Approximate Well-Supported Nash Equilibria Below Two-Thirds*

John Fearnley[1], Paul W. Goldberg[1],
Rahul Savani[1], and Troels Bjerre Sørensen[2]

[1] Department of Computer Science, University of Liverpool, UK
[2] Department of Computer Science, University of Warwick, UK

Abstract. In an ϵ-Nash equilibrium, a player can gain at most ϵ by changing his behaviour. Recent work has addressed the question of how best to compute ϵ-Nash equilibria, and for what values of ϵ a polynomial-time algorithm exists. An ϵ-*well-supported* Nash equilibrium (ϵ-WSNE) has the additional requirement that any strategy that is used with non-zero probability by a player must have payoff at most ϵ less than a best response. A recent algorithm of Kontogiannis and Spirakis shows how to compute a 2/3-WSNE in polynomial time, for bimatrix games. Here we introduce a new technique that leads to an improvement to the worst-case approximation guarantee.

1 Introduction

In a bimatrix game, a Nash equilibrium is a pair of strategies in which both players only assign probability to best responses. The apparent hardness of computing an exact Nash equilibrium [5,4] has led to work on computing approximate Nash equilibria, and two notions of approximate Nash equilibria have been developed. The first, and more widely studied, notion is of an ϵ-*approximate Nash equilibrium* (ϵ-Nash), where each player is required to achieve an expected payoff that is within ϵ of a best response. A line of work [7,6,2] has investigated the best ϵ that can be guaranteed in polynomial time. The current best result in this setting is a polynomial time algorithm that finds a 0.3393-Nash equilibrium [12].

However, ϵ-Nash equilibria have a drawback: since they only require that the expected payoff is within ϵ of a pure best response, it is possible that a player could be required to place probability on a strategy that is arbitrarily far from being a best response. This issue is addressed by the second notion of an approximate Nash equilibrium. An ϵ-*well supported approximate Nash equilibrium* (ϵ-WSNE), requires that both players only place probability on strategies that have payoff within ϵ of a pure best response. This is a stronger notion of equilibrium, because every ϵ-WSNE is an ϵ-Nash, but the converse is not true.

* This work is supported by by EPSRC grant EP/H046623/1 "Synthesis and Verification in Markov Game Structures", and EPSRC grants EP/G069239/1 and EP/G069034/1 "Efficient Decentralised Approaches in Algorithmic Game Theory." A full version of this paper is available at `http://arxiv.org/abs/1204.0707`

M. Serna (Ed.): SAGT 2012, LNCS 7615, pp. 108–119, 2012.

In contrast to ϵ-Nash, there has been relatively little work ϵ-WSNE. The first result on the subject gave a $\frac{5}{6}$ additive approximation [7], but this only holds if a certain a graph-theoretic conjecture is true. The best-known polynomial-time additive approximation algorithm was given by Kontogiannis and Spirakis, and achieves a $\frac{2}{3}$-approximation [10]. We will call this algorithm the KS algorithm. In [9], which is an earlier conference version of [10], the authors presented an algorithm that they claimed was polynomial-time and achieves a ϕ-WSNE, where $\phi = \frac{\sqrt{11}}{2} - 1 \approx 0.6583$, but this was later withdrawn, and instead the polynomial-time $\frac{2}{3}$-approximation algorithm was presented in [10]. It has also been shown that there is a PTAS for ϵ-WSNE if and only if there is a PTAS for ϵ-Nash [4].

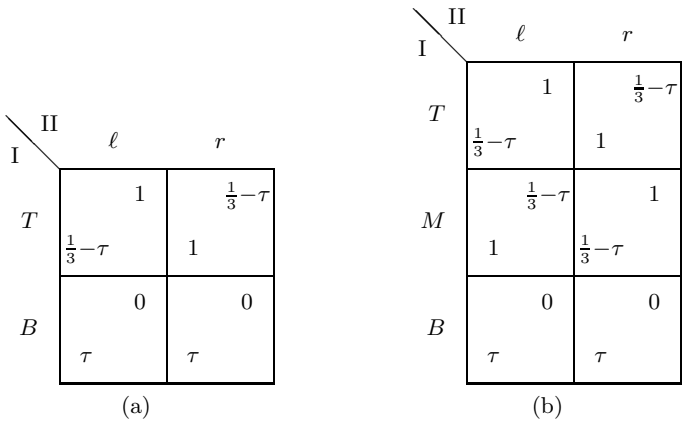


Fig. 1. Two examples that approach the worst case for the KS algorithm

Our approach. We build on the KS algorithm for finding a $\frac{2}{3}$-WSNE. Figure 1a gives a game where the KS algorithm produces a $\frac{2}{3}$-WSNE. The KS algorithm begins by checking there is a *pure* $\frac{2}{3}$-WSNE. In Figure 1a, there is a pure $\frac{2}{3}$-WSNE when $\tau = 0$, but not when $\tau > 0$, because any pure profile where both payoffs are at least $\frac{1}{3}$ is a $\frac{2}{3}$-WSNE. If no pure $\frac{2}{3}$-WSNE exists, the algorithm solves the zero-sum game $(D, -D)$, where $D = \frac{1}{2}(R - C)$, and gives the solution as a WSNE in the original game. In Figure 1a, if τ is small, then the solution to the zero-sum game has the row player playing B, and the column player mixing equally between ℓ and r. The *regret* for the row player is the difference between the payoff of a best response, and the lowest payoff of a row used by the row player. In our example, the row player's regret is the difference between the payoff of B and the payoff of T, and we can see that as $\tau \to 0$, the row player's regret approaches $\frac{2}{3}$. Since we have a ϵ-WSNE only if both players have regret smaller than ϵ, the quality of the WSNE approaches the worst-case bound of $\frac{2}{3}$.

Notice that in Figure 1a we can improve things for the row player by transferring some of the *column* player's probability from r to ℓ. The row player's regret is reduced, and the column player's regret is the same. However, consider Figure 1b. Once again this is approximately worst-case for the KS algorithm; the column player again mixes ℓ and r, while the row player uses row B, again getting regret of about $\frac{2}{3}$. This game is designed to prevent the trick of shifting some of the column player's probability so as to reduce the row player's regret.

In this case however, there is a new trick, which is to focus on rows T and M, and columns ℓ and r, where the payoffs are similar to the Matching Pennies game. By mixing uniformly on these strategies, the players both obtain average payoffs more than $\frac{1}{3}$, so that their regret in the entire game must be less than $\frac{2}{3}$.

Our main result is to show that one of these tricks can always be applied, and that we can always produce an ϵ-WSNE with $\epsilon < \frac{2}{3}$. We give an algorithm with three steps. The first step finds the best pure WSNE, and corresponds to the preprocessing step of the KS algorithm. The second step searches for the best WSNE where both players use at most two strategies, which corresponds to checking whether the Matching Pennies trick can be applied. The third step uses the KS algorithm to find a $\frac{2}{3}$-WSNE, and then finds the best possible WSNE that can be produced through our trick of shifting probabilities. We show that one of these three steps will always produce an ϵ-WSNE with $\epsilon = \frac{2}{3} - 0.004735 \approx 0.6619$.

2 Definitions

A *bimatrix game* is a pair (R, C) of two $n \times n$ matrices: R gives payoffs for the *row player*, and C gives payoffs for the *column player*. We assume that all payoffs are in the range $[0, 1]$. We use $[n] = \{1, 2, \ldots n\}$ to denote the *pure strategies* for each player. To play the game, both players simultaneously select a pure strategy: the row player selects a row $i \in [n]$, and the column player selects a column $j \in [n]$. The row player then receives $R_{i,j}$, and the column player receives $C_{i,j}$.

A *mixed strategy* is a probability distribution over $[n]$. We denote a mixed strategy as a vector $\mathbf{x}$ of length n, such that $\mathbf{x}_i$ is the probability that the pure strategy i is played. The *support* of mixed strategy $\mathbf{x}$, denoted $\text{Supp}(\mathbf{x})$, is the set of pure strategies i with $x_i > 0$. If $\mathbf{x}$ and $\mathbf{y}$ are mixed strategies for the row and column player, respectively, then we call $(\mathbf{x}, \mathbf{y})$ a *mixed strategy profile*.

Let $\mathbf{y}$ be a mixed strategy for the column player. The *best responses* against $\mathbf{y}$ for the row player is the set of pure strategies that maximize the payoff against $\mathbf{y}$. More formally, a pure strategy $i \in [n]$ is a best response against $\mathbf{y}$ if, for all pure strategies $i' \in [n]$ we have: $\sum_{j \in [n]} \mathbf{y}_j \cdot R_{i,j} \geq \sum_{j \in [n]} \mathbf{y}_j \cdot R_{i',j}$. Column player best responses are defined analogously. A mixed strategy profile $(\mathbf{x}, \mathbf{y})$ is a *mixed Nash equilibrium* if every pure strategy in $\text{Supp}(\mathbf{x})$ is a best response against $\mathbf{y}$, and every pure strategy in $\text{Supp}(\mathbf{y})$ is a best response against $\mathbf{x}$. Nash [11] showed that all bimatrix games have a mixed Nash equilibrium.

An *approximate well-supported Nash equilibrium* weakens the requirements of a mixed Nash equilibrium. For a mixed strategy $\mathbf{y}$ of the column player, a pure strategy $i \in [n]$ is an *ϵ-best response* for the row player if, for all pure

strategies $i' \in [n]$ we have: $\sum_{j\in[n]} \mathbf{y}_j \cdot R_{i,j} \geq \sum_{j\in[n]} \mathbf{y}_j \cdot R_{i',j} - \epsilon$. We define ϵ-best responses for the column player analogously. A mixed strategy profile $(\mathbf{x}, \mathbf{y})$ is an *ϵ-well-supported Nash equilibrium* (ϵ-WSNE) if every pure strategy in $\text{Supp}(\mathbf{x})$ is an ϵ-best response against $\mathbf{y}$, and every pure strategy in $\text{Supp}(\mathbf{y})$ is an ϵ-best response against $\mathbf{x}$.

3 Our Algorithm

We begin with an algorithm for finding the best WSNE on a given pair of supports. Let S_c and S_r be supports for the column and row player, respectively. We define an LP, which assumes that the row player uses a strategy with support S_r, and then finds a strategy on S_c that minimizes the row player's regret.

Definition 1. *Let $\mathbf{y}'$ be a mixed strategy for the column player. We define:*

$$\textbf{\textit{Minimize:}} \qquad \epsilon$$

$$\textbf{\textit{Subject to:}} \qquad R_{i'} \cdot \mathbf{y}' - R_i \cdot \mathbf{y}' \leq \epsilon \qquad i \in S_r,\ i' \in [n] \qquad (1)$$

$$\mathbf{y}'_j = 0 \qquad j \notin S_c \qquad (2)$$

A linear program for the row player can be defined symmetrically.

Let $(\mathbf{y}^*, \epsilon_{\mathbf{y}})$ be a solution of the LP given in Definition 1 (that is, $\mathbf{y}^*$ and $\epsilon_{\mathbf{y}}$ are the values of $\mathbf{y}'$ and ϵ that result) with parameters S_r and S_c, and let $(\mathbf{x}^*, \epsilon_{\mathbf{x}})$ be a solution of the corresponding LP for the row player. We define ϵ^* to be $\max(\epsilon_{\mathbf{x}}, \epsilon_{\mathbf{y}})$, and we have the following property.

Proposition 2. *$(\mathbf{x}^*, \mathbf{y}^*)$ is an ϵ^*-WSNE.*

More importantly, we can show that $(\mathbf{x}^*, \mathbf{y}^*)$ is at least as good, or better than, all well-supported Nash equilibria with support S_c and S_r.

Proposition 3. *For every ϵ-WSNE $(\mathbf{x}, \mathbf{y})$ with $\text{Supp}(\mathbf{x}) = S_r$ and $\text{Supp}(\mathbf{y}) = S_c$, we have $\epsilon^* \leq \epsilon$.*

Our algorithm for finding a WSNE consists of three distinct procedures.

(1) **Find the best pure WSNE.** The KS algorithm requires a preprocessing step that eliminates all pure $\frac{2}{3}$-WSNE, and this is a generalisation of that step. Suppose that the row player plays row i, and that the column player plays column j. Let: $\epsilon_r = \max_{i'}(R_{i',j}) - R_{i,j}$, and $\epsilon_c = \max_{j'}(C_{i,j'}) - C_{i,j}$. Thus i is an ϵ_r-best response against j, and that j is an ϵ_c-best response against i. Therefore, (i, j) is a $\max(\epsilon_r, \epsilon_c)$-WSNE. We can find the best pure WSNE by checking all $O(n^2)$ possible pairs of pure strategies. Let ϵ_p be the best approximation guarantee that is found by this procedure.

(2) **Find the best WSNE with 2×2 support.** We can use the linear program from Definition 1 to implement this procedure. For each of the $O(n^4)$ possible 2×2 supports, we solve the LPs to find a WSNE. Proposition 3 implies that this WSNE is at least as good as the best WSNE on those supports. Let ϵ_m be the best approximation guarantee that is found by this procedure.

(3) **Find an improvement over the KS algorithm.** The KS algorithm constructs a zero-sum game $(D, -D)$, where $D = \frac{1}{2}(R - C)$, and solves it. Kontogiannis and Spirakis showed that, if there is no pure $\frac{2}{3}$-WSNE, the min-max strategies for the zero-sum game are always a $\frac{2}{3}$-WSNE in the original game [10]. To find an improvement over the KS algorithm, we take the mixed strategy pair $(\mathbf{x}, \mathbf{y})$ that is produced by the KS algorithm, and we use the linear program from Definition 1 with parameters $S_r = \text{Supp}(\mathbf{x})$ and $S_c = \text{Supp}(\mathbf{y})$. Let $(\mathbf{x}^*, \mathbf{y}^*)$ be the mixed strategy profile returned by the LPs, and let ϵ_i be the smallest value such that $(\mathbf{x}^*, \mathbf{y}^*)$ is a ϵ_i-WSNE.

We take the smallest of ϵ_p, ϵ_m, and ϵ_i, and return the corresponding WSNE.

4 Outline

We want to show that our algorithm finds a $(\frac{2}{3} - z)$-WSNE, for some $z > 0$. The precise value of z will be determined during the proof, so for now we treat z as a parameter. At a high level, we will show that if $\epsilon_p > \frac{2}{3} - z$, and if $\epsilon_m > \frac{2}{3} - z$, then we must have $\epsilon_i \leq \frac{2}{3} - z$. Recall that Procedure (3) takes the mixed strategy profile $(\mathbf{x}, \mathbf{y})$, and finds the best WSNE on the supports of $\mathbf{x}$ and $\mathbf{y}$. Our approach is to use the assumptions that $\epsilon_p > \frac{2}{3} - z$ and $\epsilon_m > \frac{2}{3} - z$ to construct $(\mathbf{x}', \mathbf{y}')$, which is a specific $(\frac{2}{3} - z)$-WSNE on the supports of $\mathbf{x}$ and $\mathbf{y}$. The existence of $(\mathbf{x}', \mathbf{y}')$ then implies that Procedure (3) must produce at least a $(\frac{2}{3} - z)$-WSNE.

In our proof, we focus on how the mixed strategy $\mathbf{y}'$ can be constructed from $\mathbf{y}$. However, all of our arguments can be applied symmetrically in order to construct $\mathbf{x}'$ from $\mathbf{x}$. Our approach is to take the strategy $\mathbf{y}$ and to improve it. If $\mathbf{x}$ is not a $(\frac{2}{3} - z)$-best response against $\mathbf{y}$, then there must be at least one row i such that $R_i \cdot \mathbf{y} > \frac{2}{3} - z$. We call these *bad rows*, and the goal of our construction is to improve all bad rows, so that we can find a $(\frac{2}{3} - z)$-WSNE. We will first define a strategy $\mathbf{y}^{imp}$, which improves a specific bad row. Then, we define $\mathbf{y}'$ to be a convex combination of $\mathbf{y}$ and $\mathbf{y}^{imp}$. Formally, we will define $\mathbf{y}' = \mathbf{y}(t)$, where $t \in [0, 1]$, and $\mathbf{y}(t) := (1 - t) \cdot \mathbf{y} + t \cdot \mathbf{y}^{imp}$.

For the remainder of the proof, we will be concerned with finding a value of z for which the following property holds.

Definition 4. *$P(z)$ is the property of (non-negative real value) z that there exists $t \in [0, 1]$ such that, for all row player strategies $\mathbf{x}'$ with $\text{Supp}(\mathbf{x}') = \text{Supp}(\mathbf{x})$, $\mathbf{x}'$ is a $(\frac{2}{3} - z)$-best response against $\mathbf{y}(t)$.*

Since all of our arguments can also be applied to the row player, if $P(z)$ holds then there must exist a t such that $(\mathbf{x}(t), \mathbf{y}(t))$ is a $(\frac{2}{3} - z)$-WSNE. Our goal is to find the largest value of z for which $P(z)$ holds in all bimatrix games. Once we have determined the appropriate z, we will have then shown that our algorithm will always find a $(\frac{2}{3} - z)$-WSNE for all possible input games.

In the final part of our proof, we will develop a test that represents a sufficient condition for $P(z)$ to hold in all bimatrix games. If the test is passed then $P(z)$ holds in all bimatrix games, but we do not prove that $P(z)$ does not hold when

the test is failed. Our test is monotone in z, and so to complete our proof, we use binary search to find the largest z for which the test tells us that $P(z)$ holds. We find that the test is passed when $z = 0.004735$, but failed when $z = 0.004736$. Thus, we arrive at our main result.

Theorem 5. *The algorithm given in Section 3 finds a* $(\frac{2}{3} - 0.004735)$*-WSNE.*

5 The Proof

5.1 Re-analysing the KS Algorithm

The original KS algorithm uses a preprocessing step that checks for a *pure* $\frac{2}{3}$-WSNE, and stops if one is found. In our version we initially check for a pure $\frac{2}{3} - z$-WSNE, a stronger requirement that leaves more input games that have to be handled by the rest of the algorithm. The results we establish for the rest of the algorithm are given in terms of the column player's strategy; corresponding results hold when the row player is considered.

Proposition 6. *Assume that* $\epsilon_p > \frac{2}{3} - z$*, and let* $(\mathbf{x}, \mathbf{y})$ *be the WSNE returned by the KS algorithm. If the row player has regret larger than* $\frac{2}{3} - z$ *in* $(\mathbf{x}, \mathbf{y})$*, then for all rows* i' *we have both of the following:*

$$R_{i'} \cdot \mathbf{y} \leq \frac{2}{3} + 2z, \qquad R_{i'} \cdot \mathbf{y} - C_{i'} \cdot \mathbf{y} \leq 3z.$$

This proposition shows that, under our new assumptions the KS algorithm will now produce a mixed strategy pair $(\mathbf{x}, \mathbf{y})$ that is a $(\frac{2}{3} + 2z)$-WSNE. The main goal of our proof is to show that the probabilities in $\mathbf{x}$ and $\mathbf{y}$ can be rearranged to construct a $(\frac{2}{3} - z)$-WSNE. From this point onwards, we only focus on improving the strategy $\mathbf{y}$, with the understanding that all of our techniques can be applied in the same way to improve the strategy $\mathbf{x}$.

Our improvement procedure must consider the rows i whose payoff lies in the range $\frac{2}{3} - z < R_i \cdot \mathbf{y} \leq \frac{2}{3} + 2z$. We call these rows *bad* rows, because they are the rows that must be improved to produce a $(\frac{2}{3} - z)$-WSNE. We classify the bad rows according to how bad they are.

Definition 7. *A row* i *is* q-bad *if* $R_i \cdot \mathbf{y} = \frac{2}{3} + 2z - qz$.

It can be seen from Proposition 6 that every row is q bad for some $q \geq 0$, and we are particularly interested in the q-bad rows with $0 \leq q < 3$.

5.2 The Structure of a q-Bad Row

To define our improvement procedure, we must understand the structure of a q-bad row. If i is a q-bad row, then we can apply the second inequality of Proposition 6 to obtain:

$$C_i \cdot \mathbf{y} \geq \frac{2}{3} - z - qz. \tag{3}$$

Now consider a q-bad row i with $q < 3$. We can deduce the following three properties about row i.

- Definition 7 tells us that $R_i \cdot \mathbf{y}$ is close to $\frac{2}{3}$.
- Equation (3) tells us that $C_i \cdot \mathbf{y}$ is close to $\frac{2}{3}$.
- The fact that $\epsilon_p > \frac{2}{3} - z$ implies that, for each column j, we must either have $R_{i,j} < \frac{1}{3} + z$ or $C_{i,j} < \frac{1}{3} + z$, because otherwise (i, j) would be a pure $(\frac{2}{3} - z)$-WSNE.

In order to satisfy all three of these conditions simultaneously, the row i must have a very particular form, which the rows T and M in Figure 1b show: approximately half of the probability assigned by $\mathbf{y}$ must be given to columns j where $R_{i,j}$ is close to 1 and $C_{i,j}$ is close to $\frac{1}{3}$, and the other (approximately) half of the probability assigned by $\mathbf{y}$ must be given to columns j where $R_{i,j}$ is close to $\frac{1}{3}$ and $C_{i,j}$ is close to 1.

Building on this observation, we split the columns of each row i into three sets. We define the set B_i of *big* columns to be $B_i = \{j \ : \ R_{i,j} \geq \frac{2}{3} + 2z\}$, and the set S_i of *small* columns to be $S_i = \{j \ : \ C_{i,j} \geq \frac{2}{3} + 2z\}$. Finally, we have the set of *other* columns $O_i = \{1, 2, \ldots, n\} \setminus (B_i \cup S_i)$, which contains all columns that are neither big nor small. We can then formalise our observations by giving inequalities about the amount of probability that $\mathbf{y}$ can assign to these sets.

Proposition 8. *If i is a q-bad row then:*

$$\sum_{j \in O_i} \mathbf{y}_j \leq \frac{2qz}{\frac{1}{3} - 2z},$$

$$\sum_{j \in B_i} \mathbf{y}_j \geq \frac{\frac{1}{3} + z - qz - (\frac{1}{3} + z) \sum_{j \in O_i} \mathbf{y}_j}{\frac{2}{3} - z},$$

$$\sum_{j \in S_i} \mathbf{y}_j \geq \frac{\frac{1}{3} - 2z - qz - (\frac{1}{3} + z) \sum_{j \in O_i} \mathbf{y}_j}{\frac{2}{3} - z}.$$

The first inequality is obtained by an application of Markov's inequality. The second two can be proved by substituting bounds for B_i, S_i, and O_i into Definition 7 and Equation 3. The inequalities show that, if $q = 0$, then $\mathbf{y}$ must give a roughly equal split between the big and small columns. As q increases, our inequalities become weaker, and the split may become more lopsided.

5.3 The Improved Strategies $\mathbf{y}^{imp}$ and $\mathbf{y}(t)$

We now define an improved version of $\mathbf{y}$. We start by constructing $\mathbf{y}^{imp}$, which will improve the *worst* bad row. That is, we choose $\bar{\imath}$ to be the index of a row in $\arg\max_i(R_i \cdot \mathbf{y})$, and therefore $\bar{\imath}$ is a $\bar{q}$-bad row such that there is no q-bad row with $q < \bar{q}$. We fix $\bar{\imath}$ and $\bar{q}$ to be these choices for the rest of this paper. If $\bar{q} \geq 3$, then $\mathbf{y}$ does not need to be improved. Therefore, we can assume that $\bar{q} < 3$.

We aim to improve row $\bar{\imath}$ by moving the probability assigned to $B_{\bar{\imath}}$ to $S_{\bar{\imath}}$. This is a generalisation of shifting probability from the first column to the

second column in Figure 1a. Formally, we define the strategy $\mathbf{y}^{imp}$, for each j with $1 \leq j \leq n$, as:

$$\mathbf{y}^{imp}_j = \begin{cases} 0 & \text{if } j \in B_{\bar{\imath}}, \\ \mathbf{y}_j + \frac{\mathbf{y}_j \cdot \sum_{k \in B_{\bar{\imath}}} \mathbf{y}_k}{\sum_{k \in S_{\bar{\imath}}} \mathbf{y}_k} & \text{if } j \in S_{\bar{\imath}}, \\ \mathbf{y}_j & \text{otherwise.} \end{cases}$$

The strategy $\mathbf{y}^{imp}$ improves the specific bad row $\bar{\imath}$, but other rows may not improve, or even get worse in $\mathbf{y}^{imp}$. Therefore, we propose that $\mathbf{y}$ should be gradually improved towards $\mathbf{y}^{imp}$. More formally, for the parameter $t \in [0, 1]$, we define the strategy $\mathbf{y}(t)$ to be $(1 - t) \cdot \mathbf{y} + t \cdot \mathbf{y}^{imp}$.

5.4 An Upper Bound on $R_i \cdot \mathbf{y}^{imp}$

Recall that $P(z)$ checks whether there exists a t such that all row player strategies with support $\text{Supp}(\mathbf{x})$ are $(\frac{2}{3} - z)$-best responses against $\mathbf{y}(t)$. In order to perform this test, we check whether there exists a t such that $R_i \cdot \mathbf{y}(t) \leq \frac{2}{3} - z$, for all rows i. Thus, eventually, we will need an upper bound on $R_i \cdot \mathbf{y}(t)$ for each row i. Since $\mathbf{y}(t)$ is a convex combination of $\mathbf{y}$ and $\mathbf{y}^{imp}$, we begin the construction of our test by finding an upper bound on $R_i \cdot \mathbf{y}^{imp}$.

The strategy $\mathbf{y}^{imp}$ is defined by moving all probability from $B_{\bar{\imath}}$ to $S_{\bar{\imath}}$. We are interested in the effect that this can have on a q-bad row $i \neq \bar{\imath}$. If we consider the partition of the columns in $\bar{\imath}$ into $(B_{\bar{\imath}}, S_{\bar{\imath}}, O_{\bar{\imath}})$, and the partition of the columns in i into (B_i, S_i, O_i), then we have a decomposition into nine possible intersections:

Row $\bar{\imath}$	$B_{\bar{\imath}}$			$S_{\bar{\imath}}$			$O_{\bar{\imath}}$		
Row i	B_i	S_i	O_i	B_i	S_i	O_i	B_i	S_i	O_i

We cannot know the precise amount of probability that $\mathbf{y}$ assigns to each of the sets in the decomposition. However, Proposition 8 gives useful constraints on the probabilities allocated to the sets used in the decomposition. We will use these inequalities to write down a linear program that characterises $R_i \cdot \mathbf{y}^{imp}$.

The LP will have one variable for each of the sets in the decomposition. The idea is that each variable should represent the amount of probability that $\mathbf{y}$ assigns to that set. Thus, we have nine variables: d_{bb}, d_{bs}, d_{bo}, and so on, where the variable d_{bb} represents $\sum_{j \in B_{\bar{\imath}} \cap B_i} \mathbf{y}_j$, the variable d_{bs} represents $\sum_{j \in B_{\bar{\imath}} \cap S_i} \mathbf{y}_j$, and so on. For convenience, we use $\sum d_{b*}$ as a shorthand for $d_{bb} + d_{bs} + d_{bo}$, and $\sum d_{*b}$ as a shorthand $d_{bb} + d_{sb} + d_{ob}$. We also use $\sum d_{s*}$, $\sum d_{*s}$, $\sum d_{o*}$, and $\sum d_{*o}$, which have analogous definitions. Finally, we use $\sum d_{**}$ as a shorthand for $\sum d_{b*} + \sum d_{s*} + \sum d_{o*}$.

The LP is shown in Figure 2; the constraints that variables d_{ij} are non-negative, and should sum to 1 are not shown. The LP takes three parameters: z,

$\bar{q}$, and q. The inequalities of this LP are taken directly from Proposition 8, and each inequality appears twice: once for row $\bar{\imath}$, and once for row i. The objective function is intended to capture $R_i \cdot \mathbf{y}^{imp}$, and it the auxiliary function:

$$\phi(z, q) = \left(1 + \frac{\frac{1}{3} + z + qz + \frac{2qz}{\frac{1}{3}-2z}}{\frac{1}{3} - 2z - qz - (\frac{1}{3} + z)\frac{2qz}{\frac{1}{3}-2z}}\right).$$

If $\mathrm{s}(z, \bar{q}, q)$ is the solution of this LP, then we have the following proposition.

Proposition 9. *For every q-bad row i we have $R_i \cdot \mathbf{y}^{imp} \leq \mathrm{s}(z, \bar{q}, q)$.*

Maximize:
$$\phi(z, \bar{q})\left(d_{sb} + (\frac{1}{3} + z) \cdot d_{ss} + (\frac{2}{3} + 2z) \cdot d_{so}\right) + d_{ob} + (\frac{1}{3} + z) \cdot d_{os} + (\frac{2}{3} + 2z) \cdot d_{oo}$$

Subject to:
$$\sum d_{b*} \geq \frac{\frac{1}{3} + z - \bar{q}z - (\frac{1}{3} + z)(\sum d_{o*})}{\frac{2}{3} - z} \tag{4}$$
$$\sum d_{*b} \geq \frac{\frac{1}{3} + z - qz - (\frac{1}{3} + z)(\sum d_{*o})}{\frac{2}{3} - z} \tag{5}$$
$$\sum d_{s*} \geq \frac{\frac{1}{3} - 2z - \bar{q}z - (\frac{1}{3} + z)(\sum d_{o*})}{\frac{2}{3} - z} \tag{6}$$
$$\sum d_{*s} \geq \frac{\frac{1}{3} - 2z - qz - (\frac{1}{3} + z)(\sum d_{*o})}{\frac{2}{3} - z} \tag{7}$$
$$\sum d_{o*} \leq \frac{2\bar{q}z}{\frac{1}{3} - 2z} \tag{8}$$
$$\sum d_{*o} \leq \frac{2qz}{\frac{1}{3} - 2z} \tag{9}$$

Fig. 2. A linear program that gives an upper bound on $R_i \cdot \mathbf{y}^{imp}$

5.5 Applying the Matching Pennies Argument

Recall that ϵ_m is computed in stage 2 of our algorithm, and is the quality of the best WSNE with 2×2 support. So far, we have not used the assumption that $\epsilon_m > \frac{2}{3} - z$. In this section we will see how this assumption can be used to strengthen our LP. We define a matching pennies sub-game as follows.

Definition 10 (Matching Pennies). *Let i and i' be two rows, and let j and j' be two columns. If $j \in B_i \cap S_{i'}$ and $j' \in B_{i'} \cap S_i$, then we say that i, i', j, and j' form a matching pennies sub-game.*

An example of a matching pennies sub-game is given by l, r, T, and M in Figure 1b, because we have $l \in B_M \cap S_T$, and we have $r \in B_T \cap S_M$. In this example, we can obtain an exact Nash equilibrium by making the row player mix uniformly between T and M, and making the column player mix uniformly between l and r. However, in general we can only expect to obtain an $(\frac{2}{3} - z)$-WSNE using this technique.

Proposition 11. *If there is a matching pennies sub-game, then we can construct a $(\frac{2}{3} - z)$-WSNE with a 2×2 support.*

Thus, we can assume that our game does not contain a matching pennies sub-game, because otherwise Procedure (2) would have found a $(\frac{2}{3} - z)$-WSNE. Note that, by definition, if the game does not contain a matching pennies sub-game, then for all rows i we must have either $B_{\bar{\imath}} \cap S_i = \emptyset$, or $B_i \cap S_{\bar{\imath}} = \emptyset$.

We can use this observation to strengthen our LP. We define two LPs, each of which is constructed by adding an extra constraint to our existing LP. In the first LP we add the constraint $d_{bs} = 0$, and in the second LP we add the constraint $d_{sb} = 0$. We refer to the solutions of these two LPs as $\mathrm{s}_1(z, \bar{q}, q)$ and $\mathrm{s}_2(z, \bar{q}, q)$ respectively. We then obtain the following strengthening of Proposition 9.

Proposition 12. *For each q-bad row i we either have $R_i \cdot \mathbf{y}^{imp} \leq \mathrm{s}_1(z, \bar{q}, q)$, or we have $R_i \cdot \mathbf{y}^{imp} \leq \mathrm{s}_2(z, \bar{q}, q)$.*

5.6 A Linear Upper Bound for Our LPs

Now we can finally obtain our bound for $R_i \cdot \mathbf{y}^{imp}$, by proving an upper bound for $\mathrm{s}_k(z, \bar{q}, q)$. It is not difficult to show that s_k is monotonically increasing in $\bar{q}$. Since $\bar{q} < 3$, we can therefore argue that $\mathrm{s}_k(z, \bar{q}, q) \leq \mathrm{s}_k(z, 3, q)$. Then, using standard techniques from sensitivity analysis in linear programming, it is possible to bound $\mathrm{s}_k(z, 3, q)$ by a linear function.

Proposition 13. *We can compute $c_{z,k}$ and $d_{z,k}$ so that $\mathrm{s}_k(z, 3, q) \leq c_{z,k} + d_{z,k} \cdot q$.*

To obtain our final upper bound on $R_i \cdot \mathbf{y}^{imp}$, we simply take the maximum over the two LPs. That is, we set $c_z = \max(c_{z,1}, c_{z,2})$ and $d_z = \max(d_{z,1}, d_{z,2})$. This then leads to our final upper bound for $R_i \cdot \mathbf{y}^{imp}$.

Proposition 14. *We have $R_i \cdot \mathbf{y}^{imp} \leq c_z + d_z \cdot q$, for every q-bad row i.*

5.7 The Test for $P(z)$

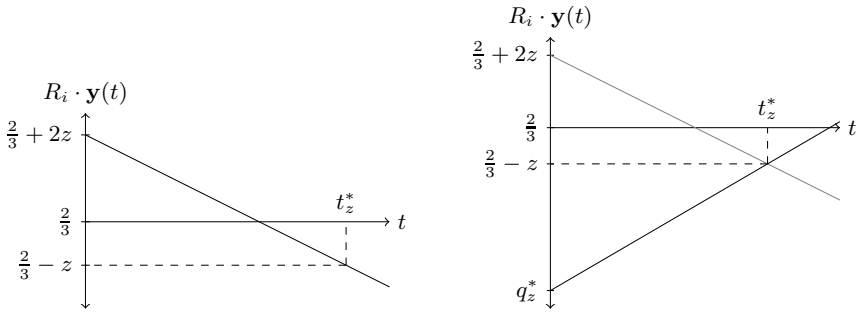
Finally, we can describe the test that determines whether $P(z)$ holds in all bimatrix games. The test constructs a point t_z^*, and then checks whether $R_i \cdot \mathbf{y}(t_z^*) \leq \frac{2}{3} - z$ holds for all rows i.

We begin by defining t_z^*, which is the smallest value of t for which, if i is a 0-bad row, then $R_i \cdot \mathbf{y}(t) \leq \frac{2}{3} - z$. By definition we have that $R_i \cdot \mathbf{y} = \frac{2}{3} + 2z$, and we also know that $R_i \cdot \mathbf{y}^{imp} \leq c_z + d_z \cdot 0$. Therefore t_z^* is the solution of:

$$(\frac{2}{3} + 2z) \cdot (1 - t_z^*) + c_z \cdot t_z^* = \frac{2}{3} - z.$$

This can be seen graphically in Figure 3a. The line in the figure starts at $\frac{2}{3} + z$ when $t = 0$, and ends at c_z when $t = 1$. The point t_z^* is the value of t at which this line crosses $\frac{2}{3} - z$. We can solve the equation to obtain the following formula:

$$t_z^* = \frac{3z}{\frac{2}{3} + 2z - c_z}. \tag{10}$$



(a) Finding t_z^*. (b) Finding q_z^*.

Fig. 3. Diagrams that show how t_z^* and q_z^* are found

Next, we define a constant q_z^*. For each row i, there is a trivial bound of:

$$R_i \cdot \mathbf{y}^{imp} \leq 1. \tag{11}$$

Note that if q is large, then this bound will be better than our bound of $c_z + d_z \cdot q$. The next step of our procedure is to find q_z^*, which is the smallest value of q such that, using this trivial bound (11), we can conclude that $R_i \cdot \mathbf{y}(t_z^*) \leq \frac{2}{3} - z$. Formally, we define q_z^* to be the solution of:

$$(\frac{2}{3} + 2z - q_z^* z) \cdot (1 - t_z^*) + t_z^* = \frac{2}{3} - z.$$

This can be seen diagrammatically in Figure 3b: we fix a line that passes through 1 when $t = 1$, and $\frac{2}{3} - z$ when $t = t_z^*$. Then, q_z^* is defined to be the point at which this line meets the y-axis of the graph, where $t = 0$. Solving the equation gives the following formula for q_z^*.

$$q_z^* = \frac{(2z - \frac{1}{3}) \cdot t_z^* - 3z}{z t_z^* - z} \tag{12}$$

For rows i that are q-bad with $q \geq q_z^*$, we can apply the trivial bound (11) to argue that $R_i \cdot \mathbf{y}(t_z^*) \leq \frac{2}{3} - z$. Therefore, we need only be concerned with rows i that are q-bad with $0 \leq q < q_z^*$. The next proposition gives a simple test that can be used to check whether all such rows will have the property $R_i \cdot \mathbf{y}(t_z^*) \leq \frac{2}{3} - z$.

Proposition 15. *If* $c_z + d_z \cdot q_z^* \leq 1$*, then* $R_i \cdot \mathbf{y}(t_z^*) \leq \frac{2}{3} - z$ *for all rows* i*.*

Thus, our test for checking whether $P(z)$ holds in all bimatrix games can be summarised as follows. First we compute the constants c_z and d_z. Then we use these to compute t_z^* and q_z^*. Finally, we check whether $c_z + d_z \cdot q_z^* \leq 1$. If the inequality holds, then Proposition 15 implies that $P(z)$ is true. To complete the proof of Theorem 5, it suffices to note that our test proves that $P(z)$ holds in all bimatrix games for $z = 0.004735$.

6 Conclusions

In Section 3, we presented a polynomial-time algorithm for computing a $(\frac{2}{3} - z)$-WSNE, where $z = 0.004735$. We do not believe that our analysis is tight, as it uses several restrictions that our algorithm does not face. For example, $\mathbf{y}(t)$ uses the same support as the strategy returned by the KS algorithm, whereas the LP given in Definition 1 can return a subset of this support. Another example is that in the analysis we only consider 2×2 subgames in which players mix uniformly, whereas Procedure 2 considers all mixtures.

An interesting open question is the following. Does every bimatrix game possess a $\frac{1}{2}$-WSNE, where both players use at most two strategies? This is known to be true with high probability in random games [1], but not known in general.

References

1. Bárány, I., Vempala, S., Vetta, A.: Nash equilibria in random games. Random Struct. Algorithms 31(4), 391–405 (2007)
2. Bosse, H., Byrka, J., Markakis, E.: New algorithms for approximate Nash equilibria in bimatrix games. Theoretical Computer Science 411(1), 164–173 (2010)
3. Bradley, S.P., Hax, A.C., Magnanti, T.L.: Applied Mathematical Programming. Addison-Wesley (1977), `http://web.mit.edu/15.053/www/`
4. Chen, X., Deng, X., Teng, S.-H.: Settling the complexity of computing two-player Nash equilibria. Journal of the ACM 56(3),14:1–14:57 (2009)
5. Daskalakis, C., Goldberg, P.W., Papadimitriou, C.H.: The complexity of computing a Nash equilibrium. SIAM Journal on Computing 39(1), 195–259 (2009)
6. Daskalakis, C., Mehta, A., Papadimitriou, C.H.: Progress in approximate Nash equilibria. In: Proceedings of ACM-EC, pp. 355–358 (2007)
7. Daskalakis, C., Mehta, A., Papadimitriou, C.H.: A note on approximate Nash equilibria. Theoretical Computer Science 410(17), 1581–1588 (2009)
8. Jansen, B., de Jong, J.J., Roos, C., Terlaky, T.: Sensitivity analysis in linear programming: just be careful! European Journal of Operational Research 101(1), 15–28 (1997)
9. Kontogiannis, S.C., Spirakis, P.G.: Efficient Algorithms for Constant Well Supported Approximate Equilibria in Bimatrix Games. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 595–606. Springer, Heidelberg (2007)
10. Kontogiannis, S.C., Spirakis, P.G.: Well supported approximate equilibria in bimatrix games. Algorithmica 57(4), 653–667 (2010)
11. Nash, J.: Non-cooperative games. The Annals of Mathematics 54(2), 286–295 (1951)
12. Tsaknakis, H., Spirakis, P.G.: An optimization approach for approximate Nash equilibria. Internet Mathematics 5(4), 365–382 (2008)