Lecture 9.2: Why PFSP?

The Learning Rules Behind Superhuman Self-Play: Accessible Companion Notes

CSCE 631-D: Intelligent Agents: Computational Game Solving
Alan Kuhnle, Texas A&M University

Fall 2026 (Module 9, Lecture 2; overall Lecture 21, Thursday, November 12, 2026)

Using these notes. These notes expand the Lecture 9.2 slides into readable prose. They give the derivations the slides only state (the $p \to \infty$ limit of PFSP; Hedge, replicator dynamics, and single-state NeuRD as one update; the R-NaD transformed reward and its fixed point; the closed form of magnetic mirror descent), pseudocode for NFSP and for PSRO with a pluggable meta-solver, a pacing plan, and suggested answers to the "pause and predict" prompts. Every figure and table is described in words. Headings follow the deck in order. Slide references use **PDF page numbers in the 47-page slide deck**, counting the title page and the maroon part dividers; the references occupy pages 45 to 47.

The question. After Lecture 9.1 a student asked: "Why PFSP? FSP was one of the weakest convergence algorithms we saw in online learning, with no theoretical rate. Why not mirror descent, FTRL, MWU?" The lecture answers in three steps. Prioritized fictitious self-play (PFSP) chooses *opponents*; it is not the rule by which the learner updates. The guarantees we know for fictitious play, Hedge, and CFR concern *averages*, which deep methods can approximate only at a cost. And since 2020 the field has in fact rebuilt the learner's update from the Module 3 toolkit, in regularized forms that make the *current* policy converge.

Figures. The deck shows nine images from primary sources: eight figures cropped from the papers' own PDFs and the 2026 CoG competition table reused from Lecture 9.1. It also has four diagrams drawn for this lecture. Each is described below at the slide where it appears. File, source, figure number, and credit for every image are in `m9-lecture2-figures.md`.

Source status. Every number, date, and venue below comes from a source opened while preparing the lecture; the claim-by-claim trace is in `m9-lecture2-claims.md`. Most sources are peer-reviewed: journals (*Nature*: AlphaStar, Ataraxos; *Science*: DeepNash) and conferences (FOCS, ICML, AAMAS, ICLR, NeurIPS, TMLR). Three are preprints: NFSP (arXiv), the Brandt, Fischer, and Harrenstein rate survey (a preprint version), and the two 2026 Kubíček, Lisý, and Sandholm papers. Gravel (NeurIPS 2026) is known to us from its abstract only. Pluto has no paper; we rely on the author's README, linked from the official competition report. Shapiro's 1958 paper was not opened; its bound is quoted from Brandt et al. Where these notes interpret, derive, or speculate, they say so.

Contents

1 Opening **3**

2 Part 0: Finishing Lecture 9.1 with Pluto **3**

3 Part I: Fictitious Play's Rate Story **5**

4 Part II: From Fictitious Play to PFSP **6**

5 Part III: Can You Average a Neural Network? **9**

6 Part IV: Whom to Play, and How to Update **10**

7 Part V: NeuRD, Hedge in Policy Space **11**

8 Part VI: Regularize to Converge (R-NaD and DeepNash) **12**

9 Part VII: Magnetic Mirror Descent and Ataraxos **14**

10 Part VIII: The 2026 Verdict, and the Answer **17**

A Pseudocode: Neural Fictitious Self-Play **19**

B Pseudocode: PSRO with a Pluggable Meta-Solver **20**

References **20**

Notation used in this lecture

- $M \in \mathbb{R}^{r \times s}$: payoff matrix of a two-player zero-sum (2p0s) matrix game, with r row and s column actions; Δ_r, Δ_s: the simplices of mixed strategies; $n = \max(r, s)$.
- $\bar{x}^t, \bar{y}^t$: empirical averages of the first t actions or strategies; g_t: the duality gap, the total amount the two players could gain by best-responding to $(\bar{x}^t, \bar{y}^t)$.
- π: a policy; π_t its value at iteration t; y: logits, with $\pi = \text{softmax}(y)$; $u_t(a)$: the utility of action a at round t (Module 3 used losses $\ell = -u$); $\bar{u}_t = \sum_a \pi_t(a) u_t(a)$. In sequential games, $q(s, a)$ is an action value and $v(s)$ a baseline.
- η: a stepsize (MWU, NeuRD, MMD). α: a regularization strength (R-NaD, MMD). The NFSP and DeepNash papers use η for their own parameters; we rename them η_{a} and α.
- A, B, $\mathcal{C}$: in PFSP, the learner, a frozen opponent, and the candidate set; $\omega_B = \Pr[A \text{ beats } B]$ (Lecture 9.1 wrote x_B); f: a weighting function; $p > 0$: the exponent in $f_{\text{hard}}(\omega) = (1 - \omega)^p$.
- Π_i: a population of policies for player i; U^{Π}: the empirical payoff table; σ_i: a meta-strategy (a mixture over Π_i). In NFSP, Π without a subscript is the average-policy network.
- π_{reg}: R-NaD's reference (regularization) policy; π_{fix}: the fixed point of the regularized dynamics; ρ: MMD's magnet policy. $D_{\text{KL}}(p \| q) = \sum_a p(a) \log(p(a)/q(a))$.
- $\text{expl}(\pi)$: exploitability, the amount a best-responding opponent gains against π (Module 5, Lecture 9.1).

1 Opening

1.1 Slide 1: Title

The title asks the student's question. The subtitle names the lecture's organizing idea: a self-play system makes two separate choices, *whom* the learner plays and *how* the learner updates. The tagline states the tension: fictitious play has no good rate, while Hedge, FTRL, and mirror descent have rates we proved in Module 3. Which rule did the superhuman systems run?

1.2 Slide 2: Learning Objectives and Plan

There are four objectives: know the status of fictitious play's rate; derive PFSP's $p \to \infty$ limit and say what PFSP does not guarantee; separate the opponent axis from the update axis; and write the NeuRD, R-NaD, and magnetic mirror descent (MMD) updates, saying which iterate each one makes converge.

Table, read as prose. The plan lists nine parts with minutes, 70 in total plus a 5-minute buffer: Part 0, finishing Lecture 9.1 with Pluto (7); Part I, fictitious play's rate (7); Part II, FSP, NFSP, and PFSP (10); Part III, the correction about averaging neural networks (6); Part IV, the two axes (6); Part V, NeuRD (7); Part VI, R-NaD and DeepNash (9); Part VII, magnetic mirror descent and Ataraxos (10); Part VIII, the 2026 verdict and the answer (8). A third column names the module each part recalls: Module 3 for fictitious play, MWU, Hedge, replicator dynamics, FTRL, and mirror descent; Module 4 for Kuhn's theorem and the sequence form; Module 5 for CFR averages and exploitability; Module 7 for PSRO.

Pacing plan for 75 minutes. Part 0, pages 3 to 7: 7 minutes (2 per content slide, 1 for the question). Part I, pages 8 to 11: 7 minutes, with 2 for the pause prompt. Part II, pages 12 to 17: 10 minutes; give the $p \to \infty$ slide 3 minutes. Part III, pages 18 to 20: 6 minutes. Part IV, pages 21 to 24: 6 minutes. Part V, pages 25 to 28: 7 minutes; do the softmax derivative on the board. Part VI, pages 29 to 32: 9 minutes. Part VII, pages 33 to 37: 10 minutes. Part VIII, pages 38 to 44: 8 minutes, leaving page 44 ("Where this is heading") as optional. Use the 5-minute buffer for discussion question 1. If time runs short, skip the pause prompt on page 24 and summarize page 40 in one sentence.

2 Part 0: Finishing Lecture 9.1 with Pluto

Lecture 9.1 did not reach its final part. Part 0 reuses its Pluto slide, its "Whom do you train against?" table, and a compressed synthesis, and then turns Pluto into the puzzle that motivates today.

2.1 Slide 4: Pluto (2026)

Pluto, by Vegard Mella, won the 2026 IEEE Conference on Games (CoG) StarCraft: Brood War AI competition: first in the round robin, then 4 to 1 over PurpleWave in the final. The organizer, David Churchill, calls it "the first Broodwar bot ever entered into a competition that was made completely by reinforcement learning." From the author's README (section "Model"): one policy network of 315M parameters, int8-quantized for CPU inference, "trained from scratch by pure self-play reinforcement learning (a PPO-style actor-critic): no replays, no imitation, no scripted behavior, no external bots. Both sides of every training game are played by the same current weights, never

older checkpoints, never other opponents." It plays with no search. It picks its opening at match time with a bandit over its record against each opponent.

Source status. Pluto has no paper. The competition report is an official organizer report; the README is author documentation. Neither is peer-reviewed. We use no press account of a ladder copy.

Figure description. The round-robin table from the competition report, first six of eleven columns: bot, games, wins, losses, win percentage, and bots beaten. Pluto is first with 2,844 wins and 40 losses in 2,884 games (98.61%) and beats all nine other bots. PurpleWave, Stardust, and BananaBrain follow in the organizer's ranking; the bottom rows are Venator and InfestedArtosis. A total row shows 14,425 round-robin games.

2.2 Slide 5: Whom Do You Train Against? Three Answers

Table, read as prose. Three columns compare AlphaStar (2019), OpenAI Five (2019), and Pluto (2026). Human data: AlphaStar imitates 971,000 replays before reinforcement learning; the other two use none. Opponents: AlphaStar trains in a league that mixes self-play (SP), PFSP, and exploiters; OpenAI Five plays its latest self in 80% of games and past versions in 20%; Pluto plays only its latest self. Learner update: AlphaStar uses an actor-critic method with V-trace and UPGO; OpenAI Five uses PPO; Pluto uses a PPO-style actor-critic. Diversity comes from exploiters and a strategy statistic in AlphaStar, from the past-opponent pool in OpenAI Five, and from sampled openings and a diversity bonus in Pluto. Sources: *Nature*; an arXiv preprint; a README and competition report.

Two terms need definitions. AlphaStar's *strategy statistic* z "summarises a strategy sampled from human data (for example, a build order)," and the policy is conditioned on it. Pluto's *diversity bonus* is "a small bonus for building less-frequently seen units and tech" among its zero-sum reward-shaping terms. The "learner update" row is new relative to Lecture 9.1; it anticipates Part IV.

2.3 Slide 6: Lecture 9.1 in One Table, and the Puzzle

Table, read as prose. Five rows from Lecture 9.1's synthesis. Libratus: a CFR blueprint with subgame solving (Nash and CFR, Modules 5 and 6). AlphaZero: Monte Carlo tree search plus self-play (minimax, policy iteration, Module 4). AlphaStar: imitation, then a league with PFSP (fictitious play and PSRO, Modules 3 and 7). OpenAI Five: PPO with 80/20 self-play (team games). Pluto: pure self-play with an opening bandit (limits of self-play; bandits).

The puzzle is AlphaStar's own finding. Its Fig. 3 caption says that naive self-play "has high Elo, but is more forgetful," where forgetting is measured as the minimum win rate against all past versions. Pluto uses only naive self-play and wins its competition by a wide margin. We do not know why (there is no paper). But the contrast suggests two separate questions: *whom* should the learner play, and *which learning rule* should it run? The rest of the lecture keeps them apart.

2.4 Slide 7: Today's Question

The slide quotes the student's question and previews the five-step answer. Treat the preview as a map: each claim is established later in the part named in parentheses.

3 Part I: Fictitious Play's Rate Story

3.1 Slide 9: Recall: Fictitious Play and Hedge

In a 2p0s matrix game with payoff matrix $M \in \mathbb{R}^{r\times s}$, fictitious play (FP) has each player, at each round, play a pure best response to the opponent's empirical average of past actions, $\bar{y}^t = \frac{1}{t}\sum_{\tau\le t} e_{j(\tau)}$ for the column player and similarly $\bar{x}^t$ for the row player. Ties among best responses are broken by some rule; Part I shows that the rule matters.

Let g_t be the duality gap of the averages: with the row player maximizing, $g_t = \max_i e_i^\top M\bar{y}^t - \min_j (\bar{x}^t)^\top M e_j$. It is nonnegative and is zero exactly at a Nash equilibrium. Robinson (1951) proved $g_t \to 0$ for every 2p0s game, without a rate in the statement (Daskalakis and Pan 2014, §I).

Module 3 gave a rate for Hedge (MWU). With payoffs scaled so that losses lie in $[0,1]$ (Module 3's standing assumption; otherwise the bound scales with the payoff range), if both players run MWU with stepsize $\eta = \sqrt{\ln n/T}$, $n = \max(r,s)$, each has regret at most about $2\sqrt{T\ln n}$, and the average strategy pair is an ε-Nash equilibrium with $\varepsilon \le 4\sqrt{\ln n/T}$ (M3, Lecture 9). Both results concern the *averages*. Module 3's matching-pennies example showed MWU's last iterates cycling (from 0.881 to 0.119 and back) while the averages converged, and Lecture 8 showed fictitious play cycling in Shapley's game, a general-sum example.

3.2 Slide 10: How Fast Does Fictitious Play Converge?

Table, read as prose. Four rows. Shapiro (1958): the game value converges at rate $O(t^{-1/(r+s-2)})$, an upper bound hiding a factor $a \cdot 2^{r+s}$ with a the largest absolute payoff. Karlin (1959): the conjecture that the true rate is $O(t^{-1/2})$, the rate no-regret methods achieve. Daskalakis and Pan (FOCS 2014): on the $n \times n$ identity matrix, FP may converge at rate $\Theta(t^{-1/n})$, but only when ties are broken arbitrarily; this refutes Karlin's strong conjecture. Abernethy, Lai, and Wibisono (2021): for diagonal payoff matrices with lexicographic tie-breaking, FP converges at $O(t^{-1/2})$, and this is tight; this proves the weak conjecture for that class only. A box below says the weak form remains open for general 2p0s matrices.

What each result says. Brandt, Fischer, and Harrenstein (2013 preprint, §1) summarize Shapiro: "in two-player constant-sum games the rate of convergence of fictitious play is at most $O(n^{-1/(r+s-2)})$, where n is the number of rounds and r and s are the numbers of actions available to the two players. Hidden in the asymptotic notation is an additional factor of $a \cdot 2^{r+s}$, where a is the largest absolute payoff value." They add that these bounds concern convergence of *payoff values*, not of strategies. We did not open Shapiro's paper (it is paywalled); the bound is cited through Brandt et al., and Daskalakis and Pan describe the same rate as the one that can be deduced by "unraveling" Robinson's induction.

Karlin's conjecture, as Daskalakis and Pan state it: "Fictitious play converges at rate $O(t^{-1/2})$ in all games" (their Conjecture 1, meaning 2p0s games). Their Theorem 1: "For every $n \ge 2$, fictitious play for I_n may converge at rate $\Theta(t^{-1/n})$, if ties are broken arbitrarily." Their Remark 1 names the two versions: with arbitrary tie-breaking the conjecture is Karlin's *strong* conjecture, which Theorem 1 disproves; with lexicographic or random tie-breaking it is the *weak* conjecture. Abernethy, Lai, and Wibisono prove the weak conjecture for diagonal matrices (abstract), and a matching lower bound in the identity case under lexicographic tie-breaking. The published version carries a 2021 SIAM copyright (doi:10.1137/1.9781611976465.84); we did not confirm which SIAM proceedings.

The comparison. For $r = s = 10$ Shapiro's guarantee is $t^{-1/18}$; Hedge's is $t^{-1/2}$ with only a $\sqrt{\ln n}$ dependence on the game size. So the student's premise is right: as a *theorem*, FP's rate is very weak.

3.3 Slide 11: In Practice, FP Often Looks Like $t^{-1/2}$

Figure description. A line plot from Abernethy et al. (2021, Fig. 2a). The horizontal axis is the iteration t from 0 to 10,000; the vertical axis is the duality gap from 0 to 500. A blue curve, "Fictitious Play," rises like a square root and stays just under an orange reference curve $y = 5\sqrt{t}$, ending near 470.

The paper's iteration uses action *counts* (unnormalized vectors $x^{(t)}$), so the plotted gap grows like $\sqrt{t}$. Dividing by t, the gap of the averages falls like $t^{-1/2}$. The curve is the maximum over 100 runs, each on a different random 10×10 Gaussian payoff matrix; the authors read it as suggesting "that Karlin's conjecture holds more generally" (§7). This is evidence, not proof.

Pause and predict: suggested answer. Every step of FP is a best response, but in a game like the identity matrix many actions tie as best responses. The tie-breaking rule decides which coordinates of the empirical average receive mass. Daskalakis and Pan build a valid FP execution in which these choices keep the players chasing each other slowly; their construction "folds" slow executions for I_2 inside executions for I_3, and so on, by induction. Their Remark 1 says it is "crucial for our construction that ties ... can be broken arbitrarily at each step."

4 Part II: From Fictitious Play to PFSP

4.1 Slide 13: Fictitious Self-Play (ICML 2015)

Heinrich, Lanctot, and Silver introduced two extensive-form versions of FP. *Full-width extensive-form FP* (XFP) updates behavioral strategies so that each iterate is realization-equivalent to normal-form FP (Module 4), so it "inherits its convergence guarantees"; its cost per iteration is linear in the number of game states instead of exponential. *Fictitious Self-Play* (FSP) replaces the two exact computations with machine learning: reinforcement learning for approximate best responses and supervised learning for the average strategy.

FSP is an instance of *generalized weakened fictitious play* (Leslie and Collins 2006, as stated in Heinrich et al., Definition 5):

$$\Pi^i_{t+1} \in (1 - \alpha_{t+1})\Pi^i_t + \alpha_{t+1}\big(b^i_{\epsilon_t}(\Pi^{-i}_t) + M^i_{t+1}\big),$$

where b^i_ϵ is the set of ϵ-best responses, with $\alpha_t \to 0$, $\epsilon_t \to 0$, $\sum_t \alpha_t = \infty$, and perturbations M_t that vanish in a specified averaged sense. Ordinary FP is the case $\alpha_t = 1/t$, $\epsilon_t = 0$, $M_t = 0$. The guarantee is asymptotic, and it needs the best-response error to vanish. The paper's conclusion (§6): "it remains an open question whether guaranteed convergence can be achieved with a finite computational budget per iteration." No rate is given; the paper cites Karlin's conjecture that FP's rate "might be $O(n^{-1/2})$."

4.2 Slide 14: Neural Fictitious Self-Play (NFSP)

Figure description. A diagram of one NFSP agent. Top: a gold box "M_{RL}: circular buffer of game transitions" feeds a red box "Best response Q, Q-learning." Bottom: a gold box "M_{SL}:

reservoir of own best-response (s, a)" feeds a green box "Average policy Π, supervised learning." Both networks feed a blue box "Act: ϵ-greedy(Q) with probability η_{a}, else Π."

NFSP (Heinrich and Silver 2016, an arXiv preprint) makes FSP deep. Each agent has two networks. The first learns an approximate best response by Q-learning from a circular replay memory M_{RL} of game transitions. The second, the average-policy network $\Pi(s, a \mid \theta^{\Pi})$, learns "a model that averages over the agent's own historical strategies" by supervised learning: it is trained "by optimizing the log-probability of past actions taken" from a memory M_{SL} to which the agent adds (s, a) only when it follows its best response. M_{SL} uses reservoir sampling, so it approximates a uniform sample of all past best-response behavior, which is exactly the average that FP needs.

The agent acts with the mixture $\sigma = (1-\eta_{\mathrm{a}})\Pi + \eta_{\mathrm{a}}\,\epsilon$-greedy$(Q)$, where the *anticipatory parameter* η_{a} (the paper's η) comes from the anticipatory dynamics of Shamma and Arslan: each agent best-responds to a short-term prediction of the opponents' average. In Leduc hold'em the paper uses $\eta_{\mathrm{a}} = 0.1$, memories of 200k (M_{RL}) and 2M (M_{SL}), and reaches exploitability 0.06, which XFP typically reaches after about 1,000 full-width iterations. Setting $\eta_{\mathrm{a}} = 1$ gives DQN with a passively learned average, and "DQN's average behaviour does not approach a Nash equilibrium either." In limit Texas hold'em NFSP was competitive with the top half of the 2014 computer poker competition agents. The paper states no convergence theorem. Pseudocode is in Appendix A.

4.3 Slide 15: PFSP, Recalled

Definition (Vinyals et al. 2019, Methods, author manuscript pp. 15 to 16). Given a learner A and a candidate set $\mathcal{C}$ of frozen opponents, PFSP samples B with probability

$$\Pr[B] = \frac{f(\omega_B)}{\sum_{C \in \mathcal{C}} f(\omega_C)}, \qquad \omega_B = \Pr[A \text{ beats } B], \tag{1}$$

for a weighting function $f : [0, 1] \to [0, \infty)$, provided some candidate has $f(\omega_C) > 0$. Uniform FSP is $f \equiv 1$. The default is $f_{\mathrm{hard}}(\omega) = (1 - \omega)^p$; since $f_{\mathrm{hard}}(1) = 0$,"no games are played against opponents that the agent already beats." The alternative $f_{\mathrm{var}}(\omega) = \omega(1 - \omega)$ prefers opponents near the learner's own level and is used "for main exploiters and struggling main agents." The paper's reading: "one can view f_{hard} as a form of smooth approximation of max-min optimisation, as opposed to max-avg that FSP imposes." The motivation is that "in highly non-transitive games like StarCraft, ... the pursuit of the mean win-rate might lead to policies that are easy to exploit."

4.4 Slide 16: Why "Max-Min"? The $p \to \infty$ Limit

Figure description. A bar chart of PFSP probabilities for three opponents that the learner beats with probability 0.9, 0.5, and 0.2. With $p = 1$ the bars are 0.07, 0.36, 0.57; with $p = 2$, 0.01, 0.28, 0.71; with $p = 8$, 0.00, 0.02, 0.98.

The slide makes the paper's phrase precise in two ways. Both are our derivation; the AlphaStar paper states the reading but not a proof.

Proposition 1 (concentration)**.** Let $\mathcal{C}$ be finite, $\omega_{\min} = \min_C \omega_C < 1$, and $\mathcal{C}^\star = \arg\min_C \omega_C$. Under f_{hard} with exponent p, $\Pr_p[B] \to 1/|\mathcal{C}^\star|$ for $B \in \mathcal{C}^\star$ and $\Pr_p[B] \to 0$ otherwise, as $p \to \infty$.

Proof. For $B \notin \mathcal{C}^\star$ and any $B' \in \mathcal{C}^\star$, $\Pr_p[B]/\Pr_p[B'] = \big((1 - \omega_B)/(1 - \omega_{\min})\big)^p$, and the base lies in $[0, 1)$ because $1 - \omega_B < 1 - \omega_{\min}$. So the ratio tends to 0. Opponents in $\mathcal{C}^\star$ have equal weights. $\square$

Proposition 2 (the objective PFSP ascends)**.** Let the learner have parameters θ, let each $\omega_B(\theta)$ be differentiable, and let the learner's expected update against the sampled opponent be $g_p(\theta) = \sum_B \Pr_p[B]\, \nabla\omega_B(\theta)$. Define $\Phi_q(\theta) = \|\mathbf{1} - \boldsymbol{\omega}(\theta)\|_q = \left(\sum_B (1-\omega_B)^q\right)^{1/q}$. If $\Phi_{p+1}(\theta) > 0$, then $-\nabla\Phi_{p+1}(\theta) = c\, g_p(\theta)$ for some $c > 0$.

Proof. Differentiating, $\nabla\Phi_q = -\Phi_q^{1-q} \sum_B (1-\omega_B)^{q-1} \nabla\omega_B$. With $q = p+1$ and $Z_p = \sum_C (1-\omega_C)^p$, $-\nabla\Phi_{p+1} = \Phi_{p+1}^{-p} Z_p\, g_p$, and $c = Z_p/\Phi_{p+1}^p > 0$. □

So the expected PFSP update is a steepest-descent direction (up to a positive factor) for the ℓ_{p+1} norm of the loss vector $\mathbf{1} - \boldsymbol{\omega}$. Two limits explain the paper's words.

- *Max-avg.* At $p = 0$ (uniform FSP), $\Phi_1 = \sum_B (1-\omega_B) = |\mathcal{C}|(1-\bar{\omega})$: minimizing it maximizes the average win rate $\bar{\omega}$.
- *Max-min.* For a finite pool, $\Phi_\infty \le \Phi_q \le |\mathcal{C}|^{1/q}\Phi_\infty$ with $\Phi_\infty = \max_B(1-\omega_B) = 1 - \min_B \omega_B$. So Φ_q is a smooth approximation of $1 - \min_B \omega_B$, with relative error at most $|\mathcal{C}|^{1/q} - 1 \approx (\ln|\mathcal{C}|)/q$. Minimizing it maximizes the worst win rate in the pool.

The variance weighting has the same structure. Since $\frac{d}{d\omega}\left(\frac{\omega^2}{2} - \frac{\omega^3}{3}\right) = \omega(1-\omega)$, f_{var} ascends $\sum_B h(\omega_B)$ with $h(\omega) = \omega^2/2 - \omega^3/3$, an increasing function whose slope peaks at $\omega = 1/2$: a curriculum that pushes all win rates up but learns most from even matches.

What this does not give. This is the idealized expected update with exact win rates and a fixed pool. The pool grows, win rates are estimated, main agents also play 35% self-play, and the learner's update is an actor-critic step, not exact gradient ascent. And the target is max-min *over the pool*, not a Nash equilibrium of the full game. One connection to Module 7 does hold (our addition): if the learner may mix over policies, ω is bilinear in the two mixtures, and by the minimax theorem $\max_A \min_{B\in\mathcal{C}} \omega_B(A) = \min_{\sigma\in\Delta(\mathcal{C})} \max_A \mathbb{E}_{B\sim\sigma}\omega_B(A)$: the max-min target is the value of best-responding to the hardest mixture of the pool, which is what a Nash meta-solver over the pool would compute.

Pause and predict: suggested answer. If A beats every candidate with probability 1, every f_{hard} weight is 0 and Eq. (1) is undefined (0/0). A rule needs a fallback, such as uniform sampling or self-play. AlphaStar's main agents do something similar for their reserved 15% share: "If there are no forgotten players or strong exploiters, the 15% is used for self-play instead."

4.5 Slide 17: Evidence for PFSP: AlphaStar's Ablation

Figure description. Three five-by-five heat maps from AlphaStar's Extended Data Fig. 5, titled "Relative Population Performance," "Min Max," and "Expected Win-rate of Last Agent." Rows and columns are Test agents, PFSP+SP, SP, PFSP, and FSP; each cell is a percentage, blue when the row beats the column and red when it loses. In the PFSP row and FSP column the values are 85%, 85%, and 89%; in the FSP row and PFSP column they are 15%, 15%, and 62%. The FSP row is red against every other row's method in the first two maps.

The caption: "PFSP based training outperforms FSP under all measures considered: it has a stronger population measured by relative population performance, provides a less exploitable solution, and has better final agent performance against the corresponding League." These are the multi-agent ablations of Fig. 3, run for 10^{10} steps with main agents only. This is strong empirical evidence. We found no published convergence theory for PFSP (the scout searched for one; see the claims table).

5 Part III: Can You Average a Neural Network?

5.1 Slide 19: Correction: Deep Methods Can Approximate the Average

A common claim is that "you can't average neural networks," so average-iterate guarantees do not transfer to deep learning. That is wrong as stated. The accurate statement is that the guarantees for MWU, FP, and CFR are about the *average* strategy, and there are three standard ways to approximate the average with networks.

1. **A second network trained on past behavior (NFSP).** The average-policy network Π is trained by supervised learning on a reservoir buffer of the agent's own best-response actions (Part II).
2. **A second network trained on weighted strategies (Deep CFR).** Brown, Lerer, Gross, and Sandholm (ICML 2019, §4) note that "it is the average strategy played over all iterations that converges to a Nash equilibrium." They keep "a separate memory M_Π of sampled infoset probability vectors"; whenever an infoset I is traversed, "the infoset probability vector $\sigma^t(I)$ is added to M_Π and assigned weight t" (Algorithm 1 initializes it as a reservoir-sampled memory), and a policy network is trained on the weighted loss. Alternatively one can store every iteration's value network; "during actual play, a value network is sampled randomly," which "eliminates the function approximation error of the final average policy network, but requires storing all prior value networks."
3. **A population.** A league's frozen snapshots, with mixture weights (uniform weights give FP's average), define a mixed strategy. It is played by sampling one snapshot at the start of each game. Under perfect recall, Kuhn's theorem (Module 4) says this mixed strategy has a realization-equivalent behavioral strategy, so the population *is* an average strategy in the game-theoretic sense.

5.2 Slide 20: What Averaging Costs, and What It Leaves Out

Averaging is possible, but it has costs, and the guarantees do not survive intact.

- **Costs.** A second network and its buffer; the supervised approximation error of fitting the average; and a lag. The uniform average of t iterates changes by a $1/(t+1)$ fraction when a new iterate arrives, so it reacts slowly late in training (our arithmetic).
- **The regret bounds do not cover the approximation.** Deep CFR's Theorem 1 bounds total regret by $(1+\frac{\sqrt{2}}{\sqrt{\rho K}})\Delta|\mathcal{I}_p|\sqrt{|A|}\sqrt{T}+4T|\mathcal{I}_p|\sqrt{|A|\Delta\epsilon_{\mathcal{L}}}$ with probability $1-\rho$, where $\epsilon_{\mathcal{L}}$ is the value network's excess loss over the best achievable. By its Corollary 1, average regret tends to a constant proportional to $\sqrt{\epsilon_{\mathcal{L}}}$, not to 0. The bound concerns the regret-matching iterates; the final strategy network adds its own error.
- **AlphaStar was not running no-regret learning on exact losses.** Its agents are updated "by the actor-critic reinforcement learning procedure" (V-trace, UPGO, TD(λ), plus a KL term toward the supervised policy) from sampled games. Even with a perfect average, the learner's update would not be Hedge. Part V makes this gap precise.
- **Last-iterate methods** take the other route: make the current network itself converge, so that no average is needed (Parts VI and VII).

The slide closes with Ataraxos (2026), whose damped self-play lets it "leverage standard policy optimization tools to regularize and control the size of its policy updates, obviating the need for more

onerous techniques, such as trajectory importance reweighting and policy averaging, for handling imperfect information" (arXiv v2, p. 13).

6 Part IV: Whom to Play, and How to Update

6.1 Slide 22: Two Axes

Figure description. A two-axis chart drawn for this lecture. The horizontal axis, "how the learner updates," has four columns: approximate best response; softmax policy gradient (PPO); Hedge in policy space (NeuRD); regularized (R-NaD, MMD). The vertical axis, "whom it plays," has four rows from bottom to top: latest self; uniform past (FSP); PFSP weights; Nash of the population. Boxes place the systems: NFSP at uniform past with approximate best response; PSRO and double oracle at Nash of the population with best response; AlphaStar at PFSP weights with policy gradient; Pluto at latest self with policy gradient; OpenAI Five slightly above Pluto, between the latest-self and uniform-past rows; DeepNash at latest self between the Hedge and regularized columns; Ataraxos at latest self in the regularized column.

This two-axis split is the *course's framing*, not a taxonomy from the literature. Its point is that the student's question mixes two choices. PFSP is a choice on the vertical axis. Mirror descent, FTRL, and MWU are choices on the horizontal axis. A system can pair any opponent rule with any update rule, and the 2022 to 2026 systems in Parts VI to VIII pair the simplest opponent rule (latest self) with a regularized update. The placements come from the papers: AlphaStar's league and actor-critic learner (Vinyals et al. 2019), OpenAI Five's 80/20 sampling and PPO (Berner et al. 2019, pp. 5, 59), Pluto's current-weights PPO-style self-play (README), DeepNash's NeuRD plus regularization (Perolat et al. 2022), and Ataraxos's damped self-play (Sokota et al. 2026).

6.2 Slide 23: PSRO: A Pluggable Meta-Solver

Figure description. A loop of four boxes: "population Π_i" feeds "payoff table U^{Π} (simulation)," which feeds a bold "meta-solver, mixture σ_{-i}," which feeds "oracle: RL best response to σ_{-i}," whose new policy π_i' is added back to the population. A side panel lists choices for σ_{-i}: latest policy gives self-play; uniform gives fictitious play; Nash of U^{Π} gives double oracle; regret matching, Hedge, and projected replicator dynamics (PRD), tried by Lanctot et al.; PFSP weights give something league-like.

PSRO (Lanctot et al. 2017, Algorithm 1) separates the two axes cleanly. The meta-solver is the opponent rule. Lanctot et al. note that independent RL and fictitious play are instances with $\sigma_{-i} = (0, \ldots, 0, 1)$ and the uniform mixture, that PSRO "generalizes Fictitious Self-Play," and that any meta-solver can be used; they tried regret matching, Hedge, and projected replicator dynamics (p. 4). The oracle's learning rule is separate: it can be DQN, policy gradient, or any of the regularized rules of Parts V to VII. AlphaStar's paper itself says PSRO "is related to League training, although League training specifies unique targets for approximate best responses (i.e. PFSP and exploiters)" (Methods, p. 13). Pseudocode with pluggable meta-solvers is in Appendix B.

6.3 Slide 24: Guarantees on the Opponent Axis

The opponent axis does have options with guarantees. *Online Double Oracle* (Dinh et al., TMLR 10/2022) combines no-regret learning with double oracle; it "is provably convergent to a Nash equilibrium," and each agent "can exploit a strategic adversary with a regret bound" that depends

on "k [which] is not the total number of pure strategies, but rather the size of effective strategy set" (abstract). *Anytime double oracle* and *anytime PSRO* (McAleer, Lanier, Wang, Baldi, Sandholm, and Fox, ICLR 2024) ensure that "exploitability does not increase between iterations"; *SP-PSRO* also adds "an approximately optimal stochastic policy into the population in each iteration" and "often converges much faster." PFSP is not one of these; it is a heuristic with an intuitive max-min reading.

Pause and predict: suggested answer. No. Plain PSRO and its deep variants "often increase exploitability between iterations, which is problematic when training in large games" (McAleer et al., abstract). The anytime variants were built to prevent exactly that.

7 Part V: NeuRD, Hedge in Policy Space

7.1 Slide 26: Softmax Policy Gradient Versus Hedge

This is the first derivation the notes promise: in one state with all actions observed, Hedge, discrete-time replicator dynamics, and tabular NeuRD are one update, and softmax policy gradient (SPG) is a different one.

Setting. One state, actions $a \in \mathcal{A}$, logits $y \in \mathbb{R}^{|\mathcal{A}|}$, policy $\pi(a) = e^{y(a)}/\sum_b e^{y(b)}$, and a utility vector u_t observed in full at each round (the all-actions setting). The softmax Jacobian is

$$\frac{\partial \pi(a')}{\partial y(a)} = \pi(a')[\mathbf{1}_{a'=a} - \pi(a)]. \tag{2}$$

Hedge. In utility form, Hedge sets $\pi_{t+1}(a) \propto \pi_t(a)e^{\eta u_t(a)}$, which in logits is $y_t = y_{t-1} + \eta u_t$ (Hennes et al., eq. (3)). Softmax is invariant to adding a constant to all logits, so the same policies come from $y_t(a) = y_{t-1}(a) + \eta[u_t(a) - \bar{u}_t]$. With $u = -\ell$ this is Module 3's MWU.

Replicator dynamics. Continuous-time replicator dynamics are $\dot{\pi}(a) = \pi(a)[u(a) - \bar{u}]$. They are exactly what the logit flow $\dot{y} = u$ produces: by Eq. (2), $\dot{\pi}(a) = \sum_b \frac{\partial \pi(a)}{\partial y(b)} \dot{y}(b) = \pi(a)\big[u(a) - \sum_b \pi(b)u(b)\big]$. Euler-discretizing the logit flow with step η gives back Hedge; Hennes et al. call this discrete-time replicator dynamics (their footnote: "given a tabular softmax policy, this definition matches the standard discrete-time RD").

Single-state tabular NeuRD. NeuRD's update (their eq. (10)) is

$$\theta_t = \theta_{t-1} + \eta \sum_a \nabla_\theta y(a; \theta_{t-1})[q(a) - v].$$

In the tabular single-state case $\theta = y$, so $\nabla_\theta y(a) = e_a$, and with $q = u_t$ and $v = \bar{u}_t$ the update is $y_t(a) = y_{t-1}(a) + \eta[u_t(a) - \bar{u}_t]$: Hedge again. This is their Statement 1: "Hedge $\equiv$ discrete-time RD $\equiv$ single state, all-actions, tabular NeuRD."

Softmax policy gradient. The all-actions PG update (their eq. (5)) replaces $\nabla_\theta y$ by $\nabla_\theta \pi$: $y_t = y_{t-1} + \eta \sum_{a'} \nabla_y \pi(a')[u_t(a') - \bar{u}_t]$. Its component a is, by Eq. (2),

$$\sum_{a'} \pi(a')[\mathbf{1}_{a'=a} - \pi(a)][u_t(a') - \bar{u}_t] = \pi(a)[u_t(a) - \bar{u}_t] - \pi(a) \underbrace{\sum_{a'} \pi(a')[u_t(a') - \bar{u}_t]}_{=0},$$

so $y_t(a) = y_{t-1}(a) + \eta\,\pi_t(a)[u_t(a) - \bar{u}_t]$ (their eq. (6)). The instantaneous regret is scaled by $\pi_t(a)$. In continuous time the same scaling gives $\dot{\pi}(a) = \pi(a)\big[\pi(a)A(a) - \sum_b \pi(b)^2 A(b)\big]$ with advantage A (their eq. (7)), which differs from replicator dynamics by the extra factor $\pi(a)$.

Why it matters. An action whose probability has become small learns slowly under SPG even after it becomes the best response, because its logit moves at rate proportional to its probability. Games are nonstationary from each player's view, so this happens often.

7.2 Slide 27: What the Factor $\pi_t(a)$ Costs: Regret

Figure description. A log-log plot (Hennes et al., Fig. 1) of regret against horizon length T from 10 to 10^6 in repeated matching pennies. Three series: SPG (crosses) grows slowly to about 15; SPG with a forfeit action (stars, with a dashed least-squares line) grows roughly linearly, past 10^5; Hedge with the forfeit action (dots) stays near 1.

The setting (§1.1): the "odd" player plays H for the first 40% of the rounds and T for the rest; the learner is the "even" player. Adding a useless "forfeit" action, whose reward is always -1, "drastically increases the regret growth-rate for SPG to linear," while Hedge's regret "remains constant."

Pause and predict: suggested answer. The forfeit logit falls fast at first. Once the forfeit action has low probability, SPG's update for its logit is scaled by that small probability, so it "is decreasing very slowly after the switch, as both the action probability and the instantaneous regret are small." This shrinks the gap between the largest and smallest logits less than it should. With the asymmetry the forfeit action creates, "the logit gap remains larger, thus leaving more probability mass on H" after the opponent has switched to T. Under Hedge the forfeit logit "keeps decreasing, thus maintaining a constant logit gap until T becomes the preferred action" (Hennes et al., §3.1, discussing their Fig. 2).

7.3 Slide 28: NeuRD: The One-Line Fix

With function approximation, NeuRD updates $\theta_t = \theta_{t-1} + \eta \sum_a \nabla_\theta y(s, a; \theta_{t-1})[q(s, a) - v(s)]$, where q is a learned critic and $v(s) = \sum_a \pi(s, a) q(s, a)$. Replacing $\nabla_\theta y$ by $\nabla_\theta \pi$ gives SPG; that is the "one-line fix" (Algorithm 1, line 10), which "simply bypasses the gradient step through the softmax." In sequential tabular games, running NeuRD at each information state inside CFR is CFR with Hedge, so CFR's regret bound applies and the average converges to an ε-Nash equilibrium (§3.3). Convergence is in the *time average*; DeepNash summarizes NeuRD as "proven to converge to a Nash equilibrium in the time average" (p. 24). Venue: AAMAS 2020, p. 492.

8 Part VI: Regularize to Converge (R-NaD and DeepNash)

8.1 Slide 30: The Problem: FTRL Cycles; Regularization Stops It

Figure description. Four phase portraits (Perolat et al. 2021, Fig. 1) on the unit square; the horizontal axis is player 2's probability of action 1 and the vertical axis player 1's. Panel (a), plain FoReL: closed orbits circle the equilibrium, a red dot near $(0.85, 0.85)$. Panels (b), (c), (d), with reward transforms at strength multipliers 0.5, 1, and 10: trajectories spiral inward (b, c) or flow directly (d) to a single rest point, which sits farther from the red dot as the strength grows.

The game is biased matching pennies, with first-player payoffs $\left[\begin{smallmatrix} 1 & -1 \\ -1 & 10 \end{smallmatrix}\right]$; its unique equilibrium has both players play action 1 with probability $11/13 \approx 0.846$ (a short indifference calculation). FoReL, "follow the regularized leader," is the continuous-time version of Module 3's FTRL, and replicator dynamics are its entropic special case. Perolat et al. generalize Poincaré recurrence (orbits return arbitrarily close to where they started) from normal-form to sequential zero-sum games: the last iterate does not converge when the equilibrium is interior. Their introduction gives the motivation for everything that follows: average-time guarantees "are not conducive to learning in large games, which rely on general function approximation techniques (e.g., deep neural networks) that are inherently difficult to time-average." Part III's correction applies: "difficult," not impossible.

8.2 Slide 31: R-NaD: Transform, Converge, Move the Reference

Figure description. DeepNash Fig. 2c: three rows, R-NaD iterations 0, 1, and 2 on matching pennies. On the left of each row, replicator trajectories on the unit square ($\pi^1(H)$ horizontal, $\pi^2(H)$ vertical) spiral into a fixed point; in iteration 0 it is near the corner set by the reference policy, and in later iterations it moves toward the center. On the right, the matching Lyapunov surface, with the fixed point marked as a red dot and passed down a dotted arrow as the next reference policy $\pi_{n+1,\text{reg}}$.

This is the second derivation: the transformed reward, its fixed point, and the reference update, for a 2p0s normal-form game. We follow DeepNash §2.2, renaming the paper's regularization parameter η to α.

Step 1: reward transformation. With reference policies π^i_{reg} and $\alpha > 0$, player i's reward becomes

$$\tilde{r}_i(\pi, a^i, a^{-i}) = r_i(a^i, a^{-i}) - \alpha \log \frac{\pi^i(a^i)}{\pi^i_{\text{reg}}(a^i)} + \alpha \log \frac{\pi^{-i}(a^{-i})}{\pi^{-i}_{\text{reg}}(a^{-i})}. \tag{3}$$

The reward is policy-dependent. Adding Eq. (3) for $i = 1, 2$, the log terms cancel in pairs, so $\tilde{r}_1 + \tilde{r}_2 = r_1 + r_2 = 0$: the transformed game is still zero-sum. In expectation, player i's payoff is

$$V_i(\pi) = \langle \pi^i, Q^i_\pi \rangle - \alpha D_{\text{KL}}(\pi^i \| \pi^i_{\text{reg}}) + \alpha D_{\text{KL}}(\pi^{-i} \| \pi^{-i}_{\text{reg}}), \qquad Q^i_\pi(a) = \mathbb{E}_{a^{-i} \sim \pi^{-i}} r_i(a, a^{-i}).$$

Step 2: the fixed point. The last term does not depend on π^i, so player i's best response maximizes $\langle \pi^i, Q^i \rangle - \alpha D_{\text{KL}}(\pi^i \| \pi^i_{\text{reg}})$. By the Gibbs variational principle (set the derivative of the Lagrangian to zero) the maximizer is

$$\pi^i(a) \propto \pi^i_{\text{reg}}(a) \exp\big(Q^i_\pi(a)/\alpha\big). \tag{4}$$

The KL terms make the regularized game strictly concave for each player in its own strategy, which is why its equilibrium π_{fix} is unique. The interior rest points of replicator dynamics on the transformed game satisfy the same equation: replicator dynamics stop when every action's transformed fitness $Q^i(a) - \alpha \log(\pi^i(a)/\pi^i_{\text{reg}}(a))$ is the same, which rearranges to Eq. (4). DeepNash states that "this system has a unique fixed point π_{fix} and convergence to it is guaranteed, which can be proven by the Lyapunov function" $H_{\pi_{\text{fix}}}(\pi) = \sum_i \sum_{a^i} \pi^i_{\text{fix}}(a^i) \log\big(\pi^i_{\text{fix}}(a^i)/\pi^i(a^i)\big)$, a sum of KL divergences; Perolat et al. (2021, Thm 5.2) show it decays exponentially in continuous time. "However, this fixed point is not yet a Nash equilibrium of the original game."

Step 3: move the reference. Set $\pi_{n+1,\text{reg}} = \pi_{n,\text{fix}}$ and repeat. The fixed points "can be proven to converge to a Nash equilibrium of the original (unmodified) game." Our reading: the outer loop is a proximal-point method with KL proximity, since each round solves the original game plus a KL penalty toward the previous solution.

Example 1 (matching pennies, checked numerically)**.** DeepNash starts from $\pi_{0,\text{reg}}[H, T] = [0.999, 0.001]$ for both players with strength 0.2 and reports the first fixed point as $[0.896, 0.104]$ and $[0.263, 0.737]$. Solving Eq. (4) for both players numerically (our computation) gives $[0.897, 0.103]$ and $[0.263, 0.737]$, matching to rounding. Iterating the reference update, our computation reaches $[0.472, 0.528]$, $[0.386, 0.614]$ after one more iteration and is within 0.01 of the uniform equilibrium after three.

8.3 Slide 32: What Is Proven, and What DeepNash Achieved

Theorem 6.1 (Perolat et al. 2021): "In a monotone game with all Nash equilibrium being interior, the sequence of policy $\{\pi_k\}_{k\geq 0}$ ... converges to a Nash equilibrium of the game." Three qualifications. *Monotone* is their Definition 3.2, a condition on paired deviations that holds for 2p0s games and zero-sum polymatrix games (their Appendix I). *Interior* means every action has positive probability at every equilibrium; the authors "were only able to prove this result for interior Nash but we conjecture that it is still true for non interior Nash equilibrium." And the inner convergence results (Theorems 5.1 and 5.2) are for the *continuous-time* dynamics: "exponential convergence rates can be achieved in continuous time." Nothing is proven for the deep implementation. ICML 2021, PMLR 139: 8525 to 8535.

DeepNash (Perolat, De Vylder, et al., *Science* 378(6623): 990 to 996, 2022) scales R-NaD with a deep network. It "uses NeuRD combined with the regularization idea ... to converge in the last iterate" (p. 24). At scale each reference update uses an approximate fixed point: "after a fixed number of learning steps, an approximate fixed point policy $\pi_{m,\text{fix}}$ is obtained, which is then used as the next regularisation policy" (p. 11). DeepNash also fine-tunes and post-processes its policy to remove very improbable actions (p. 9). Results: a win rate of over 97% against existing Stratego bots, and on the Gravon platform 42 wins in 50 ranked matches (84%) from early April 2022, a top-3 rank on the 2022 and all-time leaderboards (pp. 5, 12).

9 Part VII: Magnetic Mirror Descent and Ataraxos

9.1 Slide 34: Magnetic Mirror Descent (ICLR 2023)

Sokota, D'Orazio, Kolter, Loizou, Lanctot, Mitliagkas, Brown, and Kroer define MMD as a non-Euclidean proximal gradient method for variational inequalities with a regularizer. In reinforcement-learning form (their eq. (10)), with stepsize η, temperature α, action values q_t, and magnet ρ:

$$\pi_{t+1} = \arg\max_{\pi} \; \mathbb{E}_{a\sim\pi} q_t(a) - \alpha D_{\text{KL}}(\pi\|\rho) - \tfrac{1}{\eta} D_{\text{KL}}(\pi\|\pi_t). \tag{5}$$

In their words, the stepsize "dictates the extent to which moving away from the current iterate is penalized," and the temperature "the extent to which being far away from the magnet ... is penalized."

Closed form (third derivation). Maximize over the simplex with a multiplier λ. Setting the derivative in $\pi(a)$ to zero,

$$q_t(a) - \alpha\Big[\log\tfrac{\pi(a)}{\rho(a)} + 1\Big] - \tfrac{1}{\eta}\Big[\log\tfrac{\pi(a)}{\pi_t(a)} + 1\Big] - \lambda = 0.$$

Multiply by η and collect the $\log \pi(a)$ terms: $(1+\alpha\eta)\log\pi(a) = \eta q_t(a) + \alpha\eta \log \rho(a) + \log \pi_t(a) + c$, with c independent of a. Hence

$$\pi_{t+1}(a) \propto \left[\pi_t(a)\, \rho(a)^{\alpha\eta}\, e^{\eta q_t(a)}\right]^{1/(1+\alpha\eta)}, \tag{6}$$

their eq. (12). Three readings.

- $\alpha = 0$: $\pi_{t+1} \propto \pi_t e^{\eta q_t}$, which is Hedge, that is, mirror descent with the entropy mirror map (Module 3).
- The new policy is a weighted geometric mean of three policies: the current one, the magnet, and the softmax of the action values.
- Fixed point: setting $\pi_{t+1} = \pi_t = \pi^\star$ in Eq. (6), $\alpha\eta \log \pi^\star = \alpha\eta \log \rho + \eta q + c$, so $\pi^\star \propto \rho\, e^{q/\alpha}$ (our derivation). When both players do this with q evaluated at the other's fixed point, and ρ is uniform, this is the *logit quantal response equilibrium* (QRE) with temperature α; the paper writes it as $x^\star \propto \exp(-Ay^\star/\alpha)$, $y^\star \propto \exp(A^\top x^\star/\alpha)$ for a loss matrix A (§2.3). Compare R-NaD's Eq. (4): MMD's magnet plays the role of R-NaD's reference policy.

QRE, defined. A logit QRE is a profile in which each player plays a softmax of its expected action values against the others, with a common temperature. As the temperature goes to 0 it approaches a Nash equilibrium; at high temperature it approaches uniform play. It is the entropy-regularized Nash equilibrium, which is why a solver for regularized games finds it.

Theorem 3.4. Under their Assumptions 3.2 and 3.3 (the game operator F is monotone and L-smooth, the regularizer is 1-strongly convex relative to the mirror map) and if the solution is interior, the algorithm "converges if $\eta \le \alpha/L^2$ and guarantees" $B_\psi(z_*; z_{t+1}) \le \left(\frac{1}{1+\eta\alpha}\right)^t B_\psi(z_*; z_1)$, where $B_\psi(x;y) = \psi(x) - \psi(y) - \langle \nabla\psi(y), x - y\rangle$ is the Bregman divergence of the mirror map ψ (the KL divergence when ψ is negative entropy). For a matrix game, Proposition 2.4 gives $L = \max_{ij} |A_{ij}|$. This is a rate for the *last iterate*, toward the QRE, not toward the Nash equilibrium. The abstract calls MMD "the first quantal response equilibria solver to achieve linear convergence for extensive-form games with first order feedback." "Note $\alpha > 0$ is necessary to converge to the solution. If $\alpha = 0$ ... Algorithm 3.1 with $\psi(z) = \frac{1}{2}\|z\|^2$ becomes projected gradient descent ascent, which is known to diverge or cycle for any positive stepsize."

Example 2 (MMD on biased matching pennies, our computation)**.** Take Perolat's biased matching pennies with the row player maximizing, uniform magnet, and $\alpha = 1$, so $L = 10$ and the theorem's stepsize bound is $\alpha/L^2 = 0.01$. The α-QRE is $x = (0.675, 0.325)$, $y = (0.902, 0.098)$, far from the Nash equilibrium $(0.846, 0.846)$: the temperature is high. Running Eq. (6) for both players from uniform with $\eta = 0.01$, the summed KL divergence to the QRE falls from 0.44 to 6×10^{-2} after 100 steps, 3×10^{-5} after 500, and 2×10^{-9} after 1,000: a straight line on a log scale. With $\eta = 0.05$, outside the theorem's condition, it still converges, faster (the condition is sufficient, not necessary).

9.2 Slide 35: MMD on a Perturbed Rock-Paper-Scissors

Figure description. Three panels from Sokota et al. (2023), Fig. 4. Left, "KL Divergence to QRE": on a log scale from 10^{-1} to 10^{-15} against iterations 0 to 1,000, four straight lines for temperatures 0.2, 0.5, 1.0, and 2.0; higher temperatures fall faster (2.0 reaches 10^{-15} within about 30 iterations; 0.2 is near 10^{-13} at 1,000). Middle, "Simplex Trajectories": a triangle with an inset showing colored

trajectories spiralling into different points, one per temperature, marked with stars. Right, "Payoff Matrix": rows and columns R, P, S with entries $0, -1, 3$; $1, 0, -3$; $-3, 3, 0$.

The straight lines are linear convergence of the last iterate. The temperature trades speed for bias: a high temperature converges quickly to a QRE far from Nash, a low one slowly to a QRE close to Nash.

Pause and predict: suggested answer. The stepsize condition $\eta \leq \alpha/L^2$ fails first: as α shrinks with η fixed, the theorem stops applying. Even with η shrunk to α/L^2, the contraction factor $1/(1+\eta\alpha) = 1/(1+\alpha^2/L^2)$ tends to 1, so the number of steps to accuracy ε grows like $(L^2/\alpha^2)\ln(1/\varepsilon)$. At $\alpha = 0$ the update is Hedge (or projected gradient descent ascent in the Euclidean case), and the last iterate cycles. Ataraxos's schedule shrinks regularization and update size together for this reason.

9.3 Slide 36: Ataraxos (Nature, 2026)

Sokota, Vinitsky, Hu, Fan, Kolter, and Farina (*Nature*, published September 30, 2026, `doi:10.1038/s41586-026-11036-y`; we read arXiv:2511.07312v2) build Stratego AI from three parts: a policy-value network trained by self-play, a belief network that models hidden information, and test-time search. "The core innovation that enables effective reinforcement learning is a coordination between regularization strength and policy update size," citing MMD. "Ataraxos employs stronger regularization and more aggressive updates early in self-play, and weaker regularization and smaller policy updates late," because "keeping update sizes commensurate with regularization strength damps the otherwise cyclical, divergent, or chaotic learning dynamics engendered by imperfect information" (p. 4).

The details (p. 13): for regularization, "a myopic reverse KL penalty towards the policy that selects a movable piece uniformly at random and then selects a legal move for this piece uniformly at random" for the move network, and an entropy term for the setup network, with coefficients annealed by power laws. For update size, four mechanisms: "a reverse KL penalty to the data collection policy, importance ratio clipping, gradient norm clipping, and the learning rate of Adam," with the learning rate also annealed by a power law. At test time (p. 15), Ataraxos samples hidden states from its belief network, estimates candidate-move values by rollouts, and updates "with a tabular step of magnetic mirror descent, which regularizes and controls the size of the update with the same two reverse KL divergences used during training," applied to the current decision only. So Ataraxos is *not* plain MMD. It is MMD-style damped self-play, plus a belief model, plus a one-step MMD search.

9.4 Slide 37: Ataraxos Against Pim Niemeijer

Figure description. Twenty small line plots (Sokota et al. 2026, Fig. 3), one per game against Pim Niemeijer, each showing the network's predicted effective win probability from 0 to 1 against the percentage of the game played. Each panel is labelled with the result and the number of moves. Fifteen games are wins (blue curves that mostly rise from about 0.5 to 1), game 6 is a loss (an orange curve that falls to 0 near the end), and games 11, 14, 17, and 19 are draws (purple curves that end near 0.5).

The paper calls Niemeijer "the most decorated Stratego player of all time" and reports 15 wins, 1 loss, and 4 draws. Training used 16 H100s for one week plus 4 H100s for four days, which "costs less than $8,000 at 2025 prices." The authors estimate that DeepNash's training run, on 1,024 TPU

nodes for two to three months by the recollection of DeepNash's corresponding author, would cost "roughly between $3,000,000 and $4,500,000" at 2025 prices; DeepNash consumed "about 5.5 billion games" and Ataraxos "about 160 million games" (cost section, p. 25). These comparisons are the Ataraxos authors' estimates, not DeepNash's.

10 Part VIII: The 2026 Verdict, and the Answer

10.1 Slide 39: Rudolph et al. (ICLR 2026)

Figure description. Five panels of box-and-whisker plots (Rudolph et al. 2026, Fig. 2), one per game: 2D5F Liar's Dice, 3x3 Abrupt Dark Hex, 3x3 Dark Hex, Abrupt Phantom Tic-Tac-Toe, and Phantom Tic-Tac-Toe. In each, seven algorithms (MMD, PPO, PPG, R-NaD, NFSP, ESCHER, PSRO) have a pair of boxes showing final exploitability on a square-root scale from 0 to 1 over the hyperparameter tuning runs (left) and the evaluation runs (right). A red dashed line marks the exploitability of the uniform policy. In most panels the evaluation boxes of MMD, PPO, and PPG sit lowest; PSRO and ESCHER sit near or above the dashed line.

Rudolph, Lichtlé, Mohammadpour, Bayen, Kolter, Zhang, Farina, Vinitsky, and Sokota hypothesize that "with proper tuning, generic PG methods are competitive with or superior to FP-, DO-, and CFR-based DRL approaches." They implement exact exploitability for five games with "millions of infostates" and run "7000 total runs across 350 hyperparameter configurations, requiring over 345,000 CPU hours." Result: "NFSP, PSRO, ESCHER, and R-NaD fail to outperform the generic PG methods (MMD, PPO, PPG)." In more detail, "ESCHER was uniformly noncompetitive," PSRO was "largely noncompetitive," and "NFSP and R-NaD approached or matched the performance of generic PG methods in select cases, but typically underperformed them." The authors' own limitations (§6): five games, four "somewhat homogeneous"; one algorithm from each family (the original NFSP, the original PSRO, and one CFR-based method); their hyperparameter choices; and evaluation at 10 million steps.

Venue. The saved copy is arXiv:2502.08938v4 (May 27, 2026), whose header reads "Published as a conference paper at ICLR 2026." The ICLR 2026 proceedings page lists the paper with the same abstract, so we cite it as ICLR 2026, peer-reviewed.

How to read it. The result is less surprising after Parts V to VII. MMD is itself a policy-gradient method with regularization and a step control, and Rudolph et al. argue that PPO's improvement step does the same three things: it maximizes expected value, regularizes the policy, and controls the size of the update (§1). R-NaD, the method behind DeepNash, did not beat these simpler methods on these games. This does not contradict DeepNash's Stratego result; it says that on these benchmarks the machinery was not what made the difference.

10.2 Slide 40: Theory Is Catching Up

Kalogiannis and Farina (NeurIPS 2025) prove that "regularized alternating policy gradient with (i) direct policy parametrization, (ii) softmax policy parametrization, and (iii) softmax policy parametrization with natural policy gradient updates converge to an approximate Nash equilibrium (NE) in the last-iterate in imperfect-information perfect-recall zero-sum EFGs." (*Gradient dominance* means that the suboptimality of a policy is bounded by a function of its gradient norm, so a stationary point is a global optimum.) The key observation is that "since the individual utilities are concave

with respect to the sequence-form strategy, they satisfy gradient dominance w.r.t. the behavioral strategy"; the regularized utility then satisfies a proximal Polyak-Łojasiewicz condition, and the methods reach an ϵ-approximate equilibrium "with a number of iterations and trajectory samples that are polynomial in $1/\epsilon$ and the natural parameters of the game." We read the proceedings abstract only. These are tabular results; no theorem covers a deep network.

10.3 Slide 41: Closing Table

Table, read as prose. Seven rows and six columns: method; what it averages or regularizes; which iterate converges; rate; assumptions; deep-scale evidence. FP and XFP average best responses; the average converges; the rate is open, and $t^{-1/n}$ is possible with adversarial ties; they assume exact best responses; there is no direct deep-scale evidence. FSP and NFSP learn an average Π; the average converges (asymptotically, for FSP); no rate is given; FSP needs $\epsilon_t \to 0$; NFSP has Leduc and limit hold'em results. PFSP sets only opponent weights; no convergence is claimed; there is no rate and no stated assumption; its evidence is AlphaStar's ablation. Hedge and NeuRD: the average converges at $O(\sqrt{\ln n/T})$ in the tabular, all-actions setting; NeuRD is part of DeepNash. R-NaD regularizes by KL toward π_{reg}, which then moves; the last iterate converges, with an exponential inner rate in continuous time; it assumes a monotone game with interior equilibria; its evidence is DeepNash in Stratego. MMD regularizes by KL toward a magnet ρ; the last iterate converges linearly to a QRE; it assumes a monotone operator and $\eta \leq \alpha/L^2$; its evidence is Ataraxos, which adds search. Regularized policy gradient regularizes the utility; the last iterate converges in poly$(1/\epsilon)$ iterations in perfect-recall 2p0s games (Kalogiannis and Farina); PPO and MMD are the strongest methods in Rudolph et al.

10.4 Slide 42: So: Why PFSP, and Why Not MWU?

The answer has three parts.

1. **PFSP picks opponents; it is not the learner.** Its $p \to \infty$ limit trains against the hardest opponents in the pool (Proposition 1), and its expected update descends a smooth surrogate for the worst win rate (Proposition 2). It has strong ablation evidence and no published convergence theory. FSP's weakness as an *update* rule is real, but PFSP was never meant as one.
2. **In 2019 the deep learner could not cheaply be MWU.** Softmax policy gradient is not Hedge, as NeuRD showed in 2019 and 2020; and Hedge's guarantee is for an average, which a deep system can approximate only with a second network, buffer, and lag, outside the regret bound.
3. **The field did move to Module 3's toolkit,** in a form that makes the *current* policy converge: regularize toward a reference or magnet, and control the step (R-NaD, MMD, Ataraxos). In 2026, tuned generic policy gradient matched or beat the FP-, PSRO-, and CFR-based deep methods on exact-exploitability benchmarks.

10.5 Slide 43: Discussion Questions

Suggested directions for each question.

1. **Why PFSP for AlphaStar and pure self-play for Pluto?** No published analysis answers this, so any answer is a hypothesis. On the opponent axis, AlphaStar's league and exploiters supply the width that keeps a policy from forgetting counter-strategies; Pluto gets some width

from sampled openings and its diversity bonus. On the update axis, a PPO-style learner already regularizes and limits its step size (the property Rudolph et al. stress), which may reduce the cycling that leagues were built to fight. Students should notice that Pluto's evidence is a competition against other bots, not an exploitability measurement (Lecture 9.1, Part IV).

2. **The league as a meta-solver.** Per learner, the opponent distribution is 0.35 on the latest self, 0.50 PFSP over the pool, and 0.15 PFSP over forgotten players and exploiters (main agents); a PSRO meta-solver can output exactly this mixture. Online Double Oracle's no-regret property or anytime PSRO's monotone exploitability would be candidates to borrow; the cost is that both need meta-game payoffs and restricted-game solves at league scale.
3. **Magnets for LLM agents.** In KL-regularized RL fine-tuning the reference model plays the role of the magnet ρ, and the KL coefficient plays the role of α; annealing α trades staying near the reference against optimizing the reward. (The scout found MMD-based alignment work, Magnetic Preference Optimization, which we did not read for this lecture.)

10.6 Slide 44: Where This Is Heading

Three 2026 directions combine regularization with search. **Gravel** (Mohammadpour, Sokota, Kaplowitz, Kolter, Brown, and Farina; NeurIPS 2026, to appear) is a public-belief-state search method that "achieves game-theoretic soundness via regularization"; the abstract reports that it outperforms ReBeL "both in head-to-head competitions against Slumbot and in endgame solving." We have the abstract only. Kubíček, Lisý, and Sandholm's **test-time reinforcement learning** preprint represents the re-solving gadget game "implicitly by modified sampling and neural policy rather then explicitly by constructing it," and reports "a win rate of over 80% in Battleship" against an opponent without test-time reasoning. Their **refinements** preprint observes that gadget games "typically contain infinitely many Nash equilibria" and that choosing refined equilibria can "reduce exploitability by more than 50%." The slide's "what comes next" box follows the syllabus: Weeks 13 and 14 hold no new lectures (PA5 and projects, with presentations in Week 14).

A Pseudocode: Neural Fictitious Self-Play

Following Heinrich and Silver (2016), Algorithm 1, for one agent; every player runs the same loop.

Input: anticipatory parameter η_{a} (0.1 in Leduc); memory sizes (200k for M_{RL}, 2M for M_{SL}); learning rates (0.1 for RL, 0.005 for SL in Leduc).

1. Initialize M_{RL} (circular buffer) and M_{SL} (reservoir). Initialize the action-value network $Q(s, a \mid \theta^{Q})$, a target network $\theta^{Q'} \leftarrow \theta^{Q}$, and the average-policy network $\Pi(s, a \mid \theta^{\Pi})$.
2. For each episode:
   (a) Set the policy $\sigma \leftarrow \epsilon$-greedy$(Q)$ with probability η_{a}; otherwise $\sigma \leftarrow \Pi$.
   (b) At each step: observe s_t and r_t; sample $a_t \sim \sigma(s_t)$; store $(s_t, a_t, r_{t+1}, s_{t+1})$ in M_{RL}.
   (c) If σ is the best-response policy, store (s_t, a_t) in M_{SL} by reservoir sampling.
   (d) Periodically: update θ^{Π} by SGD on the cross-entropy loss $-\log \Pi(a \mid s)$ over a minibatch from M_{SL}; update θ^{Q} by SGD on the Q-learning loss $\big(r + \max_{a'} Q(s', a' \mid \theta^{Q'}) - Q(s, a \mid \theta^{Q})\big)^2$ over a minibatch from M_{RL}; every few hundred updates (300 in Leduc) copy $\theta^{Q'} \leftarrow \theta^{Q}$.
3. Output Π as the approximate equilibrium strategy.

In the paper's Leduc experiments each agent ran 2 stochastic gradient updates of minibatch size 128 per network every 128 game steps, and the ϵ-greedy exploration started at 0.06 and decayed to 0.

B Pseudocode: PSRO with a Pluggable Meta-Solver

Following Lanctot et al. (2017), Algorithm 1, with the meta-solver as a parameter. The menu of meta-solvers lists what each choice is called in the lecture; the PFSP entry is our illustration of how a league-style opponent rule fits this interface.

> **Input:** initial policy sets Π_i (for example, one uniform random policy per player); a meta-solver MS; a best-response oracle ORACLE (any RL algorithm, with any of the update rules of Parts V to VII).
>
> 1. Compute the empirical payoff table U^{Π} by simulating every profile of policies in Π.
> 2. $\sigma \leftarrow \mathrm{MS}(U^{\Pi})$ (one mixture σ_i over Π_i per player).
> 3. Repeat for each epoch:
>   (a) For each player i: train $\pi_i' \leftarrow \mathrm{ORACLE}(i,\ \pi_{-i} \sim \sigma_{-i})$ and add π_i' to Π_i.
>   (b) Simulate the new entries of U^{Π}.
>   (c) $\sigma \leftarrow \mathrm{MS}(U^{\Pi})$.
> 4. Output the current σ (a mixed strategy over the population).
>
> **Meta-solvers.**
>
> - LATEST: all weight on the newest policy. Self-play.
> - UNIFORM: equal weight on every policy. Fictitious play (FSP).
> - NASH: a Nash equilibrium of U^{Π}. Double oracle.
> - RM, HEDGE, PRD: run regret matching, Hedge, or projected replicator dynamics on U^{Π} (Lanctot et al., p. 4).
> - PFSP_f (our illustration): for learner i, weight each $C \in \Pi_{-i}$ by $f(\hat{\omega}_C)$, where $\hat{\omega}_C$ is the estimated probability that the current learner beats C, and normalize; fall back to LATEST if all weights are 0.

Online Double Oracle and anytime PSRO change what the meta-solver and oracle compute so that their stated guarantees hold; see the papers for the exact algorithms (we read their abstracts).

References

Each entry is marked peer-reviewed, preprint, abstract-only, or author documentation, as opened for this lecture.

- Abernethy, J., Lai, K. A., and Wibisono, A. (2021). "Fast Convergence of Fictitious Play for Diagonal Payoff Matrices." SIAM proceedings, doi:10.1137/1.9781611976465.84; arXiv:1911.08418v3 used. [peer-reviewed; the specific SIAM venue was not confirmed]
- Berner, C., Brockman, G., et al. (2019). "Dota 2 with Large Scale Deep Reinforcement Learning." arXiv:1912.06680. [preprint]
- Brandt, F., Fischer, F., and Harrenstein, P. (2013). "On the Rate of Convergence of Fictitious Play." Preprint submitted to *Theory of Computing Systems*, version of February 19, 2013. [preprint, as opened]
- Brown, N., Lerer, A., Gross, S., and Sandholm, T. (2019). "Deep Counterfactual Regret Minimization." ICML 2019, PMLR 97; arXiv:1811.00164v3 used. [peer-reviewed]
- Churchill, D. (2026). "2026 CoG StarCraft AI Competition Report." https://davechurchill.ca/starcraft/cog/results/2026/report/. [official competition report]
- Daskalakis, C., and Pan, Q. (2014). "A Counter-Example to Karlin's Strong Conjecture for Fictitious Play." FOCS 2014, pp. 11-20; arXiv:1412.4840 used. [peer-reviewed]
- Dinh, L. C., et al. (2022). "Online Double Oracle." *Transactions on Machine Learning Research*, 10/2022; arXiv:2103.07780. [peer-reviewed; abstract and introduction read]

- Heinrich, J., Lanctot, M., and Silver, D. (2015). "Fictitious Self-Play in Extensive-Form Games." ICML 2015, PMLR 37. [peer-reviewed]
- Heinrich, J., and Silver, D. (2016). "Deep Reinforcement Learning from Self-Play in Imperfect-Information Games." arXiv:1603.01121v2. [preprint] (NFSP)
- Hennes, D., Morrill, D., Omidshafiei, S., Munos, R., Perolat, J., Lanctot, M., Gruslys, A., Lespiau, J.-B., Parmas, P., Duéñez-Guzmán, E., and Tuyls, K. (2020). "Neural Replicator Dynamics: Multiagent Learning via Hedging Policy Gradients." AAMAS 2020, p. 492; arXiv:1906.00190v5 used. [peer-reviewed]
- Kalogiannis, F., and Farina, G. (2025). "Policy Gradient Methods Converge Globally in Imperfect-Information Extensive-Form Games." NeurIPS 2025. [peer-reviewed; abstract only read]
- Kubíček, O., Lisý, V., and Sandholm, T. (2026). "Equilibrium Refinements Improve Subgame Solving in Imperfect-Information Games." arXiv:2601.17131. [preprint]
- Kubíček, O., Lisý, V., and Sandholm, T. (2026). "Test-time Reinforcement Learning in Imperfect Information Games." arXiv:2608.30635. [preprint]
- Lanctot, M., Zambaldi, V., et al. (2017). "A Unified Game-Theoretic Approach to Multiagent Reinforcement Learning." NIPS 2017; arXiv:1711.00832. [peer-reviewed] (PSRO)
- McAleer, S., Lanier, J. B., Wang, K. A., Baldi, P., Sandholm, T., and Fox, R. (2024). "Toward Optimal Policy Population Growth in Two-Player Zero-Sum Games." ICLR 2024. [peer-reviewed; abstract read]
- Mella, V. (2026). *Pluto: StarCraft: Brood War AI, CoG 2026 entry* (PLUTO_README.txt, linked from the competition report). [author documentation, not peer-reviewed]
- Mohammadpour, S., Sokota, S., Kaplowitz, B., Kolter, J. Z., Brown, N., and Farina, G. (2026). "A Regularization-Based Approach to Public Belief State Search for Adversarial Games" (Gravel). NeurIPS 2026, to appear. [abstract only]
- Perolat, J., Munos, R., Lespiau, J.-B., Omidshafiei, S., Rowland, M., Ortega, P., Burch, N., Anthony, T., Balduzzi, D., De Vylder, B., Piliouras, G., Lanctot, M., and Tuyls, K. (2021). "From Poincaré Recurrence to Convergence in Imperfect Information Games: Finding Equilibrium via Regularization." ICML 2021, PMLR 139: 8525-8535; arXiv:2002.08456v1 used. [peer-reviewed]
- Perolat, J., De Vylder, B., et al. (2022). "Mastering the game of Stratego with model-free multiagent reinforcement learning" (DeepNash). *Science* 378(6623): 990-996; arXiv:2206.15378 used. [peer-reviewed]
- Rudolph, M., Lichtlé, N., Mohammadpour, S., Bayen, A., Kolter, J. Z., Zhang, A., Farina, G., Vinitsky, E., and Sokota, S. (2026). "Reevaluating Policy Gradient Methods for Imperfect-Information Games." ICLR 2026; arXiv:2502.08938v4 used. [peer-reviewed]
- Shapiro, H. N. (1958). "Note on a computation method in the theory of games." *Comm. Pure Appl. Math.* 11: 587-593. [peer-reviewed; not opened; cited through Brandt et al.]
- Sokota, S., D'Orazio, R., Kolter, J. Z., Loizou, N., Lanctot, M., Mitliagkas, I., Brown, N., and Kroer, C. (2023). "A Unified Approach to Reinforcement Learning, Quantal Response Equilibria, and Two-Player Zero-Sum Games" (MMD). ICLR 2023; arXiv:2206.05825v4 used. [peer-reviewed]
- Sokota, S., Vinitsky, E., Hu, H., Fan, Z., Kolter, J. Z., and Farina, G. (2026). "Scalable decision-making for games of imperfect information" (Ataraxos). *Nature*, published September 30, 2026, doi:10.1038/s41586-026-11036-y; arXiv:2511.07312v2 used. [peer-reviewed]
- Vinyals, O., Babuschkin, I., Czarnecki, W. M., et al. (2019). "Grandmaster level in StarCraft II using multi-agent reinforcement learning." *Nature* 575: 350-354. Author manuscript used. [peer-reviewed]