Classical and Non-Classical Reinforcement Learning for Two-Player Adverserial Signaling Games

Jad Soucar (soucar@usc.edu)
USC Viterbi Daniel J. Epstein Department of Industrial
Systems Engineering 3715 McClintock Avenue, GER 240
Los Angeles, CA 90089-0193

Mentoring Organization: Project Mesa
Keywords: Quantum Games, Control Theory, Signaling Games, Reinforcement Learning

Abstract

Quantum frameworks allow researcher to analyze game theoretic interactions where uncertainty is a latent feature and deception is a prevalent strategy. In this proposal we develop the mathematical background necessary to implement quantum two player game. We focus our attention on modeling quantum agents in deceptive environments through the development of quantum circuit and the utilization of classical tools including the Hamilton-Jacobi-Isaacs equation and deep Q learning. Our end goal is understanding the mathematical building blocks of quantum agentic modeling in order to implement an open-sourced library for implementing quantum games so that users can classically simulate and analyze quantum game-theoretic interactions. We hope that such a library can make the process of developing quantum strategies more accessible and interpretable for researchers as we prepare for true quantum hardware to become readily available.

1 Adversarial Signaling Games

We describe an adversarial Signaling game as a two player game, where each player has the capacity to manipulate a shared environment. Additionally each player can observe the others' actions along with the effect of those actions on the shared environment. Hidden intention's, however, may remain obfuscated. Within this game structure there are a few features to take note of which we describe bellow along with an examples from finance, military strategy, political strategy and cyber secruity.

- **Misleading Signals:** The purpose of each players' actions is two fold. First they seek to manipulate their environment in order to create favorable conditions that allow them to maximize some objective. Second each player may take an action to manipulate not the environment but the other player's actions. For example, player A may seek to establish a intentionally deceptive repeated pattern of actions in order to lure player B into a particular action strategy, before suddenly breaking that pattern to reap a reward. This is not an unusually strategy nor is particularly novel. Take for example the mythologized Trojan war described in Homer's Iliad where the the Achaeans intentionally established an observable pattern of peace by retreating their forces and offering the Trojans a "gift". The Trojans were lulled into a belief that the threat had passed leading them to lower their defenses and giving the

Achaeans a window to launch a final strike. In other words one player took repeated actions that created a false expectation and manipulated the actions of their competitors before breaking that pattern to reap a reward.

- **Two-Stage Learning:** note that within a adversarial signaling game there is latent information asymmetry. Namely that each player has a set of hidden intentions and observable actions. if a player were to only analyze the observable actions of their competitor then they will be manipulated and eventually beat. To avoid this outcome and fate similar to the Trojans, each player must learn to do two things. They must first, learn how to detect patterns within the observable actions of their competitor. Second they must learn to evaluate those patterns and decide whether their opponent is intentionally deceiving them or whether the pattern they are observing can exploited. For example imagine two high frequency equity traders. Trader A seeks to manipulate their competitors by placing a sequence of intermittent large buy orders whih increase the price of the equity. Trader B observes the pattern and may decide buy the equity before Trader A's anticipated buy order in order to profit from the price jump induced by Trader A's periodic buy orders. However Trader A places a sells the equity while the price is high leaving Trader B with a loss. On the other hand Trader B may decide to stay out of the market by anticipating Trader A's attempts at market manipulation. Trader B may also anticipate the Trader A's intention to offload the asset and decide to short the equity before hand in order to take advantage of Trader A's hidden intention. In other words Trader B must not only detect the patterns but must also infer the hidden intention behind the pattern in order to make a "winning" decision.

- **Iterative Structure and Early Stage Interactions:** Within the adversarial signaling game each player must observe repeated actions in order to make reasonable and educated inferences about the other player. This means that the game cannot be captured by a one-shot static model such as that proposed within optimal auction theory or a prisoner's dilemma type game. However the player may be able to bring a set of pre-existing beliefs into an interaction with a player in order to avoid the "cold-start" problem. In this scenario it may advantageous to to employ meta-strategies like randomized actions to avoid the risk of one's strategy being immediately anticipated by an opponent with an accurate set of pre-existing beliefs and to functionally nullify any pre-existing beliefs or strategies an opponent may have within the first couple of interaction of the game. By denying the opponent predictable action structures in the early phase of the game, a player can force an opponent into uncertainty, delaying exploitation or even misleading them into classifying the system incorrectly. All of which force a "cold start" and wipe out any advantage an opponent may have had when they entered the game. Take for example a hacker attempting to infiltrate a server through a given port they believe to be open. Often times the first response of the server is to randomly deny or silently drop a server request from an unknown or suspicious IP addresses, which induces ambiguity. In other words the hacker's pre-existing belief that the port is open is now called into question which forces a cold start and wipes out the hacker's informational advantage.

2 Potential Formalizations

There are various mathematical tools one could use to formalize this type of two player game, however for the sake of this proposal we choose one non-classical and one classical formalization to explore. First we explore repeated quantum games and their ability to incorporate uncertainty and non-classical signal-detection strategies. Second we examine the potential for stochastic/differential games to capture continuous time learning, environment evolution, and optimal deception.

2.1 Quantum Games

The first potential approach is the use of repeated quantum games to model the two-player adversarial game. This approach was first popularized with the introduction of the quantum prisoners dilemma, which serves as a pedagogical example for how this quantum game theory applies to the setting described above. For that purpose we spend some time discussing its implementation and how it can generalize to different scenerios we might wish to model.

2.1.1 Repeated Quantum Prisoners Dilemma (QPD)

In the classical version of the quantum prisoners dilemma there are two players who can choose to either cooperate (C) or defect (D). The joint action of the two players results in a certain payoff for each player.Bellow we illustrate a potential payoff structure.

	Player B: C	Player B: D
Player A: C	(3, 3)	(0, 5)
Player A: D	(5, 0)	(1, 1)

This game structure creates a tension between deceiving your opponent or believing your opponent will do whats best of both players. Classicaly the dominant strategy is to always defect which results in a Nash equilibrium of (D,D). Note that the Nash equilibrium does not correspond with the Pareto optimal solution which is to always cooperate in order to maximize the outcome for both players. Often times as described in section 1 a players strategy may not be as black and white as cooperating or defecting. Instead of the player may opt to cooperate for a time while hiding their true intentions of lowering their opponents defense and eventually defecting. Additionaly in the classical case the two players do not interact which we is unrealistic for modeling real world game dyanamics. In the quantum prisoners dilemma (QPD) this problem is addressed by allowing players to both defect and cooperate simultaneous (i.e the QPD allows for mixed strategies and implicit communication). The game starts with an two-qubit entangled state $|\psi_0\rangle$, where $J \in \mathcal{H}$ is some unitary operator that produces an entangled state where the two players are in a superposition of both cooperating and defecting. Note that γ affects the degree to which the two players are entangled, which we will discuss in more detail bellow.

$$|\psi_0\rangle = J|00\rangle = \cos\left(\frac{\gamma}{2}\right)|00\rangle + i\sin\left(\frac{\gamma}{2}\right)|11\rangle \tag{1}$$

note that the classical strategy basis is $\{|00\rangle \leftrightarrow (C,C), |01\rangle \leftrightarrow (C,D), |10\rangle \leftrightarrow (D,C), |11\rangle \leftrightarrow (D,D)\}$. The players then apply local unitary operators to adjust the probability of defecting or cooperating. The local unitary is often times written as $U(\theta,\phi)$ where θ,ϕ are parameters that correspond to how the players choose to adjust their states.

$$U(\theta,\phi) = \begin{bmatrix} e^{i\phi}\cos\left(\frac{\theta}{2}\right) & \sin\left(\frac{\theta}{2}\right) \\ -\sin\left(\frac{\theta}{2}\right) & e^{-i\phi}\cos\left(\frac{\theta}{2}\right) \end{bmatrix} \tag{2}$$

The update step can be written simply as the application of each player's unitary matrix $U_i(\theta_i,\phi_i)$ as applied to $|\psi_0\rangle$

$$|\psi_1\rangle \leftarrow (U_1 \otimes U_2)|\psi_0\rangle \tag{3}$$

Finally we apply $J^\dagger$ to bring the outcomes back into the classical strategy basis and take a measurement to determine the outcome. Each outcome corresponds with a classical payoff outlined an a table like that described in table 2.1.1.

$$J^\dagger |\psi_1\rangle \xrightarrow[Measure]{} |xy\rangle, \quad x,y \in \{0,1\} \tag{4}$$

The affects of the infinite number of unitaries each player can apply vary and offer players classical strategies and new non-classical strategies that are not available within the classical setting. For example $U(\pi,0)$ corresponds with the bit flip operator X which swaps the probabilities of cooperating and defecting. While $U(0,0)$ corresponds to the Identity matrix which does not change the players decision probabilities. Both of these actions are classical in nature and are available to players in a classical prisoner dilemma game. Notice that in both actions references above $\theta = 0$. This gives us the generally fact that θ controls a players classical ability to adjust their personal decision probabilities, while ϕ introduces local phase which does not affect personal decision probabilities but instead can non-classically affects joint outcomes (i.e the shared environment). We provide a few examples bellow that reflect a number of these strategies.

Example 1 ($U(0,0)$ **at** $\gamma = 0$)**.** *In this case we allow $\gamma = 0$, which means that the decision probabilities of both players are not-entangled. This can be interpreted as disallowing any implicit communication between the two players while the game is in progress. Second we let $U_1 = U_2 = U(0,0) = I$, which implies that both players wish to make no changes to their decision probabilities. In this case the two players will chose (C,C) 100% of the time since $\gamma = 0$ defaulted the players to (C,C) and both players chose to stay the course.*

$$\begin{aligned} J \to |\psi_0\rangle &= |00\rangle \\ (U_1 \otimes U_2)|\psi_0\rangle &= |00\rangle \\ J^\dagger |00\rangle &= |00\rangle \end{aligned}$$

$$\implies P(00) = 1, \quad P(01) = P(10) = P(11) = 0$$

Example 2 (**Phase sabotage at** $\gamma = 0$)**.** *In this case we let $\gamma = 0$ which means that the decision probabilities of both players are not-entangled. We also set $\theta = 0$ and let $\phi \neq 0$. This implies that the players wish to make no direct changes to their decision probabilities, but instead wish to induce some sort of change to the joint probability outcome.*

$$\begin{aligned} |\psi_0\rangle &= |00\rangle \\ U_1 = U(0,0) = I, \quad U_2 = U(0,\pi/2) &= \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix} \\ (U_1 \otimes U_2)|00\rangle &= i|00\rangle \\ J^\dagger |\psi_f\rangle &= i|00\rangle \end{aligned}$$

$$P(00) = 1, \quad others = 0$$

This example illuminates that there is no penalty for adding a phase when no entanglement is present. This shows that the phase ϕ has no strategic effect unless there is entanglement — like bluffing without an audience. Example 1 and 2 highlight that when $\gamma = 0$ and there is not entanglement, the QPD reduces to a classical PD, where mixed strategies are allowed, but communication between the two parties during the course of the game is impossible.

Example 3 ($U(0,0)$ **at** $\gamma = \pi/2$)**.** *In this case we allow* $\gamma = \frac{\pi}{2}$ *which means that the decision probabilities of both players are entangled (i.e the actions of the player 1 will impact the action player 2 choose to make and vice versa). A* γ *of* $\frac{\pi}{2}$ *also defaults the players to a equal probability of achieving any of the 4 outcomes. Second we let* $U_1 = U_2 = U(0,0) = I$*, which implies that both players wish to make no changes to their decision probabilities.*

$$
\begin{aligned}
J|00\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + i|11\rangle) \\
U_1 = U_2 &= U(0,0) = I \\
(U_1 \otimes U_2)|\psi_0\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + i|11\rangle) \\
J^\dagger|\psi_0\rangle &= \frac{1}{2}(|00\rangle - i|01\rangle + i|10\rangle + |11\rangle) \\
\implies P(00) = P(01) &= P(10) = P(11) = \frac{1}{4}
\end{aligned}
$$

Example 4 (**Phase sabotage at** $\gamma = \pi/2$)**.** *In this case we allow* $\gamma = \frac{\pi}{2}$ *which means that the decision probabilities of both players are entangled. Second we let* $U_1 = U(0,0)$*, which means that player 1 wishes to make no changes to their decision probabilities, whereas* $U_2 = U(0,\pi/2)$

$$
\begin{aligned}
|\psi_0\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + i|11\rangle) \\
U_1 = U(0,0) = I, \quad U_2 &= U(0,\pi/2) = \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix} \\
(U_1 \otimes U_2)|\psi_0\rangle &= \frac{1}{\sqrt{2}}(i|00\rangle - i|11\rangle) \\
J^\dagger|\psi\rangle &= |00\rangle \\
P(00) = 1, \quad &\textit{others} = 0
\end{aligned}
$$

The interaction between the two actions is more nuanced in this case. Since if U_2 *had I we would have had equal probabilities across the full result space like in example 3. However by adjusting* ϕ*, player 2 was able to secretly shape the joint outcome to be (C,C). This models a player's ability to adjust the joint environment instead of just their own decision probabilities. This feature is unique to QPD and is inline with the features we wish to model in an adversarial signaling game.*

Example 5 (**Strategic anticipation and covert betrayal at** $\gamma = \pi/2$)**.**

$$
\begin{aligned}
|\psi_0\rangle = J|00\rangle &= \frac{1}{\sqrt{2}}(|00\rangle + i|11\rangle) \\
U_1 = U(\pi,\pi/4), \quad U_2 &= U(0,0) = I \\
(U_1 \otimes U_2)|\psi_0\rangle &= \frac{1}{\sqrt{2}}(|10\rangle - i|01\rangle) \\
|\psi_f\rangle = J^\dagger &\left(\frac{1}{\sqrt{2}}(|10\rangle - i|01\rangle)\right)
\end{aligned}
$$

$$P(10) \approx 1, \quad others \approx 0$$

Consider a case where Player 2 has established a pattern of naively applying $U(0,0)$ which maintains equal probabilities across the result space, but Player 1 anticipates this and strategically applies $U(\pi,\pi/4)$. Then player 1's action pushes the final state toward $|10\rangle$ (D, C) which maximizes Player 1's payoff while punishing Player 2. Additionally note that unlike the $\gamma = 0$ case, player 2 can't be certain that player 1 actively defected since the from their point of view the outcome was just "bad luck" since they agreed to an equal probability of each outcome. This type of action strategy is known as "covert betrayal" and offers a new dimension to the stratagem that the players can employ.

At this point note that while (QPD) only requires 4 output state, one may want to simulate a scenario where the players have more outcomes they can induce. To this end we can employ qudits which are qubits that encode more output states. We can also expand our qubit space to n qubit $|\psi_0\rangle$ which would allow for 2^n possible outcomes. We intend to explore this expansion during my work with Red Hen Labs. We've described how the QPD framework allows players to take actions that change the shared environment along with their own personal states. We've also described cases where a player can adjust their action to exploit an action pattern they may have anticipated. This begs the question of how a player can both identify patterns in their opponents and determine whether the pattern represents an intentional deception or opportunity for exploitation. To that end we discuss a few approaches in literature that have been used to expand the QPD from a single round game to an iterative game that has the capacity to capture the features of the an adversarial signaling game as discussed in section 1.

2.1.2 Belief and Action Estimation

The first task a player has after a round has come to an end is to determine the strategy their opponent may have used in order to track whether an exploitable pattern is emerging. This can be done by leveraging tools from classical optimization. Specifically a player can estimate $U_2(\theta,\phi)$ based on observed measurements by fixing their own unitary U_1 and iterating over candidate values of (θ,ϕ) for U_2. For each candidate the player can compute the candidate final state along with its corresponding measurement probabilities $P_{ab}(\theta,\phi)$

$$|\psi_f(\theta,\phi)\rangle = J^\dagger(U_A \otimes U_B(\theta,\phi))J|00\rangle \tag{5}$$

Next the player can repeatedly compute the following loss function for all candidate final states and optimize for $(\theta^*,\phi^*) = \arg\min_{\theta,\phi} L(\theta,\phi)$ using basic grid search, Bayesian optimization or even evolutionary optimization.

$$L(\theta,\phi) = \sum_{ab} \left(P_{ab}(\theta,\phi) - P_{ab}^{\text{obs}}\right)^2 \tag{6}$$

For example if the player 1 plays 10 rounds while using $U(0,0)$, and observe the following measurement frequencies:

$$P_{00}^{\text{obs}} = 0.1, \quad P_{10}^{\text{obs}} = 0.9$$

Then player 1 can sweep over candidate $U_2(\theta,\phi)$ and find that U_2 is likely $U(\pi,\pi/4)$, which strongly suggests the opponent is defecting with a phase twist, a known stealth betrayal tactic. Once a player has this information they can feed the inferred data into a reinforcement learning model in order to determine their next action (θ',ϕ'). We provide relevant background for how RL can be applied in 3. This pipeline may induce interesting strategies such as rewarding honest opponents, testing cooperates with sudden "covert" phase perturbations and even detecting patterns in behavior that can be exploited. All of these features are

ones that we sought to incorporate in order to effectively model an adversarial signaling game. Note however the variational quantum policy is a tool used in literature that may be worth investigating as an alternative to classically optimizing over the space of probability measures.

2.2 Stochastic and Differential Games

The second potential approach is to model the two-player adversarial signaling game using classical stochastic control. Specifically we make use of the Hamilton-Jacobi-Isaacs (HJI) equations which naturally arises in the context of zero-sum differential games, where each player is allowed to employ a continuous-time dynamic strategy. We first spend some time discussing the general HJI modeling framework

2.2.1 General HJI Framework

Generally the dynamic game is simple to set up and follows the following structure. First we have an environmental state $x(t)$, along with the decision strategy of the players which are often referred to as policies or controls and denoted as $u_1(t)$ and $u_2(t)$. The environment evolves as a function of its current state, and the decisions of each player.

$$dx(t) = f(x(t), u_1(t), u_2(t), t) \tag{7}$$

This general differential is natural within the context of adversarial signaling games since each player's actions change the environment and serve to signal patterns to their opponent or to perturb the environment to a state which is favorable to their own objectives. To that end we must also define an objective function which is typically defined as some functional J. The functional is composed of a running cost function L_i evaluation over the course of the game from $[0,T]$ along with a terminal cost/reward obtained at the end of the game based on the environment at time T.

$$J(x,u_1,u_2) = g(x(T)) + \int_0^T L(x(t),u_1(t),u_2(t))dt \tag{8}$$

The cost function is again an intuitive component of modeling an adversarial signaling game, since the game is objective oriented and zero sum. In other words there is some measurable quantity the the players are competing for. To that end it is one players objective to minimize J and the other players objective to maximize J. For example if Trader A and Trader B engage in a two person market, and Trader A and B have respective wealth W_A, W_B then function J will be roughly $W_A - W_B$ where Trader A is trying to maximize J and the Trader B is trying to minimize J. We know introduce the concept of the value function $V(t,x)$ which represents the best possible objective for both players if both players behave optimally. We can express $V(t,x)$ as a basic min-max problem, namely

$$V(t,x) = \min_{u_1} \max_{u_2} J(t,x,u_1,u_2) \tag{9}$$

From the value function V we can derive the PDE the controls the dynamics of the value funciton V.

Theorem 2.1. *Given a cost functional $J(x,u_1,u_2) = g(x(T)) + \int_0^T L(x(t),u_1(t),u_2(t))dt$ which is controlled by two players with respective controls u_1, u_2 the corresponding value function $V(t,x) = \min_{u_1} \max_{u_2} J(t,x,u_1,u_2)$ follows the following PDE.*

$$\frac{\partial V}{\partial t}(t,x) + \inf_u \sup_v \{L(x,u,v) + \nabla_x V(t,x) \cdot f(x,u,v)\} = 0, \quad V(T,x(T)) = g(x(T))$$

Proof. Let Δt be a small time increment. We define a dynamic programming equation which the value function satisfies by the principle of optimality of dynamic programming:

$$V(t,x) = \inf_u \sup_v \left[\int_t^{t+\Delta t} L(x(s),u,v)\,ds + V(t+\Delta t, x(t+\Delta t)) \right] \tag{10}$$

Take the Taylor expansion we find that

$$\begin{aligned} \int_t^{t+\Delta t} L(x(s),u,v)\,ds &\approx L(x,u,v)\Delta t \\ &\approx \tfrac{\partial V}{\partial t}(t,x)\Delta t + \nabla_x V(t,x)\cdot \dot{x}(t)\Delta t \\ &= V(t,x) + \left(\tfrac{\partial V}{\partial t}(t,x) + \nabla_x V(t,x)\cdot f(x,u,v) \right)\Delta t \end{aligned} \tag{11}$$

Substitute into the dynamic programming equation:

$$\begin{aligned} V(t,x) &= \inf_u \sup_v \left[L(x,u,v)\Delta t + V(t,x) + \left(\tfrac{\partial V}{\partial t}(t,x) + \nabla_x V(t,x)\cdot f(x,u,v) \right)\Delta t \right] \\ &= V(t,x) + \inf_u \sup_v \left[\left(L(x,u,v) + \nabla_x V(t,x)\cdot f(x,u,v) + \tfrac{\partial V}{\partial t}(t,x) \right)\Delta t \right] \end{aligned} \tag{12}$$

Thus, we arrive at the Hamilton–Jacobi–Isaacs equation:

$$\frac{\partial V}{\partial t}(t,x) + \inf_u \sup_v \{ L(x,u,v) + \nabla_x V(t,x)\cdot f(x,u,v) \} = 0, \quad V(T,x(T)) = g(x(T)) \tag{13}$$

□

note that a similar HJI equation can be derived for a stochastic environment, although because the problem is a closed loop control problem the analytical solution is difficult to derive if the stochastic component is also controlled by u_1 and u_2. We can solve this system numerically and observe whether or not deception plays a part in either player's strategy. We can, however build deception into the model directly, which we discuss in the next section.

2.2.2 HJI for Two-Player Adversarial Signaling Games

The HJI framework as discussed above lends itself naturally toward signaling games, however we can make the connection more explicit by redefining the full state space as $z(t)$, where $x(t)$ is the external system state, b_i are each players beliefs about the other players likelihood of currently carrying out a deceptive action strategy, and d_i are each players current level of deception or commitment to a pattern. For example if d_1 is high then player 1 will likely switch their action strategy soon to catch player 2 off guard. As a result player 2 will likely increase b_2 to reflect player 1's deception.

$$z(t) = \begin{bmatrix} x(t) & b_1(t) & b_2(t) & d_1(t) & d_2(t) \end{bmatrix} \tag{14}$$

Each player needs some way of choosing their actions which we again as the control $u_{i,x}$, along with some controls over their beliefs and levels of deception which we denote as $u_{i,b}$ and $u_{i,d}$ respectively. So the updated dynamics are as follows.

$$dx(t) = f_x(x, u_{1,x}, u_{2,x}, d_1, d_2) \tag{15}$$

$$d(d_i) = f_d(x, d_i, u_{i,d}) \tag{16}$$

$$d(b_i) = f_d(x, b_i, u_{i,b}) \tag{17}$$

As a result we can rewrite the HJI equation to account for the enlarged state space which now includes the belief and hidden intention states for each player.

$$\frac{\partial V}{\partial t}(t,z)+\inf_{u}\sup_{v}\{L(z,u,v)+C_d(d_A,b_B)-R_b(b_A,b_B)+\nabla_z V(t,z)\cdot f(z,u,v)\}=0,\quad V(T,z(T))=g(x(T)) \tag{18}$$

$$f(z,u,v)=\begin{bmatrix} f_x(x,u_x,v_x,d_A,d_B) \\ f_b^A(b_A,d_B,v_b) \\ f_b^B(b_B,d_A,u_b) \\ f_d^A(d_A,u_d) \\ f_d^B(d_B,v_d) \end{bmatrix} \tag{19}$$

note that each the dynamics themselves including the objective function design are left up to the user and will be explored as apart of this project. It is well known however that HJI equations are difficult to solve analytically and instead are often evaluate through numerical techniques like reinforcement learning (RL). To that end we provide some technical background on RL along with a few RL architectural choices we may choose to implement during the course of this project.

3 Reinforcement Learning

Before we begin our discussion of Q learning, we will start with an exposition of the two primary categories of RL: model-based and model-free. Model-based RL systems create a model of the environment, which includes the dynamics of how actions lead to subsequent states and rewards. Essentially, it predicts the future states and rewards for actions taken from a given state. Take for example a gambler who is given a probability function $P(s',s,a)$ that provides explicit probabilities as to whether the actor will win or lose a gamble. In other words, the gambler queries the environment for its transition dynamics. The second approach is model-free, which is where a system learns a policy or value function directly from interactions with the environment without constructing an explicit model of the environment's dynamics. The system learns what to do by trial and error, adjusting its actions based on the rewards received.

With the general background of RL out of the way, its important to note that Q learning is a model-free training scheme. That means that Q learning does not rely on an environment that can produce exact transition probabilities and must instead use trial and error of near-optimal actions in order to learn the optimal policy. In order to formalize this type of training scheme we first define the quality function $Q(s,a)$, which tells you the joint value/quality of taking an action a given a current state s.

To formally define $Q(s,a)$ we first introduce some notation. Let $R(s',s,a)$ be the reward of transitioning from state s to s' through action a, γ be the discount factor of future reward, and $V(s')$ be the expected value over all possible future rewards given a current state s'. With that we define $Q(s,a)$ as

$$\begin{aligned} Q(s,a) &= \mathbb{E}\left(R(s',s,a)+\gamma V(s')\right) \\ &= \sum_{s'} P(s'|s,a)\left(R(s',s,a)+\gamma V(s')\right) \end{aligned} \tag{20}$$

In other words, Q is the expected sum of the instantaneous reward of the state-action pair (s,a) along with the discounted future rewards of being at a new state s' brought on by (s,a). Using the quality function $Q(s,a)$ we can define an optimal policy π and value function $V(s)$, that considers which action a is optimal and what the expected reward from taking that action is.

$$V(s) = \max_a Q(s,a) \tag{21}$$

$$\pi(s) = \arg\max_a Q(s,a) \tag{22}$$

We've defined what a q value is and how to construct the quality function $Q(s,a)$, but now we define a recursive equation to update $Q(s,a)$ as the agent learns through trial and error.

$$Q^{\text{new}}(s_k,a_k) = Q^{\text{old}}(s_k,a_k) + \alpha\left(r_k + \gamma \max_a Q^{\text{old}}(s_{k+1},a) - Q^{\text{old}}(s_k,a_k)\right) \tag{23}$$

Let's dissect this update equation. As we engage in trial and error learning, we update our $Q(s,a)$ by slowly nudging our q values up or down by a factor of the difference between the actualized current reward r_k in addition to the best possible future reward $r_k + \gamma \max_a Q(s_{k+1},a)$ (TD-Target) and our predicted reward $Q^{\text{old}}(s_k,a_k)$). This difference is sometimes referred to as the "Temporal Difference (TD) error". We also note that α in this case is the learning rate.

$$\underbrace{\overbrace{r_k + \gamma \max_a Q(s_{k+1},a;\theta^-)}^{\text{TD Target}} - Q^{\text{old}}(s_k,a_k;\theta)}_{\text{TD Error}} \tag{24}$$

We also note that when calculating the TD error, the agent calculates the future reward by looking one step into the future. Naturally, the degree to which the agent looks into the future can be modified using the following updated scheme which looks n steps into the future. This process is known as TD-N learning.

$$Q^{\text{new}}(s_k,a_k) = Q^{\text{old}}(s_k,a) + \alpha(r_k + R_\Sigma^{(n)} - Q^{\text{old}}(s_k,a_k)) \tag{25}$$

$$R_\Sigma^{(n)} = (\sum_{j=1}^{n} \gamma^j r_{k+j}) \tag{26}$$

where r_{k+j} is the reward derived from future applications of the agent's chosen policy π. Now we discuss the process of recasting Q Learning into training schemes that incorporate deep learning methods like neural networks. This is especially useful in settings where the state space is too large to reasonably store all q values in a Q table. Here, a deep learning approach seeks to parameterize $Q(s,a)$ as dependant on some weights θ such that

$$Q(s,a) \approx Q(s,a,\theta) \tag{27}$$

In order to optimize for the parameters θ within a Q Learning framework we use a loss function that is eerily similar to the TD error defined above.

$$\mathcal{L} = \mathbb{E}[(r_k + \gamma \max_a Q(s_{k+1},a,\theta) - Q(s_k,a_k,\theta))^2] \tag{28}$$

We can also choose to recast n-step temporal difference learning framework (TD-N) described above into the following neural network loss function.

$$\mathcal{L} = \mathbb{E}[(r_k + R_\Sigma^{(n)} - Q(s_k,a_k))^2] \tag{29}$$

With the loss function properly defined, we can move on to the neural network architecture, and how it allows us to approximate the Q function. In practice, we find that the architecture of the network varies based on the use case. For example a team of DeepMind engineers in 2013 coupled the loss function described above with several convolutions layers and fully connected layers. The inputs were several consecutive frames for the game, which represented the agent's state. And the output was one of several possible action that the agent could take. Regardless of the architecture used the value contained in each output node approximates the value $Q(s,a)$ for the associated (s,a) which corresponds to the network's $(\text{Input}, \text{output})$ pair. In practice, we follow the ε-greedy policy to catalyze off-policy actions by introducing randomness. Formally ε-greedy is implemented by replacing the $\max_a Q(s_{k+1},a)$ term with $Q(s_{k+1},\tilde{a}_{k+1})$ where

$$\tilde{a}_{k+1} = \begin{cases} \text{random action from } A(s_t), & \text{with probability } \varepsilon \\ \arg\max_a Q(s_t,a), & \text{with probability } 1-\varepsilon \end{cases} \tag{30}$$

As training continues we slowly take $\varepsilon \to 0$ using a simulated annealing strategy. The reason the ε-greedy strategy can be beneficial is that it allows the agent to explore the space of possible actions freely at the beginning of the training process while also emphasizing exploitation of the agent's accumulated knowledge toward the end of training. Please note that for the purpose of feeding the state into the deep Q network, we must first normalize by applying a soft max function to each column of the matrix s_t and then flatten the matrix. We've defined the DQN loss function but in practice the loss is implemented using various approximations. First the expectation in the Q Learning loss function is approximated using experience replay by sampling a mini-batch of experiences from the replay buffer. Given a replay buffer D that contains experiences (s,a,r,s'), a mini-batch of N experiences $B = \{(s_i,a_i,r_i,s_i')\}_{i=1}^N$ is sampled uniformly at random from D. Using this strategy we can approximate the loss function as follows.

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left[\left(r_i + \gamma \max_{a'} Q(s_i',a';\theta) - Q(s_i,a_i;\theta)\right)^2\right] \tag{31}$$

The next adjustment practitioners use is the integration of the the Huber loss. This is designed to make our loss function quadratic for small values of the error and linear for large values of the error. The parameter δ effectively determines the sensitivity of the loss function to outliers. Previously we defined TD Error, which for the sake of integrating the Huber loss we'll equate to δ. This final improvement yields the following loss function.

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} L(\delta) \tag{32}$$

where

$$\delta = \left(r + \gamma \max_{a'} Q(s',a')\right) - Q(s,a) \tag{33}$$

$$L(\delta) = \begin{cases} \frac{1}{2}\delta^2 & \text{for } |\delta| \leq 1, \\ |\delta| - \frac{1}{2} & \text{otherwise.} \end{cases} \tag{34}$$

Lastly, in practice the Adam optimizer is used to minimize the loss function. This is an adaptive optimization algorithm used in training our neural networks. It incorporates momentum to accelerate convergence in relevant directions and introduces bias correction to improve parameter update accuracy. By computing adaptive moments of gradients, Adam scales updates effectively and ensures numerical stability with a small constant in the denominator. These combined features make Adam a robust and efficient choice for optimizing our neural networks.

4 Goals and Objectives

The first course of action, is naturally to meet with my two mentors, to discuss any other potential collaborators within the Red Hen ecosystem along with finalizing the proposed schedule outlined bellow.

Goal 1: To Develop a Theoretical Framework

- **Objective 1.1:** Conduct a literature review into quantum game theory with an emphasis on quantum prisoners dilemma and variational quantum policy updating.
- **Objective 1.2:** Conduct a literature review into Hamiltonian-Bellman-Isaacs equations along with permissible formulations and numerical solution methods.

Goal 2: To Construct Computational Models

- **Objective 2.1:** Develop a quantum computational model that captures the core features of an adversarial signaling game like two-stage learning and intentional deception.
- **Objective 2.2:** Construct a HJI formulation of the adversarial signaling game and produce analytical results along with a reinforcement learning based numerical solver.

Goal 3: To Validate and Refine the Models

- **Objective 3.1:** Design and execute a series of simulations to test the models' in a test setting, then compare the results of the quantum and HJI models.
- **Objective 3.2:** Refine the models based on simulation outcomes and feedback, enhancing their accuracy and applicability to a broader range scenerios.
- **Objective 3.3:** Publish the code to collaborators of the Red Hen Lab to receive feedback about the model construction, then make relevant updates.
- **Objective 3.4: Midterm Evaluation.** From July 8 to July 12 finalize the existing code base and submit a report to the Red Hen Lab Mentor.

Goal 4: To Explore Applications and Implications

- **Objective 4.1:** Investigate the applicability of the developed models to various setting such as military, cyber-security and financial settings
- **Objective 4.2:** Analyze the efficacy of the model in competing against real world players, such as performing paper trading in a financial market.

Goal 5: Dissemination of Findings

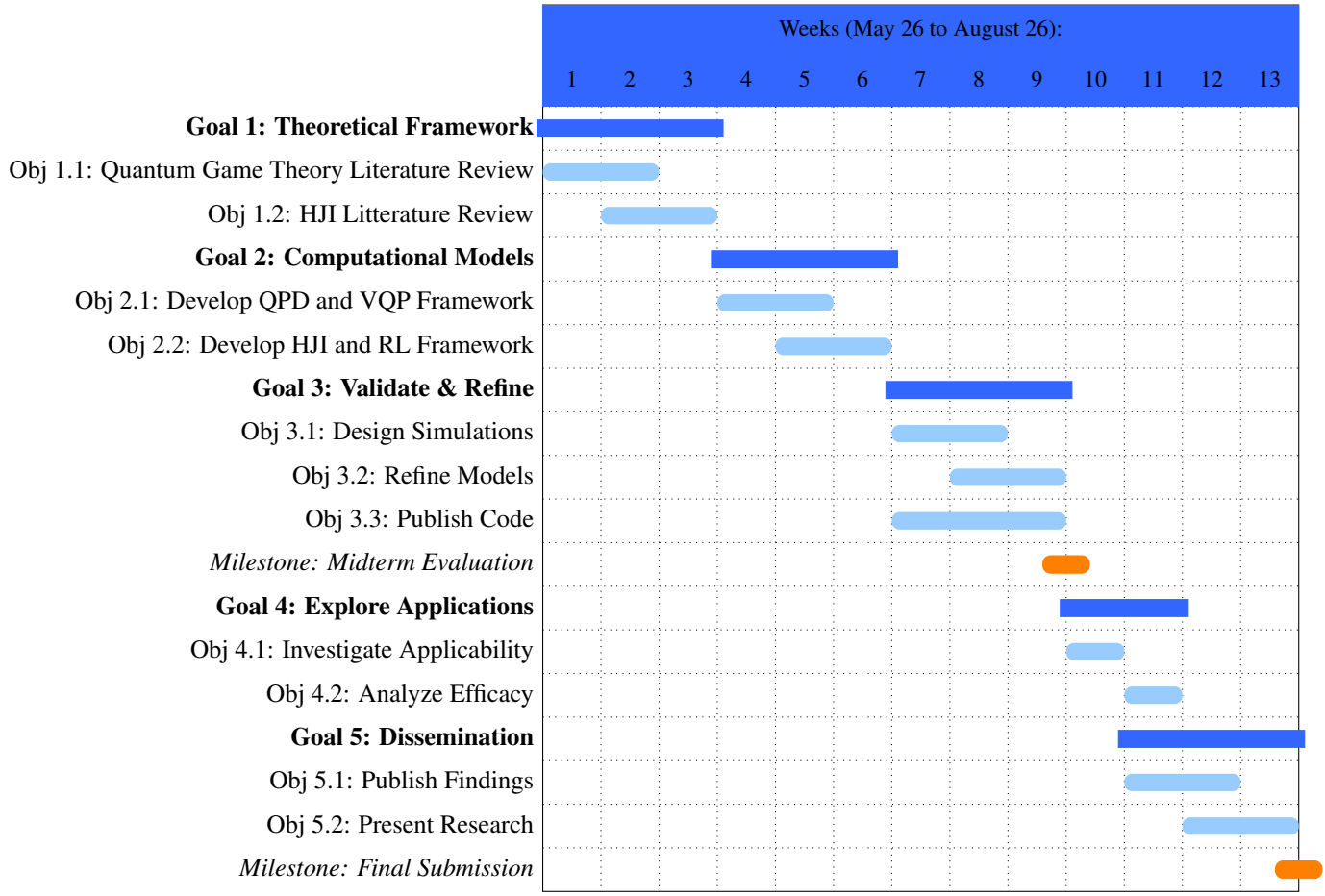
- **Objective 5.1:** Publish the findings in peer-reviewed journals to share the advancements with the academic community. Also publish the code using an open access platform like Github for anyone to use.

- **Objective 5.2:** Present the research at a conference to foster discussions and collaborations surrounding the open-sourced code base with other researchers in the field.
- **Objective 5.3: Final Week of GSoC.** From August 19-26 prepare and submit a final report and code base for submission to my mentors.

Expected Outcomes

We hope to provide a validated quantum and classical model that is capable of simulating adversarial signaling games by capturing core features such as intentional deception and two-stage learning. By the end of GSoC, we also would like to compare the efficacy of the quantum and classical models and to publish our results to the broader academic community. We also intend to develop a easy to use python library so researchers in the field can deploy simulations models for adversarial signaling games in their respective domains.

5 Tentative Timeline



References

1. Briola, Antonio, et al. Deep Reinforcement Learning for Active High Frequency Trading, 19 Aug. 2023, arxiv.org/abs/2101.07107.

2. Brunton, Steven Lee, and José Nathan Kutz. Data-Driven Science and Engineering: Machine Learning, Dynamical Systems and Control. Cambridge University Press, 2021.

3. Dai, Zhenwen, et al. In-Context Exploration-Exploitation for Reinforcement Learning, 11 Mar. 2024, arxiv.org/abs/2403.06826.

4. De Asis, Kristopher, et al. Multi-Step Reinforcement Learning: A Unifying Algorithm, 11 June 2018, arxiv.org/abs/1703.01327.

5. Hill, R. Carter, et al. Principles of Econometrics. Wiley, 2018.

6. Luo, Yunfei, and Zhangqi Duan. Agent Performing Autonomous Stock Trading under Good and Bad Situations, 6 June 2023, arxiv.org/abs/2306.03985.

7. Mnih, Volodymyr, et al. Playing Atari with Deep Reinforcement Learning, 19 Dec. 2013, arxiv.org/abs/1312.5602.

8. Pricope, Tidor-Vlad. Deep Reinforcement Learning in Quantitative Algorithmic Trading: A Review, 31 May 2021, arxiv.org/abs/2106.00123.

9. Ravichandiran, Sudharsan. Deep Reinforcement Learning with Python: Master Classic RL, Deep RL, Distributional RL, Inverse RL, and More with Openai Gym and Tensorflow. Packt, 2020.

10. Singh, Prabhsimran. "PSKRUNNER14/Trading-BOT: Stock Trading Bot Using Deep Q Learning." GitHub, 31 Dec. 2020, github.com/pskrunner14/trading-bot/.

11. Sutton, Richard S., and Andrew Barto. Reinforcement Learning: An Introduction. The MIT Press, 2020.

12. Eisert, Jens, Martin Wilkens, and Maciej Lewenstein. "Quantum Games and Quantum Strategies." Physical Review Letters, vol. 83, no. 15, 1999, pp. 3077–3080. https://doi.org/10.1103/PhysRevLett.83.3077.

13. Meyer, David A. "Quantum Strategies." Physical Review Letters, vol. 82, no. 5, 1999, pp. 1052–1055. https://doi.org/10.1103/PhysRevLett.82.1052.

14. Beer, Kati, et al. "Training Deep Quantum Neural Networks." Nature Communications, vol. 11, 2020, 808. https://doi.org/10.1038/s41467-020-14454-2.

15. Cerezo, M., et al. "Variational Quantum Algorithms." Nature Reviews Physics, vol. 3, 2021, pp. 625–644. https://arxiv.org/abs/2012.09265.

16. Jerbi, Sofiene, et al. "Variational Quantum Policies for Reinforcement Learning." Quantum Machine Intelligence, vol. 4, no. 1, 2022. https://arxiv.org/abs/2103.05577.

17. Skolik, Andrea, et al. "Quantum Agents in the Gym: A Variational Quantum Algorithm for Deep Q-Learning." Quantum Machine Intelligence, vol. 3, no. 5, 2021. https://arxiv.org/abs/2103.15084.

18. Dong, Daoyi, et al. “Quantum Reinforcement Learning.” IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 38, no. 5, 2008, pp. 1207–1220. https://doi.org/10.1109/TSMCB.2007.914849.

19. Briegel, Hans J., and Gemma De las Cuevas. “Projective Simulation for Artificial Intelligence.” Scientific Reports, vol. 2, 2012, 400. https://doi.org/10.1038/srep00400.

20. Li, Jiaming, et al. “A Deep Reinforcement Learning Approach for Quantum Gate Control.” Machine Learning: Science and Technology, vol. 1, no. 3, 2020. https://arxiv.org/abs/1701.08213.

JAD SOUCAR

EDUCATION

UCLA, B.S Mathematics
September 2020 – June 2024
- **8x Dean's Honor List**, UCLA Mock Trial Team, Data Science Union

USC, M.S Mathematical Finance
August 2024 – May 2026 (Expected)

USC, PhD Industrial & Systems Engineering w. Concentration in Statistics and Machine Learning
August 2024 – May 2028 (Expected)
- **Recipient of Annenberg Fellowship**

RESEARCH

Capital One, Research Fellow
December 2024 – Present – Los Angeles
- Received the award to support my research into robust optimization algorithms with Professor Johanes Royset. Work includes designing fault-tolerant machine learning loss functions and Rockafellians to help identify credit fraud with noisy or overcomplete datasets.

Institute of Pure and Applied Mathematics, Quantum Algorithms Researcher
June 2024 – Present – Los Angeles
- Conducting research with Dr. Jeff Marshal and Dr. Namit Anand at NASA Ames to develop efficient algorithms for the classical simulation of high depth noisy quantum circuits. Work included high performance simulation using multiprocessing and CUDA.
- Our results included the only practically situatable noisy quantum circuit of depth 10,000. Which is 700x the current state of the art

UCLA, MFG Lab, Mean Field Games Researcher
June 2023 – June 2024 – Los Angeles
- Conducting research under Professor Wilfrid Gangbo to develop a multi-population mean field game model of inter-bank lending to study the relationship between different population-risk profiles and system failure.

Red Hen Labs, Optimal Control Researcher
March 2023 – May 2024 – Los Angeles
- Conducting research under Professor Francis Stein involving the use of model predictive control (MPC) with cognition constraints to simulate prospection
- Work involves python implementation of MPC systems and cognitive science research/data fitting to determine model parameters

KAUST, NUMPDE Lab, Mixed Finite Elements Researcher
April 2023 – July 2023 – Jeddah, Kingdom of Saudi Arabia
- Working under Professor Daniele Boffi, on the numerical approximation of systems of partial differential equations using mixed finite element methods
- Implementation of a fictitious domain formulation with distributed Lagrange multiplier for fluid structure interaction problems.
- Compiled work into a standard and Mixed Finite Element Python Library

UCLA Communication News Archive, Machine Learning Engineer
August 2022 – Present – Los Angeles
- Working with Professor Tim Groeling on using single-linkage clustering algorithms to detect show boundaries in the UCLA's Rosenthal News Collection for the purpose of splicing news footage for future communications research.
- Developing a transcription and translation pipeline for non-English news footage for linguistics, communications, and public policy research with the Semel Institute for Human Behavior

UCLA, Paris Labs, Stochastic Optimization Researcher
October 2020 – June 2022 – Los Angeles
- Conducted research under Professor Mathieu Bauchy involving ML, and stochastic optimization for predicting the properties of amorphous materials.
- Work involved cleaning datasets, feature engineering, feature selection, applying different ML models and tuning their hyperparameters.
- Built a gradient decent boosted constrained particle swarm optimization algorithm with simulated annealing that could generate glass compositions with specified properties.

INDUSTRY EXPERIENCE

Google, Summer of Code Participant
May 2024 – August 2024 – Los Angeles, CA
- Used Deep Reinforcement learning and quantum circuits to model the probabilistic behavior of Stentor Roseli. Developed and implemented a novel quantum circuit to simulate protist decision making. Achieved 93% accuracy in predicting protist action.
- Implemented and tested Vanilla DQN, Target DQN, and Double DQN using OpenAI Gym.

UBS, Wealth Management/Data Analyst Intern
December 2022 – April 2024 – Los Angeles, CA
- Worked directly with wealth managers in the Pasadena office to build NLP powered pipelines to scrape news articles for client prospects.
- Built a decision tree-based model to rank client prospects by "best-fit" to UBS

Sentient.io, Machine Learning Researcher
July 2022 – January 2023 – Singapore, SG
- Sentiment is an AI API service provider based in Singapore.
- Read literature on and evaluated the performance of audio emotion recognition AI by developing validation datasets and developing testing pipelines
- Implemented an English to Phonetic (IPA → Arphabet) Finite State Transducer for use in text to speech models.
- Implemented a Human Pose Recognition training and inference pipeline with depth recognition. Deployed the model to docker and developed a API wrapper.

Origgin Venture Capital, Data Analyst Intern
June 2022 - August 2022 – Singapore, SG
- Origgin Venture is a fund focused on supporting/building startups around unique IP
- Used zero-shot learning, and various statistical analysis tools to identify trends in patent data and built automated process to produce weekly reports on emerging fields of IP. My software helped identify 20+ promising university patents in the green energy sector

Vulcan Value Partners, Machine Learning Engineer Intern
June 2021- October 2021 – Birmingham, AL

- Vulcan Value Partners are a value-oriented hedge fund with 17 Billion AUM.
- Worked on OCR, dependency parsing, topic modeling, and zero-shot learning for the purposes of harvesting, validating, and categorizing data from SEC filings. Worked with Dask, multiprocessing, threading, and CUDA to speed up pipelines.

Duffl, Project Manager, Data Analyst
October 2020 – March 2022 – Los Angeles

- Lead a team of 8 in using deep learning and customer data to build the app's recommendation engine from the ground up.
- Used NLP, and custom Bayesian autoregressive models to better predict supply needs.
- Testing algorithms to solve multiple traveling salesman problems for delivery optimization

TEACHING EXPERIENCE

Hodis Learning, Math Instructor
January 2025 – Present

- Teaching students ranging from kindergarten to 12th grade. Topics including Geometry, Calculus, Pre-Calculus, Statistics, and Arithmetic

UCLA, Teaching Assistant
September 2023 – January 2024

- Teaching assistant for MATH 32B: Calculus of Several Variables. The course covers topics related to integration in several variables, including line and work integrals, surface and flux integrals, and more.

Bruin Corps, Instructor
August 2022 – December 2022

- Taught mathematics classes to underserved school in Los Angeles, and helped establish a personalized tutoring program for students interested in extra-curricular math topics.

Global Mock Trial, Founder
August 2023 – June 2024

- Global Mock Trial is non-profit dedicated to proving public speaking and due process education. We've partnered with 20+ schools in China, Hong Kong, and South Korea, and have raised 20k.

Mayer Cressy, Speech Instructor
July 2020 – December 2022

- Developed curriculum and taught classes in Mock Trial and Speech & Debate. Class sizes ranged from 10-30 students

ORIGINAL WORK / PRESENTATIONS

Published

- **Interatomic potential parameterization using particle swarm optimization: Case study of glassy silica**
  - Journal of Chemical Physics, 2022 (https://doi.org/10.1063/5.0041183)
- **Cognitive Dynamics of Advertising**
  - Intelligent Systems Conference, CogSci2024, 2024 (https://escholarship.org/uc/item/7813v5n5)
- **Quantum Informational Model for Protist Decision Making**
  - Intelligent Systems Conference, CogSci2024, 2025

Invited Talks

- **CogSci 2024 (Rotterdam, Netherlands) –** July 26, 2024
  - Sesson**:** Dynamical Systems
  - Title: "Modeling Trust Based Markets"
- **Joint Mathematical Conference (Seatle, USA) –** January 10, 2024
  - Session: Topics in Mathematical Physics
  - Title: "Classical Simulation of Non-Clifford Noise Channels"

Posters

- **Joint Mathematical Conference (Seatle, USA) –** January 10, 2024
  - Session: PME Contributed Session
  - Title**:** "Simulating Quantum Circuits with Non-Clifford Noise using Stabilizer Formalism"
- **CogSci 2025 (San Fransico, USA) –** July 30, 2025
  - Session: Contributed Paper Session
  - Title: "Quantum Information Model for Protist Decision Making"

Pre-Print

- Multi-Population Mean Field Game representation of the Aiygari Income Model
- A Topological Model of Prospection using Model Predictive Control with Cognition Costs
- Risk Averse Deep Q Learning Loss Functions
- Efficient Classical Simulations of Non-Clifford Quantum Noise
- Dual-Boosted Dynamic Programming for Zero Norm Optimization in Linear Time

Awards & Honors

- **2025** Recognized as a Capital One Fellow in AI & Finance
- **2024** Recognized as an Anneberg Fellow at USC
- **2024** American Mock Trial Association All-American (Top 0.5%)
- **2023** American Mock Trial Association National Champion
- **2023** American Mock Trial Association All-American (Top 0.5%)
- **2022** UCI Hackathon Best AI Application
- **2022** American Mock Trial Association 2nd Place (National)
- **2021** American Mock Trial Association 3rd Place (National)
- **2021** Accenture Case Competition 2nd Place
- **2019** California State Legislature Certificate of Recognition
- **2019** SoCalGas Energy Award
- **2018** ASU Walton Sustainability Award
- **2018** PECG James E. Roberts Award