OPT2023: 15th Annual Workshop on Optimization for Machine Learning 1–11

Global CFR: Meta-Learning in Self-Play Regret Minimization

David Sychrovský SYCHROVSKY@KAM.MFF.CUNI.CZ
Charles University & Czech Technical University *Prague*

Michal Šustr MICHAL.SUSTR@AIC.FEL.CVUT.CZ
Czech Technical University & EquiLibre Technologies *Prague*

Michael Bowling MBOWLING@UALBERTA.CA
University of Alberta *Edmonton*

Martin Schmid SCHMID@EQUILIBRETECHNOLOGIES.COM
Charles University & EquiLibre Technologies *Prague*

Abstract

In real-world situations, players often encounter a distribution of similar but distinct games, like poker games with different public cards or trading varied correlated stock market assets. While these games exhibit related equilibria, current literature mainly delves into single games or their repeated versions. Recently, offline meta-learning was used to accelerate equilibrium discovery for such distributions in a single-player online setting. We build upon this, extending to a more challenging domain of two-player zero-sum *self-play* setting. Our method uniquely integrates information for next strategy selection for both players across all decision states, promoting global communication as opposed to the traditional local regret decomposition. Evaluations on distributions of matrix and sequential games reveal our meta-learned algorithms surpass their non-meta-learned variants.

1. Introduction

Regret minimization has become a widely adapted approach for finding equilibria in imperfect information games. The literature on equilibrium finding mainly focuses on isolated games or their repeated play, with a few recent exceptions (Xu et al., 2022; Harris et al., 2022; Sychrovsky et al., 2023). Nevertheless, numerous real-world scenarios feature playing similar, but not identical games, such as playing poker with different public cards or trading correlated assets on the stock market. As these similar games feature similar equilibria, it is possible to accelerate equilibrium finding on such a distribution (Harris et al., 2022).

Recently, (Sychrovsky et al., 2023) used offline meta-learning framework to accelerate online play for a distribution of two-player zero-sum games. Their motivation, similar to ours, was to make the agent more efficient in online settings, where one has limited time to make a decision. In this setting, they wanted to minimize the time required to find a single-player strategy with low one-sided exploitability, i.e. low one-sided approximation error from a mini-max equilibrium (Nisan et al., 2007).

In this paper, we extend their result to a self-play setting and show similar improvements can be gained when approaching a full Nash equilibrium profile. As the prior regret minimization algorithms typically work significantly better in the self-play setting, it is more difficult to make advances in this domain. We investigate several meta-learned algorithms and show that meta-learning can improve performance of the algorithms in almost all cases. A unique feature of our method is we meta-learn the predictions for both players and all the decision states simultaneously. We thus facilitate *global* inter-state communication, in contrast to the classic CFR regret decomposition *local* to individual states (Zinkevich et al., 2008). We evaluate the algorithms on a distribution of matrix and small sequential games. Our experiments show the meta-learned algorithms confidently surpass their non-meta-learned variants.

2. Preliminaries

Games. We now very briefly describe a game formalism based on factored-observation stochastic games (Kovařík et al., 2019).

Definition 1 *A game $\mathcal{G}$ is a tuple $\langle \mathcal{N}, \mathcal{W}, w^o, \mathcal{A}, u, \mathcal{O}\rangle$, where*

- $\mathcal{N} = \{1, 2\}$ *is a **player set**. We use symbol i for a player and -i for its opponent.*
- $\mathcal{W}$ *is a set of **world states** and $w^0 \in \mathcal{W}$ is a designated initial world state.*
- $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2$ *is a space of **joint actions**. A world state with no legal actions is **terminal**. We denote the set of terminal world states as $\mathcal{Z}$.*
- $u_i(z) = -u_{\text{-}i}(z)$ *is the **utility** player i receives when $z \in \mathcal{Z}$ is reached.*
- $\mathcal{O} = (\mathcal{O}_1, \mathcal{O}_2)$ *specifies the **observation** that i receives[1] upon the state transition.*

The space $\mathcal{S}_i$ of all action-observation sequences can be viewed as the infostate tree of player i. A **strategy profile** is a tuple $\boldsymbol{\sigma} = (\boldsymbol{\sigma}_1, \boldsymbol{\sigma}_2)$, where each **strategy** $\boldsymbol{\sigma}_i : s_i \in \mathcal{S}_i \mapsto \boldsymbol{\sigma}_i(s_i) \in \Delta^{|\mathcal{A}_i(s_i)|}$ specifies the probability distribution from which player i draws their next action conditional on having information s_i.

The expected reward (in the whole game) is $u_i(\boldsymbol{\sigma}) = \mathbb{E}_{z\sim\boldsymbol{\sigma}}\, u_i(z)$. The best-response to the other player's strategy $\boldsymbol{\sigma}_{\text{-}i}$ is $\mathcal{br}(\boldsymbol{\sigma}_{\text{-}i}) \in \arg\max_{\boldsymbol{\sigma}_i} u_i(\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_{\text{-}i})$. **Exploitability** of a strategy $\boldsymbol{\sigma}$ is the sum of rewards each player can get by best-responding to his opponent

$$\mathcal{expl}(\boldsymbol{\sigma}) = \sum_{i\in\mathcal{N}} u_i(\mathcal{br}(\boldsymbol{\sigma}_{\text{-}i}), \boldsymbol{\sigma}_{\text{-}i}).$$

A strategy profile is a Nash equilibrium if it has zero exploitability[2].

Regret Minimization. An **online algorithm** m for the regret minimization task repeatedly interacts with an **environment** g through available actions $\mathcal{A}$. The goal of regret minimization algorithm is to maximize its hindsight performance (i.e. to minimize regret). We will describe everything from the point of view of the player i acting at infostate $s \in \mathcal{S}_i$.

Formally, at each step $t \le T$, the algorithm submits a **strategy** $\boldsymbol{\sigma}^t = (\boldsymbol{\sigma}^t_1, \boldsymbol{\sigma}^t_2)$, where $\boldsymbol{\sigma}^t_i(s) \in \Delta^{|\mathcal{A}_i(s)|}$ is a strategy of i. Subsequently, it observes the **reward** $\boldsymbol{x}^t = (\boldsymbol{x}^t_1, \boldsymbol{x}^t_2), \boldsymbol{x}^t_i(\boldsymbol{\sigma}^t_{\text{-}i}) \in \mathbb{R}^{|\mathcal{A}_i(s)|}$ returned from the environment g depending on strategy of the opponent -i. The difference in reward obtained under $\boldsymbol{\sigma}^t_i$ and any fixed action strategy is measured by the **instantaneous regret** $\boldsymbol{r}_i(\boldsymbol{\sigma}^t, \boldsymbol{x}^t) = \boldsymbol{x}^t_i(\boldsymbol{\sigma}^t_{\text{-}i}) - \langle \boldsymbol{\sigma}^t_i, \boldsymbol{x}^t_i(\boldsymbol{\sigma}^t_{\text{-}i})\rangle \mathbf{1}$ of player i. The **cumulative regret** over the entire sequence is $\boldsymbol{R}^T_i = \sum_{t=1}^T \boldsymbol{r}_i(\boldsymbol{\sigma}^t, \boldsymbol{x}^t)$.

1. This observation includes both the public and the private observation of player i.
2. This is because then the individual strategies are mutual best-responses.

Connection Between Games and Regret Minimization. In two-player zero-sum games, if the **external regret** $R_i^{\text{ext},T} = \left\|\boldsymbol{R}_i^T\right\|_\infty$ grows sublinearly in T, the average strategy $\overline{\boldsymbol{\sigma}}_i^T = \frac{1}{T}\sum_{t=1}^T \boldsymbol{\sigma}_i^t$ converges to a Nash equilibrium (Nisan et al., 2007) as $T \to \infty$. In sequential games, one can decompose the regret into individual (i.e. per infostate) counterfactual regrets and minimize the separately, leading to counterfactual regret minimization (Zinkevich et al., 2008). This approach again converges to a Nash equilibrium.

3. Meta-learning framework

On a distribution of regret minimization tasks G, we aim to find an online algorithm m_θ with some parameterization θ that efficiently minimizes the expected external regret. However, what does it mean for a regret minimizer to be *good at minimizing regret on* G?

The simplest answer is to minimize the final external regret $R^{\text{ext},T}$ in expectation over G. However, even if we reduced the regret to zero, this would only guarantee that the final average strategy $\overline{\boldsymbol{\sigma}}_\theta^T$ is close to an equilibrium. Rather, we want that the regret minimizer chooses strategies close to an equilibrium along *all the points of the trajectory* $\boldsymbol{\sigma}_\theta^1, \dots \boldsymbol{\sigma}_\theta^T$. Consequently, we define the loss as expectation over the maximum instantaneous regret experienced at each step in all infostates of the game, i.e.

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{g\in G}\left[\sum_{i\in\mathcal{N}}\sum_{s_i\in\mathcal{S}_i(g)}\sum_{t=1}^{T}\left\|\boldsymbol{r}_i(\boldsymbol{\sigma}_\theta^t, \boldsymbol{x}^t(\theta))\right\|_\infty\right] \geq \mathop{\mathbb{E}}_{g\in G}\left[\sum_{i\in\mathcal{N}}\sum_{s_i\in\mathcal{S}_i(g)} R_i^{\text{ext},T}(s_i|\theta)\right]. \qquad (1)$$

This is analogous to minimizing $\sum_{t=1}^T f(x^t)$ rather than $f(x^T)$ as in (Andrychowicz et al., 2016), where the authors meta-learned a function optimizer. Note that this loss does not correspond to any kind of regret that one can hope to minimize against a black-box. See Appendix A for further discussion. For matrix games, since the game is zero-sum, $\sum_{i\in\mathcal{N}}\left\|\boldsymbol{r}_i(\boldsymbol{\sigma}_\theta^t, \boldsymbol{x}^t(\theta))\right\|_\infty = \sum_{i\in\mathcal{N}}\left\|\boldsymbol{x}_i^t(\theta)\right\|_\infty$. Minimizing (1) is thus equivalent to minimizing the expected exploitability of the selected strategy along the trajectory.

In comparison with (Sychrovsky et al., 2023) where the authors used a best-responding opponent, we meta-train the regret minimizer for both players simultaneously. There are two main reasons for why our domain is more challenging. First, prior regret minimization algorithms typically work significantly better in self-play compared to the best-response setting (Farina et al., 2021). Second, as the meta-learning continues, the 'environment' as viewed by each of the players keeps changing, akin to moving-target problems.

We train a recurrent neural network θ to minimize (1). By utilizing a recurrent architecture we can also represent algorithms that are history and/or time dependent. Furthermore, this approach allows us to combine *all infostates of the game.* This is different from standard applications of regret minimization to games in which each infostate is optimized separately (Zinkevich et al., 2008). The local information strongly depends on the strategy selected at other infostates. In our approach, this can be sidestepped by directly accessing information from all infostates of the game tree. See Section 4 and Appendix C for details.

In the rest of this section, we briefly outline two meta-learning algorithms introduced in (Sychrovsky et al., 2023). Their main difference is whether or not they provide regret minimization guarantees. Both use a neural network trained to minimize Eq. (1).

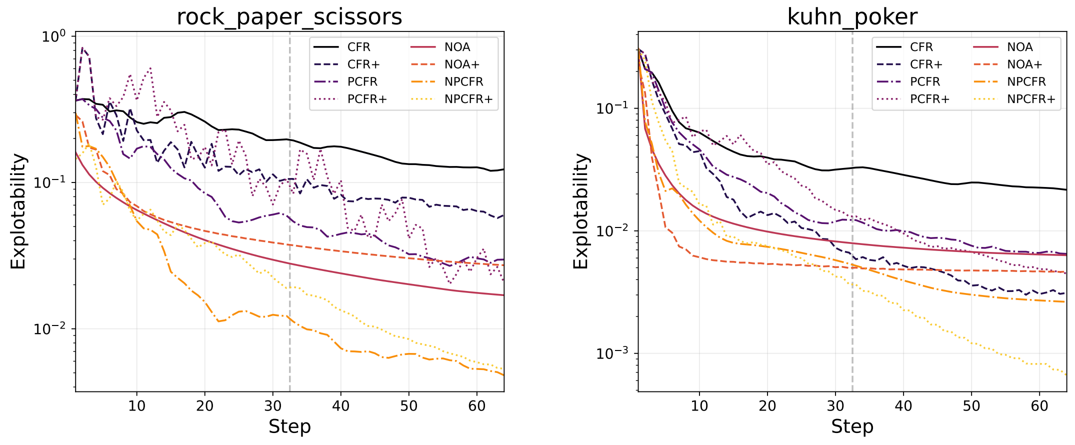

Figure 1: Comparison of non-meta-learned algorithms (CFR$^{(+)}$, PCFR$^{(+)}$) with meta-learned algorithms (NOA$^{(+)}$, NPCFR$^{(+)}$) on `rock_paper_scissors` (left) and `kuhn_poker` (right). The figures show exploitability of the average strategy $\overline{\boldsymbol{\sigma}}$. Vertical dashed lines separate two regimes: training (up to $T = 32$ steps) and generalization (from T to $2T$ steps). See Figure 5 for standard errors.

Neural Online Algorithm (NOA). This is the simplest option, which is *not guaranteed* to minimize regret. We directly parameterize the online algorithm m_θ to output strategy $\boldsymbol{\sigma}_\theta^t$.

Neural Predictive Counterfactual Regret Minimization (NPCFR). In order to provide convergence guarantees, (Sychrovsky et al., 2023) introduced meta-learning within the predictive counterfactual regret minimization (PCFR) framework (Farina et al., 2021). The PCFR is an extension of counterfactual regret minimization[3] (CFR) (Zinkevich et al., 2008) which uses an additional predictor about future regret. One can show PCFR converges faster for more accurate predictions, and (crucially for us) is guaranteed to converge[4] (Farina et al., 2021; Sychrovsky et al., 2023). The neural predictive counterfactual regret minimization (NPCFR) is an extension of PCFR which uses a predictor parameterized by a neural network θ. This approach combines the best of both words – adaptive algorithm with a small regret in the domain of interest, while keeping the regret minimization guarantees.

4. Experiments

We focus on application of regret minimization in games. We construct two distributions by adding noise to utilities of a fixed game, see Appendix B. For both NOA and NPCFR, the neural network is based on a two layer LSTM and uses all infostates of the game to produce $\boldsymbol{\sigma}_\theta^t$, see Appendix C. In addition to the last-observed instantaneous regret $\boldsymbol{r}^t$ and the cumulative regret $\boldsymbol{R}^t$, the networks also receive one-hot encoding of the infostate and

3. Here we refer to using regret matching at each infostate rather than other regret minimizers.
4. This is true regardless of the prediction under a mild assumption that they are bounded.

keeps track of its hidden state $\boldsymbol{h}^t$. Additionally, we train NOA$^+$ and NPCFR$^+$ defined as in CFR$^+$ (Tammelin, 2014) in the same way.

We minimize objective (1) for $T = 32$ iterations. For evaluation, we compute exploitability of the strategies up to $2T$ iterations to see whether our algorithms can generalize outside of the horizon T and keep reducing the exploitability. Both regimes use the self-play setting, i.e. each algorithm controls strategies of both players. Comparison of our meta-learned algorithms with (P)CFR$^{(+)}$ (Tammelin, 2014; Farina et al., 2021) is presented in Figure 1. See also Figure 5 in Appendix D for a version which includes standard errors.

4.1. Matrix Games

For evaluation in matrix setting, we use a modification of the standard `rock_paper_scissors`, perturbing two of its elements. Figure 1 shows our algorithms can converge very fast. In fact, all the meta-learned algorithms outperform their non-meta-learned counterpart, often by nearly an order of magnitude. To further illustrate their differences, we plot the current strategy $\boldsymbol{\sigma}^t$ selected by each algorithm in Appendix D, Figure 4. The meta-learned algorithms exhibit much smoother convergence.

4.2. Sequential Games

We use standard small benchmark `kuhn_poker` to evaluate our algorithms in the sequential setting. Figure 1 shows similar improvements as in the matrix setting can be achieved here. While all algorithms keep minimizing regret, NOA$^{(+)}$ initially converges fast but exhibits poor generalisation. In comparison, NPCFR$^{(+)}$ significantly outperform their non-meta-learned counterparts. As was observed before (Tammelin, 2014), the 'plus-versions' of each algorithm show better performance on this domain.

5. Conclusion

In this paper, we've built on the results of (Sychrovsky et al., 2023) who introduced meta-learning within regret minimization. We extended their results to the self-play domain in two-player zero-sum games. We evaluated the meta-learned algorithms and compared them to state-of-the-art on small two-player zero-sum games. The meta-learned algorithms considerably outperformed the prior algorithms in nearly all situations.

Future Work We would like to improve our experimental results in several ways.
(i) Consider larger games, such as the river endgame of Texas Hold'em poker.
(ii) Show our methods can be competitive even on longer instances, i.e. larger values of T.
(iii) Make ablations on the influence of the communication across the infostates.

Acknowledgements The authors would like to thank Martin Loebl, Matěj Moravčík, Viliam Lisý, and Milan Hladík for their insightful comments. This work was supported by the Czech Science Foundation grant no. GA22-26655S.

References

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

Gabriele Farina, Christian Kroer, and Tuomas Sandholm. Faster game solving via predictive blackwell approachability: Connecting regret matching and mirror descent. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 5363–5371, 2021.

Keegan Harris, Ioannis Anagnostides, Gabriele Farina, Mikhail Khodak, Zhiwei Steven Wu, and Tuomas Sandholm. Meta-learning in games. *arXiv preprint arXiv:2209.14110*, 2022.

Vojtěch Kovařík, Martin Schmid, Neil Burch, Michael Bowling, and Viliam Lisý. Rethinking formal models of partially observable multiagent decision making. *arXiv preprint arXiv:1906.11110*, 2019.

Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. Algorithmic game theory. *Google Scholar Google Scholar Digital Library Digital Library*, 2007.

David Sychrovsky, Michal Sustr, Elnaz Davoodi, Marc Lanctot, and Martin Schmid. Learning not to regret. In *OptLearnMAS*. IFAAMAS, 2023.

Oskari Tammelin. Solving large imperfect information games using CFR$^+$. *arXiv preprint arXiv:1407.5042*, 2014.

Hang Xu, Kai Li, Haobo Fu, Qiang Fu, and Junliang Xing. Autocfr: Learning to design counterfactual regret minimization algorithms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 5244–5251, 2022.

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. In *Advances in neural information processing systems*, pages 1729–1736, 2008.

Appendix A. Meta-Loss Function

In this section we elaborate on Eq. (1) and discuss one other feasible meta-loss. As stated in the main text, optimizing for final external regret only forces the average strategy at step T to be near the equilibrium. Naturally, one can thus define the meta-loss as a weighted sum of external regrets over $t \leq T$

$$\tilde{\mathcal{L}}(\theta) = \mathop{\mathbb{E}}_{g \in G} \left[\sum_{t=1}^{T} \omega_t \sum_{i \in \mathcal{N}} R_i^{\mathrm{ext},t}(\theta) \right], \qquad \omega_t \geq 0. \tag{2}$$

To better understand this loss, let us rewrite it in terms of reward using the fact the game is zero-sum

$$\sum_{i \in \mathcal{N}} R_i^{\mathrm{ext},t}(\theta) = \sum_{i \in \mathcal{N}} \left\| \sum_{l=1}^{t} \boldsymbol{x}_i^l - \langle \boldsymbol{\sigma}_i^l, \boldsymbol{x}_i^l \rangle \mathbf{1} \right\|_\infty = \sum_{i \in \mathcal{N}} \left\| \sum_{l=1}^{t} \boldsymbol{x}_i^l(\boldsymbol{\sigma}_{-i}^l) \right\|_\infty,$$

This allows us to express the gradient with respect to the strategy selected at step $\tau \leq T$

$$\frac{\partial \tilde{\mathcal{L}}}{\partial \boldsymbol{\sigma}_i^\tau} = \frac{\partial}{\partial \boldsymbol{\sigma}_i^\tau} \sum_{t=1}^{T} \omega_t \left[R_1^{\mathrm{ext},t}(\theta) + R_2^{\mathrm{ext},t}(\theta) \right] = \sum_{t=\tau}^{T} \omega_t \frac{\partial}{\partial \boldsymbol{\sigma}_i^\tau} \left\| \sum_{l=1}^{t} \boldsymbol{x}_{-i}^l(\boldsymbol{\sigma}_i^l) \right\|_\infty.$$

However, this can lead to instabilities because in general

$$\frac{\partial}{\partial \boldsymbol{\sigma}_i^\tau} \left\| \sum_{l=1}^{t} \boldsymbol{x}_{-i}^l(\boldsymbol{\sigma}_i^l) \right\|_\infty \neq \frac{\partial}{\partial \boldsymbol{\sigma}_i^\tau} \left\| \boldsymbol{x}_{-i}^\tau(\boldsymbol{\sigma}_i^\tau) \right\|_\infty.$$

Suffering large rewards in earlier optimization steps may thus result in meta-loss penalising actions, which do not lower opponent's best-response reward. Thus, the meta-gradient descent may not approach equilibrium, see Section A.1 for an example.

In contrast to loss (2), our choice (1) is clearly consistent in this sense. However, note that in sequential incomplete information games, these local strategy improvements may still lead to the overall strategy being more exploitable.

A.1. Example of Non-Smooth Convergence

Consider using Eq. (2) on matching pennies for $T = 2$. Furthermore, focus on just the first player[5] and let his selected strategies be

$$\boldsymbol{\sigma}_1^1 = (1, 0), \qquad \boldsymbol{\sigma}_1^2 = (1/3, 2/3).$$

Then the rewards of the second player are

$$\boldsymbol{x}_2^1 = (1, 0) \cdot \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = (1, -1), \qquad \boldsymbol{x}_2^2 = (1/3, 2/3) \cdot \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = (-1/3, 1/3).$$

5. We can have the second player do something analogous.

Consider the gradient with respect to the strategy of the first player at $t = 2$. To achieve 'smooth convergence', the gradient should guide $\boldsymbol{\sigma}_1^2$ closer to the uniform equilibrium. However, from the above we have

$$\frac{\partial \tilde{\mathcal{L}}}{\partial \boldsymbol{\sigma}_1^2} = \frac{\partial}{\partial \boldsymbol{\sigma}_1^2} \left\| \sum_{l=1}^{2} \boldsymbol{x}_2^l(\boldsymbol{\sigma}_1^l) \right\|_\infty \quad \text{where} \quad \left\| \sum_{l=1}^{2} \boldsymbol{x}_2^l(\boldsymbol{\sigma}_1^l) \right\|_\infty = \|(2/3, -2/3)\|_\infty = 2/3,$$

or in words, $\boldsymbol{\sigma}_1^2$ needs to be such that the *first* element of the vector is minimized. Critically, $\boldsymbol{x}_2^1$ is a constant w.r.t. $\boldsymbol{\sigma}_1^2$. But in this case, it forces us to minimize the first element of $\boldsymbol{x}_2^2$, or the reward of player two for playing only the first action. This would mean

$$\frac{\partial \tilde{\mathcal{L}}}{\partial \boldsymbol{\sigma}_1^2} = \frac{\partial}{\partial \boldsymbol{\sigma}_1^2} \left[(\boldsymbol{\sigma}_1^1 + \boldsymbol{\sigma}_1^2) \cdot \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right] = (1, -1)^T.$$

However, this is exactly the opposite direction than what we would need to approach the equilibrium.

Eq. (2) is 'consistent' since if $\boldsymbol{\sigma}^1$ is sufficiently close to an equilibrium, the associated rewards $\boldsymbol{x}^1$ are small. But finding the equilibrium in the first step is harder than in the second, as the regret minimizer has no information about the instance $g \sim G$ being solved. In contrast, gradient of Eq. (1) will always 'point in the right direction', which makes the training more stable.

Finally, considering $T > 2$ only compounds the problem – if one action is consistently overused, it may disturb the gradient of a large number of following steps.

Appendix B. Games

In this section we describe two distributions of games used in our experiments. Both can be viewed as a single game with added noise in the terminal utilities.

B.1. Rock Paper Scissors

The `rock_paper_scissors` game is a matrix game given by

$$u_1 = -u_2 = \begin{pmatrix} 0 & -1 & 3 + X \\ 1 & Y & -1 \\ -1 & 1 & 0 \end{pmatrix},$$

where $X, Y \sim \mathcal{U}(-0.5, 0.5)$. Note that the fixed variant is a biased version of the original game. We opted for this option to make the equilibrium strategy non-uniform, as in the original game (P)CFR$^{(+)}$ are initialized with the equilibrium policy.

B.2. Kuhn Poker

In `kuhn_poker`, we keep the structure of the original game structure and perturb all terminal utilities. Specifically, we generate i.i.d. $x_z \sim \mathcal{U}(-0.05, 0.05)$ for each terminal $z \in \mathcal{Z}$ and change the terminal reward to

$$\tilde{u}_z = (1 + x_z)u_z,$$

where u_z is the utility in the original game.

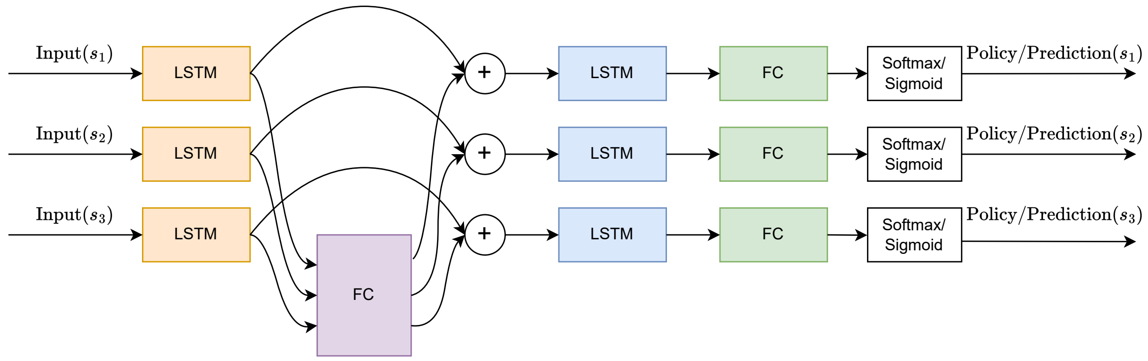

Figure 2: Network architecture used for NOA$^{(+)}$/NPCFR$^{(+)}$. This example shows a game with three infostates $\{s_k\}_{k=1}^{3}$. Same colours indicate shared parameters.

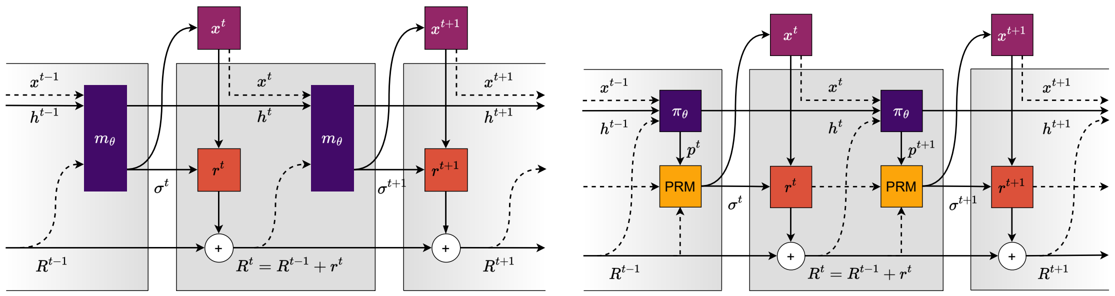

Figure 3: Computational graphs for NOA$^{(+)}$ (left) and NPCFR$^{(+)}$ (right). The gradient flows only through the solid edges.

Appendix C. Neural Network Architecture and Training Setup

Network architecture used for both NOA$^{(+)}$ and NPCFR$^{(+)}$ is shown in Figure 2. The input to the neural network is a concatenation of last-observed regret, cumulative regret, and one-hot encoding of the infostate.

The input is first processed by a LSTM layer separately for each infostate. Then we apply resnet-like gate consisting of a single fully-connected layer going over the same actions in each infostate and using ReLU activation. The rest of the network operates again per-infostate and consists of a LSTM player followed by a fully-connected layer. Finally, for NOA$^{(+)}$ we apply the softmax activation. For NPCFR$^{(+)}$, we apply the sigmoid activation and scale it by $\alpha \in \{2, 4, 8\}$, which is found via grid search[6]. The prediction of NPCFR$^{(+)}$ is the sum of of the output of the network and last-observed regret.

The computational graphs for both for NOA$^{(+)}$ and NPCFR$^{(+)}$ are presented in Figure 3. The gradient $\partial\mathcal{L}/\partial\theta$ originates in the collection of maximal instantaneous regrets $\left\|\boldsymbol{r}^{1\ldots T}\right\|_\infty$ and propagates through the strategies $\boldsymbol{\sigma}^{1\ldots T}$ (the predictions $\boldsymbol{p}^{1\ldots T}$ for NPCFR$^{(+)}$), the

6. Specifically, the size of the LSTM layer, the number of games in each batch gradient update, the L2 weight decay, and the regret prediction bound α.

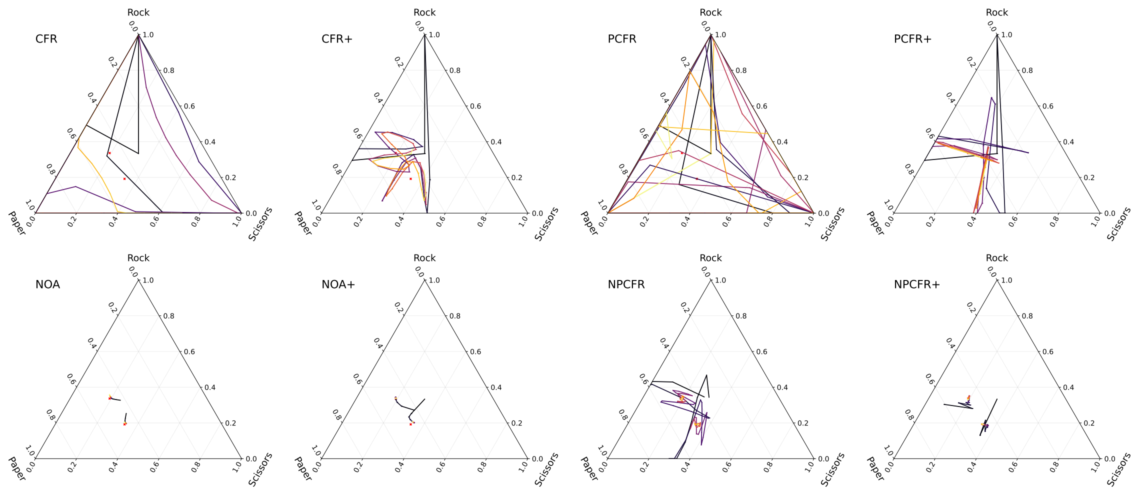

Figure 4: Comparison of the convergence in current strategy $\boldsymbol{\sigma}^t$ on a random sample of `rock_paper_scissors` over $2T = 64$ steps. The red crosses show the per-player equilibria of the sampled game. The trajectories start in dark colors and get brighter for later steps. We show both the non-meta-learned algorithms (top row) and the meta-learned algorithms (bottom row).

rewards $\boldsymbol{x}^{1\ldots T}(\boldsymbol{\sigma}_\theta^{1\ldots T})$ coming from the opponent, and hidden states $\boldsymbol{h}^{0\ldots T-1}$. We do not propagate the gradient through the inputs of the network[7].

Appendix D. Additional Results

In this section, we show a exploitability of the average strategy presented in Figure 1. However, for additional clarity, we only plot mean values with respect to the distribution of games. Clearly, all meta-learned algrithms outperformed their non-meta-learned counterparts show in the same line-style.

Furthermore, we present a representative example of current strategy evolution. We use a random sample of `rock_paper_scissors` and plot the current strategy selected by each algorithm in Figure 4. All meta-learned algorithms exhibit a much smoother convergence compared to their non-meta-learned counterparts.

7. This is similar to the “learning to learn” setup (Andrychowicz et al., 2016) and allows us to skip the expensive second-order derivative computation.

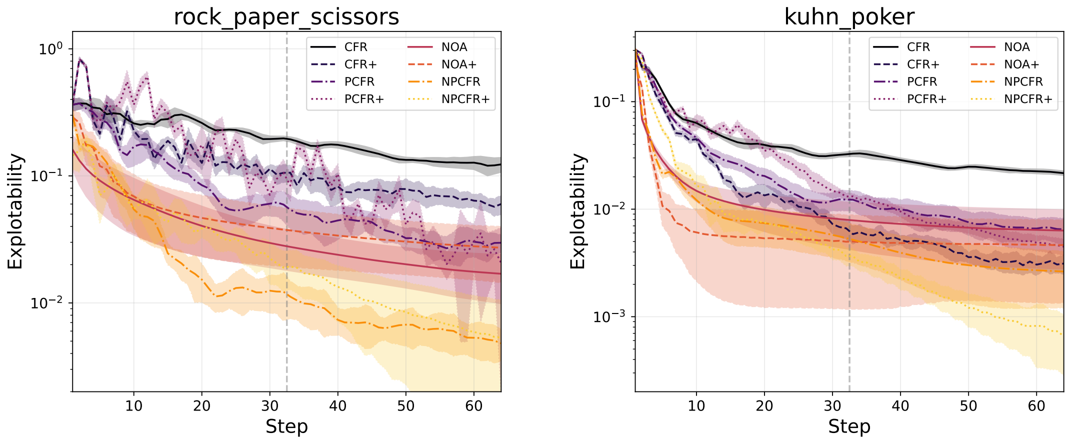

Figure 5: Comparison of non-meta-learned algorithms (CFR$^{(+)}$, PCFR$^{(+)}$) with meta-learned algorithms (NOA$^{(+)}$, NPCFR$^{(+)}$) on `rock_paper_scissors` (left) and `kuhn_poker` (right). The figures show mean exploitability of the average strategy $\overline{\boldsymbol{\sigma}}$. Vertical dashed lines separate two regimes: training (up to $T = 32$ steps) and generalization (from T to $2T$ steps). Colored areas show standard errors.