CLEANing the Reward: Counterfactual Actions Remove Exploratory Action Noise in Multiagent Learning

Chris HolmesParker
Parflux LLC
Salem, OR
chris@parflux.com

Matthew E. Taylor
Washington State University
Pullman, WA
taylorm@eecs.wsu.edu

Adrian Agogino
University of California at Santa Cruz
Mountain View, CA
Adrian.K.Agogino@nasa.gov

Kagan Tumer
Oregon State University
Corvallis, OR
kagan.tumer@oregonstate.edu

***Abstract*—Coordinating the joint-actions of agents in cooperative multiagent systems is a difficult problem in many real world domains. Learning in such multiagent systems can be slow because an agent may not only need to learn how to behave in a complex environment, but also to account for the actions of other learning agents. The inability of an agent to distinguish between the true environmental dynamics and those caused by the stochastic exploratory actions of other agents creates noise in each agent's reward signal. This learning noise can have unforeseen and often undesirable effects on the resultant system performance. We define such noise as *exploratory action noise*, demonstrate the critical impact it can have on the learning process in multiagent settings, and introduce a reward structure to effectively remove such noise from each agent's reward signal. In particular, we introduce two types of Coordinated Learning without Exploratory Action Noise (CLEAN) rewards that allow an agent to estimate the counterfactual reward it would have received had it taken an alternative action. We empirically show that CLEAN rewards outperform agents using both traditional global rewards and shaped difference rewards in two domains.**

I. Introduction

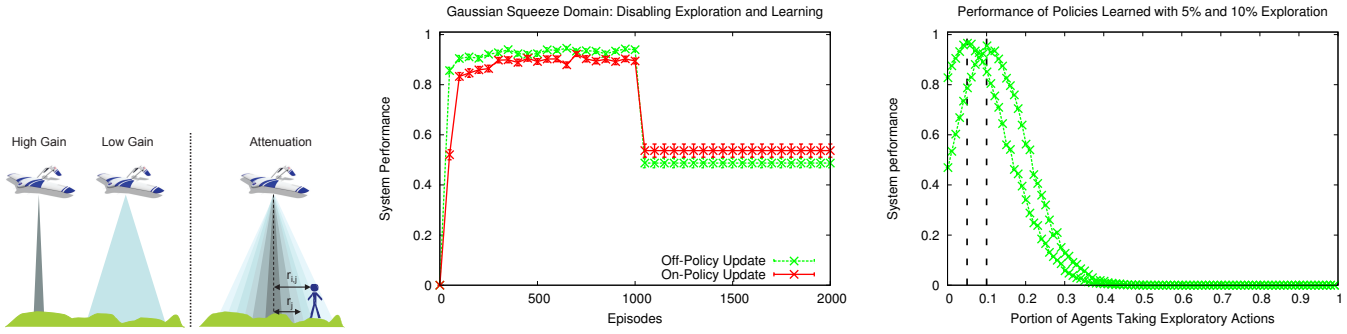
Learning in large multiagent systems is a critical area of research with applications including controlling teams of autonomous vehicles [1], managing distributed sensor networks [2], [3], and air traffic management [4]. A key difficulty of learning in such systems is that the agents in the system provide a constantly changing background in which each agent needs to learn its task. As a consequence, agents need to extract the underlying reward signal from the noise of other agents acting within the environment. This learning noise can have a significant and often detrimental impact on the resultant system performance. In this work, we first define *exploratory action noise* present in multiagent systems and then introduce Coordinated Learning without Exploratory Action Noise (CLEAN) rewards which are designed to promote coordination while removing *exploratory action noise* agents' reward signals.

There are currently two popular ways for agents to account for each other within decentralized multiagent systems: 1) explicitly modeling the other agents, and 2) treating agents as a part of the environment. Agent modeling techniques have been shown to work well in a number of settings, but they quickly become intractable as scaling increases [5], [7], [8]. Other issues arise when agents are treated as a part of the environment (e.g., *exploratory action noise*), and their exploratory actions are seen by other agents as stochastic environmental dynamics. The inability of agents to distinguish the true environmental dynamics from those caused by the stochastic exploratory actions of other agents creates noise on each agent's reward signal. This problem cannot simply be addressed by turning off exploration and acting greedily (this has been repeatedly shown to result in poor performance as agents always exploit their current knowledge which is frequently incomplete or inaccurate [9]). We address this by introducing CLEAN rewards which are designed to effectively remove much of the learning noise caused by agents taking exploratory actions.

The key innovation of our approach is that agents never explicitly take exploratory actions. Instead, exploration is accomplished by agents privately computing a "counterfactual" reward they would have received had they taken an exploratory action. Here, agents utilize a model of the system reward function and the CLEAN shaping structure in order to calculate the counterfactual rewards for actions not taken and these rewards are used to update their policies.[1] Only an agent's non-exploratory action is seen by other agents — this exploration through counterfactual reasoning does not result in noised added to the system. This paradigm promotes agent-to-agent coordination, while maintaining the exploration needed during learning (off-policy counterfactuals provide agents with rewards that approximate the impact of changing their current policies). Through using off-policy exploration, learning with CLEAN rewards effectively remove the exploratory action noise associated with learning, simplifying the learning problem for agents and improving scalability.

The primary goal of this paper is to demonstrate the impact of exploratory action noise on the multiagent learning process and to provide an approach to reduce the effect of this problem. To demonstrate the effectiveness of CLEAN, we assume agents have access to an accurate model of the system

[1] In this work, to demonstrate the ability of CLEAN rewards to remove exploratory action noise, we provided agents with an accurate model of the system reward function. In the future, reward modeling techniques will be utilized by agents to construct approximate models of the system level reward, enabling CLEAN rewards to be used in real world domains [10].



(a) (b) (c)

Fig. 1: a) The gain of an antenna in the UAVCN domain can trade off signal power with coverage. Signal strength also depends on how far the customer is from the center of the signal cone. b) When agents stop exploring in the GSD domain (episode 1000), system performance decreases due to exploratory action noise. Error bars show the standard error over 100 trials. c) After agents train with some ϵ exploration rate in the GSD, their policies are fixed. These agents policies perform best when a percentage of agents in the environment are forced to act randomly match the previous exploration rate.

reward, which they can use for internal reward calculations. This may not be a realistic assumption in some real-world domains and we plan to extend this work to learn approximate reward models in the future [10]. By demonstrating the success of CLEAN shaping rewards with a known reward model, this work takes a critical first step towards applicability in all domains, whether or not a model of the system reward is known.

The contributions of this work are to:

1) Separate environmental noise from noise caused by the exploratory actions of agents by defining exploratory action noise and show its impact on learning within multiagent systems,
2) Introduce two variations of Coordinated Learning without Exploratory Action Noise (CLEAN) rewards that promote coordination and remove the exploratory action noise associated with multiagent learning,
3) Refine the CLEAN learning algorithm to allow batch updating of these counterfactual actions, and
4) Empirically demonstrate the benefits of CLEAN rewards in two domains with 100–1000 independent agents.

II. Background and Related Work

In traditional single-agent reinforcement learning, an agent executes exploratory actions to update its policy, balancing exploration and exploitation [9], [11]. In multiagent systems, exploratory actions add stochasticity to an environment, adding noise to the learning process [12]. Agents therefore attempt to adapt their policies to account for not only the environmental dynamics, but also the exploratory actions of other agents. Such learning noise may have unpredictable and undesirable effects on the resultant system performance in multiagent systems, as we show in Section IV.

There are two common approaches to address the stochasticity of agents during learning: agent modeling and treating agents as a part of the environment. Extensive work has been done with agent modeling but such approaches become intractable as the number of agents increase [8], [13], [14]. Treating agents as a part of the environment can significantly impact both learning convergence and learning performance [12], [15], [16]. In one approach, the "Win Or Learn Fast" (WOLF) framework adjusts the learning rate of individual agents based upon the stochasticity of the environment due to other agents (effectively controlling how rapidly agents change their policies) [15], [17], [18]. Although these techniques have been shown to work well in many situations, they do not explicitly allow the agents to remove the impact of exploratory action noise from their reward signals.

A. Learning and Notation

In this paper, only multiagent stateless problems are considered, and inter-agent communication is not allowed.[2] An agent in such a setting can use an off-line or on-line update mechanism. The first, a simplification of Q-learning, is

$$Q(a_t) \leftarrow Q(a_t) + \alpha[r_{t+1} + \gamma \max_a Q(a) - Q(a_t)]$$

where α is the learning rate, γ is a discount factor, r_{t+1} is the reward provided to this agent by the environment for it's action at timestep t, a_t is the action executed by the agent on timestep t, and $Q(a_t)$ is the estimated return for executing action a on timestep t [9]. The on-line update, based on Sarsa, uses the action executed by the agent rather than the maximum possible action:

$$Q(a_t) \leftarrow Q(a_t) + \alpha[r_{t+1} + \gamma Q(a_{t+1}) - Q(a_t)]$$

In many multiagent domains, it is difficult for agents to learn by using only local rewards (r_t) when agents' independent local goals do not align with the global system goal. For instance, if members of a sports team focus on scoring goals so that they receive a higher salary, they may not optimize the team objective of winning games. One common approach to address this problem is to provide each agent with a reward

[2]If agents could communicate, existing cooperative multi-agent methods like DCEE [19] may be more appropriate than a reinforcement learning approach.

based upon the performance of the entire team as a whole, a joint global reward G. The global reward of a system on timestep t is therefore $G_t(\mathbf{a}_t)$, where $\mathbf{a}_t$ is the joint action vector of all agents in the system at time step t.

B. Difference Rewards

Within multiagent learning systems, a key problem is how best to select the reward function(s) agents use to learn. One popular reward technique, and perhaps the most straightforward, is to let each agent use the global reward G as its individual reward (i.e., each agent receives the team's reward as its individual reward). Unfortunately, in many domains, especially domains involving large numbers of agents, such a reward often leads to slow learning because each individual agent has relatively little impact on the global reward.

Previous work has shown that while learning is possible using only the global reward, a type of shaping called *difference rewards* can significantly improve both learning speed and performance [1]. Difference rewards achieve these benefits because they ensure that all agents are attempting to optimize the global reward (i.e., what is good for the agent is good for the system) while at the same time providing each agent with specific feedback on how its own actions contributed to the received reward (i.e., each agent is able to tell whether or not its action was good or bad for the system, enabling them to improve their own individual behavior). Within stateless multiagent problems (as those used in this work), the difference reward for agent i at time step t is calculated as:

$$D_i(\mathbf{a}_t) \equiv G(\mathbf{a}_t) - G(\mathbf{a}_{t,a_i \leftarrow a'_i})$$

where the first is the global reward received $G(\mathbf{a}_t)$, and the second term $G(\mathbf{a}_{t,a_i \leftarrow a'_i})$ is the global reward that would have been received if agent i had not executed action a but instead executed a different action a', a counterfactual action.[3] Difference rewards are aligned with the system objective G regardless of how the counterfactual action a' is chosen. The second term does not depend on agent i's actions, thus agents receive positive rewards for benefiting the system and negative rewards for harming the system [4].

Difference rewards are more informative than the global reward because the second term in D removes much of the effect of other agents from i's reward. In many situations it is possible to use an a'_i that is equivalent to removing agent i from the system. This causes the second term to be independent of i, and therefore D evaluates the agent's contribution directly to the global performance. Difference rewards have been shown to work well in a number of domains and conditions [4], [20], but none account for the learning noise caused by exploratory actions of agents.

III. Experimental Domains

This section introduces the two domains used in this paper. The first is a simple toy domain while the second is more realistic and complex.

[3] In the case of stateful learning, e.g., reinforcement learning, $\mathbf{a}$ is replaced with $\mathbf{z}$, the system state information vector that includes both states and actions.

A. The Gaussian Squeeze Domain

The Gaussian Squeeze Domain (GSD) is a novel problem designed to highlight the problem of exploratory action noise. There is a set of agents in which each agent contributes to a system objective in the GSD and the agents are attempting to learn to optimize the capacity of the system objective. The objective function for the domain is as follows:

$$G(x) = xe^{\frac{-(x-\mu)^2}{\sigma^2}} \tag{1}$$

where x is the cumulative sum of the actions of agents ($x = \sum_{i=0}^{n} a_i$), μ is the mean of the system objective's Gaussian, σ is the standard deviation of the system objective's Gaussian. Here, μ and σ effectively define the target capacity, x, that the agents must coordinate their actions to achieve. Note that we have dropped the t subscript for clarity, as the timestep is only used in the learning update. The goal of the agents is to choose their individual actions a_i in such a way that the sum of their individual actions optimize Equation 1. Here, each agent has 10 actions ranging in participation value from zero to nine. The GSD is a congestion domain — adjusting the variance changes the coordination complexity for agents within the system. The lower the variance, the higher the coupling of agents' joint actions and the more difficult the optimization task.

B. UAV Communication Network Domain

The UAV Communication Network Domain (UAVCN) is based on the behavior of CDMA-based communication networks using UAVs as transmitters that hover in fixed locations (rather than relying on fixed towers). In addition to having long-range line-of-sight communications, the UAVs adaptively focus their signal. The UAVs are all at similar altitudes and communicate through directional antennas pointed towards the ground. The amount of area on the ground that is covered by the UAV is determined by the gain of its antenna. Antennas with low gain, transmit over a wider area, but within that area the strength of the signal is lower (see Figure 1a). The objective is to maximize the average data rate of each of the i customers:

$$G = \frac{1}{n}\sum_{i=1}^{n} C_i$$

where there are n customers, C_i is the data rate to customer i, and G is the system evaluation function. Based on the dynamics of this domain [6], [20], we find;

$$G = \frac{1}{n}\sum_{i=1}^{n}\sum_{j\in J_i} B\log_2\left(1.0 + \frac{\frac{aP_j}{r_j^2}e^{-b\frac{r_{i,j}}{r_j}}}{\sum_{j\notin J_i}\frac{aP_j}{r_j^2}e^{-b\frac{r_{i,j}}{r_j}} + k}\right)$$

where there are j UAVs, B is the bandwidth of the channel, r_j is the signal half-power radius for UAV j, $r_{i,j}$ is the distance from customer i to the center of UAV j's transmission, k is a constant for background noise. A UAV can control its power level, P_j, and indirectly, $r_{i,j}$, through the orientation of the UAV.

Each UAV is controlled by a pair of agents: one adjusts the power level P_j and the second controls the transmitter

orientation. The solution to the full problem consists of the power level and orientation values for all the UAVs. The power agent selects from a discrete set of ten power levels. The orientation agent chooses one of ten directions: either straight down, or at 45 degrees in one of nine evenly spaced directions around the UAV. Between these two agents, each UAV has one hundred distinct power/orientation settings.

IV. Exploratory Action Noise

Agents often treat other agents as part of the environment — the exploratory actions of other agents become stochastic environmental noise. However, agents are then unable to distinguish when their peers are taking purposeful actions or are exploring. This may cause agents bias their policies such that they actually depend upon the exploratory actions of other agents to perform well. Agents learning optimal policies in the presence of exploration may not be optimal once learning completes and exploration is disabled.

Figure 1b shows learning results when a set of 300 ϵ-greedy reinforcement learning agents learn for 1000 episodes with an exploration rate of $\epsilon = 0.10$ in the GSD. After 1000 episodes, exploration and learning is disabled. The counterintuitive result is that performance *decreases* when exploration is turned off because each agent had formed policies under conditions where all the other agents were occasionally performing exploratory actions. The agents' inability to distinguish between true environmental dynamics and dynamics caused by the exploratory actions of other agents means that the agents themselves (the solution) actually end up becoming part of the problem (adding complexity due to stochastic learning noise). This result holds for both off-line and on-line updates because such update mechanisms have no knowledge about when other agents explore.[4] The goal of techniques presented in this paper is to eliminate such exploratory action noise.

Figure 1c shows an additional example. In these experiments, 300 agents again train for 1000 episodes, now using 5% and 10% exploration, respectively. The learned policies are then fixed and tested in systems where a proportion of the agents are replaced with agents that act randomly. These results show that agent performance is optimal when the proportion of agents that act randomly are equivalent to the exploration rate during training. During the learning phase, approximately ϵ portion of the agents were taking random actions per time step and the learned joint-policy accounted for this exploratory noise. Performance is averaged over 100 trials.

Learned policies are not always worse once exploration is disabled, but these examples demonstrate the potential for exploration to interfere with learning in multiagent systems. Although the exploration-exploitation trade-off has been extensively studied, e.g., in the context of multi-armed bandits and reinforcement learning, little work has been done to address the biasing issues associated with agents learning to depend upon the exploratory actions of other agents.

We define exploratory action noise to be the portion of an agent's learning signal that is impacted by the exploratory actions of other agents. N_i, the value of the exploratory action noise for agent i, is:

$$N_i = \frac{|g_i(\mathbf{a}) - g_i(\mathbf{a} - \mathbf{a}_\epsilon))|}{|g_i(\mathbf{a})|} \tag{2}$$

where g_i is agent i's reward function, $\mathbf{a}$ is the joint-action vector for all agents in the system, and $\mathbf{a}_\epsilon$ is the subset of the joint-action vector containing all elements of $\mathbf{a}$ that were exploratory actions. Intuitively, N_i is a ratio measuring how much of agent i's reward is due to other agents' exploratory actions. The two extremes, $N = 1$ and $N = 0$ would be achieved if the system exploration rate were $\epsilon = 1.0$ (all reward is from exploratory actions) or $\epsilon = 0$ (no exploration noise), respectively. CLEAN rewards, introduced in the following section, are designed to avoid dependencies upon the exploratory actions of other agents in the system by performing exploration via off-policy counterfactual actions which do not create explicit environmental noise.

V. CLEAN Rewards

This introduces Coordinated Learning without Exploratory Action Noise (CLEAN) rewards. These shaping rewards simultaneously address the structural credit assignment problem and issues arising from learning noise caused by exploration to promote learning, coordination, and scalability in multiagent systems. As with difference rewards, we assume agents have access to an accurate model of the system reward (which can be approximated from data [6], [10]). The key requirements of CLEAN are that agents have an approximation of the underlying system objective, G, and that agents in the system follow their current target policies without exploration.[5] Greedily following target policies typically leads to poor performance, but CLEAN rewards address this shortcoming via off-policy counterfactual action exploration (Equations 3 and 4).

CLEAN rewards are structured in such a way that they promote implicit coordination that leads to good system performance (agents improving their own local reward are concurrently improving the system performance, while agents harming their local reward are also harming system performance) and are also designed to address the structural credit assignment by providing each agent with specific feedback on how its own actions impacted the reward it received.

[4]This is in contrast to the example of "Cliff Walking" in section 6.5 of Sutton and Barto [9]. In the Cliff domain, an agent can improve its performance by accounting for exploratory noise, and if the learning is disabled, the Q-learning agent would outperform Sarsa. In a multiagent domain, an agent cannot reason about what action other agents would take in the absence of exploration and the performance of standard learning algorithms may decrease in the absence of exploration.

[5]The key distinctions between CLEAN rewards and difference rewards are: i) CLEAN rewards provide reward updates for the counterfactual actions $r(a_i')$, whereas difference rewards provide rewards for the actual action $r(a_i)$ (i.e., the counterfactual action appears in the first term of the CLEAN rewards as opposed to the second term of difference rewards), ii) CLEAN rewards require agents to behave greedily and exploration is achieved through the counterfactual actions — implicit exploration is achieved by calculating the reward $r(a_i')$ in the first term of these rewards, whereas explicit exploration in the actions a is required for difference rewards).

A. CLEAN 1: $\mathcal{C}1_i$

This CLEAN reward requires each agent calculate the reward for a counterfactual action. This reward is formed by calculating the difference between the reward expected if taking some counterfactual action a_i' and the global reward actually received.[6] In this case, agent i's CLEAN reward can be represented as follows:

$$\mathcal{C}1_i(\mathbf{a}) \equiv G(\mathbf{a}_{a_i \leftarrow a_i'}) - G(\mathbf{a}) \tag{3}$$

Intuitively, this gives the agent a reward that represents how the system would have performed had it not followed its policy, taking action a_i, but instead had taken some counterfactual action, a_i'. This reward is designed to allow the agent to act greedily, while still improving its evaluation for actions not taken (equivalent to off-policy exploration).

B. CLEAN 2: $\mathcal{C}2_i$

A second variation of CLEAN rewards is defined so that the CLEAN reward to agent i is

$$\mathcal{C}2_i(\mathbf{a}) \equiv G(\mathbf{a}_{a_i \leftarrow a_i'}) - G(\mathbf{a}_{a_i \leftarrow a_i''}) \tag{4}$$

where agent i took action a_i, actions a_i' and a_i'' are counterfactual actions of agent i, and G is again the system reward. Rather than using the action the agent executed, this CLEAN reward instead approximates the reward agent i would have received if it had taken either of these counterfactual actions. This form is often very useful when an action can be chosen for the second counterfactual, a_i'', that is equivalent to removing agent i from the system (as done in this work's experiments in Section VI). In this case, the CLEAN reward would provide a reward approximation that tells the agent its contribution to the system; $\mathcal{C}2_i(\mathbf{a})$ will be positive if the counterfactual action a_i' would have been beneficial to the system, and a negative reward if the counterfactual action c_i would have been harmful to the system.

C. A CLEAN Example

This section provides an example of CLEAN rewards and their benefits to ground our discussion. Consider a set of agents learning within the GSD, with the following system parameters: $\mu = 200.0$, $\sigma = 50.0$, and $\epsilon = 0.10$. For this example, each agent has a model of the system reward, as well as the ability to observe the state of the environment. Agents are unable to communicate and must implicitly coordinate through the their rewards.

GSD agents attempt to coordinate their actions to optimize the system reward (Equation 1). Using traditional rewards (e.g., global or difference rewards), during each episode of learning an average of ϵ of the agents are exploring, meaning that the system reward is really comprised of two parts:

$$x = x_{\textit{intentional}} + x_{\textit{exploration}} \tag{5}$$

[6]For this work, we selected the counterfactual actions a_i' and a_i'' uniformly randomly during each episode in the same way that exploratory actions are selected in learning, unless otherwise specified. More informed counterfactual action exploration strategies will be explored in the future.

where $x_{\textit{intentional}}$ represents the contribution to the system of agents who are executing their current best policy, and $x_{\textit{exploration}}$ represents the contribution to the system performance of agents that are acting randomly. This exploratory action noise will impact agents learning with any explicit exploration strategy, e.g., uniform random, Boltzman, etc. As discussed earlier, such exploration noise can have a detrimental impact on overall system performance. Agents learning with CLEAN rewards always behave greedily with respect to their current best policy, eliminating the $x_{\textit{exploration}}$ term in Equation 5. Agents use CLEAN rewards to implicitly explore via internal models of the system reward and observations of the environment.

For example, given the domain setting described above, suppose that an agent that executed an action of 10 and observed that the cumulative value for the system was 250. This agent could then calculate its CLEAN reward (Equation 3) in two steps. First, it would calculate $G(\mathbf{a_T})$:

$$G(\mathbf{a}_T) = \mathbf{a_T} e^{\frac{-(\mathbf{a_T}-\mu)^2}{\sigma^2}} = 250 e^{\frac{-(250-200)^2}{50^2}} = 679.57$$

where $\mathbf{a}_T$ is the system action vector of all of the agents acting greedily (i.e., following their target policies), and $G(\mathbf{a}_T)$ is the performance of the system given that all agents are behaving greedily (including agent i). Next, the agent must calculate $G(\mathbf{a}_T - a_i + a_i')$, where a_i' represents the agent's counterfactual exploratory action. In this case, suppose the agent's exploratory action was to execute the action 3 instead of the action 10 that it actually took (i.e., $a_i' = 3$). Therefore:

$$\begin{aligned} G(\mathbf{a}_T - a_i + a_i') &= (\mathbf{a}_T - a_i + a_i') e^{\frac{-((\mathbf{a}_T - a_i + a_i') - \mu)^2}{\sigma^2}} \\ &= (250 - 10 + 3) e^{\frac{-((250-10+3)-200)^2}{50^2}} = 509.11 \end{aligned}$$

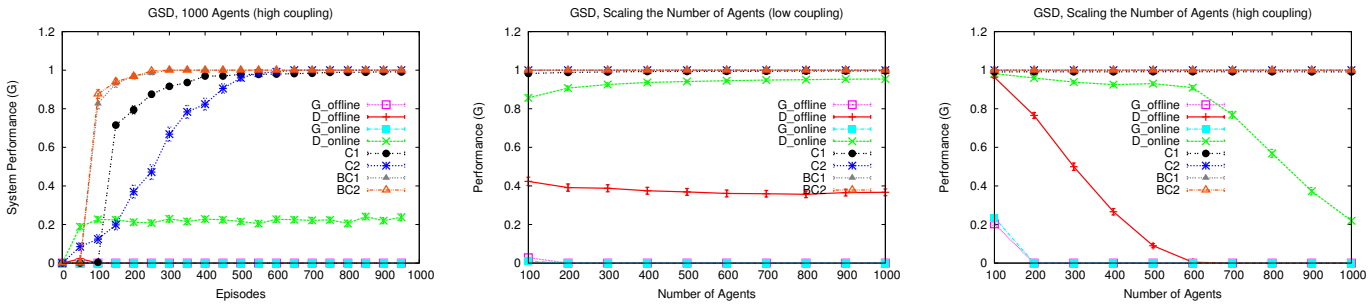
which is the system performance if agent i had taken action 3 instead 10. The agent then combines these two calculations in order to determine its CLEAN reward, as follows:

$$\mathcal{C}1_i = G(\mathbf{a}_T - a_i + a_i') - G(\mathbf{a}_T) = 509.11 - 679.57 = -170.46$$

Because the reward is negative, action 3 is worse than action 10. CLEAN rewards thus allow agents to make updates based upon how good or bad potential actions seem in comparison to the agent's current best action. This shaping reward is still aligned with the global system reward, allow agents to more quickly learn than if they maximized G directly. This ability to receive a reward update based upon a counterfactual action is the critical difference between CLEAN rewards and difference rewards. In the next section, we discuss an extension which enables agents to further improve learning speed at the expense of additional computation cost.

D. Batch Updates with CLEAN Rewards

Agents using CLEAN rewards are able to calculate a value for a reward associated with some action a_i' that they did not explicitly take within the environment. We further leverage this capability by calculating updates for multiple values of a_i' during each time step of learning, enabling agents to make a trade-off of more exploration during each time step at the cost



(a) (b) (c)

Fig. 2: **Gaussian Squeeze Domain Results**: a) 1000 agents using global or difference rewards learn policies that depend upon exploratory action noise, while agents learning with CLEAN rewards do not perform worse once learning is disabled. b) The offline performance for agents using G and D is frequently worse than the online performance, and is eclipsed by the performance agents using of CLEAN rewards. c) As the number of agents increases, the problem becomes more difficult and the effect of exploratory action noise becomes increasingly impactful.

of increased computation (linearly increasing computation cost for each additional reward calculation). Our latter empirical results show that such Batch-CLEAN reward updates can result in significantly faster learning in terms of episodes of learning, though they come with increased computational costs.

In this section, we will introduce two formulations of Batch-CLEAN rewards, which are based upon the two CLEAN rewards introduced previously in Equations 3 and 4. First, consider the CLEAN rewards outlined in Equation 3, where each agent's reward compares an exploratory counterfactual action against it's current best policy. Batch-CLEAN rewards offer an extension by enabling each agent to perform reward calculations associated with multiple potential counterfactual actions a_i' during each individual episode of learning. In particular, during each time step the agent calculates a set of CLEAN rewards as follows:

$$\overline{\mathcal{BC}1}_i = \{\mathcal{C}1_i(a_i')\}\forall a_i' \in A_i \tag{6}$$

where $\overline{\mathcal{BC}1}_i$ is the set of rewards agent i receives during each time step, associated with a set of potential counterfactual actions A_i. Agents using Batch-CLEAN rewards perform value updates for each of the individual rewards during each time step (performing value updates for each counterfactual action a_i' respectively). When there are relatively few actions in each agents' action space, agents can frequently calculate CLEAN rewards for the entire set of potential counterfactual actions. In situations where there is a large action space, agents using Batch-CLEAN may choose to calculate CLEAN rewards for only a selected portion of their potential action space.

Next, we consider Batch-CLEAN rewards based upon Equation 4 which enable agents to approximate and compare the impact of two freely selected counterfactual actions a_i' and a_i'' on the overall system performance. In this work, we select a second counterfactual a_i'' that is equivalent to removing agent i from the system, and we hold this counterfactual constant in all reward calculations based upon Equation 4. Selecting the counterfactual action a_i'' this way causes the second term of the CLEAN reward to approximate the performance of the system in the case where agent i did not exist. Hence, an agent can determine how beneficial its presence and interaction was to the system as a whole. Batch-CLEAN rewards are defined as:

$$\overline{\mathcal{BC}2}_i = \{C2_i(a_i')\}\forall a_i' \in A_i \tag{7}$$

where $\overline{\mathcal{BC}2}_i$ is the set of Batch-CLEAN rewards agent i receives each time step when the counterfactual a_i'' is selected to approximate removing agent i from the system. Again, during each time-step agents calculate value updates for each of the individual CLEAN rewards. [7]

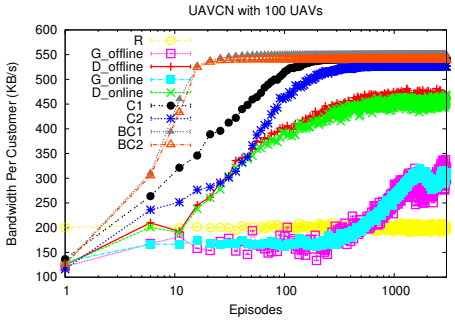


Fig. 3: UAVCN Domain with 100 agents. Agents using CLEAN rewards outperform those using global or difference rewards. Batch-CLEAN rewards allow agents to learn even faster and achieve higher asymptotic performance.

VI. Results

Experiments evaluate agents learning via off-policy Q-learning and acting independently (without communication), relying on implicit coordination via couple reward functions. We explore both example domains using seven types of agents:

- R : Agents taking random actions
- G : Agents learning with Global (G) rewards
- D : Agents learning with Difference (D) rewards
- C1 : Agents learning with CLEAN 1 ($\mathcal{C}1_i$) rewards
- C2 : Agents learning with CLEAN 2 ($\mathcal{C}2_i$) rewards

[7]Further discussion of the the Batch-CLEAN rewards and their derivation in the UAVCN domain can be found elsewhere [6], [20].

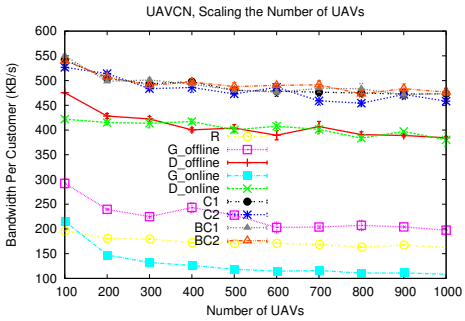


Fig. 4: UAVCN Domain with agent scaling. Agents using CLEAN reward techniques significantly outperform all other methods.

BC1 : Agents learning with Batch-CLEAN ($\overline{\mathcal{BC}1}_j$) rewards
BC2 : Agents learning with Batch-CLEAN ($\overline{\mathcal{BC}2}_j$) rewards

Experiments consist of $r = 30$ independent trials, Q-tables are initialized stochastically, and the error in the mean $\sigma/\sqrt{r}$ is plotted (although it often too small to be visible). Results were statistically significant as verified by t-tests with $p = 0.05$. In both domains the agents were using off-policy updates. All agents used Q-learning. The learning rate was $\alpha = 0.10$, the exploration rate was $\epsilon = 0.10$, and the discount factor was $\gamma = 0.10$ in all experiments unless otherwise specified.

Throughout these experiments we plotted both online and offline performance for agents using global and difference rewards. This was done to demonstrate the impact of exploratory action noise on the learning process.[8] The difference between the online and offline performance using G and D is due to the impact of exploratory action noise on learning.

A. The Gaussian Squeeze Domain

Figure 2a shows the results for 1000 agents learning in the Gaussian Squeeze Domain with $\mu = 175$, $\sigma = 175$. This is a case of high congestion and is meant to simultaneously show the potential benefits in learning speed associated with Batch-CLEAN rewards and the impact that exploratory action noise can have on the learning process when using traditional reward structures. Here, the performance of agents using global rewards G is poor in both the online and the offline settings. This is because global rewards do not provide individual agents with specific feedback on how their individual actions impacted the system performance compared to the actions of all of the other agents in the system (i.e., each agent's reward signal gets lost in the "noise" of the rest of the system). Agents using difference rewards D performed better than agents using G because difference rewards provide each agent with a reward that is reflective of it's own individual impact on the system performance. Unfortunately, difference rewards do not address the issues associated with exploratory action noise. The disparity in performance between CLEAN rewards and difference rewards can be directly attributed to the impact of exploratory action noise on the learning process. As seen here, exploratory action noise can have a massive impact on learning performance, particularly in large, tightly coupled multiagent systems. Agents using CLEAN rewards all converge to near-optimal performance, maintaining 5 times the performance of the next best technique (i.e., difference rewards).

Figure 2b shows the results of an experiment where the number of agents, the mean, and the variance of the target function were proportionally scaled. Here, the mean and the variance were scaled proportionally to the number of agents such that $\mu = 0.50N$ and $\sigma = 0.50N$. This experiment was set-up with low coupling compared to the previous experiment and was conducted to show that even when coupling is relatively low, the impact of exploratory action noise on the learning process can be significant. Agents using global rewards, G, continue to perform poorly due to their inability to assign credit to agents in the system for their individual contributions to the system performance. As observed in Figure 2b, the gap between the online and offline performance of agents using difference rewards grows as scaling is increased. This is because more and more agents are exploring during each time step, and exploratory actions of agents in the system have an increasingly large impact on the overall system dynamics. Again, all forms of CLEAN rewards outperform all other techniques, converging to nearly optimal performance for the Gaussian Squeeze Domain.

The final GSD experiment considers how performance changes as complexity increases. In the GSD, as the variance decreases, the agents become more coupled and the exploratory action noise has a larger effect. Figure 2c shows the results of using a fixed mean and variance ($\mu = 175$ and $\sigma = 175$) with different numbers of agents in the system. As expected, CLEAN rewards are much more robust when the problem complexity is increased (more agents and higher congestion) relative to global and difference rewards because agents' exploratory actions play a larger role in system performance.

B. UAV Communication Network Domain

In the following experiments, only the online performance of agents using global and difference rewards are plotted for brevity. The first UAVCN experiment considered set of 100 UAVs and 100 customers. As see in in Figure 3, agents selecting power and orientation actions randomly perform poorly. Similarly, agents using global rewards are slow to learn and learn a policy with performance far worse than other agents. Global rewards provide a noisy learning signal to agents because of the coupling with other agents' actions. Difference rewards address this shortcoming by effectively filtering off much of the impact of other agents on the learning signal. This additional information (which addresses structural credit assignment) results increased performance compared to agents using G, but difference rewards do not account for the impact of exploratory action noise.

To address the shortcomings of global and difference rewards, we implemented CLEAN rewards ($\mathcal{C}1$, and $\mathcal{C}2$), which

[8]The online and offline performance of CLEAN rewards are identical since agents explore via off-policy counterfactual actions while continually following their target policies.

not only address the structural credit assignment problem as difference rewards do, but also address exploratory action noise. CLEAN rewards outperform difference rewards by up to 25% and global rewards by over 100% in this domain. However, CLEAN rewards still learn slowly in comparison to Batch-CLEAN rewards — they are limited by only receiving a single reward update per episode of learning.

Batch-CLEAN rewards maintain the high performance of CLEAN rewards while significantly improving learning speed over standard CLEAN rewards. The speed-up in learning stems from the fact that Batch-CLEAN rewards leverage the properties of privatized exploration. This extension effectively enables learning speed to be improved over traditional reward techniques such as global or difference rewards, as well as standard CLEAN rewards, which only allow a single reward update per learning episode.

It is interesting to note that although the computational increase for agents' reward calculations using $\mathcal{BC}1$ and $\mathcal{BC}2$ rewards compared to G, D, $\mathcal{C}1$, and $\mathcal{C}2$ rewards is twenty fold, these agents learn one hundred fold faster (showing a nonlinear increase in learning performance compared to computational costs in this particular case). Here, Batch-CLEAN rewards $\mathcal{BC}1$ and $\mathcal{BC}2$ outperform agents using D by over 25% and agents using G by over 100%, while maintaining the performance of standard CLEAN rewards.

The next set of experiments involved scaling the number of UAVs and customers in the system, while keeping the ratio fixed at $1:1$. As seen in Figure 4, the general performance of all techniques decreases with increased task complexity. This is not surprising, given that within a CDMA-based communication network, the bandwidth is directly dependent upon the signal-to-noise ratio, where the signal noise increases as the number of communication towers (UAVs) broadcasting increases (see the discussion on the dynamics of a CDMA-based communication system in Section III-B for details). In a shared-channel network, this background noise necessarily reduces the amount of signal throughput to any individual node in the system. It is important to note, however, that the total system bandwidth increases as the number of UAVs is increased, even though the signal per customer degrades. Again, agents using random policies continue to perform poorly as scaling increases. Agents using global rewards experience a significant drop in performance when the number of UAVs and customers is increased by an order of magnitude. Agents using difference rewards experience a slight decrease in performance as scaling increases. Agents using CLEAN and Batch-CLEAN rewards also experience a slight performance drop (characteristic of adding more cell towers within a CDMA network), but continue to significantly outperform all other methods. Agents using Batch-CLEAN rewards learn significantly faster than agents using CLEAN rewards with single counterfactual reward updates in all of these cases.

VII. Discussion and Conclusion

There has been a lot of research involving the exploration-exploitation tradeoff within the multiagent learning literature. However, relatively little work has been done to directly address the impact of learning noise caused by the exploratory actions of agents. In this work, we first showed the potential impact of such exploratory action noise on learning, demonstrating that exploratory actions can cause agents to bias their policies to depend upon the exploratory actions of others, which can lead to suboptimal learning. We defined this learning noise as *exploratory action noise*. We then introduced CLEAN rewards, which are shaped rewards designed specifically to promote coordination and scalability in multiagent systems by addressing both the structural credit assignment problem, as well as the *exploratory action noise* caused by agent exploration. Additionally, we introduced an extension of these rewards called Batch-CLEAN, which leveraged the concept of counterfactual actions in order to enable agents to calculate multiple reward updates per episode of learning. We provided empirical results demonstrating the performance of CLEAN rewards in two multiagent domains including a Gaussian Squeeze Domain (toy domain) and a more realistic UAV Communication Network domain based upon CDMA-based network dynamics. We showed that CLEAN rewards are highly robust to exploration and scaling, significantly outperforming both global rewards and difference rewards in both domains.

This work was aimed at demonstrating the presence and potential impact of exploratory action noise within multiagent learning problems, and providing a potential solution to addressing this problem. Though CLEAN rewards showed much promise in this work, they operated under the assumption that a complete and accurate model of the reward structure is known by the agents. This assumption is violated in many multiagent domains where the reward structure may be unknown or may be too complex for an analytical model. As an extension of this work, we will extend CLEAN rewards into domains where agents are not provided with a model of the underlying system reward structure, and instead must construct an approximate model of the system reward, and then use this approximate model to calculate their CLEAN rewards. This will significantly improve the real-world applicability of these rewards.

Acknowledgements This work was partially supported by the National Energy Technology Laboratory under grant DE-FE0011403 and by the National Science Foundation under grant IIS-1149917.

References

[1] A. Agogino and K. Tumer, "Analyzing and visualizing multi-agent rewards in dynamic and stochastic domains," *J. of Autonomous Agents and Multi-Agent Systems*, pp. 320–338, 2008.

[2] A. Farinelli, A. Rogers, and N. Jennings, "Maximising sensor network efficiency through agent-based coordination of sense/sleep schedules," in *Proc. of the Int'l Workshop on Energy in Wireless Sensor Networks*, 2008.

[3] S. Williamson, E. Gerding, and N. Jennings, "Reward shaping for valuing communications during multi-agent coordination," in *Proc. of the Int'l Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, 2009.

[4] K. Tumer and A. Agogino, "Distributed agent-based air traffic flow management," in *Proc. of the Int'l Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2007.

[5] L. Kaelbling, M. Littman, and A. Moore, "Reinforcement learning - a survey," *J. of Artificial Intelligence Research*, 1996.

[6] C. HolmesParker, "CLEAN Learning to Improve Coordination and Scalability in Multiagent Systems ," *Ph.D. Dissertation*, Oregon State University, 2013.

[7] S. Ross and J. Pineau, "Model-based bayesian reinforcement learning in large structured domains," in *Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2008.

[8] P. Stone and M. Veloso, "Multiagent systems: A survey from a machine learning perspective," *Autonomous Robotics*, vol. 8, no. 3, July 2000.

[9] R. Sutton and A. Barto, *Reinforcement Learning An Introduction*. Cambridge, MA: MIT Press, 1998.

[10] S. Proper and K. Tumer, "Modeling difference rewards for multiagent learning," *Proc. of the Int'l Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, 2012.

[11] H. Maei, C. Szepesvari, S. Bhatnagar, and R. Sutton, "Toward off-policy learning control with function approximation," in *Proc. of the Int'l Conf. on Machine Learning (ICML)*, 2010.

[12] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *Proc. of the American Associated for Artificial Intelligence Conf. (AAAI)*, 1998.

[13] G. Chalkiadakis and C. Boutilier, "Sequentially optimal repeated coalition formation under uncertainty," *J. of Autonomous Agents and Multi-Agent Systems*, 2012.

[14] L. Panait and S. Luke, "Cooperative multi-agent learning - the state of the art," *J. of Autonomous Agents and Multi-Agent Systems*, 2005.

[15] M. Bowling and M. Veloso, "Simultaneous adversarial multi-robot learning," in *Proc. of the Int'l Joint Conf. on Artificial Intelligence (IJCAI)*, 2003.

[16] S. Jensen, D. Boley, M. Gini, and P. Schrater, "Non-stationary policy learning in 2-player zero sum games," in *Proc. of the National Conf. on Artificial Intelligence (AAAI)*, 2005.

[17] M. Bowling, "Convergence and no-regret in multiagent learning," *Advances in Neural Information Processing Systems (NIPS)*, 2005.

[18] V. Conitzer and T. Sandholm, "Bl-wolf - a framework for loss-bounded learnability in zero-sum games," in *Proc. of the Int'l Conf. on Machine Learning (ICML)*, 2003.

[19] M. E. Taylor, M. Jain, Y. Jin, M. Yooko, and M. Tambe, "When should there be a "me" in "team"? Distributed multi-agent optimization under uncertainty," in *Proc. of the Int'l Conf. on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2010.

[20] A. Agogino, C. HolmesParker, and K. Tumer, "Evolving large scale UAV communication system," in *Proc. of the Genetic and Evolutionary Computation Conf. (GECCO)*, July 2012.