Reinforcement Learning Algorithms for MFG and MFC

Ryan Yick

Professors: Nils Detering, Jean-Pierre Fouque | Graduate Mentor: Jimin Lin

STATISTICS & APPLIED PROBABILITY — CENTER FOR FINANCIAL MATHEMATICS & ACTUARIAL RESEARCH

Problem

What is Reinforcement Learning? Reinforcement learning (RL) is a type of machine learning that involves training an intelligent agent to make decisions in an environment by receiving feedback in the form of rewards or costs. The agent learns to take actions that minimize their cumulative cost over time.

What are Mean Field Games (MFG) and Mean Field Control (MFC)? MFG and MFC are models used to study the behavior of large populations of interacting agents, where the distribution of the population is a critical factor. In MFG, one agent aims to minimize their individual cost by observing the population, while in MFC, a 'controller' develops a strategy for the entire population to minimize costs.

MFG is a competitive model, whereas MFC is a cooperative one.

Research Question: Can we write an RL algorithm that captures optimal decision-making for MFG and MFC models?

Relevance: MFG and MFC are becoming more and more accurate models for portfolio trading, predicting optimal times to buy and sell shares. In this model, our 'agents' can be seen as investors and our 'population' distribution would track the trades of shares from the other participating investors.

Problem Formulation

Our research involved studying a very rudimentary model, a state space, X, with two states and an action space, A, with two actions:

$$x \in \{x_0, x_1\} = X \qquad a \in \{0, 1\} = A$$

where the agent resides in one of the states and can either choose **a=0 to stay or a=1 to switch states.** Each action has a corresponding cost, denoted by the **cost function:**

$$f(x, a) = (1 + c\mu(x))(x + a)$$

where c is the congestion seeking/averse parameter and $\mu(x)$ is the proportion of the population in state x. It should be noted that in practice, we only get the outputs of f and don't know the actual function itself.

Over time, these accumulated costs are captured in an **aggregate cost function, Q:**

$$Q^*(x, a) = \min_{\alpha} \left[\sum_{n=0}^{\infty} \gamma^n f(X_n, \alpha(X_n)) | X_0 = x, A_0 = a \right]$$

where γ is a discounting factor and $\alpha \in A$. In our model, we represent Q^* as the following matrix:

$$Q^*(x, a) = \begin{array}{c} \\ x_0 \\ x_1 \end{array} \begin{array}{c} \begin{array}{cc} a{=}0 & a{=}1 \end{array} \\ \begin{pmatrix} q(0,0) & q(0,1) \\ q(1,0) & q(1,1) \end{pmatrix} \end{array}$$

Using dynamic programming, it can be shown that Q* is the solution to the **Bellman Equation:**

$$Q^*(x, a) = f(x, a) + \gamma \sum_{x' \in X} p(x'|x, a) \min_{a'} Q^*(x', a')$$

where the probabilities in the equation are denoted by an **epsilon matrix:**

$$\varepsilon_{00} = p(x_1|x = x_0, a = 0) \qquad \varepsilon_{01} = p(x_0|x = x_0, a = 1)$$
$$\varepsilon_{10} = p(x_0|x = x_1, a = 0) \qquad \varepsilon_{11} = p(x_1|x = x_1, a = 1)$$

One can think of ε_{01} as the probability of starting in x_0, choosing $a = 1$ to switch, but actually ending in state x_0 as a result of unknown noise. Our goal is to create an algorithm to minimize Q^* that captures this noise.

The RL Algorithm

Algorithm 1 Unified Two Timescales Mean Field Q-learning - Tabular version

Require: T : number of time steps in a learning episode,
$\mathcal{X} = \{x_0, \dots, x_{|\mathcal{X}|-1}\}$: finite state space,
$\mathcal{A} = \{a_0, \dots, a_{|\mathcal{A}|-1}\}$: finite action space,
μ_0 : initial distribution of the representative player,
ϵ : parameter related to the $\epsilon-$greedy policy,
tol_μ, tol_Q : break rule tolerances.

Initialization: $Q^0(x, a) = 0$ for all $(x, a) \in \mathcal{X} \times \mathcal{A}$, $\mu_n^0 = \left[\frac{1}{|\mathcal{X}|}, \dots, \frac{1}{|\mathcal{X}|}\right]$ for $n = 0, \dots, T$
for each episode $k = 1, 2, \dots$ **do**
  Initialization: Sample $X_0^k \sim \mu_T^{k-1}$ and set $Q^k \equiv Q^{k-1}$
  for $n \leftarrow 0$ to $T - 1$ **do**
    Update μ:
    $\mu_n^k = \mu_n^{k-1} + \rho_k^\mu \boldsymbol{\delta}(X_n^k) - \mu_n^{k-1})$ where $\boldsymbol{\delta}(X_n^k) = \left[\mathbf{1}_{x_0}(X_n^k), \dots, \mathbf{1}_{x_{|\mathcal{X}|-1}}(X_n^k)\right]$
    Choose action A_n^k using the ϵ-greedy policy derived from $Q^k(X_n^k, \cdot)$
    Observe cost $f_{n+1} = f(X_n^k, A_n^k, \mu_n^k)$ and state X_{n+1}^k provided by the environment
    Update Q:
    $Q^k(X_n^k, A_n^k) = Q^k(X_n^k, A_n^k) + \rho^Q_{k,n,X_n^k,A_n^k}[f_{n+1} + \gamma \min_{a' \in \mathcal{A}} Q^k(X_{n+1}^k, a') - Q^k(X_n^k, A_n^k)]$
  end for
  if $\delta(\mu_T^{k-1}, \mu_T^k) \leqslant tol_\mu$ and $\|Q^k - Q^{k-1}\|_{1,1} < tol_Q$ **then**
    break
  end if
end for
return (μ^k, Q^k)

This single algorithm is capable of modeling both MFG and MFC, depending on the values of two critical ρ parameters:

$$\rho^Q_{k,n,x,a} = \frac{1}{(1 + \#|(x, a, k, n)|)^{\omega^Q}} \qquad \rho_k^\mu = \frac{1}{(1 + k)^{\omega^\mu}}$$

when we are at the kth time step. $\omega^Q \in [\frac{1}{2}, 1]$ and ω^μ can be chosen to make our problem an MFG or MFC:

$$\boldsymbol{\omega^Q < \omega^\mu} \qquad \text{for } \textbf{MFG}$$
$$\boldsymbol{\omega^Q > \omega^\mu} \qquad \text{for } \textbf{MFC}$$

where intuitively, $\omega^Q < \omega^\mu$ means that $\rho^Q_{k,n,x,a}$ will update Q faster than ρ_k^μ updates the population. This can be likened to the behavior of an MFG agent, which competitively aims to primarily minimize individual cost, adapting as the population distribution changes.

By contrast, in the context of MFC, the population distribution updates faster than the cost. This is because the 'controller' selects cooperative strategies that modify the population distribution, and then observes the resulting cost.

The Numerical Solver

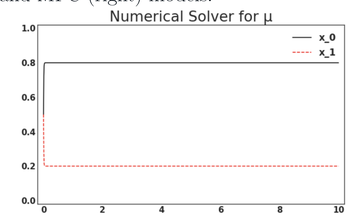
To verify the credibility of the algorithm, we developed code to numerically solve the Bellman Equation, which serves as the theoretical solution for Q. In the RL algorithm, the MFG and MFC models are only differentiated by their ω^Q and ω^μ values- values not used in the numerical solver. This means our solver will provide solutions to both models.

It should be noted that although MFG models are competitive and MFC models are cooperative, they may very well share the same optimal decision-making solution. Moreover, there may exist many parameter combinations such that no solutions exist for either model. The following graphs will show these findings in greater detail.

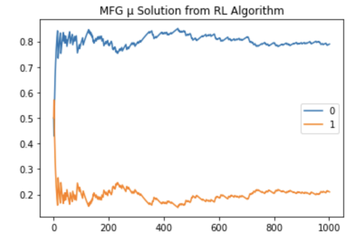
For the graphs, the Q values directly come from our cost function, $f(x, a)$, with which the only unknown is the population distribution, $\mu(x)$. Thus, comparing the convergence of $\mu(x)$ over time is sufficient to make conclusions about the accuracy of the convergence of Q.

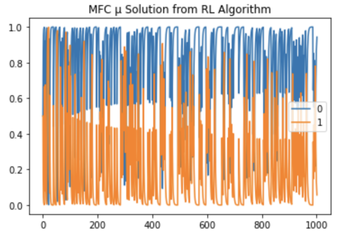
Visualization

Here we compare the numerical solution (shown on top) with both the MFG (left) and MFC (right) models:

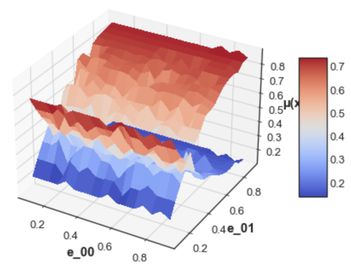


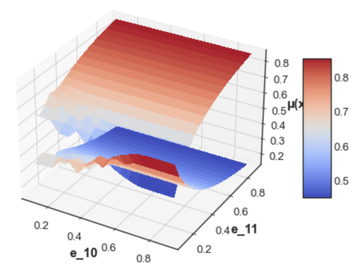

```
#Parameters
x_list = [0,1]
a_list = [0,1]
c = 1.5
gamma = 0.9
mu_initial = pd.Series([.5,.5])
ε_list = [[.15,.35],[.6,.2]]
```





The matching μ convergence for the MFG model suggests an optimal solution for minimizing cost exists. On the other hand, **the MFC model struggles to converge**, implying no solution exists under these parameters.





For each of these 3D graphs, the two surfaces represent how $\mu(x_0)$ and $\mu(x_1)$ move with changes in ε. The left graph tracks the epsilon probabilities associated with state x_0 and the right graph with those associated with x_1.

The first observation is that the right graph is much smoother than the left. Our model is structured in a way where cost is almost always minimized when the agent can get to x_0. The right graph keeps the x_0 epsilon probabilities constant which yields a constant probability matrix and hence a smoother curve. On the other hand, the left graph shows when the x_0 epsilon probabilities move, causing different probability matrices that cause more variance in the resulting population distribution.

References

- Fouque, Jean-Pierre, et al. "Unified Reinforcement Q-Learning for Mean Field Game and Control Problems." ArXiv.org, 31 May 2021, https://arxiv.org/abs/2006.13912.
- Zaman, Muhammad Aneeq uz, et al. "Oracle-Free Reinforcement Learning in Mean-Field Games along a Single Sample Path." ArXiv.org, 26 Aug. 2022, https://arxiv.org/abs/2208.11639.