A Reproduction of Correlated-Q Learning

Xinli Yu
xyu350@gatech.edu

ABSTRACT

Greenwald's 2003 paper "Correlated-Q Learning" [1] introduced the Correlated-Q (CE-Q), a framework of multiagent Q-learning algorithms based on the correlated equilibrium (CE). Unlike Nash-Q models that independently optimizes each player's policy, CE-Q optimizes the joint distribution of all players' actions. Therefore, CE-Q is a generalization of both Nash-Q and Friend-and-Foe-Q. The paper also introduced four variants of CE-Q objective functions and run experiments to show CE-Q converges to equilibrium under general-sum Markov game settings.

In this report, we first mathematically convert Foe-Q and CE-Q to standard linear programming so that they can be efficiently solved by existing optimization tools like cvxopt. For the experiments, a Soccer Game environment is implemented and then used to reproduce the results of the Soccer Game experiments in [1] and obtain similar Q-value difference graphs for four different agents, CE-Q, Foe-Q, Friend-Q and Q-learner, which supports the paper's claim that the first three agents converge but not the Q-learner. Moreover, we show our agents' equilibrium also to further confirm the discussion in [1] about the agents' final policy. The code can be found at https://github.com/XinliYu/RL-Projects.

1. MARKOV GAMES

In Greenwald's 2003 paper [1], we are concerned with the Markov game problem, a stochastic game with Markov property. We consider the stochastic game of the form

$$\langle I, S, \left(A_i(s)\right)_{s\in S, 1\le i\le n}, P, (R_i)_{1\le i\le n}\rangle$$

where I is a set of n players, S is a set of states, $A_i(s)$ is the ith player's set of actions at state s, P is the set of transition function that describes state transitions, conditioned on past states and joint actions and $R_i(s, \vec{a})$ is the ith player's reward for state $s \in S$ and joint actions $\vec{a} \in A(s) = A_1(s) \times \ldots \times A_n(s)$. A Markov game is a stochastic game with the Markov property

$$P(s_{t+1}|s_t, \vec{a}_t, \ldots, s_0, \vec{a}_0) = P(s_{t+1}|s_t, \vec{a}_t)$$

2. Q-LEARNING

A MDP is a one-player Markov game, with Bellman's equations that describes the optimal state and action for a single agent MDP, $Q^*(s,a)$ is the long-term reward as sum of immediate reward and discounted future reward by following optimal-policy if the agent take action a at state s, the value function $V^*(s)$ is the optimal action for the agent by taking the action that maximize $Q^*(s,a)$ over all actions a, $\pi^*(s)$ is the action that maximize $Q^*(s,a)$ at each state s:

$$Q^*(s,a) = (1-\gamma)R(s,a) + \gamma\sum_{s'} P(s'|s,a)V^*(s')$$

$$V^*(s) = \max_{a\in A(s)} Q^*(s,a)$$

$$\pi^*(s) \in \operatorname{argmax}_{a\in A(s)} Q^*(s,a)$$

In Markov games, player i's Q-values are defined over states and action vectors $\vec{a} = (a_1, \ldots, a_n)$, rather than the state-action pairs:

$$Q_i(s,a) = (1-\gamma)R_i(s,\vec{a}) + \gamma\sum_{s'} P(s'|s,\vec{a})V_i(s')$$
$$V_i^*(s) = \max_{a\in A(s)} Q_i^*(s,a) \tag{1}$$

For the state-value function, the notion also makes sense when we generalize MDP to the Markov games, but the analogue in MDP that all players maximize their respective rewards with respect to other players' actions no longer makes sense because there might not exist actions that satisfy all these equations. One example is our later Soccer Game at the state of Fig 1. We now examine different definitions of state-value functions in the following sections.

3. FRIEND/FOE-Q

Littman's paper [2] considers a two-player, zero-sum Markov games and von Neumann minimax value function This is nicknamed the Foe-Q learner in [1]. Foe-Q learner utilizes adversary strategy, in which each agent picks an action that minimizes opponent's rewards, even if this action is not the best choice for himself if not in a zero-sum game. Foe-Q uses minimax function, in which an agent first finds opponent's action that minimizes his rewards, and then maximizes his expected reward based on such distribution.

Let $\sum_i(s)$ be the probabilistic action space of player i at state s. Now the state-value function is

$$V_1(s) = \max_{\sigma_1\in\sum_1(s)} \min_{a_2\in A_2(s)} Q_1(s,\sigma_1,a_2) = -V_2(s) \tag{2}$$

where

$$Q(s,\sigma_1,a_2) = \textstyle\sum_{a_1\in A_1} \sigma_1(a_1)Q(s,a_1,a_2).$$

[2] also defines the Friend-Q learning for coordination game, which has equivalent reward functions for all the players. The state-value function for Friend-Q is defined as

$$V_i(s) = \max_{\vec{a}\in A(s)} Q_i(s,\vec{a})$$

Friend/Foe-Q only guarantees to work on zero-sum games and may not apply to some general-sum games.

4. NASH-Q

For the general case of n-player, general-sum games, Hu and Wellman [3] proposed the state-value function as

$$V_i(s) \in \text{Nash}_i(Q_1(s), \ldots, Q_n(s))$$

$\text{Nash}_i(X_1, \dots, X_n)$ denotes the ith player's reward according to some Nash equilibrium in the general sum game determined by reward matrices $X_1, \dots, X_n$.

Nash-Q assumes independent probability distributions over actions and All agents optimize with respect to one another probabilities. When equilibrium is reached, no agent would prefer a different action. This is a generalization of the minmax value function, since Nash equilibrium and minimax equilibrium coincide in zero-sum games.

5. CE-Q

The *Correlated-Q Learning* paper [1] proposes an alternative definition to the state-value function in Markov games as

$$V_i(s) \in \text{CE}_i(Q_1(s), \dots, Q_n(s)) = \text{CE}_i\left(\vec{Q}(s)\right)$$

where $\text{CE}_i(X_1, \dots, X_n)$ denotes the ith player's long-term reward according to some *correlated equilibrium* (CE) in the general sum game determined by the rewards $X_1, \dots, X_n$. The paper [1] defines a *correlated equilibrium* as "a probability distribution over the joint space of actions"; for example, let σ_s^* be the set of all such joint equilibrium distributions, then

$$\text{CE}_i\left(\vec{Q}(s)\right) = \left\{ \sum_{\vec{a} \in A(s)} \sigma_s^*(\vec{a}) Q_i(s, \vec{a}) \right\}$$

The paper *claims* Nash-Q is a special case of CE-Q for the following reasons,

- Nash-Q assumes independent probability distributions over actions and the agents optimize their own action probabilities.
- CE-Q does not assume agents' probability distribution over actions are independent; on the contrary, it considers a joint distribution over all actions taken by all players. All agents optimize "with respect to one another's probabilities, conditioned on their own", and therefore players' marginal distributions over actions can be correlated.

Objectives. Let σ denote a joint distribution over the action vector $\vec{a}$ of all players; use I to denote the set of all players. The paper provides four variants of objective functions to optimal σ_s,

1. *Utilitarian* (uCE-Q): maximizes the sum of the players' rewards,
$$\sigma_s^* \in \arg\max_{\sigma} \sum_{i \in I} \sum_{\vec{a} \in A(s)} \sigma(\vec{a}) Q_i(s, \vec{a}) \tag{3}$$
2. *Egalitarian* (eCE-Q): maximizes the minimum of the players' rewards,
$$\sigma_s^* \in \arg\max_{\sigma} \min_{i \in I} \sum_{\vec{a} \in A(s)} \sigma(\vec{a}) Q_i(s, \vec{a})$$
3. *Republican* (rCE-Q): maximizes the maximum of the players' rewards:
$$\sigma_s^* \in \arg\max_{\sigma} \max_{i \in I} \sum_{\vec{a} \in A(s)} \sigma(\vec{a}) Q_i(s, \vec{a})$$
4. *Libertarian* (lCE-Q): maximize the maximum of each individual player i's rewards:
$$\sigma_s^* = \prod_{i \in I} (\sigma^i)^*, (\sigma_s^i)^* \in \arg\max_{\sigma} \sum_{\vec{a} \in A(s)} \sigma(\vec{a}) Q_i(s, \vec{a})$$

Rationality Constraints. In addition, according to [1], CE-Q must satisfy some rationality constraints. For example, in a two-player game, it can be mathematically written as

$$\begin{aligned} &\sum_{a_2 \in A_2} \pi(a_2|a_1)\, Q_1(a_1, a_2) \geq \sum_{a_2 \in A_2} \pi(a_2|a_1)\, Q_1(a, a_2), \forall a \in A_1 \\ &\sum_{a_1 \in A_1} \pi(a_1|a_2)\, Q_1(a_1, a_2) \geq \sum_{a \in A_1} \pi(a_1|a_2)\, Q_1(a_1, a), \forall a \in A_2 \end{aligned} \tag{4}$$

where notations like $\pi(a_1|a_2)$ denotes the conditional probability of player 1 taking action a_1 given another player 2 taking action a_2.

> ***Remarks***: $\pi(a_2|a_1)$ can be viewed as player1's optimal "*analysis*" of the likelihood of player2's action if player1 takes action a_1, and summation $\sum_{a_2 \in A_2} \pi(a_2|a_1)\, Q_1(a_1, a_2)$ is the long-term reward the player1 would expect to have after action a_1 is executed.
>
> On the contrary, summation $\sum_{a_2 \in A_2} \pi(a_2|a_1)\, Q_1(a, a_2), a \neq a_1$ is the expected reward when player1 executes another action $a \neq a_1$ with the same "analysis" $\sigma(a_2|a_1)$ in mind, if this is higher reward, then a rational player would not have taken action a_1, and hence a contradiction.

Multiply both sides of the first inequality of (4) by $\pi(a_1)$, the probability of player1 taking action a_1, and multiply both sides of the second inequality of (4) by $\pi(a_2)$, and notice $\pi(a_2|a_1)\pi(a_1) = \pi(a_1|a_2)\pi(a_2) = \sigma(a_1, a_2)$, then (4) can be re-written in terms of σ as

$$\begin{aligned} &\sum_{a_2 \in A_2} \sigma(a_1, a_2)\, Q_1(a_1, a_2) \geq \sum_{a_2 \in A_2} \sigma(a_1, a_2)\, Q_1(a, a_2), \forall a \in A_1 \\ &\sum_{a_1 \in A_1} \sigma(a_1, a_2)\, Q_1(a_1, a_2) \geq \sum_{a \in A_1} \sigma(a_1, a_2)\, Q_1(a_1, a), \forall a \in A_2 \end{aligned} \tag{5}$$

> ***Difficulty***: one major effort of this report is to properly convert Foe-Q and CE-Q to linear programming. Foe-Q is a minimax optimization problem, and the procedure to convert such optimization to standard linear programming is discussed in Sec.7.
>
> All four variants of objective functions can be written in a linear programming format. uCE-Q and lCE-Q is to maximize linear sum(s), while eCE-Q and rCE-Q can use the same tricks that convert minimax optimization linear programming. Conversation of rationality constraints to linear programming is non-trivial and analyzed in Sec.7.

6. SOCCER GAME

The Soccer Game in [1] uses a 2×4 grid. It has two players A, B, one with the ball. The leftmost column is the goal for player A, and the rightmost column is the goal for player B. *Note* although most game setup is borrowed from [2], the field size and rewards are changed.

State: The state of the game is represented by 3 values: 1) who has the ball; 2) position of player A; position of player

B. Since two agents cannot occupy the same cell, so there are $2 \times 8 \times 7 = 112$ states in total.

Action: Every player can move north, south, west, east or stand still, and we represent these five actions by letters N,S,W,E,O.

Transitions & Rewards: each iteration can be summarized as the following,

- Randomly, one player moves first.
- If an action moves the player towards the filed border, then the player stand still.
- If one player moves into the other player, i.e. a collide happens, then 1) if the moving player has ball, then the ball changes possession; 2) if not, nothing happens.
- If a player with ball moves to someone's goal, then "someone" gets 100 reward, and the other player gets -100 reward, and the game ends immediately. This means the game is a zero-sum game.

Paper [1] is particularly interested in how different agents handle the following state: B has the ball while A blocks B's way to goal. This state also means this simple soccer game does not have deterministic policies (any deterministic policy of B can be indefinitely blocked by player A). More discussion is offered in Sec. 8.

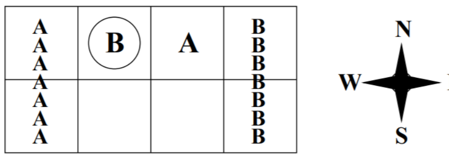


Fig 1. The paper [1] is particularly interested in checking how different agents handle this state.

7. LINEAR PROGRAMMING FORMULATION

Linear programming (LP) implementation by tools like *cvxopt* or *scipy.optimize* requires a problem being formatted in a standard format,

$$\min_{\mathbf{x}} \mathbf{c}^{\mathrm{T}}\mathbf{x} \quad \text{s.t.} \quad \begin{matrix} \mathbf{Gx} \leq \mathbf{h} \\ \mathbf{Ax} = \mathbf{b} \end{matrix} \tag{6}$$

This section discusses how to formulate the Foe-Q and CE-Q models as a linear programming problem, which is necessary for the problem to be efficiently[1] solved using existing LP tools. We ***focus*** on the zero-sum two-player case with the same action space A just like the soccer game. Let $|A|$ denote the number of available actions. Suppose the state s transits to state s'.

<u>FOE-Q</u>: The optimization of $V_i(s'), i = 1,2$ by Eq. (2) can be re-written as the following.

$$\begin{aligned} V_1(s') &= \max_{\sigma_1} \min_{a_2} \sum_{a_1} \sigma_1(s', a_1) Q_1(s', a_1, a_2) \\ &= -V_2(s') \end{aligned} \tag{7}$$

Use matrix notation: 1) $\boldsymbol{\sigma}_{1,s'}^{\mathrm{T}} \coloneqq \left(\sigma_1(s', a_1), \dots, \sigma_1(s', a_{|A|})\right)$ denotes the policy of player 1 for state s'; 2) $\mathbf{Q}_{1,s'} \coloneqq [Q_1(s', a_1, a_2)]_{a_1, a_2 \in A}$ denotes an $|A| \times |A|$ matrix, which is player 1's Q-table under state s'. the above (7) can be re-written as

$$V_1(s') = \max_{\sigma_1} \min_{a_2} \boldsymbol{\sigma}_{1,s'}^{\mathrm{T}} \mathbf{Q}_{1,s'} = \max_{\sigma_1} \min_{a_2} \mathbf{Q}_{1,s'}^{\mathrm{T}} \boldsymbol{\sigma}_{1,s'}$$

Let $x_0 = \min_{a_2} \mathbf{Q}_{1,s'}^{\mathrm{T}} \boldsymbol{\sigma}_{1,s'}$ be the "inner" optimization, then x_0 is smaller than every element in the column vector $\mathbf{Q}_{1,s'}^{\mathrm{T}} \boldsymbol{\sigma}_{1,s'}$. To write down this relationship in a matrix format, let $\mathbf{x}_0 = (x_0, \dots, x_0)^{\mathrm{T}}$ be a column vector with all its elements being x_0, then we have $\mathbf{x}_0 - \mathbf{Q}_{1,s'}^{\mathrm{T}} \boldsymbol{\sigma}_{1,s'} \leq \mathbf{0}$. This can be further written as

- Constr. 1: $\mathbf{G}_1 \mathbf{x} \leq \mathbf{0}$ where $\mathbf{G}_1 = \left(\mathbf{1}, -\mathbf{Q}_{1,s'}^{\mathrm{T}}\right)$ and $\mathbf{x} = \begin{pmatrix} x_0 \\ \boldsymbol{\sigma}_{1,s'} \end{pmatrix}$.

$\boldsymbol{\sigma}_{1,s'}$ is a probability distribution, its elements should be non-negative and sum to 1. This can be written in terms of $\mathbf{x}$ as

- Constr. 2: $\mathbf{a}^{\mathrm{T}}\mathbf{x} = 1$ where $\mathbf{a} = (0,1, \dots, 1)^{\mathrm{T}}$
- Constr. 3: $\mathbf{G}_2 \mathbf{x} \leq 0$ where $\mathbf{G}_2 = \begin{pmatrix} 0 & \mathbf{0} \\ \mathbf{0} & -\mathbf{I} \end{pmatrix}$ where $\mathbf{I}$ is an $|A| \times |A|$ identity matrix.

Constraint 1 and 3 can be merged into a single constraint:

- Combined Constr. 1 & Constr. 3: $\mathbf{Gx} \leq \mathbf{0}$ where $\mathbf{G} \coloneqq \begin{pmatrix} \mathbf{G}_1 \\ \mathbf{G}_2 \end{pmatrix}$.

The "outer" optimization wants to maximize x_0, equivalent to minimize $-x_0$. Notice $-x_0 = \mathbf{c}^{\mathrm{T}}\mathbf{x}$ where $\mathbf{c}^{\mathrm{T}} = (-1, \dots, 0)$. Finally, the minimax optimization in (7) is transformed into the LP format of (6).

$$\min_{\mathbf{x}} \mathbf{c}^{\mathrm{T}}\mathbf{x} \quad \text{s.t.} \quad \begin{matrix} \mathbf{Gx} \leq \mathbf{0} \\ \mathbf{a}^{\mathrm{T}}\mathbf{x} = 1 \end{matrix} \tag{8}$$

The complete LP formulation for FOE-Q is summarized as Algorithm 1 using Numpy-like pseudocode.

```
Algorithm 1: Linear programming formulation for FOE-Q
Input  : Player 1's Q-table under state $s'$, denoted by matrix $\mathbf{Q}_{1,s'}$, the
         number of actions $|A|$
Output: A linear programming problem
$c = \text{zeros}(|A| + 1)$;
$c[0] = -1$;
$G_1 = \text{ones}(|A|, |A| + 1)$;
$G_1[:, 1:] = -\mathbf{Q}_{1,s'}^{\mathrm{T}}$;
$G_2 = -\text{identity}(|A| + 1, |A| + 1)$;
$G_2[0, 0] = 0$;
$G = \text{vstack}(G1, G2)$;
$h = \text{zeros}(2 * |A| + 1)$ $A = ones(1, |A| + 1)$;
$A[0, 0] = 0$;
$b = \text{ones}(1)$;
Return lp(c=$c$,G=$G$,h=$h$,A=$A$,b=$b$);
```

<u>CE-Q</u>: As discussed in Sec.5, the optimization of a CE-Q model consists of the rationality constraints and an objective function like (3). We first convert the rationality constraints to LP format, which is common for all CE-Q models.

Let $\mathbf{Q}_{i,s} \coloneqq Q_i(s, \cdot, \cdot)$ denote a player's Q-table under state s. In the two-player case, the first type of inequality constraints in (5) (i.e. the rationality constraints for player1) can be written as

$$\mathbf{Q}_{1,s}(i,:)\boldsymbol{\sigma}^{\mathrm{T}}(i,:) \geq \mathbf{Q}_{1,s}(k,:)\boldsymbol{\sigma}^{\mathrm{T}}(i,:), \forall k = 1, \dots, |A|, k \neq i$$

This is equivalent to a single matrix-form inequality

[1] Some optimization frameworks like *cvxopt.modeling* allows input of a general convex optimization problem with their original objective function and constraints; this saves our manual effort to transform the problem. However, the algorithm will not be aware of the linear nature of the problem and will apply a general algorithm that is much slower than LP. By our experiment, CE-Q finishes its 1M iterations by LP in less than 1 hour, while the *cvxopt.modeling* framework takes 12 hours to finish and arrive at the same equilibrium.

$$\mathbf{G}_1\tilde{\boldsymbol{\sigma}} \le 0$$

where $\tilde{\boldsymbol{\sigma}}$ denotes the row-wise vectorization of $\boldsymbol{\sigma}$, $\mathbf{G}_1 = \begin{pmatrix} \mathbf{G}_{1,1} & & \\ & \ddots & \\ & & \mathbf{G}_{1,|A|} \end{pmatrix}$ is a block diagonal matrix, and each block $\mathbf{G}_{1,i} = \begin{pmatrix} \vdots \\ \mathbf{Q}_{1,s}(k,:) - \mathbf{Q}_{1,s}(i,:) \\ \vdots \end{pmatrix}, k \ne i$ is a $(|A| - 1) \times |A|$ matrix.

The second type of inequality constraints in (5) (i.e. the rationality constraints for player2) can be written as

$$\mathbf{Q}_{2,s}^{\mathrm{T}}(:,j)\boldsymbol{\sigma}(:,j) \ge \mathbf{Q}_{2,s}^{\mathrm{T}}(:,k)\boldsymbol{\sigma}(:,j), \quad \forall k = 1,\dots,|A|, k \ne j \tag{9}$$

This is also equivalent to some $\mathbf{G}_2\tilde{\boldsymbol{\sigma}} \le 0$ for a matrix $\mathbf{G}_2$, but the description of $\mathbf{G}_2$ is bit trickier. Let's suppose $|A| = 5$ like in the soccer game, then

- For $j = 1$, (9) gives
$$\mathbf{G}'_{2,1}\boldsymbol{\sigma}(:,1) \le \mathbf{0}$$
$$\mathbf{G}'_{2,1} = \left(\mathbf{Q}_{2,s}(:,k) - \mathbf{Q}_{2,s}(:,1), k = 2,3,4,5\right)^{\mathrm{T}}$$
where $\mathbf{G}_{2,1}$ is a transpose of four columns $\mathbf{Q}_{2,s}(:,k) - \mathbf{Q}_{2,s}(:,1)$, $k = 2,3,4,5$. However, these constraints are over $\boldsymbol{\sigma}(:,1)$, which corresponds to every five elements in $\tilde{\boldsymbol{\sigma}}$ starting at the first element; therefore, to write the constraints in term of $\tilde{\boldsymbol{\sigma}}$, we should insert 4 columns of zeros after each column in $\mathbf{G}'_{2,1}$ to have a matrix $\mathbf{G}_{2,1}$ compatible with $\tilde{\boldsymbol{\sigma}}$. An illustration is in Fig 2.
- We can continue the same construction of $\mathbf{G}'_{2,j}$ for $j = 2,3,\dots$, and insert 4 columns of zeros after each column in a "circulant" way to have $\mathbf{G}_{2,j}$. Finally, $\mathbf{G}_2$ is a vertical stack of these $\mathbf{G}_{2,j}$ matrices. An illustration is shown in

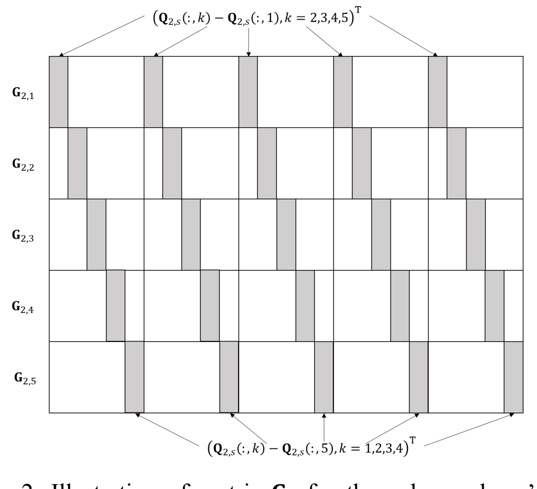

$\left(\mathbf{Q}_{2,s}(:,k) - \mathbf{Q}_{2,s}(:,1), k = 2,3,4,5\right)^{\mathrm{T}}$
$\mathbf{G}_{2,1}$
$\mathbf{G}_{2,2}$
$\mathbf{G}_{2,3}$
$\mathbf{G}_{2,4}$
$\mathbf{G}_{2,5}$
$\left(\mathbf{Q}_{2,s}(:,k) - \mathbf{Q}_{2,s}(:,5), k = 1,2,3,4\right)^{\mathrm{T}}$

Fig 2. Illustration of matrix $\mathbf{G}_2$ for the column-player's rationality constraints in LP format $\mathbf{G}_2\tilde{\boldsymbol{\sigma}} \le \mathbf{0}$.

The remaining constraints are similar to FOE-Q, we need to ensure $\tilde{\boldsymbol{\sigma}}$ is a probability vector, and thus need 1) $-\mathbf{I}\tilde{\boldsymbol{\sigma}} \le \mathbf{0}$, where $\mathbf{I}$ is a $|A|^2 \times |A|^2$ identity matrix, together with $\mathbf{G}_1\tilde{\boldsymbol{\sigma}} \le 0, \mathbf{G}_2\tilde{\boldsymbol{\sigma}} \le 0$, we have $\mathbf{G}\tilde{\boldsymbol{\sigma}} \le 0, \mathbf{G} = \begin{pmatrix} G_1 \\ G_2 \\ -\mathbf{I} \end{pmatrix}$; 2) $\mathbf{a}^{\mathrm{T}}\tilde{\boldsymbol{\sigma}} = 1$ where a is a vector of length $|A|^2$ consisting of all 1s.

The discussion until now is true for all CE-Q. In particular, the soccer game experiment is done with the *Utilitarian* objective function (3), which is equivalent to $\min \mathbf{c}^{\mathrm{T}}\tilde{\boldsymbol{\sigma}}$ where $\mathbf{c} = \mathrm{vec}\left(\mathbf{Q}_{1,s} + \mathbf{Q}_{2,s}\right)$. As a result, the whole uCE-Q model is equivalent to a LP problem

$$\min_{\tilde{\boldsymbol{\sigma}}} \mathbf{c}^{\mathrm{T}}\tilde{\boldsymbol{\sigma}} \quad \mathrm{s.t.} \quad \begin{matrix} \mathbf{G}\tilde{\boldsymbol{\sigma}} \le \mathbf{0} \\ \mathbf{a}^{\mathrm{T}}\tilde{\boldsymbol{\sigma}} = 1 \end{matrix}$$

The complete LP formulation for CE-Q (Utilitarian) is summarized as Algorithm 2 using Numpy-like pseudocode.

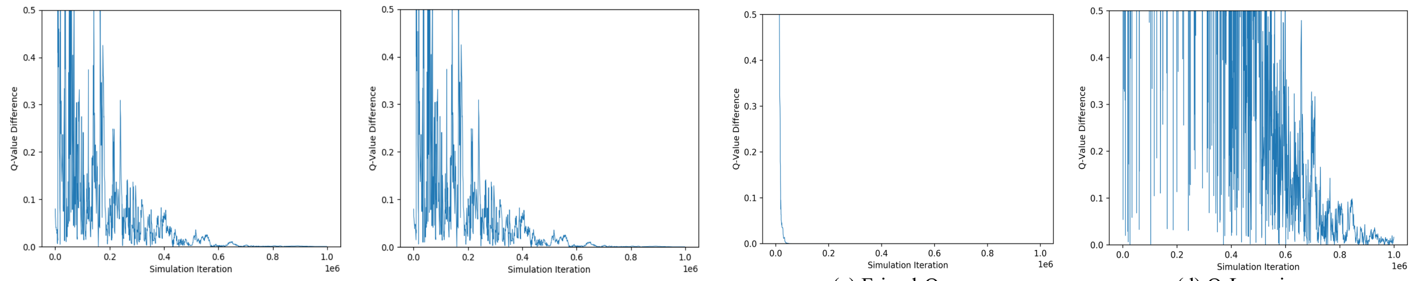
```
Algorithm 2: Linear programming formulation for uCE-Q
Input  : Both players' Q-tables under state $s'$, denoted by matrix
         $\mathbf{Q}_{1,s'}, \mathbf{Q}_{2,s'}$, the number of actions $|A|$
Output: A linear programming problem
$c = \mathrm{flatten}(\mathbf{Q}_{1,s'} + \mathbf{Q}_{2,s'})$;
$G_1, G_2 = \mathrm{zeros}((|A|-1) \times |A|, |A|^2)$;
for $i = 0, \dots, |A|-1$ do
    cols = range($i \times |A|, (i+1) \times |A|$);
    for $k = 0, \dots, |A|-1, k \ne i$ do
        row = $i \times (|A|-1) + k - 1$ if $k > i$ else $k$;
        $G_1$[row,cols]=$\mathbf{Q}_{1,s'}$[k] − $\mathbf{Q}_{1,s'}$[i];
    end
end
for $j = 1, \dots, |A|-1$ do
    cols = range($j, |A|^2, |A|$);
    for $k = 1, \dots, |A|-1, k \ne j$ do
        row = $i \times (|A|-1) + k - 1$ if $k > j$ else $k$;
        $G_2$[row, cols] = $(\mathbf{Q}_{2,s'}[:,k] - \mathbf{Q}_{2,s'}[:,j])^{\mathrm{T}}$ ;
    end
end
$G = \mathrm{vstack}(G_1, G_2, -\mathbf{I})$;
$h = \mathrm{zeros}(G.shape[0])$;
$A = ones(1, |A|)$;
$b = \mathrm{ones}(1)$;
Return lp(c=$c$,G=$G$,h=$h$,A=$A$,b=$b$);
```

8. EXPERIMENTS

This section demos our reproduction of the Soccer Game experiment in [1] with four agents uCE-Q, Foe-Q, Friend-Q and Q-learning, and reproduce their figure 3. In addition, we show the equilibrium evolution and the agents' final policies to further confirm the paper [1]'s discussion of its soccer game results.

Implementation: A Soccer Game environment is coded according to Sec. 6. Policy updates of CE-Q and Foe-Q are using cvxopt linear programming function.

Difficulties: The paper is unclear about parameters. It only states $\gamma = 0.9, \alpha \to 0.001, \epsilon \to 0.001$; however, it does not further mention the initial learning rate α_0, the initial epsilon ϵ_0, and how fast both parameters decay to 0.001, and we found not any parameter setup can lead to graphs similar to the paper's.



(a) CE-Q (Utilitarian) (b) Foe-Q (c) Friend-Q (d) Q-Learning

Fig 3. Reproducing the Q-value difference graphs.

As a result, we must do some rough parameter turning on our end. We start with Friend-Q for this purpose as it is a fast agent, only needs a few minutes to train for 1M iterations and plot the Q-value difference. We then test the good parameter setup for Friend-Q on other agents and found $\alpha_0 \approx 0.2$, $\epsilon_0 = 1$ with decay factor 0.99999 is good for all agents. We also tested random initialization of Q matrices, or simply initialize all values as 1, but this does not make much difference.

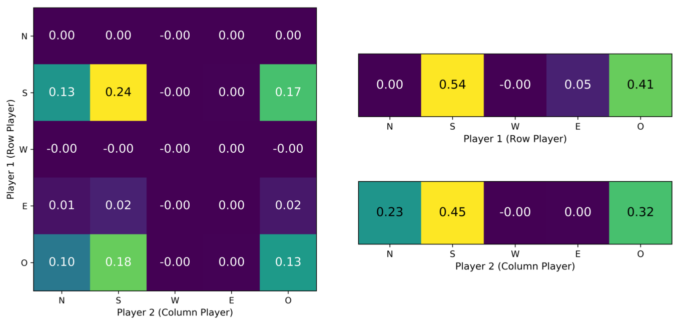
Result 1: **Q-Value Difference Graphs**. We run the four agents with parameter setup $\gamma = 0.9, \alpha_0 = 0.2, \epsilon_0 = 1$, and $\alpha, \epsilon \rightarrow 0.001$ by a decay factor 0.99995. The experiment runs for 1 million iterations from tens of thousands of episodes. At the end of each episode, the game ends and we reset the players to a random non-goal position with random ball ownership. Moreover, we pre-generate the training episodes including the initial ball ownerships, so that all agents are given the same training data and the results are comparable.

Before each iteration, if the game is the same state as in Fig 1, and if then player A heads south while player B stands still, then we record the Q-value before and after the iteration for that state, and plot the difference in Fig 3.

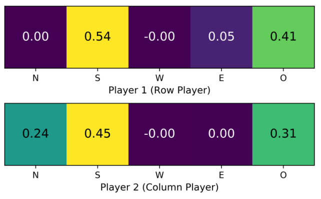
- Convergence of CE-Q (Utilitarian), Foe-Q and Friend-Q is confirmed, and their trends and fluctuations well resemble the results in [1].
- Results for CE-Q (Utilitarian) and Foe-Q are nearly the same except for numerical noise, confirming the same claim in [1].
- The Q-leaner in Fig 3 (d) has much larger fluctuations just like in [1], and it does not converge within 1M iterations (still with noticeable fluctuations at the end in the plot). The Q-value decreases, though, and we agree this is due to the diminishing learning rate. However, our Q-learner has smaller fluctuations than in [1]. We tried a few times but were unable to replicate fluctuations of similar magnitude for the Q learner. This difference could be caused by different parameter setups, different training episodes, or even numerical stability of the optimization software.

Result 2: **Equilibrium**. [1] states that CE-Q and Foe-Q "converge to nondeterministic policies for both players", where each one either heads south or sticks. This claim is confirmed by the following Fig 4 (a), (b), where the majority of probabilities are over states 'N', 'S', 'O' for both players. Note at the state of Fig 1, state 'N' is the same as the standing-still action 'O'. Moreover, we can observe again that CE-Q and Foe-Q learn the same equilibrium.

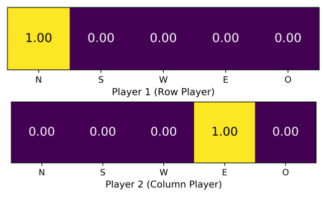
[1] also state that Friend-Q converges to a deterministic policy for both A and B, where A chooses to stand still ('N' is the same as 'O' at the state of Fig 1), and B choose to bump into A so that A will the ball. [1] explains this as that B falsely believes A will take the ball to B's goal, and similarly A is indifferent to B's actions because she assumes B "plans to score a goal for her immediately".



(a) CE-Q; left: joint distribution, right: marginal distribution



(b) FOE-Q



(c) Friend-Q

Fig 4. Equilibrium of different agents. Note action 'N' is equivalent to action 'O' at the state of Fig 1.

REFERENCES

1. Greenwald, Amy, Keith Hall, and Roberto Serrano. "Correlated Q-learning." ICML. Vol. 3. 2003.
2. Littman, Michael L. "Markov games as a framework for multi-agent reinforcement learning." Machine learning proceedings 1994. Morgan Kaufmann, 1994. 157-163.
3. Hu, Junling, and Michael P. Wellman. "Multiagent reinforcement learning: theoretical framework and an algorithm." ICML. Vol. 98. 1998.