2段階平均による二人ゼロ和不完全情報ゲーム ナッシュ均衡近似精度の向上

平野 瑞紀[1,a)]　鶴岡 慶雅[1,b)]

概要：二人ゼロ和ゲームでナッシュ均衡を求める多くのアルゴリズムでは, 戦略の平均化によってナッシュ均衡を近似する必要がある. アルゴリズムを大規模ゲームへ適用するために, 戦略をニューラルネットワークで表現し計算コストを削減するようになった. しかし, ニューラルネットワークによって戦略の平均を学習すると表現誤差が増幅されてしまい近似精度に影響を及ぼす. 大規模ゲームを想定した実験を実施したところ, 平均の近似が安定しないことが確認された. 近似の不安定性はアルゴリズム全体を通しての近似精度を低下させる可能性がある. そこで本研究では, 平均化された戦略をさらに平均することで安定性を向上させる 2 段階平均化を提案する. 実験によって, 提案手法が平均の近似を安定させるだけでなく, 近似精度の向上にもつながることが示された.

Improving Performance of Approximating Nash Equilibrium in Two-Player Zero-Sum Games via Two-Stage Averaging

Mizuki Hirano[1,a)]　Yoshimasa Tsuruoka[1,b)]

Abstract: Many algorithms for finding Nash equilibrium in two-player zero-sum games need to approximate the Nash equilibrium by averaging strategies. In order to apply these algorithms to large-scale games, the strategies are represented by neural networks to reduce the computational cost. However, learning the average of strategies with neural networks amplify the representation error and affects the approximation performance. Experiments for large-scale games showed that the approximation of the average is not stable. The instability of the approximation may degrade the approximation performance through the algorithm. We propose a two-stage averaging method in which the stability is improved by further averaging the averaged strategy. Experiments showed that the proposed method not only stabilises the approximation of the average, but also improves the approximation performance.

1. はじめに

二人ゼロ和不完全情報ゲームでナッシュ均衡を求めるアルゴリズムの多くは平均収束性と呼ばれる性質を持つ [4,18]. 平均収束性を持つアルゴリズムでは, 反復的に戦略を更新した際に, 反復の過程で生成された戦略の列の平均である平均戦略のみが反復回数に応じてナッシュ均衡への収束を保証されている. そのため, 平均戦略を求めることによってナッシュ均衡を近似する. 更新された戦略自体は一般的にナッシュ均衡へ収束しない [11].

ゲーム木が大きい大規模ゲームでナッシュ均衡を求めるためにアルゴリズムの計算コストを削減する手法が提案されている [2,7,8]. 特に空間計算量の点では, 戦略をテーブルで保持すると (情報集合数) × (行動の種類数) の確率値をメモリ上に記憶する必要がある. 例えば Heads-up no-limit texas hold'em は 10^{160} 以上の情報集合を持つとされるため実行不可能となる. それに対して, ニューラルネットワークで戦略を表現し, 情報集合から戦略への対応をパラメータに集約することでメモリ使用量を削減することができる.

[1] 東京大学大学院 情報理工学系研究科 電子情報学専攻
Graduate School of Information Science and Technology, The University of Tokyo
[a)] MizukiHirano@logos.t.u-tokyo.ac.jp
[b)] tsuruoka@logos.t.u-tokyo.ac.jp

しかし, ニューラルネットワークによる戦略の表現は平均収束性に適していないことが問題視される [1, 10]. 戦略をニューラルネットワークで表現すると戦略更新の際に学習誤差が発生する. その上, 戦略の平均を求める際に誤差を含んだ戦略の平均を学習するため, 誤差が増幅し近似精度に影響してしまう.

そこで, 更新される戦略自体がナッシュ均衡に収束する最終反復収束と呼ばれる性質を持つアルゴリズムが研究されるようになった. 最終反復収束を実現するためのアプローチとして報酬変換 [13] がある. アルゴリズムの計算コストを制限していない状況において, 報酬変換は最終反復収束だけでなく近似精度の向上をもたらす [10]. また, 計算コストを削減した状況においても Regularized Nash Dynamics (R-NaD) [12] のように報酬変換を取り入れたアルゴリズムが提案されている. ただし, 計算コストの削減によって発生する誤差の影響で, 最終反復収束を目標としていても戦略の平均化を行う必要がある.

大規模ゲームを想定した実験を実施したところ, R-NaD アルゴリズムでは報酬変換による近似精度の向上が見られなかった. さらに, 実験結果では近似精度が乱高下しており, 近似が安定していないことが確認された. 近似精度の評価にはゲーム木の大きさに比例する計算コストが必要であり, アルゴリズムの実行で最も高い近似精度を持つ結果を得ることができない. そのため, 実行終了時に近似精度が急落するとアルゴリズム全体を通しての近似精度が低下してしまう.

そこで本研究では, 安定した近似を実現するために, 平均化された戦略をさらに平均する 2 段階平均を提案する. 実験によって, 提案手法が平均化を安定させるだけでなく, ナッシュ均衡の近似精度向上をもたらすことが示された.

2. 背景

2.1 二人ゼロ和不完全情報ゲームとナッシュ均衡

二人ゼロ和不完全情報ゲームは $(N \cup \{c\}, H, Z, P, A, u, \mathcal{I})$ によって構成される. $N = \{1, 2\}$ はプレイヤー集合を, c はチャンスプレイヤー表す. プレイヤー i を除くすべてのプレイヤーを $-i$ で表記する. H は履歴集合であり, 履歴 $h \in H$ は実現し得るゲーム状況を表す. $Z \subseteq H$ は終端集合であり, 終端 $z \in Z$ はゲームが終了するゲーム状況を表す. $H \setminus Z$ は手番集合であり, 手番 $h \in H \setminus Z$ でただ一人のプレイヤーが意思決定を行う. P はプレイヤー関数であり, $P(h)$ は手番 h で意思決定を行うプレイヤーを表す. A は行動集合であり, $A(h)$ は手番 h で選択可能な行動の集合を表す. 手番 h で行動 a を選択すると履歴が ha に遷移する. u は利得関数であり, $u_i(z) \in [0, 1]$ は終端 $z \in Z$ でプレイヤー $i \in N$ が得る利得を表す. $I \in \mathcal{I}$ は情報集合を表す. 情報集合 I で意思決定を行うプレイヤーを $P(I)$, 選択可能な行動の集合を $A(I)$ で表記する.

プレイヤーの戦略の組を $\sigma = (\sigma_1, \sigma_2, \sigma_c)$ で表記する. 以降, 戦略の組を単に戦略と呼ぶ. 戦略 σ に従った際に, 手番 h で行動 a を選択する確率を $\sigma(h, a)$, 情報集合 I で行動 a を選択する確率を $\sigma(I, a)$ で表記する. プレイヤー $-i$ の戦略の組 σ_{-i} に対して $(\sigma_i, \sigma_{-i}) = \sigma$ という表記が成り立つ. 戦略 σ に従ってゲームをプレイした際に手番 h に到達する確率 $x^\sigma(h)$ を到達確率という.

報酬は手番で行動を選択した際に得られる値であり, 手番 h で行動 a を選択した際にプレイヤー i が得る報酬を $r_i(h, a)$ で表記する. 二人ゼロ和ゲームでは常に $r_1(h, a) = -r_2(h, a)$ が成り立つ. ある手番からゲームが終端するまでプレイした際に得られる報酬の和を収益という. ゲームを初期状態から終端 z までプレイした際に得られる収益と利得 $u_i(z)$ は一致する. 手番 h から戦略 σ に従ってゲームをプレイした際にプレイヤー i が得る収益の期待値 $v_i^\sigma(h)$ および手番 h で行動 a を選択しその後は戦略 σ に従ってゲームをプレイした際にプレイヤー i が得る収益の期待値 $q_i^\sigma(h, a)$ を期待収益という. 期待収益に対して式 (1) の関係性が成り立つ

$$\begin{aligned} v_i^\sigma(h) &= \sum_{a \in A(h)} \sigma(h, a) q_i^\sigma(h, a) \\ q_i^\sigma(h, a) &= r_i(h, a) + v_i^\sigma(ha) \end{aligned} \tag{1}$$

また, 情報集合 I に関する期待収益は式 (2) で計算される.

$$\begin{aligned} v_i(I) &= \frac{\sum_{h \in I} x^\sigma(h) v_i^\sigma(h)}{\sum_{h \in I} x^\sigma(h)} \\ q_i(I, a) &= \frac{\sum_{h \in I} x^\sigma(h) q_i^\sigma(h, a)}{\sum_{h \in I} x^\sigma(h)} \end{aligned} \tag{2}$$

特にゲームの初期状態での期待収益を v_i^σ で表記する.

ナッシュ均衡とは, ある戦略の組であって「どのプレイヤーも自身の戦略のみを変更することで自身が得る期待利得を大きくすることができない」という性質を満たすものである. つまり, σ^* がナッシュ均衡ならば, 任意のプレイヤー i とプレイヤー i の任意の戦略 σ_i に対して式 $v_i^{\sigma^*} \geq v_i^{(\sigma_i, \sigma^*_{-i})}$ が成り立つ. ある戦略 σ に対する可搾取量 exploitability(σ) は式 (3) で定義される.

$$\text{exploitability}(\sigma) = \sum_{i \in N} \left(\max_{\sigma'_i} v_i^{(\sigma'_i, \sigma_{-i})} - v_i^\sigma \right) \tag{3}$$

exploitability は戦略がナッシュ均衡からどの程度離れているかを表し, ナッシュ均衡 σ^* においてのみ exploitability$(\sigma^*) = 0$ となる.

なお, 本研究ではゲームが完全記憶性を持つと仮定する. 完全記憶性を持つゲームにおいて, プレイヤーはそのゲーム中の自身の意思決定の内容をすべて記憶している. 完全記憶性によってナッシュ均衡の存在が保証される.

2.2 Neural Replicator Dynamics

Neural Replicator Dynamics (NeuRD) [4] は平均収束性を持つアルゴリズムである．NeuRD では戦略をニューラルネットワークで表現する．式 (4) のように, パラメータ θ に対してニューラルネットワークがロジット y^θ を出力し, softmax 関数を通じて戦略 σ^θ に変換する．

$$\sigma^\theta(I,a) = \frac{e^{y^\theta(I,a)}}{\sum_{a\in A(I)} e^{y^\theta(I,a)}} \tag{4}$$

NeuRD では式 (5) のようにパラメータを更新することで戦略を更新する．

$$\theta_{t+1} = \theta_t + \alpha_t \sum_{I\in\tau_t} \sum_{a\in A(I)} \nabla_\theta y^{\theta_t}(I,a) \left(q^{\sigma^{\theta_t}}_{P(I)}(I,a) - v^{\sigma^{\theta_t}}_{P(I)}(I) \right) \tag{5}$$

ここで α は学習率を, $\tau \subseteq \mathcal{I}$ はゲーム木からサンプルされた情報集合の集合を表す．式 (1) および (2) のように, 期待収益の計算はゲーム木上で再帰的に行われるため計算コストが高い．そこで期待収益を推定し, 誤差の含まれる勾配でパラメータを更新する．NeuRD では更新された戦略の列 $\{\sigma^{\theta_t}\}_{t=1}^T$ の時間平均が反復回数 T に応じてナッシュ均衡に収束する．

2.3 Regularized Nash Dynamics

Regularized Nash Dynamics (R-NaD) [12] は NeuRD に報酬変換を適用したアルゴリズムである．報酬変換のアプローチでは通常の戦略 σ^θ とは別に正則化戦略 $\sigma^{\theta_{\text{reg}}}$ を導入し報酬を式 (6) のように変換する．

$$\begin{aligned} r_i^{\sigma^{\theta_{\text{reg}}}}(\sigma,h,a) = r_i(h,a) &- \mathbf{1}_{i=P(h)}\eta \log \frac{\sigma^\theta(h,a)}{\sigma^{\theta_{\text{reg}}}(h,a)} \\ &+ \mathbf{1}_{i\neq P(h)}\eta \log \frac{\sigma^\theta(h,a)}{\sigma^{\theta_{\text{reg}}}(h,a)} \end{aligned} \tag{6}$$

ここで η は変換の強度を決めるパラメータである．式 (6) の変換はゼロ和性を失わない．変換した報酬に対して NeuRD を実行すると, 更新されたパラメータ θ が η および θ_{reg} によって定まるパラメータ θ' に収束する．報酬を変換したことで $\sigma^{\theta'}$ は変換前のゲームのナッシュ均衡から外れるが, $\sigma^{\theta'}$ は $\sigma^{\theta_{\text{reg}}}$ よりも変換前のゲームのナッシュ均衡に近づく, つまり $\text{exploitability}(\sigma^{\theta'}) \leq \text{exploitability}(\sigma^{\theta_{\text{reg}}})$ となることが保証される．よって, $\sigma^{\theta'}$ を新たな正則化戦略としてアルゴリズムを繰り返すことで, 正則化戦略が報酬を変換していないゲームのナッシュ均衡に収束する．ただし, NeuRD の戦略更新の勾配に誤差が含まれることから, T 回の更新を行った際に収束しないどころか, $\text{exploitability}(\sigma^{\theta'}) \leq \text{exploitability}(\sigma^{\theta_{\text{reg}}})$ とならない可能性がある．そこで, 式 (7) のように戦略パラメータ θ の指数移動平均をターゲットパラメータ θ_{target} として計算し, 正則化戦略をターゲット戦略で更新することで問題に対処する．

$$\theta_{\text{target},n+1} = \gamma\theta_{n+1} + (1-\gamma)\theta_{\text{target},n} \tag{7}$$

以上を踏まえると R-NaD はアルゴリズム 1 で記述できる．

アルゴリズム 1 R-NaD

```
入力 T, K, M
初期化 θ_1, θ_{reg,1}, θ_{target,1}
m ← 1
for t = 1, 2, ..., T do
    τ_t ← φ
    for k = 1, 2, ..., K do
        σ^{θ_t} に従ってゲームをプレイし, サンプリングした情報集合や報酬を τ_t に追加
    式 (6) で σ^{θ_t}, σ^{reg,θ_m} を用いて τ_t の報酬を変換
    式 (5) で θ_t を θ_{t+1} に更新
    式 (7) で θ_{t+1} を用いて θ_{target,t} を θ_{target,t+1} に更新
    if t が M の倍数 then
        θ_{reg,m+1} ← θ_{target,t+1}
        m ← m + 1
return θ_{reg,m}
```

3. 関連研究

3.1 戦略表現のメモリ使用量削減

Abstruction [5] は類似した情報集合を同一視して戦略のテーブルサイズを小さくする．しかし, 適切な abstraction にはゲーム特有の知識が必要であり, 適用できるゲームの規模にも限りがある．Deep CFR [2] および Double Neural CFR [8] は Counterfactual Regret Minimization (CFR) [18] という平均収束性を持つアルゴリズムの戦略をニューラルネットワークで表現する．どちらの手法についても, 戦略更新の時間計算量が大きいことが問題となる．Single Deep CFR [15] は Deep CFR の平均学習誤差を取り除くために戦略のパラメータ列を記憶媒体に保存し, 評価時にはランダムにサンプルしたパラメータでゲームをプレイすることで平均戦略を近似する．本研究はパラメータを平均することで戦略の平均を求めるので, Single Deep CFR からパラメータ列保存の必要性を省略したと考えられる．

3.2 戦略更新の分散削減

戦略更新の誤差は平均戦略の近似精度に影響を与えるので, 推定値の分散削減が重要視される．Monte Calro CFR (MCCFR) [7] は, CFR の戦略更新の差分をモンテカルロ法によって推定し, NeuRD と同様に時間計算量を削減する．推定値は不偏推定量であり, ナッシュ均衡への収束性は失われないが, 分散が大きく近似精度に影響をもたらすことが問題になる．そこで, Variance Reduction MCCFR [14]

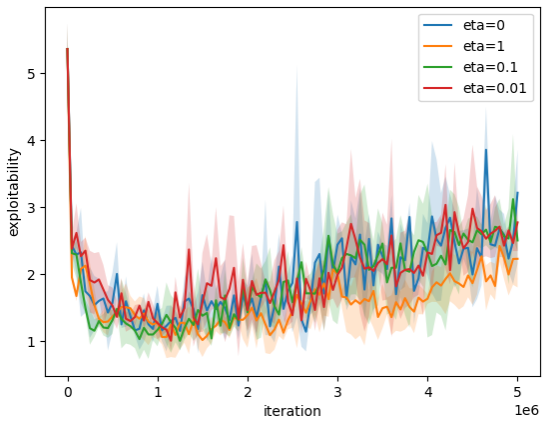

図 1 R-NaD において η を変化させた際の実行結果 (横軸は反復数 T).

アルゴリズム 2 2 段階平均 R-NaD

入力 T, K, M
初期化 θ_1, $\theta_{\text{reg},1}$, $\theta_{\text{target},1}$
$m \leftarrow 1$
for $t = 1, 2, ..., T$ **do**
  $\tau_t \leftarrow \phi$
  for $k = 1, 2, ..., K$ **do**
    σ^{θ_t} に従ってゲームをプレイし, サンプリングした情報集合や報酬を τ_t に追加
  式 (6) で $\sigma^{\theta_t}, \sigma^{\text{reg},\theta_m}$ を用いて τ_t の報酬を変換
  式 (5) で θ_t を θ_{t+1} に更新
  式 (7) で θ_{t+1} を用いて $\theta_{\text{target},t}$ を $\theta_{\text{target},t+1}$ に更新
  if t が M の倍数 **then**
    $\theta_{\text{reg},m+1} \leftarrow (\theta_{\text{reg},m}, \theta_{\text{target},t+1}$ を 4 章で示す方法で平均したパラメータ)
    $m \leftarrow m + 1$
return $\theta_{\text{reg},m}$

はベースライン関数によって, 収束性を維持したまま推定値の分散を削減した. また, DREAM [16] はニューラルネットワークによって情報集合ごとに適したベースライン関数を学習することで Deep CFR の時間計算量を削減した. ただし, DREAM では手番への到達確率によって推定値を割る計算が行われるため, ゲーム木の深さが深く到達確率が小さくなるようなゲームでは分散を減らすことができない. ESCHER [9] は推定値に到達確率の項を含まず, DREAM と比べて分散が小さいが, 価値関数によって推定値を導出するため, 推定値の質が価値関数の近似精度に影響される. NeuRD の戦略更新は ESCHER と同様に到達確率の項を含まない. また, R-NaD では V-trace [3] と呼ばれる手法で推定値の分散削減が行われる.

3.3 最終反復収束

Reward Transformation CFR+ (RT-CFR+) [10] は CFR に報酬変換を適用することで, ゲーム木の全探索を行い戦略をテーブルで表現する状況下において, 報酬変換が最終反復収束だけでなく高い近似精度を提供することを示した.

4. 2 段階平均化

RT-CFR+ のようにゲーム木の全探索が可能で戦略がテーブルで表現されるならば報酬変換はナッシュ均衡の近似精度向上に大きく寄与するが, 図 1 に示す実験結果によって, R-NaD では報酬変換を用いない場合と同程度の近似精度にしか達しない可能性があることが分かった. 実験の詳細は 5.1 節で述べる.

図 1 を見ると, exploitability が急激に振動しナッシュ均衡の近似が安定していないことがわかる. この不安定さは NeuRD の勾配の誤差やニューラルネットワークの学習誤差, 式 (7) の平均化誤差によって生じている. 近似の不安定さはアルゴリズムを通しての近似精度の低下を引き起こす可能性がある. exploitability の計算にはゲーム木の全探索が必要であり, 大規模なゲームでは計算することができない. そのため, アルゴリズムの実行を通して exploitability が最も小さい戦略を採用するといったことはできず, 実行終了時の戦略を実行結果として返さなければならない.

そこで, 本研究では近似の安定化を実現するためのアプローチとして 2 段階平均を提案する. R-NaD では正則化戦略を直接ターゲット戦略で更新しているが, 2 段階平均ではターゲット戦略をさらに平均化させた戦略で更新する. ターゲット戦略を平均することで正則化戦略のパラメータの変動が小さくなり, 近似を安定させることが期待される. 2 段階平均を取り入れた R-NaD はアルゴリズム 2 で記述される. 戦略を平均化させる方法として以下の 3 通りが考えられる.

(1) 確率分布平均

Deep CFR で用いられる方法. 各反復でバッファ $\mathcal{B}$ に情報集合 I と行動選択の確率分布 $\sigma^{\theta_t}(I)$ のペアを格納する. そして, 式 (8) を最小化するようにパラメータ θ_{average} を学習することで, $\sigma^{\theta_{\text{average}}}$ が平均戦略を近似する.

$$\mathbb{E}_{(I,\sigma(I))\sim\mathcal{B}}\left[\sum_{a\in A(I)}\left(\sigma^{\theta_{\text{average}}}(I,a)-\sigma(I,a)\right)^2\right] \tag{8}$$

情報集合と戦略のペアは非常に多くなりバッファに格納しきれない可能性がある. そこで Deep CFR では reservoir sampling [17] を用いてバッファに格納しきれないペアを捨てるようにしている. 実験によって確率分布平均は後述するパラメータ算術平均およびパラメータ指数移動平均と比べて収束速度が遅いことが分

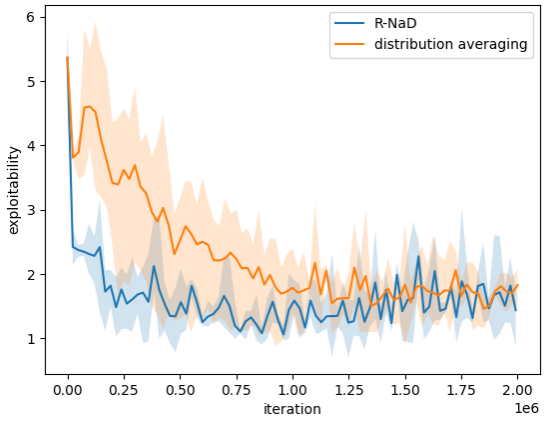

図 2 R-NaD の平均化方法を EMA から確率分布平均に変えた場合の近似精度の比較 (R-NaD が EMA, distribution averaging が確率分布平均).

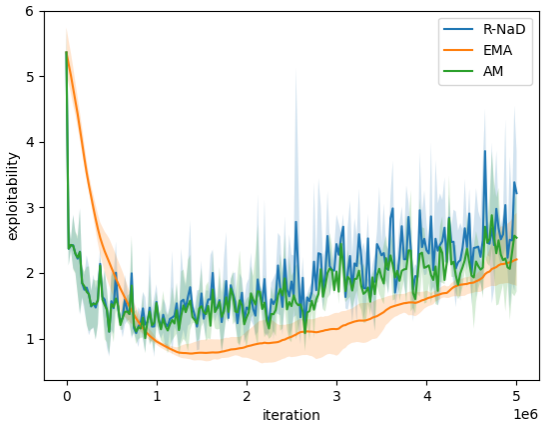

図 3 ベースラインと提案手法の近似精度の比較.

かった. 結果は図 2 に示す. これはニューラルネットワークのミニバッチ学習でサンプルされるミニバッチのサイズがゲームの情報集合数と比べて小さいことが原因であると考えられる. 収束速度が遅いため本研究では採用しない.

(2) パラメータ算術平均 (AM)
式 (9) のようにパラメータの算術平均で戦略を平均する.

$$\begin{aligned}\bar{\theta}_n &= \frac{1}{n}(\theta_1 + \theta_2 + \cdots + \theta_n) \\ &= \frac{1}{n}((n-1)\bar{\theta}_{n-1} + \theta_n)\end{aligned} \tag{9}$$

学習が進むにつれて加えられるパラメータの寄与が小さくなる. 例えば, $\bar{\theta}_1$ から $\bar{\theta}_2$ へ更新する際に加えられる θ_2 の寄与は 1/2 だが, $\bar{\theta}_{99}$ から $\bar{\theta}_{100}$ へ更新する際に加えられる θ_{100} の寄与は 1/100 である. そのため, 学習初期では exploitability が大きく変動し, 学習が進むにつれて変化しなくなることが予想される.

(3) パラメータ指数移動平均 (EMA)
式 (7) で表される方法. 算術平均と違って加えられるパラメータの寄与は常に一定である.

2 段階平均以外に式 (7) の平均化速度 γ を小さくすること, および式 (7) の代わりに式 (9) を使用することで, 正則化戦略のパラメータの変動が小さくなり, 近似精度を安定させられる可能性がある. そこで, 実験ではこれらの手法に対する比較も行う.

5. 実験

以下の 4 つの研究課題を解き明かすことを目的として実験を行う.

(1) 2 段階平均によってナッシュ均衡の近似が安定するか.
(2) 平均化に用いる方法が近似精度にどう影響するか.
(3) EMA によって導入されるハイパーパラメータの値によって近似精度が左右されるか.
(4) R-NaD の平均化方法を修正することで近似が安定するか.

5.1 実験設定

実験は Leduc Poker で行い, ナッシュ均衡の近似精度を exploitability によって評価する. Leduc Poker は情報集合数が 936, 実現可能な行動列が 5520 あるゲームである.

R-NaD の実装は OpenSpiel [6] を使用する. 実験のハイパーパラメータは基本的に OpenSpiel のデフォルト値を使用するが, 行動列のサンプル数 K を 1, ある θ_{reg} に対して戦略を更新する回数 M を 500 とした. 大規模ゲームでは終端の数に対してわずかな割合の行動列しかサンプルできないので K を小さく, ある θ_{reg} に対して $M \times K$ の行動列をサンプルするが, これも終端の数に対してわずかな割合の行動列しかサンプルできないため M を 500 としている. また, 式 (7) の平均化速度 γ は断りのない限り R-NaD と 2 段階目の平均化ともに 0.001 とする.

それぞれの実験設定では乱数のシードを $[0, 1, 2]$ の 3 種類で実行し, 平均値および最大値と最小値で囲まれる範囲をプロットする.

5.2 2 段階平均による平均化の安定および近似精度の向上

ベースライン手法である R-NaD と, 2 段階目の平均として AM を用いた方法および EMA を用いた方法の近似精度を比較した. 図 3 に結果を載せる. EMA において近似が安定していることが確認できる. さらに EMA ではベースライン手法と比べて近似精度が向上していることが分かる. 一方, AM では学習の進行に関わらず近似の安定および近似精度の向上が示されなかった. すべての手法について, 学習が進むにつれ近似精度が悪化する現象が見られる. この点に関しては今後調査を行う.

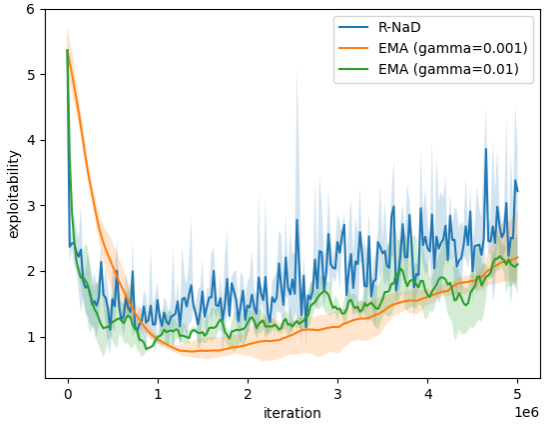

図 4 パラメータ指数移動平均の平均速度 γ を変化させた場合の近似精度の比較.

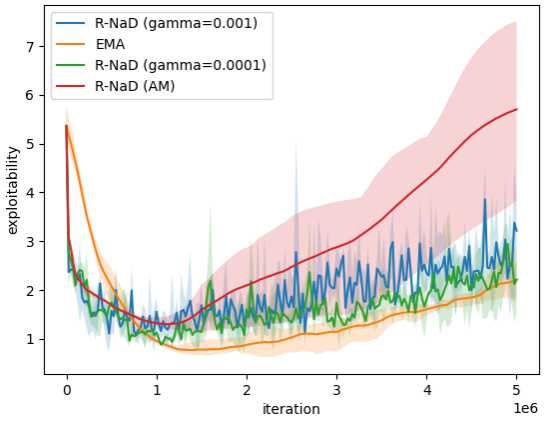

図 5 R-NaD の平均化方法を修正した場合の近似精度の比較.

5.3 ハイパーパラメータ γ による比較

EMA では平均化の速度を表すハイパーパラメータ γ が導入される. γ の設定が近似精度に及ぼす影響を調査するために $\gamma = 0.01$ の結果を追加して近似精度を比較した. 図 4 に結果を載せる. $\gamma = 0.01$ の結果において, 近似精度は $\gamma = 0.001$ の場合と同様にベースラインより高くなっているが, 近似の安定性は $\gamma = 0.001$ の場合と比べて低下している. 一方, $\gamma = 0.001$ の結果では学習初期段階での収束速度が遅いことがわかる. 近似精度の安定性を保ったまま収束速度を上げる方法として γ を徐々に低下させる方法が考えられる. 例えば, 式 (7) において, $\gamma = \max(1/n + 1, 0.001)$ とすると, $n = 999$ までは AM として, $n = 1000$ 以降は $\gamma = 0.001$ の EMA として実行することができる.

5.4 R-NaD の平均化方法を修正した場合との比較

R-NaD において γ を 0.0001 に小さくした場合と, 平均化として AM を使用した場合について近似精度がどう変化するかを検証する. 図 5 に結果を示す. γ を小さくした場合に関してはベースライン手法と比べて近似の安定性および近似精度に大きな変化は見られなかった. 一方, 平均化として AM を使用した場合では EMA を使用するベースラインと比べて高い速度で近似精度が悪化した. 5.2 節の結果も合わせると, パラメータ同士の加算では小さなパラメータの変化であっても確率分布が大きく変化してしまう可能性が考えられる.

6. おわりに

本研究では, 平均化された戦略をさらに平均化することでナッシュ均衡近似の安定化を実現する 2 段階平均を提案し, R-NaD アルゴリズムに適用した. Leduc Poker での実験によって, 2 段階平均はナッシュ均衡の近似を安定させるだけでなく近似精度の向上をもたらすことが確認できた. また, 2 種類の平均化方法を試したところ, 2 段階の平均化でともにパラメータの指数移動平均を用いることが最も優れた結果を導くことが示された.

今後の展望としては, パラメータ指数移動平均の収束速度を上げること, 学習が進むにつれて近似精度が悪化する原因の調査, 異なるゲームやアルゴリズムでの評価が挙げられる. R-NaD 以外にも平均収束性を持つアルゴリズムなどで近似精度の向上につながることが考えられる.

参考文献

[1] Kenshi Abe, Kaito Ariu, Mitsuki Sakamoto, Kentaro Toyoshima, and Atsushi Iwasaki. Last-iterate convergence with full and noisy feedback in two-player zero-sum games. In *International Conference on Artificial Intelligence and Statistics*, pages 7999–8028, 2023.

[2] Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *International Conference on Machine Learning*, pages 793–802, 2019.

[3] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416, 2018.

[4] Daniel Hennes, Dustin Morrill, Shayegan Omidshafiei, Remi Munos, Julien Perolat, Marc Lanctot, Audrunas Gruslys, Jean-Baptiste Lespiau, Paavo Parmas, Edgar Duenez-Guzman, et al. Neural replicator dynamics. *arXiv preprint arXiv:1906.00190*, 2019.

[5] Michael Johanson, Neil Burch, Richard Valenzano, and Michael Bowling. Evaluating state-space abstractions in extensive-form games. In *International Conference on Autonomous Agents and Multiagent Systems*, pages 271–278, 2013.

[6] Marc Lanctot, Edward Lockhart, Jean-Baptiste Lespiau, Vinicius Zambaldi, Satyaki Upadhyay, Julien Pérolat, Sriram Srinivasan, Finbarr Timbers, Karl Tuyls, Shayegan Omidshafiei, et al. Openspiel: A framework for reinforcement learning in games. *arXiv preprint arXiv:1908.09453*, 2019.

[7] Marc Lanctot, Kevin Waugh, Martin Zinkevich, and Michael Bowling. Monte carlo sampling for regret mini-

mization in extensive games. *Advances in Neural Information Processing Systems*, 22, 2009.

[8] Hui Li, Kailiang Hu, Zhibang Ge, Tao Jiang, Yuan Qi, and Le Song. Double neural counterfactual regret minimization. In *International Conference on Learning Representations*, 2020.

[9] Stephen McAleer, Gabriele Farina, Marc Lanctot, and Tuomas Sandholm. ESCHER: Eschewing importance sampling in games by computing a history value function to estimate regret. In *International Conference on Learning Representations*, 2023.

[10] Linjian Meng, Zhenxing Ge, Wenbin Li, Bo An, and Yang Gao. Efficient last-iterate convergence in solving games. *arXiv preprint arXiv:2308.11256*, 2023.

[11] Panayotis Mertikopoulos, Christos Papadimitriou, and Georgios Piliouras. Cycles in adversarial regularized learning. In *Proceedings of the twenty-ninth annual ACM-SIAM symposium on discrete algorithms*, pages 2703–2717, 2018.

[12] Julien Perolat, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, Jerome T Connor, Neil Burch, Thomas Anthony, et al. Mastering the game of stratego with model-free multiagent reinforcement learning. *Science*, 378(6623):990–996, 2022.

[13] Julien Perolat, Remi Munos, Jean-Baptiste Lespiau, Shayegan Omidshafiei, Mark Rowland, Pedro Ortega, Neil Burch, Thomas Anthony, David Balduzzi, Bart De Vylder, et al. From poincaré recurrence to convergence in imperfect information games: Finding equilibrium via regularization. In *International Conference on Machine Learning*, pages 8525–8535, 2021.

[14] Martin Schmid, Neil Burch, Marc Lanctot, Matej Moravcik, Rudolf Kadlec, and Michael Bowling. Variance reduction in monte carlo counterfactual regret minimization (vr-mccfr) for extensive form games using baselines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2157–2164, 2019.

[15] Eric Steinberger. Single deep counterfactual regret minimization. *arXiv preprint arXiv:1901.07621*, 2019.

[16] Eric Steinberger, Adam Lerer, and Noam Brown. Dream: Deep regret minimization with advantage baselines and model-free learning. *arXiv preprint arXiv:2006.10410*, 2020.

[17] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985.

[18] Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. Regret minimization in games with incomplete information. *Advances in Neural Information Processing Systems*, 20, 2007.

