Approximating Equilibria in Restricted Games

Jack W. Doyle
Phillips Academy Andover, nycjackdoyle@gmail.com

Follow this and additional works at: https://scholar.rose-hulman.edu/rhumj

Part of the Numerical Analysis and Computation Commons

Recommended Citation
Available at: https://scholar.rose-hulman.edu/rhumj/vol25/iss1/7
Approximating Equilibria in Restricted Games

Cover Page Footnote
I would like to thank Dr. Legat for his advice and support on this paper.
Approximating Equilibria in Restricted Games

By Jack Doyle

Abstract. We consider optimal play in restricted games with linear constraints, and use $\epsilon$-equilibria to find near-equilibrium states in these games. We present three mathematical optimization formulations – a mixed-integer linear program (MILP), a quadratic program with linear constraints (QP), and a quadratically constrained program (QCP) – to both approximate and identify these states. The MILP has a short runtime relative to the QP and QCP for large games (a factor 100 faster for $|S| = 9$) and exhibits linear growth in run time, but provides only relatively weak upper bound. The QP and QCP provide a tight bound and the precise value respectively, and outperform the LP for small games ($|S| \leq 5$), but they exhibit an exponential growth of the required runtime.

1 Introduction

Nash equilibrium, first introduced in [5], is the main solution concept in game theory. In a Nash equilibrium, both agents implement the best possible strategy they have, meaning their strategy maximizes their utility given their opponent's strategy. A pair of optimal strategies like this is stable, since no agents benefit from changing their strategy.

Unrestricted games are a class of games in which there are no constraints on the set of mixed strategies. In these games, a Nash equilibrium is guaranteed to exist [5]. It is possible to find Nash equilibria in these games using mixed-integer linear programming (MILP). Several formulations of MILP exist to identify Nash equilibria [6].

However, if there are restrictions on mixed strategies, this is no longer the case. Restricted games are introduced in [1]. In these games, the set of allowable strategies is limited to $\Pi$, a subset of the set of mixed strategies allowed in an unrestricted game. Since Nash equilibria do not necessarily exist in a restricted game, the use of $\epsilon$-equilibria to evaluate the optimality of a pair of strategies is recommended.

An $\epsilon$-equilibrium is a pair of strategies where neither player can gain more than $\epsilon$ utility by switching strategies. In other words, an $\epsilon$-equilibrium is only $\epsilon$ “worse” than a Nash equilibrium. While an $\epsilon$-equilibrium provides some notion of “proximity” to optimality, and allows us to compare the optimality of two game states, it is not clear how to minimize them over a game. While given a fixed value $\epsilon$ and an unrestricted game, it

Mathematics Subject Classification. 90C33
Keywords. Nash Equilibrium, Mathematical optimization, Non-linear programming
Approximating Equilibria in Restricted Games

is possible to find an $\epsilon$-equilibrium in quasi-polynomial time [2]. However, this result does not hold generally for restricted games.

Although a restricted game does not necessarily have a Nash equilibrium, it is still valuable to analyze optimality in these games. Conceptually, we wish to find the state closest to a Nash equilibrium in a restricted game. To do so, we mathematical optimization to find the minimum value of $\epsilon$ such that there exists an $\epsilon$-equilibrium.

2 Formulation

We define a 2-player unrestricted game $G$ as follows.

Each player $i \in \{0, 1\}$ has a set of pure strategies $S_i$ they may use. Let $s_{i,j} \in S_i$ denote the $j$-th strategy in $S_i$. The function $U_i(s, s')$ represents player $i$’s utility where player 0 and 1 play strategies $s$ and $s'$, respectively.

Each player also has a support, which is the set of probabilities for which they play each possible pure strategy. Let $p_{s_{i,j}}$ denote the probability with which player $i$ uses strategy $s_{i,j} \in S_i$. The sum of the probabilities each player $i$ places must be equal to 1.

We define a game state as a fixed pair of supports. In standard form, a matrix is used to represent the utility (payoff) of each player given what strategies are played. The utility matrix in Table 1 is an example of such a matrix. Here, the $i$-th element in cell $(j, k)$ (row $j$, column $k$) is equal to $U_i(s_{i,j}, s_{(2,k)})$, the payoff of player $i$ given strategies $s_{(1,j)}$ and $s_{(2,k)}$ are played.

A restricted game is defined as a pair $(G, C)$, where $G$ is an unrestricted game, and $C$ is a set of constraints on the values of $p_{s_{i,j}}$. We only consider linear constraints in this paper.

Let $u_i$ be the payoff of the pure strategy that maximizes the payoff of player $i$ given the opposing player's strategy. Note that for unrestricted games, the “optimal,” payoff-maximizing response is always a pure strategy, so this value is equal to the maximum payoff player $i$ can achieve given his opponent’s strategy. However, in restricted games, this is not the case, since player $i$ is not necessarily allowed to play the optimal pure strategy with probability 1.

We define the regret of a player’s current mixed strategy as the difference in utility between their optimal and current strategy. Formally, we have

$$r_i = u_i - \sum_{s_{i,j} \in S_i} \sum_{s_{(1-i),j} \in S_{1-i}} p_{s_{i,j}} p_{s_{(1-i),j}} U_i(s_{i,j}, s_{(1-i),j})$$

By definition, a game state is a Nash equilibrium if both players have regret of 0. In contrast, a game state is said to be an $\epsilon$-equilibrium for some value of $\epsilon$ if both players

---

1This definition of $u_i$ effects our later definition of an $\epsilon$-equilibrium, which diverges from the definition used by [1]. We make this choice since otherwise, determining $u_i$ (and therefore $r$ and $\epsilon$) becomes much more difficult. Also, note that our definition of $u_i$ is strictly greater than those used [1], meaning that an $\epsilon$-equilibrium under our definition is always an $\epsilon$-equilibrium under the more standard definition.
have regret of at most $\epsilon$. Formally, a game state is an $\epsilon$-equilibrium if for all players $i$, $\epsilon \geq r_i$.

We define the value $\epsilon_0$ for each game as the minimum value of $\epsilon$ such that the game has an $\epsilon$-equilibrium. We are interested in estimating the $\epsilon_0$ value of games in order to find the game state that is the closest to being a Nash equilibrium.

As context for our definitions, consider the following situation. Suppose we have two manufacturers, both of which can choose to allocate their resources among producing 3 products. The row player (player 0) can produce products $a_0$, $b_0$, and $c_0$, while the column player (player 1) choose from $a_1$, $b_1$, and $c_1$. The revenue they receive from producing any of these products is dependent on how the other manufacturer allocates their resources, as defined in the standard form payoff matrix (Table 1).

<table>
<thead>
<tr>
<th></th>
<th>$a_1$</th>
<th>$b_1$</th>
<th>$c_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_0$</td>
<td>4, 3</td>
<td>6, 2</td>
<td>3, 2</td>
</tr>
<tr>
<td>$b_0$</td>
<td>2, 5</td>
<td>8, 6</td>
<td>5, 4</td>
</tr>
<tr>
<td>$c_0$</td>
<td>3, 0</td>
<td>2, 8</td>
<td>9, 7</td>
</tr>
</tbody>
</table>

Table 1: Payoff matrix

First, suppose that there are no restrictions on how the manufactures allocate their resources. Since this is an unrestricted game, it is guaranteed that a Nash equilibrium exists [5]. By inspection, we note that if both manufacturers only produce product $a$, neither manufacturers can increase their revenue by reallocating their resources, so this is a Nash equilibrium. By our definition, this is also an $\epsilon$-equilibrium, with $\epsilon_0$.

Now, suppose that the manufacturers follow these strategies, and only produce product $a$. This would lead to a shortage of products $b$ and $c$. In order to prevent this situation, the government mandates that neither manufacturer can allocate more than 40% of their resources to producing any one product. This is now a restricted game, so it is no longer guaranteed that a Nash equilibrium will exist (in fact, none exists). Even though neither manufacturer can allocate resources with 0 regret, they are still interested in the optimal way to allocate their resources. One way to do this is to find the minimum value of $\epsilon$ for which the game has an $\epsilon$-equilibrium.\(^2\) As a justification for this, consider our situation, where two manufacturers allocate their resources to maximize profit. Since there is no Nash equilibrium, both manufacturers will continue reallocating their resources for eternity in order to maximize their own profit. This reallocation of resources is practically costly. If this reallocation cost is more than $\epsilon$, then both manufacturers would settle in an $\epsilon$-equilibrium, as this would maximize their profit.\(^2\)

\(^2\)An alternative is to instead minimize the sum of the regret of the two manufacturers. In fact, we use this sum as the objective value in our QP formulation, although we do so only to estimate the minimum $\epsilon$-equilibrium. However, minimizing the sum of regrets only makes sense if the manufacturers are able to pool and divide up their regret in some way, which seems unlikely in our situation.
Approximating Equilibria in Restricted Games

profits. Finding the value of $\epsilon_0$ and the corresponding game state in this restricted game is equivalent to finding the distribution of resources that minimizes the potential gain either manufacturer can get by redistributing supplies. If it is mutually beneficial not to redistribute resources, then this is the best possible resource distribution, since neither manufacturer will have too much of an incentive to do so.

We now discuss how to approximate and find these regret-minimizing game states.

## 3 Strategies

We introduce 3 formulations for approximating $\epsilon_0$ in an unrestricted game. In all formulations, any linear constraints are added directly to the model. It is possible to use non-linear constraints with these formulations, but doing so would limit the performance benefits of the MILP and QP.

### 3.1 MILP

First, we describe a linear program that provides a weak upper bound on the value of $\epsilon_0$. The formulation is identical to the second formulation described in [6], although its intended use is different.

The idea here is to “punish” a player for playing a strategy with non-zero regret. The formulation does not take into account the values of regret or the probabilities placed on strategies. This simplification, which yields a MILP, allows us to quickly place a loose upper bound on the minimal $\epsilon$-equilibrium.

Let $D_i$ be the difference between the maximum and minimum values in player $i$’s payoff matrix. $b_{s(i,j)}$ is a boolean variable stating whether the strategy $s(i,j)$ is played with non-zero probability. $f_{s(i,j)}$ is an upper bound on the regret with which $s(i,j)$ is played. This property is utilized in Theorem 3.1.

\[
\text{Minimize } X = \sum_{i=0}^{1} \sum_{s(i,j) \in S_i} f_{s(i,j)} - D_i b_{s(i,j)}
\]
where

\[
\sum_{s(i,j) \in S_i} p_{s(i,j)} = 1 \tag{1}
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad u_{s(i,j)} = \sum_{s(1-i,j) \in S_{1-i}} p_{s(1-i,j)} U_i(s(0,j), s(1,j)) \tag{2}
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad u_i \geq u_{s(i,j)} \tag{3}
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad r_{s(i,j)} = u_i - u_{s(i,j)} \tag{4}
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad p_{s(i,j)} \leq 1 - b_{s(i,j)} \tag{5}
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad f_{s(i,j)} \geq r_{s(i,j)} \tag{6}
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad f_{s(i,j)} \geq D_i b_{s(i,j)} \tag{7}
\]

with \(p_{s(i,j)} \geq 0, u_{s(i,j)} \geq 0, u_i \geq 0, r_{s(i,j)} \geq 0, b_{s(i,j)} \in \{0, 1\}, f_{s(i,j)} \geq 0\).

Conceptually, this formulation approximates each player’s regret by treating any positive probability as the same as a probability of 1. Equation 5 sets \(b_{s(i,j)}\) equal to 0 if \(s(i,j)\) is played with positive probability, and 1 otherwise. Then, equations 6 and 7 ensure that \(f_{s(i,j)} = D_i\) if \(s(i,j)\) is played with zero probability, and \(f_{s(i,j)} = r_{s(i,j)}\) otherwise. This means that the corresponding term in the objective term is \(r_{s(i,j)}\) if the strategy is played, and 0 otherwise. So the objective function always overestimates regret.

We claim that the objective value \(X\) is an upper bound on the value of \(\epsilon_0\).

**Theorem 3.1.** For any game with minimum objective value \(X, X > \epsilon_0\).

**Proof.** We claim that for all \(s(i,j)\), \(f_{s(i,j)} - D_i b_{s(i,j)} \geq p_{s(i,j)} \cdot r_{s(i,j)}\).

First, consider the case in which \(p_{s(i,j)} > 0\). Then by (5), \(b_{s(i,j)} = 0\), since \(b\) is a binary variable. Thus, the term \(D_i b_{s(i,j)} = 0\). Then from (6), we see \(f_{s(i,j)} - D_i b_{s(i,j)} = f_{s(i,j)} \geq r_{s(i,j)}\). Since \(p < 1\), and \(r_{s(i,j)} \geq 0\), we have \(p \cdot r_{s(i,j)} \leq r_{s(i,j)}\). So our claim holds.

Alternatively, consider if \(p_{s(i,j)} = 0\). Then \(p \cdot r_{s(i,j)} = 0\). Then, from (7), we have \(f_{s(i,j)} - D_i b_{s(i,j)} \geq 0\). So our claim holds.

Thus, for all \(s(i,j)\), \(f_{s(i,j)} - D_i b_{s(i,j)} \geq p \cdot r_{s(i,j)}\). Summing over all \(s(i,j)\), we have

\[
\sum_{l=0}^{1} \sum_{s(i,j) \in S_l} f_{s(i,j)} - D_i b_{s(i,j)} \geq \sum_{l=0}^{1} \sum_{s(i,j) \in S_l} p_{s(i,j)} \cdot r_{s(i,j)}
\]

Since \(r_{s(i,j)} \geq 0\) for all \(s(i,j)\), for both players \((k = 0\) and \(k = 1\)) we have

\[
\sum_{l=0}^{1} \sum_{s(i,j) \in S_l} p_{s(i,j)} \cdot r_{s(i,j)} \geq \sum_{s(k) \in S_k} p_{s(k)} \cdot r_{s(k)}
\]

Note that the RHS is the minimum value of \(\epsilon\) so that a given game state is an \(\epsilon\)-equilibrium.

So we see that for every game state satisfying the constraints, the game state is an \(\epsilon\)-equilibrium where \(X \geq \epsilon\). Since \(\epsilon_0\) is at most \(\epsilon\), we have \(X \geq \epsilon_0\). ∎
Note that the LP will always return 0 as the objective value if \( \epsilon_0 = 0 \), or equivalently, if there is a Nash equilibrium. If \( \epsilon_0 = 0 \), both players have 0 regret, so they must only play strategies with 0 regret. These strategies would yield an objective value of 0 in the LP, so some Nash equilibrium will be returned. Equivalently, if the objective value of the LP is positive, then no Nash equilibrium exists. Also, given the game state that minimizes \( X \), directly calculating the regrets of each player in this game state provides a much tighter bound on \( \epsilon_0 \) than just the value of \( X \) alone. These facts are practically valuable, as we discuss later.

### 3.2 QP

We describe a quadratic programming (QP) formulation to approximate the lowest \( \epsilon \)-equilibrium.

\[
\text{Minimize} \quad Y = u_1 + u_2 - \sum_{s_0 \in S_0} \sum_{s_1 \in S_1} p_{s_0} p_{s_1} (U_1(s_0, s_1) + U_2(s_0, s_1))
\]

subject to:

\[
(\forall i) \quad \sum_{s(i,j) \in S_i} p_{s(i,j)} = 1 \quad (8)
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad u_{s(i,j)} = \sum_{s(1-i,j) \in S_1-i} p_{s(1-i,j)} \cdot U_i(s(i,j), s(1-i,j)) \quad (9)
\]

\[
(\forall i)(\forall s(i,j) \in S_i) \quad u_i \geq u_{s(i,j)} \quad (10)
\]

**Theorem 3.2.** Given a game with minimum objective value \( Y \), \( Y/2 \leq \epsilon_0 \leq Y \).

*Proof.* Consider a game state with objective value \( Y \), where \( Y \) is the minimum value the objective function takes.

Observe that \( Y = u_1 + r_2 = u_2 + r_1 \). Then since \( r_1, r_2 \geq 0 \), we have \( r_1, r_2 \leq Y \). Therefore, \( \max(r_1, r_2) \leq Y \). So the state with objective \( Y \) is an \( \epsilon \)-equilibrium for \( \epsilon = Y \). Thus, \( \epsilon_0 \) is at most \( Y \).

Assume \( \epsilon_0 < Y/2 \). Then there exists a set of probabilities for which \( \max(r'_1, r'_2) = \epsilon_0 < Y/2 \). So \( r'_1, r'_2 < Y/2 \). Summing, we see \( r'_1 + r'_2 < Y \) for this game state. So \( Y \) is not the minimum value of the objective function. This is a contradiction. \( \square \)

This bound can be further strengthened with the following observation.

**Theorem 3.3.** For a game state which minimizes the objective of the QP, \( \epsilon_0 \leq \max(r_1, r_2) \).

*Proof.* The game state is an \( \epsilon \)-equilibrium for \( \epsilon = \max(r_1, r_2) \). By definition, \( \epsilon_0 \leq \epsilon \). So \( \epsilon_0 \leq \max(r_1, r_2) \). \( \square \)

This upper bound is significantly stricter, as \( \max(r_1, r_2) = Y - \min(r_1, r_2) \). Note that \( r_1 \) and \( r_2 \) can be calculated directly from the optimal values yielded by the QP, which can be retrieved from the solver.
3.3 QCP

The problem of minimizing $\epsilon_0$ can be formulated exactly as follows. We let $\epsilon_0$ denote the minimum $\epsilon$ for which the current game state is an $\epsilon$-equilibrium.

Minimize $Z = \epsilon$ subject to:

$$(\forall i) \sum_{s(i,j) \in S_i} p_{s(i,j)} = 1$$

$$(\forall s(i,j) \in S_i) \quad u_{s(i,j)} = \sum_{s(1-i,j) \in S_{1-i}} p_{s(1-i,j)} \cdot U_i(s(i,j), s(1-i,j))$$

$$(\forall i) \quad u_i \geq u_{s(i,j)}$$

$$(\forall i) \quad \epsilon \geq u_i - \sum_{s(i,j) \in S_i} \sum_{s(1-i,j) \in S_{1-i}} p_i \cdot p_{s(1-i,j)} \cdot U_i(s(i,j), s(1-i,j))$$

The minimum objective value of this function must be the value of $\epsilon_0$. Note that 14 bounds $\epsilon$ to be greater than the regret in the current game state, meaning the game state is an $\epsilon$-equilibrium. Thus, this formulation yields the exact value of $\epsilon_0$. However, this formulation contains two constraints with $|S_i|^2$ non-convex quadratic terms each, making it computationally expensive relative to the other formulations.

4 Results

The linear program runs much faster than both the quadratic and quadratically constrained programs for (except for $|S| = 5$, when the QP is faster). Additionally, the run-times of both the QP and QCP grow exponentially, while the LP grows only linearly, as shown in figures 1 and 2. For larger value of $|S|$, both the QP and QCP may be computationally infeasible, although the QP runtime appears to grow slightly slower than the QCP.

For all values of $|S|$ tested, the median run time was significantly shorter than the mean run time for both the QP and QCP. For larger $|S|$, the mean is more than twice the median. This is due to the very slow worst-case run times for these models. As shown in figure 3, most cases have a very low run time. So for a single problem, optimization using the QP or QCP will likely take significantly below the mean runtime. Another potential benefit of the QP is that it contains no quadratic constraints. A solver which does not optimize for quadratic constraints would make the QCP infeasible.

In general, the benefits of the QP formulation are limited. The QP is less accurate and has a slower average runtime than the QCP. However, at $|S| = 5$, the QP is significantly faster than the QCP, and always finds the precise value as the upper bound. Also, the QP is on average only around 0.1 from the true value of $\epsilon_0$, and bounds the true value of $\epsilon_0$ within a relatively small interval.
<table>
<thead>
<tr>
<th></th>
<th>Speed (s)</th>
<th>Accuracy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Median</td>
<td>Bound</td>
</tr>
<tr>
<td>LP</td>
<td>0.00214</td>
<td>0.00185</td>
<td>15.05</td>
</tr>
<tr>
<td>QP</td>
<td>0.00157</td>
<td>0.00134</td>
<td>1.848</td>
</tr>
<tr>
<td>QCP</td>
<td>0.00220</td>
<td>0.00191</td>
<td>1.848</td>
</tr>
</tbody>
</table>

Table 2: Formulation performance (avg. over 1000 samples, $|S| = 5$)

<table>
<thead>
<tr>
<th></th>
<th>Speed (s)</th>
<th>Accuracy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Median</td>
<td>Bound</td>
</tr>
<tr>
<td>LP</td>
<td>0.0132</td>
<td>0.0116</td>
<td>10.35</td>
</tr>
<tr>
<td>QP</td>
<td>0.0113</td>
<td>0.00589</td>
<td>1.266</td>
</tr>
<tr>
<td>QCP</td>
<td>0.0129</td>
<td>0.00697</td>
<td>1.128</td>
</tr>
</tbody>
</table>

Table 3: Formulation performance (avg. over 1000 samples, $|S| = 6$)

<table>
<thead>
<tr>
<th></th>
<th>Speed (s)</th>
<th>Accuracy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Median</td>
<td>Bound</td>
</tr>
<tr>
<td>LP</td>
<td>0.0317</td>
<td>0.0319</td>
<td>7.578</td>
</tr>
<tr>
<td>QP</td>
<td>0.1111</td>
<td>0.0363</td>
<td>0.924</td>
</tr>
<tr>
<td>QCP</td>
<td>0.104</td>
<td>0.0345</td>
<td>0.760</td>
</tr>
</tbody>
</table>

Table 4: Formulation performance (avg. over 100 samples, $|S| = 7$)

<table>
<thead>
<tr>
<th></th>
<th>Speed (s)</th>
<th>Accuracy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Median</td>
<td>Bound</td>
</tr>
<tr>
<td>LP</td>
<td>0.0468</td>
<td>0.0431</td>
<td>5.580</td>
</tr>
<tr>
<td>QP</td>
<td>0.933</td>
<td>0.251</td>
<td>0.616</td>
</tr>
<tr>
<td>QCP</td>
<td>0.449</td>
<td>0.225</td>
<td>0.499</td>
</tr>
</tbody>
</table>

Table 5: Formulation performance (avg. over 100 samples, $|S| = 8$)

<table>
<thead>
<tr>
<th></th>
<th>Speed (s)</th>
<th>Accuracy</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Median</td>
<td>Bound</td>
</tr>
<tr>
<td>LP</td>
<td>0.0590</td>
<td>0.0582</td>
<td>4.456</td>
</tr>
<tr>
<td>QP</td>
<td>3.347</td>
<td>1.553</td>
<td>0.468</td>
</tr>
<tr>
<td>QCP</td>
<td>2.797</td>
<td>1.363</td>
<td>0.378</td>
</tr>
</tbody>
</table>

Table 6: Formulation performance (avg. over 100 samples, $|S| = 9$)
Figure 1: Mean solve time

Figure 2: Mean solve time (logarithmic scale)

Figure 3: QP and QCP performance (avg. over 1000 samples, |S| = 6)
Note that we excluded games with $\epsilon_0 = 0$ (i.e. they have a Nash equilibrium). This is due to two reasons. First, linear programs, such as our LP, can identify a Nash equilibrium if one exists much faster than either the QP or QCP (on the order of 100 times faster). Additionally, both the QP and QCP performed very poorly on these cases, often $10^{-100}$ times slower than the average values shown. This is further discussed in the next section.

The samples were randomly generated using the Julia \texttt{Random} module.

## 5 Conclusions

Approximating $\epsilon$-equilibrium using LP has been previously investigated in the context of anytime algorithms. Specifically, the value of the objective function in a linear formulation intended to find a Nash equilibrium in an unrestricted game places an upper bound on $\epsilon_0$ [6]. The formulation can be applied directly to a restricted game. The downside of this method is that the upper bound on $\epsilon$ is very weak, and there is no lower bound provided.\footnote{Note that neither of these factors are problematic for the intended use of the formulation. Since a Nash equilibrium is guaranteed to exist in unrestricted games, the upper bound can always be strengthened with increased computation time, as the objective will approach 0. Similarly, a lower bound on $\epsilon_0$ is unnecessary since it is already known that $\epsilon_0 = 0$.}

Note that it is impossible to formulate either an LP or a QP with linear constraints to find $\epsilon_0$. This is because the utility of each player in a given state is dependent on the supports of both players.

As mentioned earlier, in practical situations, we recommend running the LP before running either the QP or QCP. While the LP generally does not give a tight bound on $\epsilon_0$, it can quickly identify a Nash equilibrium if one exists. This can save a lot of time, especially given that the other formulations frequently stall when a Nash equilibrium exists. Additionally, for large $|S|$ the runtime of the LP is negligible in comparison to that of the QP or QCP.

In applications, it is also worth considering the maximum acceptable $\epsilon_0$ of a solution. The QP and QCP often find a game state near the true value of $\epsilon_0$ very quickly, and spend most of the runtime either confirming optimality or marginally improving this original candidate solution. Terminating the QP and QCP after reaching this acceptable $\epsilon$, or even placing a run-time limit on the models, are likely satisfactory for applications of these models. However, it should be noted that the lower bound the QP places on $\epsilon_0$ does not hold if the model is terminated prematurely.

Note that the QP we give will always give a stricter bound on $\epsilon_0$ than this linear formulation (since the LP overestimates regret in order to achieve linearity).

For future research, one possibility is to use the LP or QP objective value as an initial value for the QCP. Due to the long runtimes of the QP and QCP, comparing the objective values of the models after constant time is also an area for investigation.
References


Jack Doyle
Phillips Academy Andover
nycjackdoyle@gmail.com