David Silver
DeepMind
{zhongwen,hado,mtthss,junhyuk,baveja,davidsilver}@google.com

| Algorithm | Algorithm properties | | | What is meta-learned? |
|---|---|---|---|---|
| IDBD, SMD [30, 27] | † | □ | → | learning rate |
| $SGD^2$ [1] | ††† | ■ | ← | optimiser |
| $RL^2$, Meta-RL [9, 39] | ††† | ■ | X | recurrent network |
| MAML, REPTILE [11, 23] | ††† | □ | ← | initial params |
| Meta-Gradient [43, 46] | † | □ | → | $\gamma$, $\lambda$, reward |
| Meta-Gradient [38, 44, 40] | † | □ | ← | auxiliary tasks, hyperparams, reward weights |
| $ML^3$, MetaGenRL [2, 19] | ††† | ■ | ← | loss function |
| Evolved PG [16] | ††† | ■ | X | loss function |
| Oh et al. 2020 [24] | ††† | ■ | ← | target vector |
| This paper | † | ■ | ← | target |

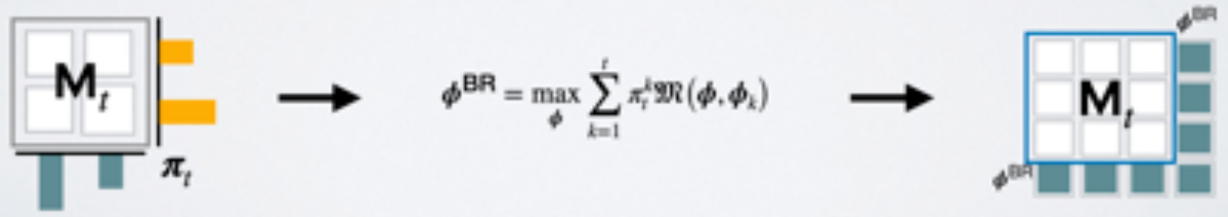
□ white box, ■ black box, † single lifetime, ††† multi-lifetime
← backward mode, → forward mode, X no meta-gradient

# Recent Advance (4): Auto-PSRO Framework



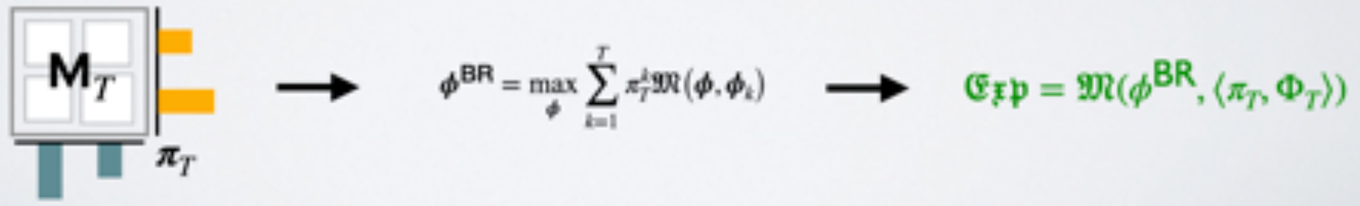

## 3 The Best-Response Oracle

- Algorithm component that controls the iterative expansion of the population
  - Given a curriculum $\pi_t \in \Delta_{|\Phi_t|}$ the goal becomes to solve a best-response to this distribution
  - Goal is the following: $\phi_t^{BR} = \text{argmax}_\phi \sum_{k=1}^{t} \pi_t^k \mathfrak{M}(\phi, \phi_k)$
    - Perform the optimisation in anyway desired, but this will impact the meta-gradient calculation



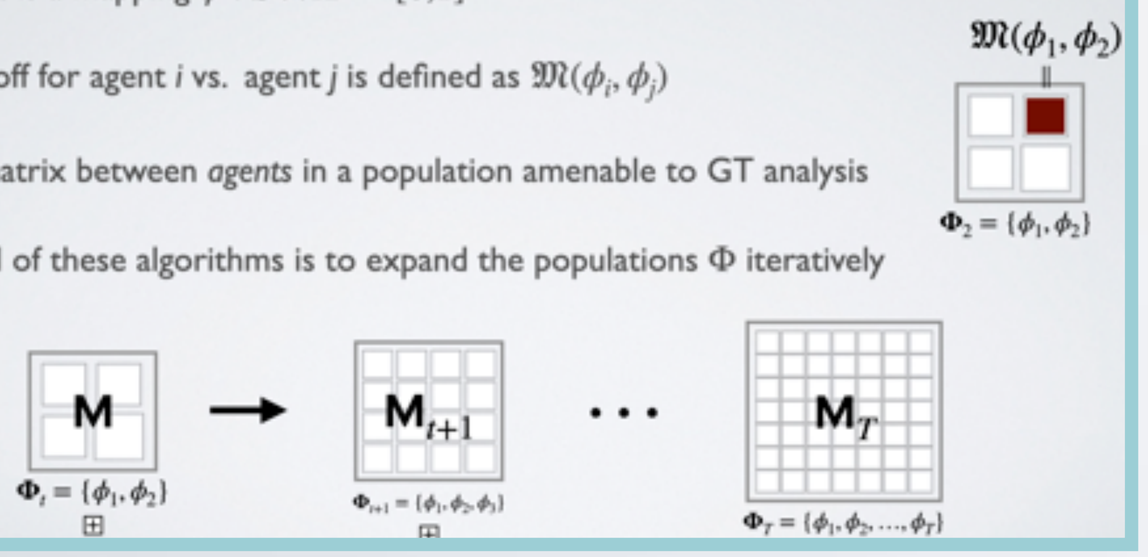

## 4 The Learning Objective

- What is the goal of the iterative update procedure?
  - Given a curriculum $\pi_T = f_\theta(\mathbf{M}_T)$ and a population $\Phi_T$ we want to be as close to a Nash equilibrium as possible.
  - Distance to Nash measured as the *exploitability*: $\mathfrak{Exp} := \max_\phi \mathfrak{M}(\phi, \langle \pi_T, \Phi_T \rangle)$
    - i.e. How good is the best-response to the curriculum? If 0, it is a Nash equilibrium




## 1 The Meta-Game

- Main component of population-based methods - **The meta-game**
  - An agent is a mapping $\phi : S \times A \rightarrow [0,1]$
  - The payoff for agent *i* vs. agent *j* is defined as $\mathfrak{M}(\phi_i, \phi_j)$
  - Payoff matrix between *agents* in a population amenable to GT analysis
  - The goal of these algorithms is to expand the populations $\Phi$ iteratively




## 2 The Meta-Solver

- Algorithm component that controls the auto-curricula of *who to compete with*
  - General examples: Nash equilibrium, Uniform distribution, Last agent
  - Need to parameterise the process so that we can learn it
    - A network with parameters $\theta$ maps $f_\theta : \mathbf{M}_t \rightarrow [0,1]^t$ so that $\pi_t = f_\theta(\mathbf{M}_t)$


## 5 Optimisation through meta-gradients

- Recall the learning objective of the player: $\mathfrak{Exp} := \max_\phi \mathfrak{M}(\phi, \langle \pi_T, \Phi_T \rangle)$
  - Also recall that $\pi_T = f_\theta(\mathbf{M}_T)$, which allows us to define the meta-solver optimisation as:

$$\theta^* = \text{argmin}_\theta J(\theta), \text{ where } J(\theta) = \mathbb{E}_{G \sim P(G)}\left[\mathfrak{Exp}(\pi, \Phi \mid \theta, G)\right]$$

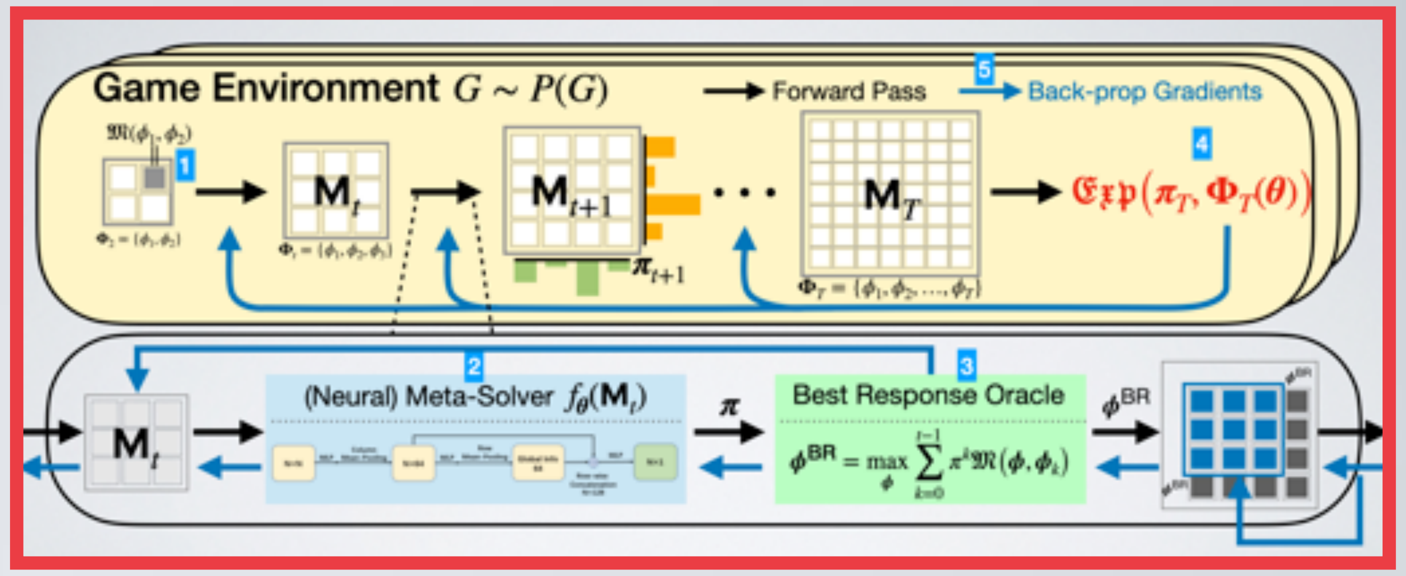
- What does the gradient boil down to then?

$$\nabla_\theta J(\theta) = \mathbb{E}_G\left[\frac{\partial \mathfrak{M}_{T+1}}{\partial \phi_{T+1}^{BR}} \frac{\partial \phi_{T+1}^{BR}}{\partial \theta} + \frac{\partial \mathfrak{M}_{T+1}}{\partial \pi_T} \frac{\partial \pi_T}{\partial \theta} + \frac{\partial \mathfrak{M}_{T+1}}{\partial \Phi_T} \frac{\partial \Phi_T}{\partial \theta}\right]$$

Gradient of most interest decomposes to

$$\frac{\partial \phi_{T+1}^{BR}}{\partial \theta} = \frac{\partial \phi_{T+1}^{BR}}{\partial \pi_T} \frac{\partial \pi_T}{\partial \theta} + \frac{\partial \phi_{T+1}^{BR}}{\partial \Phi_T} \frac{\partial \Phi_T}{\partial \theta}$$

# Recent Advance (4): Auto-PSRO Objective




1. Overall, the objective is give by:

The goal of LMAC is to find an auto-curricula that after $T$ best-response iterations returns a meta-strategy and population, $\langle \boldsymbol{\pi}_T, \boldsymbol{\Phi}_T \rangle$, that helps minimise the exploitability, written as:

$$\min_{\boldsymbol{\theta}} \mathfrak{Exp}(\boldsymbol{\pi}_T(\boldsymbol{\theta}), \boldsymbol{\Phi}_T(\boldsymbol{\theta})), \text{ where } \mathfrak{Exp} := \max_{\boldsymbol{\phi}} \mathfrak{M}\left(\boldsymbol{\phi}, \langle \boldsymbol{\pi}_T, \boldsymbol{\Phi}_T \rangle\right), \quad (3)$$

$$\boldsymbol{\pi}_T = f_{\boldsymbol{\theta}}(\mathbf{M}_T), \boldsymbol{\Phi}_T = \left\{\boldsymbol{\phi}_T^{\text{BR}}(\boldsymbol{\theta}), \boldsymbol{\phi}_{T-1}^{\text{BR}}(\boldsymbol{\theta}), ..., \boldsymbol{\phi}_1^{\text{BR}}(\boldsymbol{\theta})\right\}. \quad (4)$$

Based on the *Player's* learning objectives in Eq. (3), we can optimise the meta-solver as follows:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}), \text{ where } J(\boldsymbol{\theta}) = \mathbb{E}_{G \sim P(G)}\left[\mathfrak{Exp}\left(\boldsymbol{\pi}, \boldsymbol{\Phi} | \boldsymbol{\theta}, G\right)\right]. \quad (5)$$

2. When optimising the meta-solver $\theta$, the format of best-response oracle matters due to back-propagation!

- one-step gradient descent oracle

$$\phi_{t+1}^{\text{BR}} = \phi_0 + \alpha \frac{\partial \mathfrak{M}(\phi_0, \langle \pi_t, \Phi_t \rangle)}{\partial \phi_0}, \quad \frac{\partial \phi_{t+1}^{\text{BR}}}{\partial \pi_t} = \alpha \frac{\partial^2 \mathfrak{M}\left(\phi_0, \langle \pi_t, \Phi_t \rangle\right)}{\partial \phi_0 \partial \pi_t}, \quad \frac{\partial \phi_{t+1}^{\text{BR}}}{\partial \Phi_t} = \alpha \frac{\partial^2 \mathfrak{M}\left(\phi_0, \langle \pi_t, \Phi_t \rangle\right)}{\partial \phi_0 \partial \Phi_t}.$$

- N-step gradient descent oracle (via implicit gradient)

$$\frac{\partial \phi_{t+1}^{\text{BR}}}{\partial \Phi_t} = -\left[\frac{\partial^2 \mathfrak{M}(\phi_{t+1}^{\text{BR}}, \langle \pi_t, \Phi_t \rangle)}{\partial \phi_{t+1}^{\text{BR}} \partial \phi_{t+1}^{\text{BR}^T}}\right]^{-1} \frac{\partial^2 \mathfrak{M}(\phi_{t+1}^{\text{BR}}, \langle \pi_t, \Phi_t \rangle)}{\partial \phi_{t+1}^{\text{BR}} \partial \Phi_t}$$

- policy-gradient based oracle (via DICE)

$$\phi_1 = \phi_0 + \alpha \frac{\partial \mathcal{J}^{\text{DICE}}}{\partial \phi_0}, \text{ where } \mathcal{J}^{\text{DICE}} = \sum_{k=0}^{H-1} \left( \prod_{k'=0}^{k} \frac{\pi_{\phi_1}(a_{k'}^1 | s_{k'}^1) \, \pi_{\phi_2}(a_{k'}^2 | s_{k'}^2)}{\perp\left(\pi_{\phi_1}(a_{k'}^1 | s_{k'}^1) \, \pi_{\phi_2}(a_{k'}^2 | s_{k'}^2)\right)} \right) r_k^1$$

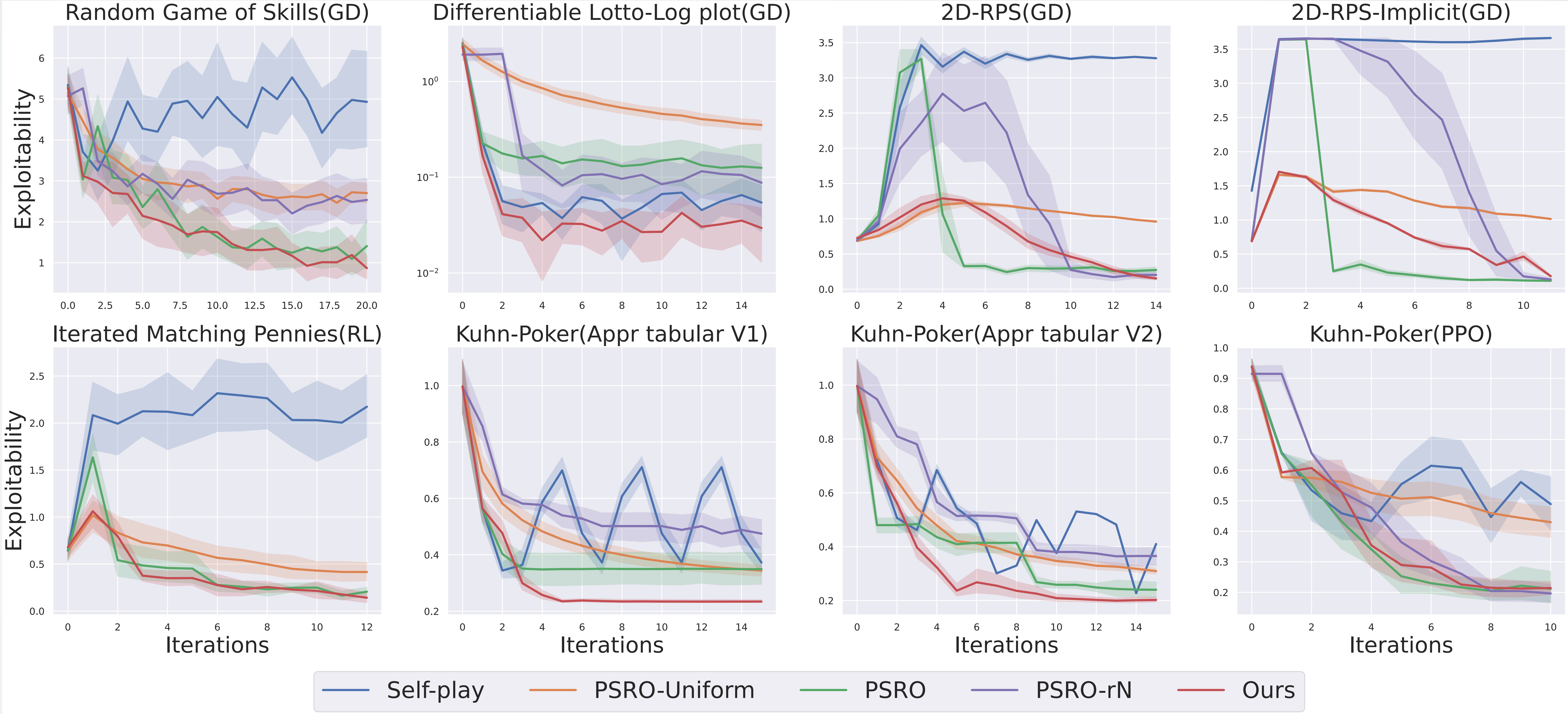
- general type of oracle (via ES)

$$\nabla_\theta \hat{J}_\sigma(\theta) = \mathbb{E}_{G \sim P(G), \epsilon \sim \mathcal{N}(0, I)}\left[\frac{1}{\sigma}\left(\mathfrak{Exp}_T(\pi_T, \Phi_T) \middle| \theta + \epsilon, G\right)\epsilon\right]$$

# Recent Advance (4): Auto-PSRO Result

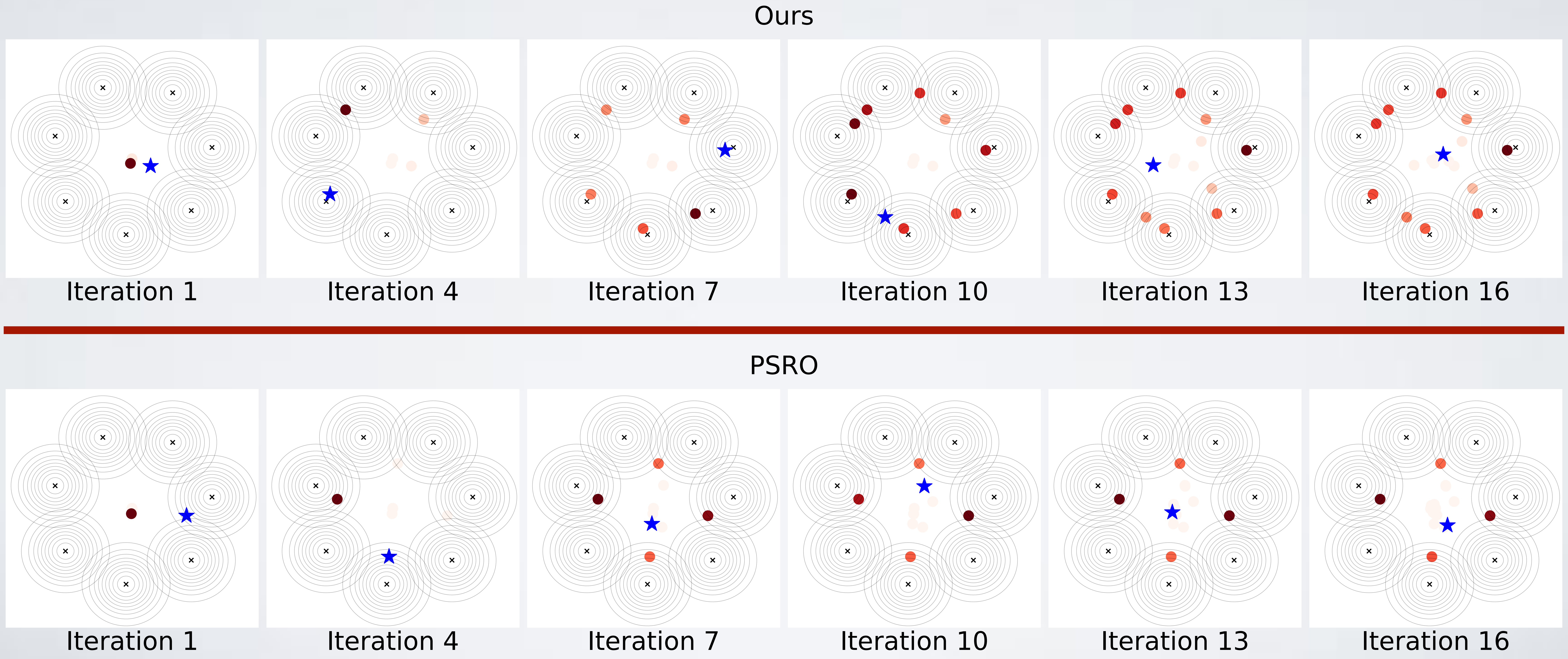
- **1st question**: is our method any good on the environments where it is trained?
  - Due to long-trajectory issues, we also focus on the *approximate* best-response setting

- Performance *at least* as good as baseline measures
- Outperforms PSRO in multiple settings



# Recent Advance (4): Auto-PSRO Result

- **2nd question**: What is the learned auto-curricula ?
  - Compare agents found and their respective densities in the meta-distribution



# Recent Advance (4): Auto-PSRO Result

- **3rd question**: Can the learned solver generalise over different games?
  - the most promising and striking aspect of LMAC - Train on small games and generalise to large game, e.g., train on Kukn Poker and test on Leduc Poker


Figure 5: (a) Exploitability when trained on Kuhn Poker with an exact tabular BR oracle using ES-LMAC and tested on Leduc Poker. (b) Same as (a) with approximate tabular BR V2 (c) Exploitability when trained on GoS with a GD oracle and tested on the AlphaStar meta-game from [8] (d) Final exploitability when trained on 200 Dimension GoS and tested on a variety of dimension size GoS.

# Additional Resources:

- If you want to know more details about PSRO and its variations, please refer to
  - Talk: https://www.bilibili.com/video/av969218959/
  - Slides: https://rlchina.org/lectures/lecture11.pdf
- A self-contained MARL survey from game theoretical perspective:
  - https://arxiv.org/abs/2011.00583
- If you want to get hands on to solving some two-player zero-sum games, e.g., Poker/Chess
  - https://arxiv.org/pdf/2103.00187.pdf
  - https://github.com/aicenter/openspiel_reproductions

# MALib: A Bespoke Library for Efficient PSRO Methods

https://github.com/sjtu-marl/malib