Perturbing Best Responses in Zero-Sum Games

Adam Dziwoki, Rostislav Horčík

Czech Technical University in Prague
dziwoada@fel.cvut.cz, xhorcik@fel.cvut.cz

Abstract

This paper examines the impact of perturbations on best-response-based algorithms that approximate Nash equilibria in zero-sum games, specifically Double Oracle and Fictitious Play. More precisely, we assume that the oracle computing the best responses perturbs the utilities before selecting the best response. We show that using such an oracle reduces the number of iterations for both algorithms. For some cases, suitable perturbations ensure the expected number of iterations is logarithmic. Although the utility perturbation is computationally demanding as it requires iterating through all pure strategies, we demonstrate that one can efficiently perturb the utilities in games where pure strategies have further inner structure.

Code —
https://github.com/geoborek/perturbing-best-responses
Extended version — https://arxiv.org/abs/2511.12523

Introduction

Computing Nash equilibria (NE) in two-player zero-sum games with huge strategy spaces is a computationally demanding problem. Among algorithms approximating NE in such games, a substantial role is played by those based on *best-response oracles* (BROs) due to their ability to consider only a subspace of the strategy spaces. Prominent examples of such algorithms are *Fictitious Play* (FP) introduced in (Brown 1951) and *Double Oracle* (DO) (McMahan, Gordon, and Blum 2003). The latter served as a basis for algorithms leveraging deep reinforcement learning to approximate best responses, such as *Policy Space Response Oracles* (Lanctot et al. 2017; McAleer et al. 2022; Bighashdel et al. 2024).

It is known that each two-player zero-sum game has an approximated ε-NE of logarithmic size in the number of pure strategies n for given $\varepsilon > 0$ (Althöfer 1994; Lipton and Young 1994). On the other hand, it is not difficult to construct games where FP and DO need at least n iterations to find an ε-NE. This raises a natural question whether there is a BRO-based algorithm computing ε-NE in a logarithmic number of iterations. (Hazan and Koren 2016) answered this question negatively as they proved that any (randomized) BRO-based algorithm needs at least $\Omega(\sqrt{n}/\log^3 n)$ iterations. At the same time, they introduced quite a complicated BRO-based algorithm providing a quadratic speed up $O(\sqrt{n}/\varepsilon^2)$ (up to a poly-logarithmic factor). Consequently, it follows that one needs to make BRO more powerful to achieve logarithmic number of iterations.

One way to make BRO stronger is by adding perturbations. Such an oracle perturbs the utilities before computing a best response for a given mixed strategy. We call such an oracle *perturbed BRO* (PBRO). A PBRO-variant of FP known as *Stochastic Fictitious Play* (SFP) was introduced in (Fudenberg and Kreps 1993). Its convergence to NE in zero-sum games was proven in (Hofbauer and Sandholm 2002). In this paper, we prove that SFP achieves the logarithmic complexity in the number of pure strategies n in expectation (note that PBRO makes SFP a randomized algorithm). Analogously, we define a PBRO-based variant of DO called *Stochastic Double Oracle* (SDO). Although its complexity in terms of n is still an open problem, we prove that perturbations ensure logarithmic behavior in expectation for some examples where DO needs $O(n)$ iterations, namely for examples introduced in (Zhang and Sandholm 2024). We also tested experimentally on other games that perturbations reduce the number of iterations. However, it turned out that the perturbations do not accelerate the convergence in random games. Implementing perturbations into a BRO for normal-form games requires iterating through all pure strategies to perturb the utility, which is not computationally efficient. However, for games whose strategy spaces have an inner structure like *partially-observable stochastic game* (POSG), one can efficiently perturb only the rewards for transitions or terminal states. Even though such perturbations do not precisely correspond to the perturbations in normal-form games, we experimentally show that they are able to reduce the number of iterations. More precisely, we demonstrate that on POSGs from (Zhang and Sandholm 2024) and a path-planning game where one player looks for the shortest path in a grid, while the other player might choose any edge and multiply its cost by a fixed coefficient.

Related Work

The convergence of FP was studied in several papers focusing mainly on the convergence rate w.r.t. ε instead of the

size of the game. FP converges to NE in zero-sum games, as proved in (Robinson 1951). Its convergence rate is one of the oldest open problems in game theory, known as Karlin's conjecture (Karlin 1959). However, there are several partial results (Abernethy, Lai, and Wibisono 2021; Daskalakis and Pan 2014). Recently, *Anticipatory Fictitious Play* (AFP) was introduced (Cloud, Wang, and Kerr 2023). AFP still needs $O(n)$ iterations in the worst case, but provides often faster convergence. Moreover, it calls the BRO four times in an iteration, whereas FP only calls the BRO two times. It is known that perturbations are related to the regularization (Hofbauer and Sandholm 2002; Abernethy et al. 2014). A regularized versions of AFP (PU and OMWU) were investigated in (Cen, Wei, and Chi 2024). Both variants need at most $O(\log n)$ iterations, but they maintain probability distributions over all pure strategies, making them unsuitable for large games.

It is easy to see that DO needs $\Theta(n)$ iterations. However, to our knowledge, there is no deeper theoretical study of its convergence rate. Exponential lower bounds for DO applied to POSGs were obtained in (Zhang and Sandholm 2024). Further variants of DO were investigated in (McAleer et al. 2022) but without any convergence guarantees.

Background

This section introduces our notation and the necessary background for the paper. The set of natural numbers $\{1, \dots, n\}$ is denoted $[n]$. The real-valued vectors are denoted by bold lowercase letters, e.g. $\boldsymbol{p} \in \mathbb{R}^n$. Its i-th component is denoted p_i. The projection into the i-th component is the map $\pi_i(\boldsymbol{p}) = p_i$. The vectors from the standard basis of $\mathbb{R}^n$ are denoted $\boldsymbol{e}_1, \dots, \boldsymbol{e}_n$. Thus $\pi_i(\boldsymbol{e}_j) = 1$ iff $i = j$ and $\pi_i(\boldsymbol{e}_j) = 0$ otherwise. Similarly, bold uppercase letters denote matrices, e.g. $\boldsymbol{M} \in \mathbb{R}^{m \times n}$. The entry of $\boldsymbol{M}$ in a row $i \in [m]$ and a column $j \in [n]$ is denoted $\boldsymbol{M}(i, j) = \boldsymbol{e}_i^\top \boldsymbol{M} \boldsymbol{e}_j$. The symbol Δ_n stands for the simplex of probability distributions over $[n]$. We identified the members of Δ_n with vectors $\boldsymbol{p} \in [0, 1]^n$ such that $\sum_{i=1}^m p_i = 1$. Given sets of indexes $R \subseteq [m]$ and $C \subseteq [n]$, $\boldsymbol{M}[R, C]$ is the submatrix of $\boldsymbol{M}$ having only rows with indexes in R and columns with indexes in C.

This paper focuses on 2-player finite zero-sum games that can be identified with matrix games if we index the sets of pure strategies by natural numbers. A *matrix game* is given by a matrix $\boldsymbol{M} \in \mathbb{R}^{m \times n}$. The row indexes $[m]$ correspond to the set of pure strategies of the *row player*. Analogously, the column indexes $[n]$ correspond to the pure strategies of the *column player*. The entry $\boldsymbol{M}(i, j)$ denotes the game's outcome if the row player chooses the i-th strategy and the column player the j-th strategy. A *mixed strategy* for the row player is a probability distribution $\boldsymbol{p} \in \Delta_m$ and analogously for the column player. The expected outcome of the game when the row (resp. column) player plays $\boldsymbol{p} \in \Delta_m$ (resp. $\boldsymbol{q} \in \Delta_n$) can be expressed as $\boldsymbol{M}(\boldsymbol{p}, \boldsymbol{q}) = \boldsymbol{p}^\top \cdot \boldsymbol{M} \cdot \boldsymbol{q}$. We assume that the entries in $\boldsymbol{M}$ represent losses (resp. rewards) for the row (resp. column) player. Thus, the row player chooses her strategy to minimize the expected outcome, whereas the column player wants to maximize it.

The FP and DO algorithms rely on a best-response oracle for each player. Given a mixed strategy $\boldsymbol{q} \in \Delta_n$, the oracle for the row player computes the index of the best response and the corresponding value, namely

$$\mathsf{BR}_r(\boldsymbol{q}) \in \operatorname*{argmin}_{i \in [m]} \pi_i(\boldsymbol{M}\boldsymbol{q}), \quad \mathsf{BRVal}_r(\boldsymbol{q}) = \min_{i \in [m]} \pi_i(\boldsymbol{M}\boldsymbol{q})$$

Analogously for $\boldsymbol{p} \in \Delta_m$, the column player's oracle computes

$$\mathsf{BR}_c(\boldsymbol{p}) \in \operatorname*{argmax}_{j \in [n]} \pi_j(\boldsymbol{p}^\top \boldsymbol{M}), \quad \mathsf{BRVal}_c(\boldsymbol{p}) = \max_{j \in [n]} \pi_j(\boldsymbol{p}^\top \boldsymbol{M})$$

Note that $\mathsf{BR}_r(\boldsymbol{q})$ and $\mathsf{BR}_c(\boldsymbol{p})$ are pure strategies. In general, there might be several best responses for a given mixed strategy, and any of them can be returned by the oracle.

Given $\varepsilon \geq 0$, a pair of strategies $\langle \boldsymbol{p}^*, \boldsymbol{q}^* \rangle$ is said to be *ε-Nash equilibrium* (ε-NE) of the game $\boldsymbol{M}$ if the following inequalities hold:

$$\boldsymbol{M}(\boldsymbol{p}^*, \boldsymbol{q}^*) - \mathsf{BRVal}_r(\boldsymbol{q}^*) \leq \varepsilon, \mathsf{BRVal}_c(\boldsymbol{p}^*) - \boldsymbol{M}(\boldsymbol{p}^*, \boldsymbol{q}^*) \leq \varepsilon$$

A *Nash equilibrium* (NE) of the game $\boldsymbol{M}$ is 0-NE. Given a pair of strategies $\langle \boldsymbol{p}, \boldsymbol{q} \rangle$, a sufficient condition for the pair to form ε-NE is

$$\mathsf{BRVal}_c(\boldsymbol{p}) - \mathsf{BRVal}_r(\boldsymbol{q}) \leq \varepsilon \tag{1}$$

that we use as a termination condition for FP and DO.

This paper investigates the influence of perturbations before selecting the best response. Let $\boldsymbol{u}$ and $\boldsymbol{v}$ be random vectors with respective dimensions m, n whose components are i.i.d. random variables coming from a given distribution. *Perturbed best responses* are defined as follows:

$$\widetilde{\mathsf{BR}}_r(\boldsymbol{q}) \in \operatorname*{argmin}_{i \in [m]} \pi_i(\boldsymbol{M}\boldsymbol{q} - \boldsymbol{u}),$$

$$\widetilde{\mathsf{BR}}_c(\boldsymbol{p}) \in \operatorname*{argmax}_{j \in [n]} \pi_j(\boldsymbol{p}^\top \boldsymbol{M} + \boldsymbol{v})$$

When we use the perturbed best responses in FP or DO, we assume that the probability distribution for the perturbations vectors $\boldsymbol{u}, \boldsymbol{v}$ is fixed.

This paper considers two common distributions *uniform* $\mathcal{U}(a, b)$ on the interval $[a, b]$ and *Gumbel* $G(\mu, \beta)$. The Gumbel distribution $G(\mu, \beta)$ is a continuous probability distribution with a CDF given by $F(x) = e^{-e^{-(x-\mu)/\beta}}$ where μ is a location and $\beta > 0$ a scale. The Gumbel distribution is tightly related to the well-known function $\mathsf{softmax}$, see e.g. (Goodfellow, Bengio, and Courville 2016), mapping $\boldsymbol{x} \in \mathbb{R}^n$ to a probability distribution $\mathsf{softmax}(\boldsymbol{x}) \in \Delta_n$ defined by

$$\pi_i(\mathsf{softmax}(\boldsymbol{x})) = \frac{e^{x_i}}{\sum_{j=1}^n e^{x_j}}$$

The following lemma is due to (Gumbel 1954) and is called *Gumbel-max trick* in the machine learning community; for details see (Train 2009, Chapter 3) or (Franke and Degen 2023). Given $\boldsymbol{x} \in \mathbb{R}^n$, the Gumbel-max trick allows to sample from $\mathsf{softmax}(\boldsymbol{x})$ by taking argmax of the perturbed values x_i.

Lemma 1. *Let $\boldsymbol{x} \in \mathbb{R}^n$ and $i^* \in \{1, \dots, n\}$. Let $\boldsymbol{z} \in \mathbb{R}^n$ be a random vector whose components z_i be drawn i.i.d. from $G(0, \beta)$. Then the probability*

$$P(\operatorname*{argmax}_{i \in [n]} \pi_i(\boldsymbol{x} + \boldsymbol{z}) = i^*) = \pi_{i^*}(\mathsf{softmax}(\boldsymbol{x}/\beta))$$

Algorithm 1: Stochastic Fictitious Play

Require: Initial strategies k, l; $\varepsilon > 0$; a probability distribution for $\widetilde{\mathsf{BR}}_r$ and $\widetilde{\mathsf{BR}}_c$
Ensure: ε-NE $\langle \boldsymbol{p}^*, \boldsymbol{q}^* \rangle$
$t \leftarrow 1$
$\boldsymbol{p} \leftarrow \boldsymbol{e}_k, \boldsymbol{q} \leftarrow \boldsymbol{e}_l$
$lb \leftarrow \mathsf{BRVal}_r(\boldsymbol{q}), ub \leftarrow \mathsf{BRVal}_c(\boldsymbol{p})$
while $ub - lb > t\varepsilon$ **do**
  $t \leftarrow t + 1$
  $i \leftarrow \widetilde{\mathsf{BR}}_r(\boldsymbol{q}), j \leftarrow \widetilde{\mathsf{BR}}_c(\boldsymbol{p})$
  $\boldsymbol{p} \leftarrow \boldsymbol{p} + \boldsymbol{e}_i, \boldsymbol{q} \leftarrow \boldsymbol{q} + \boldsymbol{e}_j$
  $lb \leftarrow \mathsf{BRVal}_r(\boldsymbol{q}), ub \leftarrow \mathsf{BRVal}_c(\boldsymbol{p})$
end while
return $\langle \boldsymbol{p}^*, \boldsymbol{q}^* \rangle = \langle \boldsymbol{p}/t, \boldsymbol{q}/t \rangle$

Dually, it follows from Lemma 1 that

$$P(\operatorname*{argmin}_{i \in [n]} \pi_i(\boldsymbol{x} - \boldsymbol{z}) = i^*) = \pi_{i^*}(\mathsf{softmax}(-\boldsymbol{x}/\beta))$$

as $\operatorname{argmin}_i \pi_i(\boldsymbol{x} - \boldsymbol{z}) = i^*$ iff $\operatorname{argmax}_i \pi_i(-\boldsymbol{x} + \boldsymbol{z}) = i^*$. Consequently, if we perturb the utilities with the Gumbel distribution $G(0, \beta)$, the perturbed best-response $\widetilde{\mathsf{BR}}_r(\boldsymbol{q})$ is sampled from $\mathsf{softmax}(\boldsymbol{M}\boldsymbol{q}/\beta)$ and analogously $\widetilde{\mathsf{BR}}_c(\boldsymbol{p})$ from $\mathsf{softmax}(-\boldsymbol{p}^\top \boldsymbol{M}/\beta)$.

To prove our result on SFP, we need to recall the *randomized exponentially weighted forecaster* (REWF), introduced in (Cesa-Bianchi and Lugosi 2006, Chapter 4). Let $\boldsymbol{M} \in [0,1]^{m \times n}$ be a $m \times n$-matrix viewed as a loss function $[m] \times [n] \to [0,1]$. Consider a "sort of" game between a player and an opponent with $T \geq 1$ rounds. In the round t, the player chooses an action $i_t \in [m]$ and the opponent an action $j_t \in [n]$, making the player to suffer the loss $\boldsymbol{M}(i_t, j_t)$. The player's goal is to minimize the cumulative loss $\sum_{t=1}^{T} \boldsymbol{M}(i_t, j_t)$. REWF is a randomized strategy for the player based on the opponent's previously selected actions. In the round t, REWF samples i_t based on $j_1, \ldots, j_{t-1}$ from the distribution $\mathsf{softmax}(-\eta \sum_{s=1}^{t-1} \boldsymbol{M}\boldsymbol{e}_{j_s})$, where $\eta > 0$ is a parameter. Following REWF ensures an upper bound on the regret, i.e., the difference between the cumulative loss and the best fixed player's action i; it immediately follows from (Cesa-Bianchi and Lugosi 2006, Theorem 2.2 and Lemma 4.1).

Corollary 2. *Let $\boldsymbol{M} \in [0,1]^{m \times n}$, $T \geq 1$, $\eta > 0$, and $\delta \in (0,1)$. REWF satisfies, with probability $1 - \delta$,*

$$\sum_{t=1}^{T} \boldsymbol{M}(i_t, j_t) - \min_{i \in [m]} \sum_{t=1}^{T} \boldsymbol{M}(i, j_t) \leq \frac{\ln m}{\eta} + \frac{T\eta}{8} + \sqrt{\frac{T}{2} \ln \frac{1}{\delta}}$$

Stochastic Fictitious Play

Stochastic Fictitious Play (SFP) is a randomized version of *Fictitious Play* (FP). Its pseudocode is shown in Algorithm 1. We expand it with the termination condition (1) so that it stops after reaching ε-NE. Note that the best-response values $\mathsf{BRVal}_r(\boldsymbol{q})$, $\mathsf{BRVal}_c(\boldsymbol{p})$ are not perturbed, hence the termination condition is not affected by the perturbations. The algorithm maintains two vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ representing multisets of already played pure strategies. In each iteration, SFP finds perturbed best responses w.r.t. $\boldsymbol{p}$ and $\boldsymbol{q}$ respectively[1]. When the termination condition is satisfied, SFP returns the average of all the played strategies so that the resulting vectors $\boldsymbol{p}^*, \boldsymbol{q}^*$ form probability distributions. Note that even though Algorithm 1 operates with vectors $\boldsymbol{e}_i \in \mathbb{R}^m$, $\boldsymbol{e}_j \in \mathbb{R}^n$, SFP can be implemented efficiently without considering all the dimensions by storing only the multisets of pure strategies played by the players up to the current iteration.

Using the Gumbel-max trick (see Lemma 1), we show that SFP combined with Gumbel perturbations corresponds to the computation when both players apply the randomized exponentially weighted forecaster (REWF) against each other. W.l.o.g. we assume that the matrix game $\boldsymbol{M} \in \mathbb{R}^{m \times n}$ is normalized into $[0,1]$ and $m \leq n$.

Theorem 3. *Let $\boldsymbol{M} \in [0,1]^{m \times n}$ be a matrix game such that $m \leq n$ and $\varepsilon > 0$. An ε-NE can be computed by SFP with perturbations sampled from $G(0, \beta)$ in $O(\frac{\log n}{\varepsilon^2})$ expected iterations, where $\beta = \frac{2 + \sqrt{2 \ln n}}{\varepsilon \sqrt{8 \ln n}}$.*

Proof. We first prove that SFP finds ε-NE in $O(\frac{\log n}{\varepsilon^2})$ iterations with probability at least $1/2$. We let SFP to iterate for $T = \left(\frac{2 + \sqrt{2 \ln n}}{\varepsilon}\right)^2 \in O(\frac{\log n}{\varepsilon^2})$ iterations.

Let $\boldsymbol{p}_t$, $\boldsymbol{q}_t$ denote the vectors $\boldsymbol{p}, \boldsymbol{q}$ at the iteration $t \leq T$. Further, let $i_t = \widetilde{\mathsf{BR}}_r(\boldsymbol{q}_{t-1})$ and $j_t = \widetilde{\mathsf{BR}}_c(\boldsymbol{p}_{t-1})$. At the iteration t, we have $\boldsymbol{q}_{t-1} = \sum_{s=1}^{t-1} \boldsymbol{e}_{j_s}$. Thus $\boldsymbol{M}\boldsymbol{q}_{t-1} = \sum_{s=1}^{t-1} \boldsymbol{M}\boldsymbol{e}_{j_s}$. By Lemma 1, i_t is sampled from the distribution $\mathsf{softmax}(-\eta \sum_{s=1}^{t-1} \boldsymbol{M}\boldsymbol{e}_{j_s})$ for $\eta = 1/\beta$. By our choice of β and T, we have $\eta = \sqrt{8 \ln n / T}$. Consequently, Corollary 2 for $\delta = 1/4$ implies with probability at least $3/4$, using also $m \leq n$ and $\ln 4 \leq 2$,

$$\begin{aligned}\sum_{t=1}^{T} \boldsymbol{M}(i_t, j_t) - \min_{i \in [m]} \sum_{t=1}^{T} \boldsymbol{M}(i, j_t) &\leq \frac{\ln m}{\eta} + \frac{T\eta}{8} + \sqrt{\frac{T}{2} \ln 4} \\ &\leq \frac{\ln n}{\eta} + \frac{T\eta}{8} + \sqrt{T} = \sqrt{T}\left(1 + \sqrt{\frac{\ln n}{2}}\right)\end{aligned}$$

Applying Corollary 2 to the matrix $\mathbf{1} - \boldsymbol{M}^\top$, we can derive an analogous bound for the column player with probability at least $3/4$

$$\max_{j \in [n]} \sum_{t=1}^{T} \boldsymbol{M}(i_t, j) - \sum_{t=1}^{T} \boldsymbol{M}(i_t, j_t) \leq \sqrt{T}\left(1 + \sqrt{\frac{\ln n}{2}}\right)$$

Consequently, both inequalities hold simultaneously with probability at least $1/2$ using the well-known lower bound $P(A \cap B) \geq P(A) + P(B) - 1$ on the probability of the intersection of two events A, B. Summing the above inequalities, we get

$$\max_{j \in [n]} \sum_{t=1}^{T} \boldsymbol{M}(i_t, j) - \min_{i \in [n]} \sum_{t=1}^{T} \boldsymbol{M}(i, j_t) \leq \sqrt{T}(2 + \sqrt{2 \ln n})$$

[1] Formally, $\boldsymbol{p}$, $\boldsymbol{q}$ are not probability distributions. We assume w.l.o.g. that the best-response oracle works for them as well.

Algorithm 2: Stochastic Double Oracle

Require: Initial strategies k, l; $\varepsilon > 0$; a probability distribution for $\widetilde{\mathsf{BR}}_r$ and $\widetilde{\mathsf{BR}}_c$
Ensure: ε-NE $\langle \boldsymbol{p}^*, \boldsymbol{q}^* \rangle$

```
t ← 1
R ← {k}, C ← {l}
p ← e_k, q ← e_l
lb ← BRVal_r(q), ub ← BRVal_c(p)
while ub − lb > ε do
    t ← t + 1
    ⟨p, q⟩ ← getNash(M[R, C])
    i ← BR̃_r(q), R ← R ∪ {i}
    j ← BR̃_c(p), C ← C ∪ {j}
    lb ← BRVal_r(q), ub ← BRVal_c(p)
end while
return ⟨p*, q*⟩ = ⟨p, q⟩
```

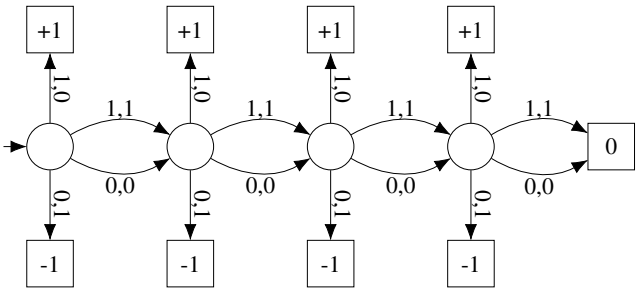

Figure 1: An example of the 4-bit stochastic game.

SFP returns the pair $\langle \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{e}_{i_t}, \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{e}_{j_t} \rangle$. Note that

$$ub = \mathsf{BRVal}_c\left(\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{e}_{i_t}\right) = \max_{j\in[n]} \pi_j \left(\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{e}_{i_t}^\top \boldsymbol{M}\right) = \frac{1}{T}\max_{j\in[n]} \sum_{t=1}^{T} \boldsymbol{M}(i_t, j)$$

Analogously, $lb = \frac{1}{T}\min_{i\in[m]} \sum_{t=1}^{T} \boldsymbol{M}(i, j_t)$. Combining the above facts, we get $ub - lb \le (2 + \sqrt{2\ln n})/\sqrt{T} = \varepsilon$. Thus $\langle \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{e}_{i_t}, \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{e}_{j_t} \rangle$ forms ε-NE with probability at least $1/2$.

To obtain an algorithm with a logarithmic expected number of iterations, we execute SFP for T many iterations until we find ε-NE. It occurs in the first or second run in expectation. □

Stochastic Double Oracle

This section introduces *Stochastic Double Oracle* (SDO), a randomized variant of Double Oracle (DO) with perturbed best responses. Its pseudocode is shown in Algorithm 2. As DO, it maintains two sets of pure strategies R, C for the row and column player, respectively. At each iteration, it computes NE for the induced subgame $\boldsymbol{M}[R, C]$. The termination condition is the same as in SFP.

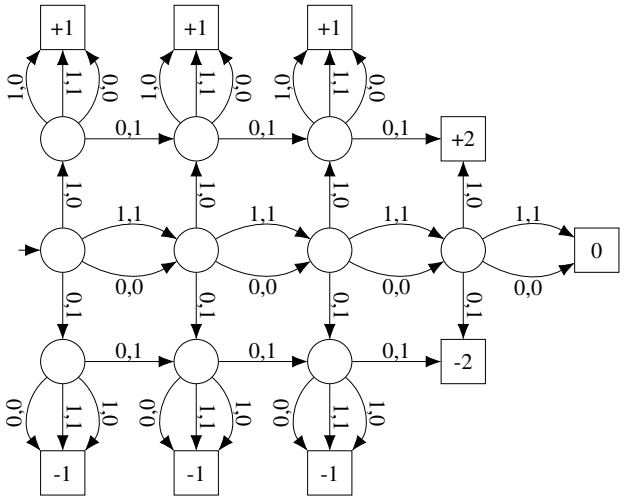

Figure 2: An example of the 4-bit POSG.

In this paper, we take the first steps in investigating convergence for SDO. We start with two matrix games introduced in (Zhang and Sandholm 2024) that are difficult to solve for DO, i.e., DO needs to iterate through all pure strategies to find ε-NE. We prove that SDO with suitable perturbations finds ε-NE in a logarithmic number of iterations in expectation for both.

Example 1. Let $n \ge 1$. In the first game, both players choose a number from $[n]$. The player who chooses the greater number wins. If the chosen numbers are equal, it is a draw. The corresponding square matrix $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ is defined as follows:

$$\boldsymbol{L} = \begin{pmatrix} 0 & 1 & \cdots & 1 \\ -1 & 0 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & 0 \end{pmatrix}$$

The game $\boldsymbol{L}$ has a unique pure ε-NE consisting of the last row and column for any $0 \le \varepsilon < 1$. Thus finding ε-NE for $\varepsilon < 1$ is the same as finding NE. If DO is initialized with $k = l = 1$ and the best responses are chosen adversarially (i.e., we always take the best response with the least index), DO needs n iterations to find the NE. Note that there are several candidates when DO looks for a best response to a row or column. On the other hand, when we apply SDO to $\boldsymbol{L}$, the perturbations allow us to select a candidate uniformly randomly.

We prove our result on SDO by applying drift theory (Kötzing and Krejca 2019). For technical reasons, we prove it w.l.o.g. for a dual game where the player who chooses the smaller number wins. Its matrix $\boldsymbol{S} = \boldsymbol{L}^\top$ has a unique NE consisting of the first row and column.

Let $k > 1$. Consider the k-th row $\boldsymbol{x}$ of $\boldsymbol{S}$, i.e., $x_i = 1$ for $i < k$, $x_k = 0$, and $x_i = -1$ for $i > k$. The following crucial lemma shows that $\widetilde{\mathsf{BR}}_c(\boldsymbol{e}_k)$ with uniform perturbations selects a best response uniformly among $\{1, \ldots, k-1\}$. Analogous lemma can be proven for the k-th column.

Lemma 4. *Let $\boldsymbol{x} = \langle 1, \dots, 1, 0, -1, \dots, -1 \rangle \in \mathbb{R}^n$ be the k-th row of $\boldsymbol{S}$, $k > 1$, and $\boldsymbol{z} = \langle z_1, \dots, z_n \rangle$ a random vector whose components $z_i \sim \mathcal{U}(-1/2, 1/2)$. Let I be the random variable $I = \operatorname{argmax}_{i \in [n]} \pi_i(\boldsymbol{x} + \boldsymbol{z})$. Then $E[I] = \frac{k}{2}$.*

Proof. Note that $x_i + z_i \sim \mathcal{U}(1/2, 3/2)$ for $i < k$, $x_i + z_i \sim \mathcal{U}(-1/2, 1/2)$ for $i = k$, and $x_i + z_i \sim \mathcal{U}(-3/2, -1/2)$ for $i > k$. Thus we have $E[I] = \sum_{i=1}^{n} i \cdot P(I = i) = \sum_{i=1}^{k-1} i \cdot P(I = i)$ because $P(I = i) = 0$ for $i \geq k$. Moreover, $P(I = i) = \frac{1}{k-1}$ is the uniform distribution on $\{1, \dots, k-1\}$ as the first $k-1$ components of $\boldsymbol{x}$ are equal. Consequently, $E(I) = \frac{1}{k-1} \sum_{i=1}^{k-1} i = \frac{k}{2}$. □

Next, we need to understand equilibria of the subgame $\boldsymbol{S}[R, C]$ in every iteration of SDO. For notational simplicity, we index rows and columns in $\boldsymbol{S}[R, C]$ with the same indexes as in the original matrix $\boldsymbol{S}$. Further, we denote the probability simplices over R, C by Δ_R, Δ_C, respectively.

Lemma 5. *Let $R, C \subseteq [n]$ and $r = \min R$, $c = \min C$.*

1. *If $r < c$, then $\langle \boldsymbol{e}_i, \boldsymbol{q} \rangle$ is NE in $\boldsymbol{S}[R, C]$ for any $i \in \{k \in R \mid k < c\}$ and $\boldsymbol{q} \in \Delta_C$. There are no other NEs.*
2. *Dually, if $c < r$, then $\langle \boldsymbol{p}, \boldsymbol{e}_j \rangle$ is NE in $\boldsymbol{S}[R, C]$ for any $j \in \{k \in C \mid k < r\}$ and $\boldsymbol{p} \in \Delta_R$. There are no other NEs.*
3. *If $r = c$, then $\langle \boldsymbol{e}_r, \boldsymbol{e}_c \rangle$ is the unique NE in $\boldsymbol{S}[R, C]$.*

Proof. For the first item, let $\boldsymbol{x}$ be an i-th row of $\boldsymbol{S}$ for $i \in \{k \in R \mid i < c\}$. Note that $x_j = -1$ for all $j \in C$. Thus the i-th row in $\boldsymbol{S}[R, C]$ is equilibrial strategy for the row player. A column player can play an arbitraty strategy. The second item is proven analogously as the first one.

The last claim follows if we note that the r-th row in $\boldsymbol{S}[R, C]$ is of the form $\langle 0, -1, \dots, -1 \rangle$ and c-th column in $\boldsymbol{S}[R, C]$ is of the form $\langle 0, 1, \dots, 1 \rangle$. □

Theorem 6. *SDO finds NE of $\boldsymbol{S}$ in $O(\log n)$ expected number of iterations.*

Proof. We assume $k, l = n$, which is the worst possible start for SDO on $\boldsymbol{S}$. Let R_t, C_t be the row and column strategies in the iteration t. Define $r_t = \min R_t$ and $c_t = \min C_t$. Next, we define $X_t = \max\{r_t, c_t\}$. Apparently, we have $X_{t+1} \leq X_t$ since $r_{t+1} \leq r_t$ and $c_{t+1} \leq c_t$. Once $X_t = 1$, SDO will find the NE in the next iteration. Thus, we want to show $E[T] \in O(\log n)$ where $T = \inf\{t \mid X_t \leq 1\}$. This can be proven by (Kötzing and Krejca 2019, Corollary 17), if we show that $X_t - E[X_{t+1} \mid X_1, \dots, X_t] \geq \delta X_t$ for some $\delta > 0$. Regarding the sequence $X_1, X_2, \dots$, we will consider only subsequence $X_1, X_3, \dots$ of odd iterations t as SDO needs often two iterations to improve strategies for both players. In other words, we will prove $X_t - E[X_{t+2} \mid X_1, X_3 \dots, X_t] \geq \delta X_t$ for some $\delta > 0$. To simplify the notation, we denote the conditional expectations without the condition in the rest of proof. For instance, $E[X_{t+2}]$ means $E[X_{t+2} \mid X_1, X_3, \dots, X_t]$.

Let t be an odd iteration. There are two cases. We prove the case if $r_t \leq c_t$. The proof for the case when $r_t \geq c_t$ is analogous. If $r_t \leq c_t$, then $\langle \boldsymbol{e}_i, \boldsymbol{q} \rangle$ is NE for $\boldsymbol{S}[R_t, C_t]$ for some $i \leq c_t$ and some $\boldsymbol{q} \in \Delta_C$ by Lemma 5 (items 1 and 3). Consequently, the best response for the column player is $j = \widetilde{\mathsf{BR}}_c(\boldsymbol{e}_i) < i \leq c_t$. Thus j becomes the new minimum of C_{t+1}, i.e., $c_{t+1} = j$. Applying Lemma 4, we get $E[c_{t+1}] = i/2 \leq c_t/2$.

In the iteration $t+1$, we either have $r_{t+1} \leq c_{t+1}$ or $r_{t+1} > c_{t+1}$. In the first case, $X_{t+2} \leq X_{t+1} = c_{t+1}$. Thus $E[X_{t+2}] \leq E[c_{t+1}] \leq c_t/2 \leq \max\{r_t, c_t\}/2 = X_t/2$. If $r_{t+1} > c_{t+1}$, then $\langle \boldsymbol{p}, \boldsymbol{e}_j \rangle$ for some $j < r_{t+1}$ and any $\boldsymbol{p} \in \Delta_R$ by Lemma 5. Consequently, the row player's best response $i = \widetilde{\mathsf{BR}}_r(\boldsymbol{e}_j) < j < r_{t+1}$ becomes the minimum of R_{t+2}, i.e., $r_{t+2} = i$. Applying the column analog of Lemma 4, we get $E[r_{t+2}] = j/2 \leq r_{t+1}/2$. As $c_{t+2} \leq c_{t+1}$, we have $X_{t+2} \leq \max\{r_{t+2}, c_{t+1}\}$. Consequently, $E[X_{t+2}] \leq \max\{r_{t+1}/2, c_t/2\} \leq \max\{r_t/2, c_t/2\} \leq X_t/2$.

To sum up, we have $X_t - E[X_{t+2}] \geq X_t/2$. (Kötzing and Krejca 2019, Corollary 17) implies $E[T] \leq 2 \ln n$ for the subsequence of odd iterations. Thus SDO needs at most $1 + 4 \ln n$ iterations to find the NE in expectation. □

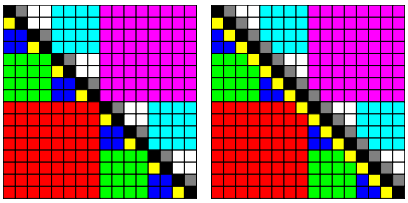

Figure 3: The clusters corresponding to the terminal states.

Example 2. The second example from (Zhang and Sandholm 2024) is a modification of the game from Example 1 where the best responses are unique. We will again use its dual variant. Its matrix $\boldsymbol{U} \in \mathbb{R}^{n \times n}$ is defined as follows:

$$\boldsymbol{U} = \begin{pmatrix} 0 & -2 & -1 & \cdots & -1 \\ 2 & 0 & -2 & \cdots & -1 \\ 1 & 2 & 0 & \cdots & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \cdots & 0 \end{pmatrix}$$

Again, $\boldsymbol{U}$ has unique ε-NE for $0 \leq \varepsilon < 1$ consisting of the first row and column. DO needs n iterations to find the NE, if initialized with $k = l = n$.

The following theorem can be proven analogously as Theorem 6. The main difference is a modification of Lemma 4; Lemma 8 below (its proof is in the extended version of the paper). It shows that the perturbed best responses w.r.t. a single row $\boldsymbol{e}_k$ are not selected uniformly from $\{1, \dots, k-1\}$ as was the case for the game $\boldsymbol{S}$. The distribution is due to the unique best responses shifted towards larger indexes. However, the expected index is less than $3/4k$ which is sufficient to apply (Kötzing and Krejca 2019, Corollary 17).

Theorem 7. *SDO finds NE of $\boldsymbol{U}$ in $O(\log n)$ expected number of iterations.*

Lemma 8. *Let $\boldsymbol{x} = \langle 1, \dots, 1, 2, 0, -2, -1 \dots, -1 \rangle \in \mathbb{R}^n$ be the k-th row of $\boldsymbol{U}$, $k > 1$, and $\boldsymbol{z} = \langle z_1, \dots, z_n \rangle$ a*

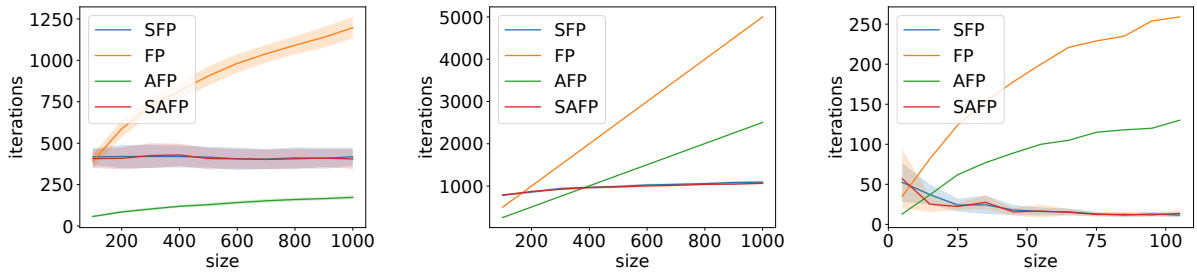

Figure 4: SFP iterations on random $[0, 1]$-matrix games (left), $\boldsymbol{U}^\top$ (middle), and the f-finger Morra game (right) for $\varepsilon = 0.1$.

random vector whose components $z_i \sim \mathcal{U}(-1, 1)$. *Let* I *be the random variable* $I = \operatorname{argmax}_{i\in[n]} \pi_i(\boldsymbol{x} + \boldsymbol{z})$. *Then* $E[I] \leq 3/4k$.

Efficient Perturbations

Let $\boldsymbol{M} \in \mathbb{R}^{m\times n}$ be a matrix game and $\boldsymbol{q} \in \Delta_n$. Suppose we have a best-response oracle computing BR_r. To implement a perturbed best response $\widetilde{\mathsf{BR}}_r(\boldsymbol{q})$ using BR_r, it requires perturbing every component of $\boldsymbol{Mq}$. That cannot be done if SDO maintains only a fraction of $\boldsymbol{M}$. Nevertheless, suppose $\boldsymbol{M}$ represents a game with inner structure like an extensive-form game (EFG) or a partially-observable stochastic game (POSG). In that case, we can implement an efficient PBRO using a best-response oracle and perturbation of utilities/rewards for terminal states or transitions.

Consider the (fully observable) stochastic game from Figure 1. In this game, each player chooses a sequence of four bits that corresponds to the player's actions taken in non-terminal (circular) states. The terminal (squared) states contain their corresponding reward. This game and its generalization for any number of bits n was defined in (Zhang and Sandholm 2024). Considering the n-bit game, the equivalent matrix game is of size $2^n \times 2^n$ and has the form of the matrix $\boldsymbol{L}$ from Example 1. To perturb best responses efficiently in this game, one can perturb only the rewards of the terminal states before applying the standard best-response oracle.

We experimentally tested that this kind of efficient perturbation speeds up the convergence of SDO for the stochastic game from Figure 1. Furthermore, we did the same for the POSG from Figure 2 also coming from (Zhang and Sandholm 2024). In this POSG, the players cannot observe the state of the game. So they again choose only four bits. The n-bit generalization of this game has an equivalent matrix game of size $2^n \times 2^n$ and has the form of matrix $\boldsymbol{U}^\top$; see Example 2. The experimental results are shown in the next section. Here, we explain the details of our implementation.

One possibility to test our hypothesis for the game in Figure 1 would be implementing the best response oracle as an MDP solver, for instance using the value iteration algorithm (VI). However, VI would eliminate the existence of multiple best responses as it computes the optimal decision for each non-terminal state. For example, if the second player chooses 0 as her first bit, any sequence starting with 1 is a best response for the first player. However, VI would return $1, 1, 1, 1$ as the only best response.

Thus, we tested the hypothesis on the corresponding matrix game $\boldsymbol{L}$. Each matrix entry corresponds to a terminal state. However, this correspondence is not one-to-one, as several entries correspond to the same terminal state. So, it induces a clustering of entries in $\boldsymbol{L}$ according to the terminal states. The clustering for $\boldsymbol{L}$ is shown in Figure 3 (left). Each color represents a single cluster.

Now, perturbing the reward of a terminal state is the same as perturbing its corresponding cluster by the same random value. Let K be the number of clusters and $\boldsymbol{z}$ a K-dimensional random vector whose components z_i are i.i.d. random variables. For each cluster $k \in [K]$, we denote the $\{0, 1\}$-matrix $\boldsymbol{B}_k \in \mathbb{R}^{m\times n}$ that masks the cluster k, i.e., $\boldsymbol{B}_k(i, j) = 1$ iff the i, j-entry belongs to the cluster k and $\boldsymbol{B}_k(i, j) = 0$ otherwise. Finally, for a column player's mixed strategy $\boldsymbol{q} \in \Delta_n$, we compute the perturbed best-response as follows:

$$\widetilde{\mathsf{BR}}_r(\boldsymbol{q}) = \operatorname*{argmin}_{i\in[m]} \pi_i \left(\left(\boldsymbol{L} + \sum_{k=1}^{K} z_k \boldsymbol{B}_k \right) \boldsymbol{q} \right) \qquad (2)$$

The best response for the column player is defined analogously. The same clustering can be applied also for the POSG in Figure 2. The corresponding clustering is shown in Figure 3 (right).

Experiments

All our experiments were implemented in the programming language Python and experimentally evaluated on the CPU AMD Ryzen 7 PRO 7840U. All randomized algorithms were repeated ten times to report the mean and the standard deviation. The initialization of SFP and SDO was deterministic, with the algorithm starting at the worst possible indexes. For tie-breaking best responses, the one with the least index was taken. To generate pseudorandom values, we used the numpy library with the initial seed 1.

SFP was tested in combination with Gumbel perturbations $G(0, \beta)$ where β was set according to Theorem 3. We ran SFP with $\varepsilon = 0.1$ on random square $[0, 1]$-matrix games, the matrix games $\boldsymbol{L}$ (Example 1) and $\boldsymbol{U}^\top$ (Example 2), and f-finger Morra game (Good 1965) all normalized to the interval $[0, 1]$; Figure 4. We compared SFP with FP, AFP, and

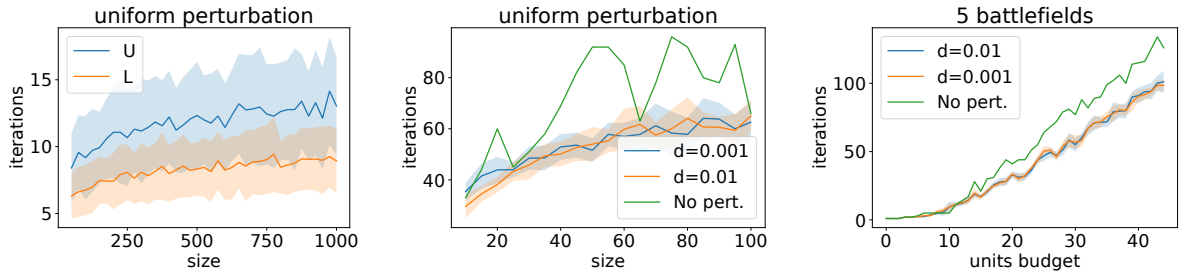

Figure 5: SDO iterations on $\boldsymbol{L}$ and $\boldsymbol{U}^\top$ with perturbations from $\mathcal{U}(-1, 1)$ (left), f-finger Morra game (middle), and Colonel Blotto (right) with perturbations from $\mathcal{U}(-d, d)$ for $\varepsilon = 0.1$.

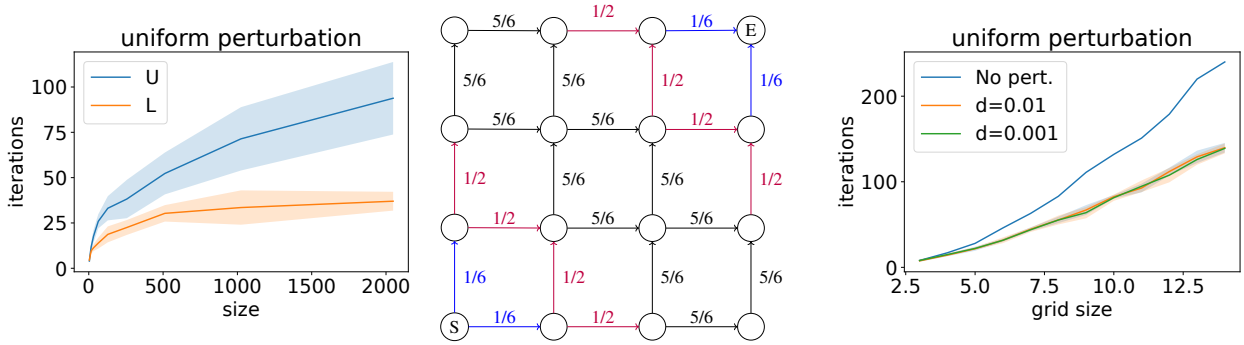

Figure 6: SDO iterations for $\boldsymbol{L}$ and $\boldsymbol{U}^\top$ with efficient perturbations from $\mathcal{U}(-1, 1)$ (left), 4×4-grid with the transition costs for the path-planning game (middle), and the corresponding SDO iterations with uniform perturbations $\mathcal{U}(-d, d)$ (right).

AFP with perturbed best responses (SAFP). The left graph shows the number of iterations on random $n \times n$-matrix games. In the experiment, we generated 100 random games for each n. AFP outperforms all the other methods; SFP and SAFP behave similarly on random games. The middle graph presents the number of iterations for the game $\boldsymbol{U}^\top$. An analogous graph for $\boldsymbol{L}$ looks almost identical, so we omit it. FP and AFP show their the worst-case $O(n)$ complexity. SFP and SAFP again have a similar performance. Finally, the right graph shows the number of iterations for the f-finger Morra game. The horizontal axis represents the number of fingers f. The size of the corresponding matrix is $f^2 \times f^2$. Interestingly, SFP and SAFP are able to quickly find an ε-NE for large values of f. However, this is not the case for FP and AFP.

We did similar experiments for DO and SDO combined with the uniform perturbations. The SDO iterations for matrix games $\boldsymbol{L}$, $\boldsymbol{U}^\top$, f-finger Morra game, and the Colonel Blotto game with five battlefields for different numbers of units are shown in Figure 5. We omit the DO iterations in the left graph as the DO needs n iterations for the size n. In all these cases, SDO outperforms DO. However, we noticed this is not the case for random games where perturbations do not provide a faster convergence or even degrades the convergence rate if the perturbations are too large. This outcome is somewhat expected, considering that the best responses in random games are inherently stochastic.

Further, we tested the efficient perturbations of clusters, see Equation (2), on the matrix games $\boldsymbol{L}$ and $\boldsymbol{U}^\top$ of size $2^n \times 2^n$ corresponding to stochastic games from Figure 1 and 2. The results are shown in Figure 6 (left). Although SDO with efficient perturbations requires more iterations in comparison with Figure 5 (left), still the obtained results are much better than the linear complexity of DO.

To test the efficient perturbations further, we defined a path-planning game on an $n \times n$-grid. An example of such a grid for $n = 4$ is shown in Figure 6 (middle). The colors and numbers denote the costs for particular transitions. The path-planning player looks for the shortest path in the grid starting in the node S and finishing in the node E. The other player selects a single edge and multiplies its cost by a given coefficient; 10 in our experiments. We implemented the perturbed best-response oracle by perturbing the transition costs every time the oracle is called. Figure 6 (right) show that uniform perturbations reduce the number of iterations with increasing grid size.

Conclusions

To summarize, perturbing best responses improves the convergence of FP and DO. We proved that SFP has a logarithmic complexity in the number of pure strategies n. Although DO needs $\Theta(n)$ many iterations to find ε-NE in the worst case, it is still possible that SDO finds an ε-NE in $O(\log n)$ expected number of iterations. We leave this open problem for future research.

Acknowledgments

The authors were supported by the Czech Science Foundation grant–no. 24-12046S.

References

Abernethy, J. D.; Lai, K. A.; and Wibisono, A. 2021. Fast Convergence of Fictitious Play for Diagonal Payoff Matrices. In Marx, D., ed., *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, 1387–1404. SIAM.

Abernethy, J. D.; Lee, C.; Sinha, A.; and Tewari, A. 2014. Online Linear Optimization via Smoothing. In Balcan, M.; Feldman, V.; and Szepesvári, C., eds., *Proceedings of The 27th Conference on Learning Theory, COLT 2014, Barcelona, Spain, June 13-15, 2014*, volume 35 of *JMLR Workshop and Conference Proceedings*, 807–823. JMLR.org.

Althöfer, I. 1994. On sparse approximations to randomized strategies and convex combinations. *Linear Algebra and its Applications*, 199: 339–355.

Bighashdel, A.; Wang, Y.; McAleer, S.; Savani, R.; and Oliehoek, F. A. 2024. Policy Space Response Oracles: A Survey. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, 7951–7961. ijcai.org.

Brown, G. W. 1951. Iterative Solution of Games by Fictitious Play. In Koopmans, T. C., ed., *Activity Analysis of Production and Allocation*. New York: Wiley.

Cen, S.; Wei, Y.; and Chi, Y. 2024. Fast Policy Extragradient Methods for Competitive Games with Entropy Regularization. *J. Mach. Learn. Res.*, 25: 4:1–4:48.

Cesa-Bianchi, N.; and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge University Press.

Cloud, A.; Wang, A.; and Kerr, W. 2023. Anticipatory Fictitious Play. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, 73–81. ijcai.org.

Daskalakis, C.; and Pan, Q. 2014. A Counter-example to Karlin's Strong Conjecture for Fictitious Play. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, 11–20. IEEE Computer Society.

Franke, M.; and Degen, J. 2023. The softmax function: Properties, motivation, and interpretation. https://doi.org/10.31234/osf.io/vsw47.

Fudenberg, D.; and Kreps, D. M. 1993. Learning Mixed Equilibria. *Games and Economic Behavior*, 5(3): 320–367.

Good, R. A. 1965. f-Finger Morra. *SIAM Review*, 7(1): 81–87.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep Learning*. MIT Press.

Gumbel, E. J. 1954. *Statistical Theory of Extreme Values and Some Practical Applications: A Series of Lectures*, volume 33 of *Applied mathematics series*. U.S. Government Printing Office.

Hazan, E.; and Koren, T. 2016. The computational power of optimization in online learning. In Wichs, D.; and Mansour, Y., eds., *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, 128–141. ACM.

Hofbauer, J.; and Sandholm, W. H. 2002. On the Global Convergence of Stochastic Fictitious Play. *Econometrica*, 70(6): 2265–2294.

Karlin, S. 1959. *Mathematical methods and theory in games, programming, and economics*. Reading, U. S.

Kötzing, T.; and Krejca, M. S. 2019. First-hitting times under drift. *Theor. Comput. Sci.*, 796: 51–69.

Lanctot, M.; Zambaldi, V. F.; Gruslys, A.; Lazaridou, A.; Tuyls, K.; Pérolat, J.; Silver, D.; and Graepel, T. 2017. A Unified Game-Theoretic Approach to Multiagent Reinforcement Learning. In Guyon, I.; von Luxburg, U.; Bengio, S.; Wallach, H. M.; Fergus, R.; Vishwanathan, S. V. N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, 4190–4203.

Lipton, R. J.; and Young, N. E. 1994. Simple strategies for large zero-sum games with applications to complexity theory. In Leighton, F. T.; and Goodrich, M. T., eds., *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, 734–740. ACM.

McAleer, S.; Lanier, J.; Wang, K.; Baldi, P.; Fox, R.; and Sandholm, T. 2022. Self-Play PSRO: Toward Optimal Populations in Two-Player Zero-Sum Games. arXiv:2207.06541.

McMahan, H. B.; Gordon, G. J.; and Blum, A. 2003. Planning in the Presence of Cost Functions Controlled by an Adversary. In Fawcett, T.; and Mishra, N., eds., *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA*, 536–543. AAAI Press.

Robinson, J. 1951. An Iterative Method of Solving a Game. *Annals of Mathematics*, 54(2): 296–301.

Train, K. E. 2009. *Discrete Choice Methods with Simulation*. Cambridge University Press.

Zhang, B. H.; and Sandholm, T. 2024. Exponential Lower Bounds on the Double Oracle Algorithm in Zero-Sum Games. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, 3032–3039. ijcai.org.