Multilinear formulations for computing Nash equilibrium of multi-player matrix games

Miriam Fischer[1] and Akshay Gupte[2]

[1] Department of Computing, Imperial College London, UK
[2] School of Mathematics, University of Edinburgh, UK

Abstract. We present multilinear and mixed-integer multilinear programs to find a Nash equilibrium in multi-player strategic-form games. We compare the formulations to common algorithms in Gambit, and conclude that a multilinear feasibility program finds a Nash equilibrium faster than any of the methods we compare it to, including the quantal response equilibrium method, which is recommended for large games. Hence, the multilinear feasibility program is an alternative method to find a Nash equilibrium in multi-player games, and outperforms many common algorithms. The mixed-integer formulations are generalisations of known mixed-integer programs for two-player games, however unlike two-player games, these mixed-integer programs do not give better performance than existing algorithms.

Keywords: Noncooperative n-person games · Nash equilibrium · Multilinear functions · Nonconvex problems · Mixed-integer optimization

1 Introduction

Finding a Nash equilibrium in two-player and multi-player games is PPAD-complete [1]. Conditions for existence of equilibria have been analysed [17]. This paper deals with the question of algorithmic and numerical computation of equilibria. For two-player games, the Lemke-Howson algorithm [8] is an established exact method to find a Nash equilibrium that gives a very good computational performance on many instances in practice, although its worst-case performance can take exponentially many pivoting steps [16]. A classical result of Mangasarian and Stone [9] is that equilibria of two-person games are optimal solutions to a bilinear optimization problem. Sandholm et al. [15] formulated a mixed-integer program for two-player games which for some instances was faster than the Lemke-Howson method. However, for multi-player games, there do not seem to be known optimization approaches for computing the equilibrium. Although there is a generalisation of the Lemke-Howson method to n-person games [13], popular algorithmic approaches include a global Newton method [5], an iterated polymatrix approximation approach [6], a simplicial subdivision method [7], a simple search algorithm aiming to find an equilibrium with small support size [12], and a quantal response equilibrium method which gives an approximation to a Nash equilibrium [18]. Many of these methods are implemented in the game-theoretic library `Gambit` [10]. Experiments comparing different methods have been undertaken [15, 12], however it is rather unclear which of the methods is best for multi-player strategic-form games. For example, the global Newton method gives solid performance for small games, however does not to scale well to larger games [4, 19]. Support enumeration algorithms are fast for games with pure equilibria but will be much slower for a game that only has equilibria of medium to large support size. There are also approximation algorithms, which tend to approximate a Nash equilibrium for large games [18, 2, 4].

In this work, we follow the optimization approach, and propose different optimization formulations for computing a Nash equilibrium for n-person games for $n > 2$. Particularly, we present a multilinear polynomial continuous feasibility program of degree m (= number of players), which is a generalisation of the bilinear problem for 2 players. Further, we extend the mixed-integer formulations of [15] to multi-player games, and give a large variety of mixed-integer formulations to find a Nash equilibrium in multi-player strategic form games. All our formulations find a Nash equilibrium of a $m \geq 2$ player strategic-form game. We compare our programs to `gambit-gnm` (global Newton), `gambit-simpdiv` (simplicial subdivision), `gambit-logit` (quantal response equilibrium) algorithms in `Gambit`, focussing on random games and covariant games with negative covariance. We find that the mixed-integer formulations do not give better performance than existing algorithms, and our analysis of those is aimed to get an understanding of which mixed-integer formulations are most suited for finding a Nash equilibrium. We find that our multilinear continuous feasibility program is faster than all the methods in Gambit we compare it to, including the `gambit-logit` method, which is so far recommended for large games. Thus, we argue that we provide an alternative approach to computing Nash equilibrium in multi-player strategic-form games.

2 Formulations for n-player games

The multi-player multilinear formulation is an extension of a bilinear formulation for bimatrix games. To motivate the multilinear formulation, we shortly recall the bilinear programme that is equivalent to finding a Nash equilibrium in a bimatrix game. To do so, we introduce some notation. Let $A, B \in \mathbb{R}^{m \times n}$ be the payoff matrix of player 1 and player 2, with m pure strategies of player 1 and n pure strategies of player 2. Let $\boldsymbol{x} \in \mathbb{R}^m$ with $\boldsymbol{x} \geq 0$ and $\sum_{i=1}^{m} x_i = 1$ be a (mixed) strategy of player 1, with x_s being the probability placed on pure strategy s. Let $\boldsymbol{y} \in \mathbb{R}^n$ with $\boldsymbol{y} \geq 0$ and $\sum_{j=1}^{n} y_j = 1$ be a (mixed) strategy of player 2. Let $\mathbf{1}_n$ and $\mathbf{1}_m$ denote vectors of all ones of dimension n (m). Any globally *optimal* solution (x, y, p, q) to the bilinear optimization problem (1a) is equivalent to a Nash equilibrium in a bimatrix game.

$$\max_{x,y,p,q} \quad x^\top Ay + x^\top By - p - q \tag{1a}$$

$$\text{s.t.} \quad Ay \leq p\mathbf{1}_m, \quad B^\top x \leq q\mathbf{1}_n \tag{1b}$$

$$\sum_{i=1}^{m} x_i = 1, \quad \sum_{j=1}^{n} y_j = 1, \quad x, y \geq \mathbf{0}. \tag{1c}$$

It is easy to see that any feasible mixed strategies x, y will have objective function value less or equal to zero, as given player 2's (mixed) strategy, any pure strategy of player 1 can give payoff at most p, and given player 1's (mixed) strategy, any pure strategy of player 2 can give payoff at most q. This implies that any combination of pure strategies (i.e. any mixed strategy) of player 1 gives payoff at most p, and any combination of pure strategies of player 2 gives payoff at most q. Further, any Nash equilibrium x^*, y^* has objective function value equal to zero, thus maximises the objective function. This is because players play best responses, and thus $p^* = {x^*}^\top Ay^*$ and $q^* = {x^*}^\top By^*$. Importantly, *only* the Nash equilibria which have objective function value of zero. This is because for any optimal solution (x^*, y^*, p^*, q^*) and any (x, y) with $x \geq 0$, $\sum_{i=1}^m x_i = 1$, $y \geq 0$, $\sum_{i=1}^n y_i = 1$, $x^\top Ay^* \leq p^*$, ${x^*}^\top By \leq q^*$, and thus ${x^*}^\top Ay^* \leq p^*$, ${x^*}^\top By^* \leq q$. As a Nash equilibrium has objective function value of zero and is guaranteed to exist, the optimal value of the bilinear formulation must be zero (as it is non-positive). Thus ${x^*}^\top Ay^* = p^*$, ${x^*}^\top By^* = q^*$. This implies $x^\top Ay^* \leq {x^*}^\top Ay^*$, ${x^*}^\top By \leq {x^*}^\top By^*$

Although an extension to multi-player games is straightforward, to our knowledge an equivalent optimization programme to finding a Nash equilibria for a multi-player matrix game with more than two players has not yet been explicitly formulated. In this work, we extend the bilinear formulation to a multilinear formulation, equivalent to finding a Nash equilibrium in a multi-player matrix game with $m \geq 2$ players. We compare our formulation to established algorithms used to find an equilibrium in multi-player games. We find that our multilinear programme is faster than a variety of algorithms in Gambit [10].

2.1 Multilinear formulation

Let us define some notation. Let $m \in \mathbb{N} \geq 2$ be the number of players, and $[m] = \{1, \ldots, m\}$ the set of players. Every player i comes with a finite set of pure strategies S_i, with $|S_i| = m_i$. Let $\mathcal{S} = S_1 \times S_2 \times \cdots \times S_m$ be the set of all m-tuples of pure strategy combinations of all players. We will further denote $\mathcal{S}_{-i} = S_1 \times \cdots \times S_{i-1} \times S_{i+1} \times \cdots \times S_m$ the set of all pure strategy tuples of all players except i. Let $\boldsymbol{s} = (s_1, \ldots, s_m) \in \mathcal{S}$ be a pure strategy tuple of all players and $\hat{\mathbf{s}} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_m) \in \mathcal{S}_{-i}$ be a pure strategy tuple of all players other than i. We define payoff matrix $A_i : \mathbf{s} \in \mathcal{S} \to \mathbb{R}$ for player i. As an example, if we had three players, $A_1[s_1, s_2, s_3]$ denotes player 1's payoff when player 1 plays pure strategy s_1, player 2 plays pure strategy s_2 and player

3 plays pure strategy s_3. Likewise, $A_2[s_1, s_2, s_3]$ and $A_3[s_1, s_2, s_3]$ denote player 2 and 3's payoff for the strategy combination $(s_1, s_2, s_3) \in \mathcal{S}$. For pure strategy s of player i and pure strategies $(s_1, \ldots, s_{i-1}, \ldots, s_{i+1}, \ldots, s_m) = \hat{\mathbf{s}} \in \mathcal{S}_{-i}$ of the other players, we write $A_i[s, \hat{\mathbf{s}}]$ to denote the payoff of player i when player i plays pure strategy $s \in \mathcal{S}_i$ and the other players play pure strategies $\hat{\mathbf{s}} \in \mathcal{S}_{-i}$. For every player i, we define strategy vector $\mathbf{x}^i \in \mathbb{R}^{m_i}$, with $\mathbf{x}^i \geq \mathbf{0}$ and $\sum_{s \in S_i} x^i_s = 1$. $\mathbf{x}^i$ is a probability distribution over player i's pure strategies, and thus a *mixed strategy.* Let $\mathbf{x} = (\mathbf{x}^1, \ldots, \mathbf{x}^m)$ be a mixed strategy profile of all players, and $\mathbf{x}^{-i} = (\mathbf{x}^1, \ldots, \mathbf{x}^{i-1}, \mathbf{x}^{i+1}, \mathbf{x}^m)$ be a mixed strategy profile of all players other than i. $\prod_{s_j \in \hat{\mathbf{s}}} x^j_{s_j}$ for $\hat{\mathbf{s}} \in \mathcal{S}_{-i}$ denotes the combined probability of all players except i to play the pure strategy tuple $\hat{\mathbf{s}} \in \mathcal{S}_{-i}$. As an example, if we have three players, player 2 has pure strategies $s_{2,1}$ and $s_{2,2}$ and player 3 has pure strategies $s_{3,1}$ and $s_{3,2}$, then $\mathcal{S}_{-1} = \{(s_{2,1}, s_{3,1}), (s_{2,1}, s_{3,2}), (s_{2,2}, s_{3,1}), (s_{2,2}, s_{3,2})\}$. If player 2 plays $s_{2,1}$ with probability 1/2 and player 3 plays $s_{3,1}$ with probability 1/4, then $\prod_{s_j \in (s_{2,1}, s_{3,1})} = 1/2 * 1/4$, $\prod_{s_j \in (s_{2,1}, s_{3,2})} = 1/2 * 3/4$, $\prod_{s_j \in (s_{2,1}, s_{3,1})} = 1/2 * 1/4$, $\prod_{s_j \in (s_{2,2}, s_{3,2})} = 1/2 * 3/4$. Further, we define vector $\mathbf{p} \in \mathbb{R}^m$. p^i corresponds to player i's highest expected payoff.

Definition 1. *Let* $\Gamma = \langle \{1, \ldots, m\}, (S_i), (A_i) \rangle$ *be a strategic-form game, with* m, S_i, A_i, $\mathbf{x}^i$ *defined as above. Let* $\mathbf{x}^i \geq 0$ *with* $\sum_{s \in S_i} x^i_s = 1$ *be a mixed strategy of player* i. *Then,* $\mathbf{x}^* = (\mathbf{x}^{*1}, \ldots, \mathbf{x}^{*m})$ *with* $\mathbf{x}^{*i} \geq 0$ *and* $\sum_{s \in S_i} {x^*}^i_s = 1$ *for all players* i *is a (mixed) Nash equilibrium if for all players* i *and every mixed strategy* $\mathbf{x}^i$, *we have* $E\left[A_i[\mathbf{x}^*]\right] \geq E\left[A_i[\mathbf{x}^i, \mathbf{x}^{*-i}]\right]$.

We now present our multilinear optimization formulation.

$$
\begin{aligned}
\text{maximize:} \quad & \sum_{i=1}^{m} \left(\sum_{\substack{(s,\hat{\mathbf{s}}) \\ \in S_i \times S_{-i}}} A_i[s, \hat{\mathbf{s}}] x^i_s \prod_{s_j \in \hat{\mathbf{s}}} x^j_{s_j} \right) - \sum_{i=1}^{m} p^i && \text{(2a)} \\
\text{subject to:} \quad & \sum_{\hat{\mathbf{s}} \in S_{-i}} A_i[s, \hat{\mathbf{s}}] \prod_{s_j \in \hat{\mathbf{s}}} x^j_{s_j} \leq p^i \qquad \forall i \in [m], s \in S_i && \text{(2b)} \\
& \sum_{s \in S_i} x^i_s = 1 \qquad \forall i \in [m] && \text{(2c)} \\
& 0 \leq x^i_s \leq 1 \qquad \forall i \in [m], s \in S_i && \text{(2d)}
\end{aligned}
$$

Fig. 1. MLP1: Multilinear programme for multi-player games

Theorem 1. *A (mixed) strategy* $(\mathbf{x}^1, \ldots, \mathbf{x}^m)$ *is a (mixed) Nash equilibrium of the* m*-player game* $(A_1, \ldots, A_m)$ *if and only if there exists numbers* $p^1, \ldots, p^m$ *such that* $(\mathbf{x}^1, \ldots, \mathbf{x}^m, p^1, \ldots, p^m)$ *is an optimal solution to the above problem.*

Proof. We first show that if $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m)$ is a Nash equilibrium to $(A_1,\dots,A_m)$, then there exist numbers $\bar{p}^1,\dots,\bar{p}^m$ such that $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$ is an optimal solution to the programme in Figure 1. Assume that $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m)$ is a Nash equilibrium. For any feasible solution $(\mathbf{x}^1,\dots,\mathbf{x}^m,p^1,\dots,p^m)$ of Figure 1, constraints (2b), (2c) imply

$$\sum_{s\in S_i}\left(x_s^i\sum_{\hat{\mathbf{s}}\in S_{-i}}A_i[s,\hat{\mathbf{s}}]\prod_{s_j\in\hat{\mathbf{s}}}x_{s_j}^j\right)\le p^i$$

for all $i\in[m]$. This implies (2a) ≤ 0 for any feasible solution. Set

$$\bar{p}^i=\sum_{s\in S_i}\left(\bar{x}_s^i\sum_{\hat{\mathbf{s}}\in S_{-i}}A_i[s,\hat{\mathbf{s}}]\prod_{s_j\in\hat{\mathbf{s}}}\bar{x}_{s_j}^j\right)$$

for every $i\in[m]$. We show that $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$ is feasible and optimal to Figure 1.

As $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m)$ is a Nash equilibrium, we have

$$\sum_{s\in S_i}\left(\bar{x}_s^i\sum_{\hat{\mathbf{s}}\in S_{-i}}A_i[s,\hat{\mathbf{s}}]\prod_{s_j\in\hat{\mathbf{s}}}\bar{x}_{s_j}^j\right)\ge\sum_{s\in S_i}\left(x_s^i\sum_{\hat{\mathbf{s}}\in S_{-i}}A_i[s,\hat{\mathbf{s}}]\prod_{s_j\in\hat{\mathbf{s}}}\bar{x}_{s_j}^j\right)$$

for all (mixed) strategies $\mathbf{x}^i\ge 0$ with $\sum_{s\in S_i}x_s^i=1$. Choosing $\mathbf{x}^i=\mathbf{e}_k$, with $k\in\{1,\dots,|S_i|\}$, hence the unit vector with all zeros except one in the k-th component, we have

$$\bar{p}^i=\sum_{s\in S_i}\left(\bar{x}_s^i\sum_{\hat{\mathbf{s}}\in S_{-i}}A_i[s,\hat{\mathbf{s}}]\prod_{s_j\in\hat{\mathbf{s}}}\bar{x}_{s_j}^j\right)\ge\sum_{\hat{\mathbf{s}}\in\mathcal{S}_{-i}}A_i[k,\hat{\mathbf{s}}]\prod_{s_j\in\hat{\mathbf{s}}}\bar{x}_{s_j}^j\quad\forall k\in\{1,\dots,|S_i|\},$$

satisfying constraint 2b. As we can apply this for all players $i\in[m]$ (and constraints 2c, 2d hold as $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m)$ is a Nash equilibrium), it follows that $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$ is feasible. Further, the objective function value is zero at the point $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$. As the objective function value is at most zero and $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$ is feasible, it follows that it is optimal.

To show that if $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$ is an optimal solution to Figure 1, $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m)$ is a Nash equilibrium, we assume that $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$ indeed is optimal to Figure 1. Since a Nash equilibrium exists in this game and has objective value of zero, and all feasible solutions have non-positive value, it follows that the objective value of $(\bar{\mathbf{x}}^1,\dots,\bar{\mathbf{x}}^m,\bar{p}^1,\dots,\bar{p}^m)$ must be zero. For any $\mathbf{x}^i\ge 0$ with $\sum_{s\in S_i}x_s^i=1$, for all players $i\in[m]$, constraints 2b, 2c imply

$$\sum_{s\in S_i}\left(x_s^i\sum_{\hat{\mathbf{s}}\in S_{-i}}A_i[s,\hat{\mathbf{s}}]\prod_{s_j\in\hat{\mathbf{s}}}\bar{x}_{s_j}^j\right)\le\bar{p}^i\quad\forall i\in[m].$$

Particularly,

$$\sum_{s \in S_i} \left(\bar{x}^i_s \sum_{\hat{\mathbf{s}} \in S_{-i}} A_i[s, \hat{\mathbf{s}}] \prod_{s_j \in \hat{\mathbf{s}}} \bar{x}^j_{s_j} \right) \le \bar{p}^i \quad \forall i \in [m].$$

As further objective value of $(\bar{\mathbf{x}}^1, \ldots, \bar{\mathbf{x}}^m, \bar{p}^1, \ldots, \bar{p}^m)$ is zero, we have

$$\sum_{s \in S_i} \left(\bar{x}^i_s \sum_{\hat{\mathbf{s}} \in S_{-i}} A_i[s, \hat{\mathbf{s}}] \prod_{s_j \in \hat{\mathbf{s}}} \bar{x}^j_{s_j} \right) = \bar{p}^i \quad \forall i \in [m].$$

Hence,

$$\sum_{s \in S_i} \left(x^i_s \sum_{\hat{\mathbf{s}} \in S_{-i}} A_i[s, \hat{\mathbf{s}}] \prod_{s_j \in \hat{\mathbf{s}}} \bar{x}^j_{s_j} \right) \le \sum_{s \in S_i} \left(\bar{x}^i_s \sum_{\hat{\mathbf{s}} \in S_{-i}} A_i[s, \hat{\mathbf{s}}] \prod_{s_j \in \hat{\mathbf{s}}} \bar{x}^j_{s_j} \right) = \bar{p}^i$$

$\forall i \in [m], \forall \mathbf{x}^i \ge 0 : \sum_{s \in S_i} x^i_s = 1$. Therefore, $(\bar{\mathbf{x}}^1, \ldots, \bar{\mathbf{x}}^m)$ is a Nash equilibrium, as the constraint states that given the other players mixed strategies $\bar{\mathbf{x}}^j$, no strategy of player i can give higher payoff than strategy $\bar{\mathbf{x}}^i$, for all players. □

It is easy to see that for two players, MLP1 equals the bilinear formulation in (1a), with $\mathbf{x}^1, \mathbf{x}^2$ instead of $\mathbf{x}, \mathbf{y}$, p^1, p^2 instead of p, q, and A_1, A_2 instead of A, B. Computational experiments on small instances reveal that the solver takes significant time to solve Figure 1 to optimality. However, further inspections reveal that it is more the verification of an optimal solution, rather than finding an optimal solution, that is the reason for this. Particularly, the solver finds a solution with objective function value 0 (i.e. a Nash equilibrium relatively quickly, but spends a lot of time verifying that there is no feasible solution with objective function value larger than zero. However, as there cannot be a feasible solution with strictly positive objective value, it is sufficient for the solver to find a feasible solution with objective function value zero, instead of verifying that the upper bound to the optimisation program is zero. Thus, we reformulate the program into a feasibility program, for which the aim is to find a feasible solution for which the objective function (2a) of program 1 is non-negative. As a strictly positive solution is not possible, any feasible solution to 2 will have a value of zero, and thus be a Nash equilibrium.

2.2 Mixed-integer formulations

Sandholm et al. [15] presented four mixed-integer formulations, which solutions are equivalent to a Nash equilibrium in a two-player, strategic-form game. We generalise the formulations to multi-player strategic-form games.

The notation we use is similar to the notation introduced in the multilinear formulations. Further, we introduce $U_i = \max_{s^l, s^h \in S_i, \hat{\mathbf{s}}^{\mathbf{l}}, \hat{\mathbf{s}}^{\mathbf{h}} \in S_{-i}} A_i[s^h, \hat{\mathbf{s}}^{\mathbf{h}}] - A_i[s^l, \hat{\mathbf{s}}^{\mathbf{l}}]$ be the maximum difference of any two payoffs of player i for any pure strategies of all players.

$$\text{maximize:}\ 0 \tag{3a}$$

$$\text{subject to:} \sum_{\hat{\mathbf{s}} \in S_{-i}} A_i[s, \hat{\mathbf{s}}] \prod_{s_j \in \hat{\mathbf{s}}} x^j_{s_j} \le p^i \quad \forall i \in [m], s \in S_i \tag{3b}$$

$$\sum_{i=1}^{m} \left(\sum_{\substack{(s,\hat{\mathbf{s}}) \\ \in S_i \times S_{-i}}} A_i[s, \hat{\mathbf{s}}] x^i_s \prod_{s_j \in \hat{\mathbf{s}}} x^j_{s_j} \right) - \sum_{i=1}^{m} p^i \ge 0 \tag{3c}$$

$$\sum_{s \in S_i} x^i_s = 1 \qquad \forall i \in [m] \tag{3d}$$

$$0 \le x^i_s \le 1 \quad \forall i \in [m], s \in S_i \tag{3e}$$

Fig. 2. MLP2: Multilinear feasibility programme for multi-player games

We have four mixed-integer multilinear formulations, of which one is a feasibility program and three are optimisation programmes. All programmes have five sets of variables. $x^i_s, r^i_s, u^i_s, \overline{u}_i$ are real variables, and b^i_s is binary. The MIMLPs have the same interpretation and range of values for variables $x^i_s \ge 0, \overline{u}_i, u^i_s, r^i_s \ge 0$, further they also come with constraints (4b), (4c), (4d), (4e) (which are mostly such that variables $x^i_s, \overline{u}_i, u^i_s, r^i_s$ are defined as desired). x^i_s, for all players $i \in [m]$ and all pure strategies $s \in S_i$ of player i, denotes the probability with which player i plays pure strategy s. Hence, variables x^i_s give us the mixed strategy played by each player. In order to be valid (mixed) strategies, all pure strategies of a player must be played with non-negative probability (Eq. 4h) and sum up to one (Eq. 4b), for all players. $\overline{u}_i$ denotes the highest utility player i can achieve by playing any strategy, given the other players mixed strategies. u^i_s is the expected utility of player i of playing pure strategy s, given the other players play their (mixed) strategies (Eq. 4c). Naturally, $\overline{u}_i \ge u^i_s$ (Eq. 4d). $r^i_s = \overline{u}_i - u^i_s$ (Eq. 4e) is the regret of player i of playing pure strategy s. It is defined as the difference of the highest utility of any strategy for the player to the utility of playing strategy s, given the other players' mixed strategies. By definition, the regret of any pure strategy must be non-negative (4h). Further, in any Nash equilibrium, every pure strategy that is played with strictly positive probability must have zero regret. If there was a pure strategy which the player plays and that has positive regret, the player can increase their payoff by putting more probability on a pure strategy with no regret and putting less probability on the pure strategy with regret. Hence, it would not be a Nash equilibrium.

The meaning of binary variables b^i_s is different in all formulations, with not all constraints of MIMLP 1 regarding this variable (Eq. 4f, 4g) present in MIMLP 2,3,4. In formulation 1, if b^i_s is 1, strategy s of player i is not played, hence $x^i_s = 0$. If $b^i_s = 0$, the probability on strategy s is allowed to be positive, however the regret of the strategy must be zero. (4f) ensures that b^i_s can only be set to 1 if zero probability is on s. Further, (4g) ensures that b^i_s can only be set to zero if the strategy's regret is zero (if $b^i_s = 1$, this constraint does not restrict any variable, as $r^i_s \leq U_i$ by definition).

Proposition 1. *The set of feasible solutions to MIP 1 (Figure 3) is precisely the set of Nash equilibria for the multi-player strategic-form game $A_1, \ldots, A_m$.*

Proof. For any player $i \in [m]$ and any pure strategy $s \in S_i$ of player i, x^i_s denotes the probability with which player i plays pure strategy s. Constraint 4b,4h guarantee $\boldsymbol{x}^i$ to be a valid mixed strategy for each player i, as all pure strategies are played with non-negative probability and sum up to one. Constraint 4c defines the expected payoff u^i_s of player i of playing pure strategy s (given the other players' mixed strategies), and 4d defines the highest possible expected payoff $\overline{u}_i$ of any (mixed) strategy of player i given the other players' (mixed) strategies. Constraint 4e,4h define the regret r^i_s of player i of playing pure strategy $s \in S_i$. The regret of a pure strategy is the difference of player i's highest possible expected payoff $\overline{u}_i$ and i's payoff of playing pure strategy s and is non-negative. Constraint 4i introduces binary variable b^i_s for any pure strategy s of any player i. Constraint 4f requires that b^i_s can only be set to one if player i puts zero probability on pure strategy s. Further, constraint 4g ensures that b^i_s can only be set to zero if the strategy's regret is zero (if $b^i_s = 1$, this constraint does not restrict any variable, as $r^i_s \leq U_i$ by definition). Thus, if b^i_s is 1, strategy s of player i is not played, hence $x^i_s = 0$. If $b^i_s = 0$, the probability on strategy s is allowed to be positive, however the regret of the strategy must be zero. Hence, only pure strategies with zero regret can be played with positive probability, which is precisely the definition of a Nash equilibrium. □

$$
\begin{aligned}
\text{minimize:} \quad & 0 && && \text{(4a)}\\
\text{subject to:} \quad & \sum_{s \in S_i} x^i_s = 1 && \forall i \in [m] && \text{(4b)}\\
& u^i_s = \sum_{\hat{\mathbf{s}} \in S_{-i}} \prod_{s_j \in \hat{\mathbf{s}}} x^j_{s_j} A_i[s, \hat{\mathbf{s}}] && \forall i \in [m], \forall s \in S_i && \text{(4c)}\\
& \overline{u}_i \geq u^i_s && \forall i \in [m], \forall s \in S_i && \text{(4d)}\\
& r^i_s = \overline{u}_i - u^i_s && \forall i \in [m], \forall s \in S_i && \text{(4e)}\\
& x^i_s \leq 1 - b^i_s && \forall i \in [m], \forall s \in S_i && \text{(4f)}\\
& r^i_s \leq U_i b^i_s && \forall i \in [m], \forall s \in S_i && \text{(4g)}\\
& x^i_s, r^i_s \geq 0,\ u^i_s, \bar{u}_i \in \mathbb{R} && \forall i \in [m], \forall s \in S_i && \text{(4h)}\\
& b^i_s \in \{0, 1\} && \forall i \in [m], \forall s \in S_i && \text{(4i)}
\end{aligned}
$$

Fig. 3. MIMLP1 formulation for multiplayer games

MIMLP1 is a feasibility program, for which only Nash equilibria are feasible solutions. MIMLP2, MIMLP3, MIMLP4 (Figures 4, 5, 6) have larger feasible regions, and pure strategies with positive probability can have positive regret, and pure strategies with positive regret can be played with positive probability. The formulations minimise a penalty, and it is only Nash equilibria for which the penalty is minimal. Thus, only Nash equilibria are optimal solutions. MIMLP2 penalises the regret of a pure strategy that is played with positive probability in the objective function, and thus for optimal solutions, the regret of pure strategies with positive probability should be zero. MIMLP3 penalises the probability placed on pure strategies with positive regret, and thus optimal solutions will have zero probability on pure strategies with positive regret. MIMLP4 combines the normalised regret and the probability as a penalty, and the solver can choose whether the regret or the probability should be minimised. As [15] noted, these formulations can be used to find approximate Nash equilibria.

The advantage of these formulations is that, since finding a Nash equilibrium is assumed to be computationally intractable, these formulations can be used to stop the program before an equilibrium has been calculated, and thus give solutions which are close to an equilibrium, also called approximate equilibria. However, it is more difficult with these formulations to find a specific equilibrium among all equilibria, rather than just an arbitrary equilibrium.

Figure 4 presents MIMLP2 and aims to minimise the regret of pure strategies that are played with positive probabilities. Particularly, the regret of a pure strategy played with positive probability serves as a penalty to the objective function. This is done by introducing variable f^i_s for all $i \in [m], s \in S_i$, which represents a pure strategy's regret if the strategy has positive probability and zero otherwise.

Proposition 2. *The set of Nash equilibria minimises the objective function of MIMLP 2 (Figure 4).*

Proof. Constraints 4b,4c,4d,4e,4h,4i guarantee that $\boldsymbol{x}^i, \boldsymbol{r}^i, \boldsymbol{u}^i, \overline{u}_i$ are correctly defined for all players. Due to constraint (4f), b^i_s can only be set to one if the probability on the pure strategy is zero. Then, due to minimising f^i_s in (5) and Equation (7), f^i_s must be set to U_i. In that case, f^i_s and $U_i b^i_s$ cancel out in the objective, and hence strategies with zero probability have no penalty. If $b^i_s = 0$, f^i_s equals r^i_s, due to minimising f^i_s and Equation (6), and as $U_i b^i_s = 0$, the penalty of the pure strategy equals the regret of the strategy, and pure strategies that have no regret do not have a penalty. Thus, due to the objective function, it is encouraged to play pure strategies which have no regret, and to not play strategies with regret. Thus, any pure strategy will only contribute to the objective function if it has positive regret *and* probability. The Nash equilibria minimise the objective function, with optimal objective of zero. As any pure strategy in a Nash equilibrium will either have zero probability (hence no penalty) or zero regret (hence no penalty), the objective function will equal zero. Solutions which do not equal a Nash equilibrium have higher objective value, as for some strategies, $f^i_s > 0$ (as r^i_s and U_i are non-negative). □

$$\text{minimize:} \quad \sum_{i=1}^{m} \sum_{s \in S_i} f_s^i - U_i b_s^i \tag{5}$$

$$\text{subject to:} \quad (4b), (4c), (4d), (4e), (4f), (4h), (4i)$$

$$f_s^i \geq r_s \qquad \forall i \in [m], \forall s \in S_i \tag{6}$$

$$f_s^i \geq U_i b_s \qquad \forall i \in [m], \forall s \in S_i \tag{7}$$

Fig. 4. MIMLP2 formulation, objective function penalises regret on pure strategies played with positive probability

Figure 5 describes MIMLP 3. It is similar to MIMLP 2, however instead of minimising the regret of pure strategies played with positive probability, the probabilities of pure strategies with positive regret is minimised. To do so, variables g_s^i are introduced, which are set such that a strategy's penalty in the objective function is zero if the strategy's regret is zero, and x_s^i (the probability with which it is played) otherwise. The set of Nash equilibria minimises the objective, as strategies with positive regret are not played.

Proposition 3. *The set of Nash equilibria minimises the objective function of MIMLP 3 (Figure 5).*

Proof. We recall that because of constraint (4g), b_s^i can only be set to zero if the strategy's regret r_s^i is zero. By constraint (8c) and minimising g_s^i, if $b_s^i = 0$, then $g_s^i = 1$. Thus, g_s^i and $1 - b_s^i$ cancel out in the objective function and the penalty of strategy s is zero. If $b_s^i = 1$, due to constraint (8b) and minimising g_s^i, $g_s^i = x_s^i$, and $1 - b_s^i = 0$. Hence, the penalty of strategy s equals x_s^i. Therefore, the probability a pure strategy is played with only contributes to the objective function if the strategy has positive regret. Nash equilibria minimise the objective function, and come with optimal value of zero. Constraint (4f) of MIMLP 1 (namely, $x_s^i \leq 1 - b_s^i$) is no longer in this formulation, and it is possible to set $b_s^i = 1$ even if some probability is placed on s. However, in a Nash equilibrium, b_s^i will only be set to 1 if the probability on s is indeed zero, as pure strategies with positive regret are not played. □

MIMLP 4, seen in Figure 6, combines MIMLP 2 and MIMLP 3. Instead of penalising all pure strategies' regret (MIMLP 2) or penalising all pure strategies' probabilities if they have positive regret (MIMLP 3), this formulation lets the solver decide for each pure strategy whether to penalise the regret or the probability. The penalised regret is expressed with variables f_s^i, the penalised probabilities are expressed with variables g_s^i. when using both the regret and the probabilities, the regret must be normalised, as the probability of a pure strategy is between zero and one, but a pure strategy's regret can generally be larger than one. Hence, f_s^i uses normalised regret r_s^i/U_i, which is between zero and one.

$$\text{minimize:} \quad \sum_{i=1}^{m} \sum_{s \in S_i} g_s^i - (1 - b_s^i) \tag{8a}$$

$$\text{subject to:} \quad (4b), (4c), (4d), (4e), (4g), (4h), (4i)$$

$$g_s^i \geq x_s^i \qquad \forall i \in [m], \forall s \in S_i \tag{8b}$$

$$g_s^i \geq 1 - b_s^i \qquad \forall i \in [m], \forall s \in S_i \tag{8c}$$

Fig. 5. MIMLP3 formulation, objective function penalises probability of pure strategies played with positive regret

Proposition 4. *The set of Nash equilibria minimises the objective function of MIMLP 4 (Figure 6).*

Proof. Constraint (9b) demands that if $b_s^i = 0$, then $f_s^i = r_s^i/U_i$, which is at most 1. Further, due to (9e), $g_s^i = 1$. If $b_s^i = 1$, then $f_s^i = 1$ (constraint (9c)) and $g_s^i = x_s^i$ (constraint (9d)), which is at most 1. Hence, $f_s^i + g_s^i$ is at least 1 for every pure strategy s, and additional penalties (either the normalised regret or the probability of the strategy) contribute to the objective function if a strategy has positive probability and positive regret. Any feasible solution that is not a Nash equilibrium has $f_s^i + g_s^i > 1$ for some strategies, as not all strategies have either no regret or zero probability. Nash equilibria minimise the objective function, with value of $\sum_{i=1}^{m} |S_i|$, as the normalised regret is zero, or the probability of strategy is zero.

$$\text{minimize:} \quad \sum_{i=1}^{m} \sum_{s \in S_i} f_s^i + g_s^i \tag{9a}$$

$$\text{subject to:} \quad (4b), (4c), (4d), (4e), (4h), (4i)$$

$$f_s^i \geq r_s^i/U_i \qquad \forall i \in [m], \forall s \in S_i \tag{9b}$$

$$f_s^i \geq b_s^i \qquad \forall i \in [m], \forall s \in S_i \tag{9c}$$

$$g_s^i \geq x_s^i \qquad \forall i \in [m], \forall s \in S_i \tag{9d}$$

$$g_s^i \geq 1 - b_s^i \qquad \forall i \in [m], \forall s \in S_i \tag{9e}$$

Fig. 6. MIMLP4 formulation, objective function penalises either regret of or probability on pure strategy

Continuous and feasibility formulations

For potential performance improvements of the mixed-integer multilinear programs, we further give continuous as well as feasibility formulations for the

MIMLPs. Particularly, for all MIMLPs, we introduce continuous formulations[3] MIMLP1(C), MIMLP2(C), MIMLP3(C), MIMLP4(C), for which constraint 4i, i.e. constraints ($b^i_s \in \{0,1\}$) is replaced by $0 \leq b^i_s \leq 1$, $b^i_s = (b^i_s)^2$. Trivially, for $0 \leq b^i_s \leq 1$, $b^i_s = (b^i_s)^2$ to be satisfied, it must be $b^i_s = 0$ or $b^i_s = 1$, and thus the continuous formulations are equivalent to the MIMLPs. The continuous formulation MIMLP1(C) for MIMLP1 is given in Figure 7, likewise MIMLP2(C), MIMLP3(C), MIMLP4(C) are simply MIMLP2, MIMLP3, MIMLP4 but constraint 4i replaced by 10 and 11.

$$
\begin{aligned}
&\text{minimize:} && (4a) \\
&\text{subject to:} && (4b), (4c), (4d), (4e), (4h) \\
&0 \leq b^i_s \leq 1 && \forall i \in [m], \forall s \in S_i \quad (10) \\
&b^i_s = (b^i_s)^2 && \forall i \in [m], \forall s \in S_i \quad (11)
\end{aligned}
$$

Fig. 7. MIMLP1(C)

For MIMLP 2,3,4 we also introduce equivalent feasibility formulations MIMLP2(F), MIMLP3(F), MIMLP4(F), by introducing a constraint which requires the objective function of the respective MIMLP to be equal to the optimal value of the MIMLP. Particularly, MIMLP2 and MIMLP3 have optimal objective function of zero, and thus we introduce constraints (5) $= 0$, i.e. $\sum_{i=1}^m \sum_{s \in S_i} f^i_s - U_i b^i_s = 0$ (MIMLP2) and (8a) $= 0$, i.e. $\sum_{i=1}^m \sum_{s \in S_i} g^i_s - (1 - b^i_s) = 0$ for MIMLP3. MIMLP4 has optimal value $\sum_{i=1}^m |S_i|$, and thus we introduce constraint (9a) $= \sum_{i=1}^m |S_i|$, i.e. $\sum_{i=1}^m \sum_{s \in S_i} f^i_s + g^i_s = \sum_{i=1}^m |S_i|$. For all feasibility formulations, the objective function is changed to 0. MIMLP3(F) is given in Figure 8.

Further, we introduce MIMLP2(C,F), MIMLP3(C,F), MIMLP4(C,F), which combine the continuous and feasibility formulations of MIMLP2,3,4, and are thus continuous multilinear formulations[4]. MIMLP3(C,F) is given in Figure 9.

$$
\begin{aligned}
&\text{minimize:} && 0 \\
&\text{subject to:} && (4b) - (4e), (4g) - (4i), (8b), (8c) \\
& && (8a) = 0
\end{aligned}
$$

Fig. 8. MIMLP3(F)

$$
\begin{aligned}
&\text{minimize:} && 0 \\
&\text{subject to:} && (4b) - (4e), (4g) - (4h), (8b), (8c) \\
& && (10), (11) \\
& && (8a) = 0
\end{aligned}
$$

Fig. 9. MIMLP3(C,F)

[3] We note that due to this, the formulations are no longer mixed-integer, however we will still refer to MIMLP(C), to make clear that they belong to the respective MIMLP

[4] MIMLP2(C,F), MIMLP3(C,F), MIMLP4(C,F) are thus no longer mixed-integer

MIMLP	1	2	3	4	1(C)	2(C)	3(C)	4(C)	2(F)	3(F)	4(F)	2(C,F)	3(C,F)	4(C,F)
1 [1]	F	O	O	O	F	O	O	O	F	F	F	F	F	F
2 [2]	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	N	N	N

[1] : [1]: Feasibility (F) or optimality (O) program
[2] : [2]: Mixed-Integer (Yes/No)

Table 1. Overview of all formulations

3 Computational Experiments

All experiments are run on a MacBook Pro with 8GB RAM and Intel i5 CPU. Multilinear and mixed-integer formulations are implemented in AMPL [3]. We use BARON 21.1.13 [14] as the solver, which uses FilterSD and FilterSQP as nonlinear subsolvers. As the multilinear formulation MLP2 (Figure 2) is much faster than any of the MIMLPs (see Figure 3), we decide to only compare MLP2 against common algorithms for multui-player strategic-form games. The MIMLPs do not seem to give better performance than existing algorithms, and hence the analysis of those is focussed on comparing the MIMLPs to each other, to get an understanding which MIMLP formulation is best. Thus, the experiments consist of two parts: an comparison of MLP2 with commong algorithms in `Gambit` [10] (results in Figure 2), and an comparison of the different MIMLPs (results in Figure 3). All games are instanced in GAMUT [11] and have integer payoffs. We focus on *random games* and *covariance games* with negative covariance, as previous work [15],[12] indicates that covariance games with negative covariance are challenging to solve experimentally for a variety of algorithms as they tend to only have few equilibria with small support size. For all games, we take the average of 10 randomly generated instances of that game, and if a method did not find a Nash equilibrium before the timeout (which, depending on the game, is 300 or 900 seconds), we add the timeout to the average.

Table 2 compares MLP2 to algorithms in Gambit. The results can be summarised as follows: The simplicial subdivision algorithm is the slowest, and already small instances are sufficient for the algorithm to not find a Nash equilibrium in less than 15 minutes. The global Newton method, although fast on the instances for which it finds an equilibrium, in many instances terminates without giving an equilibrium back. In these cases, we put the timeout towards the average. As the global Newton method is not directly implemented in Gambit, we believe that this is some bug in code. The logit algorithm and the multilinear formulation have similar runtime for smaller instances, but for larger games, our formulations seems to be faster. Thus, to conclude, our algorithm is faster than the algorithms in Gambit we test it with, and can be an alternative.

Table 3 presents the results for the MIMLPs and the reformulations. It should be pointed out than any of the MIMLPs takes much longer to find an equilibrium than MLP2, and thus none of the MIMLPs is suited to find an equilibrium for *large* multi-player games. This is different to the mixed-integer formulations for

Instance	MLP2	GN	SD	QRE	
CG(5,5,$\rho = -0.2$)[1]	2.35	810.09	900	1.9	average (in seconds)
	100%	10%	0%	100%	percentage solved
	2.53	0.91	–	1.9	average on solved (in seconds)
CG(3,10,$\rho = -0.2$)[1]	0.57	271.56	518.96	0.36	average (in seconds)
	100%	70%	50%	100%	percentage solved
	0.57	2.22	137.9	0.36	average on solved (in seconds)
RG(5,5)[2]	2.23	540.57	632.98	2.08	average (in seconds)
	100%	40%	40%	100%	percentage solved
	2.23	1.43	232.45	2.08	average on solved (in seconds)
RG(3,10)[2]	0.329	91.325	382.9	0.362	average (in seconds)
	100%	90%	70%	100%	percentage solved
	0.329	1.47	161.287	0.362	average on solved (in seconds)
CG(5,10,$\rho = -0.2$)[1]	250.28	825.52	900	361.46	average (in seconds)
	100%	10%	0%	100%	percentage solved
	250.28	155.21	–	361.46	average on solved (in seconds)
RG(5,10)[2]	208.79	900	900	564.32	average (in seconds)
	100%	0%	0%	90%	percentage solved
	208.79	–	–	527.02	average on solved (in seconds)

[1] CG(m,$|S_i|$,ρ) = Covariance Game with m players and $|S_i|$ actions per player, covariance $\rho = -0.2$

[2] RG(m,$|S_i|$) = Random Game with m players and $|S_i|$ actions per player

GN = Global Newton method, SD = Simplicial subdivision method, QRE = quantal response equilibrium

The time is the average over 10 instances of this game in seconds - if no solution is found after the timeout of 15 minutes, the timeout is evaluated as time for the instance.

Table 2. Comparison of multilinear feasibility program to state-of-the-art algorithms

two-player games, for which [15] showed better performance on some instances than existing algorithms. Therefore, the analysis of the MIMLPs aims more to get an understanding what type of formulation is best to find a Nash equilibrium in a multi-player game, than to compare the MIMLPs to common algorithms.

First, the continuous formulations MIMLP2(C), MIMLP3(C), MIMLP4(C) of MIMLP2, MIMLP3, MIMLP4 don't give much performance improvement compared to MIMLP2,3,4. For MIMLP2 and MIMLP3, both the feasibility formulations MIMLP2(F) and MIMLP3(F) and the combined continuous and feasibility formulations MIMLP2(C,F) and MIMLP3(C,F) give better performance than MIMLP2 and MIMLP3, but whether MIMLP2(C,F) and MIMLP3(C,F) are better than MIMLP2(F) and MIMLP3(C,F) depends very much on the game. For MIMLP4, whether MIMLP4(F) or MIMLP4(C,F) are better than MIMLP4 depends on the game. Further, compared over all games, MIMLP1(C), i.e. the continuous formulation of the feasibility formulation MIMLP1 seems to give the best performance.

Method	RG(3,5)[2]	RG(3,10)[2]	CG(3,5,-0.2)[1]	CG(5,3,-0.2)[1]	RG(5,3)[2]
	average Time	Time	Time	Time	Time
MIMLP1	8.41	229.86	107.4	122.73	112.5
MIMLP1(C)	2.876	231.57	41.4	47.96	66.3
MIMLP2	19.11	202.38	16.16	538.17	660.5
MIMLP2(C)	55.93	272.15	165.5	469.5	453.14
MIMLP2(F)	14.4	150.33	29.72	410.94	345.45
MIMLP2(C,F)	14.3	279.03	68.516	198.7	218.16
MIMLP3	46.79	265.53	161.4	392.45	535.9
MIMLP3(C)	75.93	300	200.59	575.32	430.03
MIMLP3(F)	17.67	225.66	18.8	188.94	115.95
MIMLP3(C,F)	9.16	260.98	76.71	54.56	90.91
MIMLP4	5.84	220.969	79.4	359.54	49.75
MIMLP4(C)	110.3	300	69.6	479.0	462.62
MIMLP4(F)	56.5	221.26	129.78	129.59	56.88
MIMLP4(C,F)	59.19	270.69	65.12	248.85	84.52
MLP 2	0.03	0.36	0.035	0.12	0.09

[1] CG(m,$|S_i|$,ρ) = Covariance Game with m players and $|S_i|$ actions per player, covariance $\rho = -0.2$

[2] RG(m,$|S_i|$) = Random Game with m players and $|S_i|$ actions per player

The time is the average over 10 instances of this game in seconds - if no solution is found after the timeout, the timeout is evaluated as time for the instance

RG(3,5), RG(3,10), CG(3,5,-0.2): Timeout after 300 seconds [5 minutes]

CG(5,3,-0.2), RG(5,3): Timeout after 900 seconds [15 minutes]

Table 3. MIMLP results

4 Future Work

Further questions include using different nonlinear solvers for the multilinear formulation. The solver we use finds a Nash equilibrium faster than any of the other algorithms we compare it to, other solvers should only improve the performance of the multilinear feasibility formulation. We also propose generating hard-to-solve instances. Even though GAMUT [11] offers many different types of games, many of these are easy to solve even for large multi-player games. Covariant games are among the few types of games that are (relatively) difficult to solve in the game generator GAMUT, and therefore we particularly use these instances. However, due to this, there is not much variety in the hard-to-solve instances we can use. Recent work has focussed on hard-to-solve instances for polymatrix games (see [2] and `http://polymatrix-games.github.io`), and so more hard-to-solve instances is a direction to explore.

References

[1] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. “The Complexity of Computing a Nash Equilibrium”. In: *SIAM Journal on Computing* 39.1 (2009), pp. 195–259.

[2] A. Deligkas, J. Fearnley, T. P. Igwe, and R. Savani. “An Empirical Study on Computing Equilibria in Polymatrix Games”. In: *Proceedings of the 16st International Conference on Autonomous Agents and Multiagent Systems.* AAMAS ’16. 2016, pp. 186–195.

[3] R. Fourer, D. M. Gay, and B. W. Kernighan. *Ampl: A Modeling Language for Mathematical Programming.* DUXBURY, 2002. 540 pp.

[4] I. Gemp, R. Savani, M. Lanctot, Y. Bachrach, T. Anthony, R. Everett, A. Tacchetti, T. Eccles, and J. Kramár. “Sample-Based Approximation of Nash in Large Many-Player Games via Gradient Descent”. In: *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems.* AAMAS ’22. 2022, pp. 507–515.

[5] S. Govindan and R. Wilson. “A global Newton method to compute Nash equilibria”. In: *Journal of Economic Theory* 110.1 (2003), pp. 65–86.

[6] S. Govindan and R. Wilson. “Computing Nash equilibria by iterated polymatrix approximation”. In: *Journal of Economic Dynamics and Control* 28.7 (2004), pp. 1229–1241.

[7] G. van der Laan, A. J. J. Talman, and L. van der Heyden. “Simplicial Variable Dimension Algorithms for Solving the Nonlinear Complementarity Problem on a Product of Unit Simplices Using a General Labelling”. In: *Mathematics of Operations Research* 12.3 (1987), pp. 377–397.

[8] C. E. Lemke and J. T. Howson Jr. “Equilibrium Points of Bimatrix Games”. In: *SIAM Journal on Applied Mathematics* 12.2 (1964), pp. 413–423.

[9] O. Mangasarian and H. Stone. “Two-person nonzero-sum games and quadratic programming”. In: *Journal of Mathematical Analysis and Applications* 9.3 (1964), pp. 348–355.

[10] R. D. McKelvey, A. M. McLennan, and T. L. Turocy. *Gambit: Software Tools for Game Theory, Version 16.0.2.*

[11] E. Nudelman, J. Wortman, Y. Shoham, and K. Leyton-Brown. “Run the GAMUT: A Comprehensive Approach to Evaluating Game-Theoretic Algorithms”. In: *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2.* AAMAS ’04. USA: IEEE Computer Society, 2004, pp. 880–887.

[12] R. Porter, E. Nudelman, and Y. Shoham. “Simple search methods for finding a Nash equilibrium”. In: *Games and Economic Behavior* 63.2 (2008), pp. 642–662.

[13] J. Rosenmüller. “On a generalization of the Lemke–Howson algorithm to noncooperative N-person games”. In: *SIAM Journal on Applied Mathematics* 21.1 (1971), pp. 73–79.

[14] N. V. Sahinidis. *BARON 21.1.13: Global Optimization of Mixed-Integer Nonlinear Programs,* User’s Manual. http://www.minlp.com/downloads/docs/baron%20manual.pdf. 2017.

[15] T. Sandholm, A. Gilpin, and V. Conitzer. “Mixed-Integer Programming Methods for Finding Nash Equilibria”. In: *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 2*. AAAI'05. Pittsburgh, Pennsylvania: AAAI Press, 2005, pp. 495–501.

[16] R. Savani and B. von Stengel. “Hard-to-Solve Bimatrix Games”. In: *Econometrica* 74.2 (2006), pp. 397–429.

[17] K.-K. Tan, J. Yu, and X.-Z. Yuan. “Existence theorems of nash equilibria for non-cooperative n-person games”. In: *International Journal of Game Theory* 24.3 (1995), pp. 217–222.

[18] T. L. Turocy. “A dynamic homotopy interpretation of the logistic quantal response equilibrium correspondence”. In: *Games and Economic Behavior* 51.2 (2005). Special Issue in Honor of Richard D. McKelvey, pp. 243–263.

[19] T. L. Turocy. *Answer to question regarding time limits of algorithms in Gambit*. https://github.com/gambitproject/gambit/issues/261#issuecomment-660894391.