Learning to Control Neurons using Aggregated Measurements

Yao-Chi Yu[1], Vignesh Narayanan[1], ShiNung Ching[1,2] and Jr-Shin Li[1,2]

Abstract— **Controlling a population of neurons with one or a few control signals is challenging due to the severely underactuated nature of the control system and the inherent nonlinear dynamics of the neurons that are typically unknown. Control strategies that incorporate deep neural networks and machine learning techniques directly use data to learn a sequence of control actions for targeted manipulation of a population of neurons. However, these learning strategies inherently assume that perfect feedback data from each neuron at every sampling instant are available, and do not scale gracefully as the number of neurons in the population increases. As a result, the learning models need to be retrained whenever such a change occurs. In this work, we propose a learning strategy to design a control sequence by using population-level aggregated measurements and incorporate reinforcement learning techniques to find a (bounded, piecewise constant) control policy that fulfills the given control task. We demonstrate the feasibility of the proposed approach using numerical experiments on a finite population of nonlinear dynamical systems and canonical phase models that are widely used in neuroscience.**

I. Introduction

Neurons in the brain form complex networks of interconnected dynamical systems and communicate with each other using action potentials or spikes [1]. These spikes are also used to represent information regarding an external stimulus (e.g., sensory inputs) from the physical world. The mechanism through which the neurons convert an external stimulus into a coded internal representation in the form of spike trains is not yet fully understood. On the other hand, due to the rapid development in neuro-stimulation technologies, it is now possible to both finely manipulate neurons with external stimuli and also to record the activity of a population of neurons [2]. With these developments, it is now feasible to carefully design experiments for precisely manipulating the neural activity, which can potentially shed some light on the neural coding mechanism [3].

Precise manipulation of the spiking activity in a population of neurons requires strategies that allow for an efficient and systematic synthesis of the stimulation signals. In this context, interpretable modeling of the dynamic behavior of a neural population and tractable design of control inputs using the neural measurements/recordings are critical. To this end, several system theoretic approaches to control neural systems at different spatio-temporal scales are available (e.g., [3]). For instance, to induce synchronization/desynchronization of spiking activity in a population of neurons, ensemble control was proposed [4]. Similarly, optimal controllers were designed to elicit a given spike sequence or a spike pattern in a network of neurons [3], [5], [6]. Alternatively, model-free strategies to learn control signals directly from data to manipulate the neural dynamics were reported [7]–[9]. These learning strategies make use of tools such as deep neural networks and deep reinforcement learning to learn a representation of the input-output behavior of a network of neurons, and to design control signals in a fully data-driven setting.

However, several challenges persist in the context of controlling a population of neurons. These include: (i) the need for perfect feedback data from each neuron at every sampling instant; (ii) noisy recording and missing measurement data during the process of signal recording [10]; (iii) the unknown nonlinearities in the neural dynamics. In this context, the inability to track the spiking activity of each neuron simultaneously remains a fundamental challenge and prohibits the use of traditional feedback or reward to design closed-loop controls for precisely manipulating a neural population.

To mitigate this challenge, in this work, we propose an aggregated Q-learning approach in which we design the instantaneous reward and the control objectives using the population-level *aggregated measurements or snapshots*. In particular, we define an output sequence that can be computed using the aggregated measurements and learn a control policy by using the time-series data of this output sequence. Different from the existing approaches [7]–[9], the proposed learning framework does not require measurements from each neuron in the population at every sampling instant. We demonstrate the feasibility of the proposed approach using numerical examples using phase-models.

The remainder of this paper is organized as follows. In Section II, we introduce the control problem, motivate the idea of aggregated measurements, and provide a brief background on Q-learning. In Section III, we present the details of the proposed method along with two cases of numerical simulations to illustrate the feasibility of the proposed learning scheme.

II. Background and Problem Formulation

In this section, we will begin by formally introducing the class of systems that is considered in this paper and introduce the notion of aggregated measurements. We will then provide a brief background on the learning problem addressed in this paper.

*This work was supported in part by the NSF awards, ECCS-1509342, CMMI-1933976, and CMMI-1763070, and the NIH grant 1R01GM131403.

[1]Y.-C. Yu, V. Narayanan, S. Ching, and J.-S. Li are with the Department of Electrical and Systems Engineering, Washington University in St. Louis, St. Louis, MO, 63130, USA. y.yu, vignesh.narayanan, shinung, jsli@wustl.edu;

[2]S. Ching and J.-S. Li are with the Division of Biology and Biomedical Sciences, Washington University in St. Louis, St. Louis, MO, 63130, USA.

Individual neurons in a population modulate their spiking activity and collectively perform complex tasks such as encoding and decoding. A population of such neurons can be modeled as a parameterized family of dynamical systems or an *ensemble system*, where the differences in the individual neurons that may arise due to variations in their channel conductances or membrane-capacitance, etc., are captured by a dispersion parameter. In this work, we will consider a class of input-affine nonlinear ensemble system governed by the dynamics

$$\frac{d}{dt}x(t,\beta) = f(\beta,x(t,\beta)) + g(\beta,x(t,\beta))u(t), \qquad (1)$$

where $\beta \in K \subset \mathbb{R}$ is the dispersion parameter, $x(t,.) \in \mathbb{R}^n$ denotes the states of the ensemble system, $u(t) \in \mathbb{R}^m$ is the parameter independent control, the functions f and g are smooth nonlinear maps representing the drift dynamics and the control-coefficient, respectively, and K is a compact set [4]. The majority of the existing biophysical models of a neuron or the phase-models, e.g., the Hodgkin-Huxley model [1], [11], belong to the class of input-affine nonlinear systems, and a population of neurons with their dynamics described using such models can be represented as in (1).

A. Aggregated Measurements

Typically, neural recordings obtained through an extracellular electrode will contain spiking activity from a large number of neurons (see Fig. 1). If the recorded signals from multiple neurons do not overlap temporally or if the overlap is minimal, spikes recorded at the same electrode can be sorted and assigned to individual neurons [2]. However, in many cases, it may not always be possible to sort these spikes recorded at an electrode and assign them to individual neurons. As a result, the traditional approaches that use feedback data from each subsystem in the population to design closed-loop controls are not feasible. In this context, it is of practical significance to directly use the measurements at the population-level without having to resolve the measured signals to individual neurons in the population for control synthesis. In this paper, we use these population-level *aggregated measurements*, and propose a learning strategy to design control signals for manipulating the neural population.

Consider the system in (1), let the sequence $\{t_0, t_1, \ldots\} \subset t$ with $t_0 \geq 0$ denote the sampling instants when the measurements are recorded. Formally, we call $Y(t_i)$ as an aggregated measurement at the sampling instant t_i for $i \in \mathbb{N}$, where Y is defined as a set given by

$$Y(t_i) := \{x(t_i,\beta) \mid \beta \in K_i\}, \qquad (2)$$

with $Y(t_i) \in \mathbb{R}^{p_i}$, $p_i = |K_i|$ (cardinality of the set K_i) and $K_i \subset K$. We emphasize that the measurements (see Fig. 1) at each sampling time (i) may not have the same number of recorded data (i.e., K_i changes with the sampling time t_i); (ii) cannot be associated uniquely to a specific neuron in the population; and (iii) do not account for all the neurons at each of the sampling time (i.e., K_i is a proper subset of K).

Before describing the learning problem investigated in this work, in the following, we briefly review the traditional

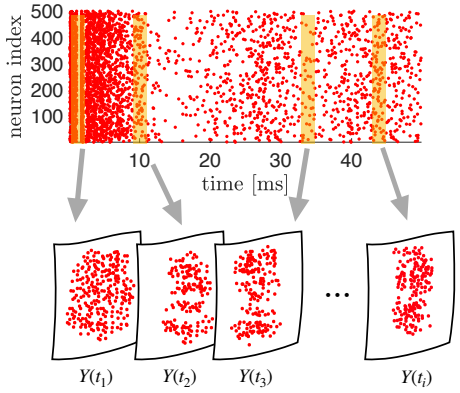

Fig. 1. Aggregated measurements: Spiking activity of 500 neurons over a period of 50ms. Each red dot denotes an action-potential or a spike. At each sampling instants, an aggregated measurement consists of snapshots of the spiking activity of the neural population.

reinforcement learning (RL) framework. Most commonly, the RL algorithms are viewed in terms of discrete-time problems, and so we will present a discrete-time description of the learning problem. In the following, for ease of exposition, the parameter β is suppressed in the notations, i.e., $x(t,\beta)$ is denoted as $x(t)$.

B. Model-free learning

In a reinforcement learning (RL) framework [12]–[14], the agent (or the controller) learns a sequence of actions (or control inputs) that optimizes a given performance measure. In the ideal case, when the controller has access to perfect measurements, i.e., when all the states are measured at each sampling instants ($K_i = K$ for each $i = 0,1,\ldots$ and $Y(t_i) = x(t_i)$), the RL agent tries to find a control policy that optimizes a given performance measure. For instance, at the i^{th} sampling instant t_i, the RL agent perceives the system state, $x(t_i) \in \mathbb{R}^n$ and chooses a control action $u(t_i) \in U$, where U is a set of admissible control inputs. Due to this control action, the system state transits to the next state $x(t_{i+1})$ as determined by the dynamics of the system (1), and this transition yields a reward (or the one-step cost) $r(t_i)$ (in general, it is a function of the system states, control or time). The performance measure is defined using this instantaneous reward as

$$V_\pi(x(t_s)) = \sum_{i=s}^{\infty} \gamma^{i-s} r(x(t_i),u(t_i)), \qquad (3)$$

where V_π denotes the *cost-to-go* from state $x(t_s)$, and it describes the sum of discounted future costs from the current step t_s to the future (infinite time-horizon) and $0 < \gamma < 1$ is the discount factor that quantifies the importance of the cost at the future states. Here $V_\pi(x(t))$ is called *value function* associated with the control policy π and $r(x(t_i),u(t_i))$ is the one-step cost induced by the control $u(t_i)$ resulting in the transition from $x(t_i)$ to $x(t_{i+1})$. The policy π defines the rule based on which the control action is selected, i.e., $u(t) = \pi(x(t),V_\pi)$. For example, the cost for a standard linear quadratic regulator (LQR) is of the form $r(x(t),u(t)) =$

$x^T(t)Lx(t) + u^T(t)Ru(t)$, where the matrices L and R are positive-definite. Thus the goal of the RL agent is to learn a control sequence that optimizes the cost-to-go as in (3).

One of the popular RL algorithms is the model-free Q-learning approach [15], where the learning agent utilizes the data $(x(t_i), u(t_i), x(t_{i+1}), r(t_i))$ at each i, to iteratively update a Q-function, denoted as $Q(x,u)$, which is a scalar function, i.e., $Q : \mathbb{R}^n \times U \to \mathbb{R}$, based on the update rule

$$Q_{k+1}(x(t_i), u(t_i)) = Q_k(x(t_i), u(t_i)) + \alpha \left[r(t_i) + \gamma \max_u Q_k(x(t_{i+1}), u) - Q_k(x(t_i), u(t_i)) \right], \quad (4)$$

where $k = 0, 1, \ldots$, is the iteration index and $0 < \alpha < 1$ is the learning rate. With (4), the Q-value for each state-action pair will converge to the optimal Q^* values asymptotically as the iterations $k \to \infty$ under certain conditions, where Q^* satisfies the Bellman optimality equation, and it corresponds to the optimal value function ($V^*_{\pi^*}$) associated with the optimal policy (π^*) [15], [16].

Note that the Q- learning framework directly uses the data to learn a control policy that achieves a desired objective. However, unlike the ideal case, in practice, we have access only to the population-level aggregated measurements as defined in (2). This introduces a bottleneck when using such algorithms to learn control sequences for a population of systems as in (1). To mitigate this challenge and to systematically design control signals without the accurate knowledge of the system dynamics or perfect feedback information, in this work, we propose an aggregated measurement-based Q-learning algorithm or the *aggregated Q-learning* that makes use of the population-level feedback instead of the perfect feedback from each neuron in the population. In particular, in the context of steering the system (1) using (2), we assume that the aggregated measurements are available as feedback at the (discrete) sampling instants defined in (2) and propose a learning scheme, which is detailed in the next section.

III. Proposed Strategy and Main results

In this section, we present our aggregated Q-learning scheme for learning a control sequence to steer the ensemble system given in (1) from an initial state to a desired final state by using the aggregated measurements (2). In particular, we introduce an output sequence that can be computed using the aggregated measurements and use this output sequence to design the reward function in our Q-learning framework.

A. Outputs induced by aggregated measurements

In our application, we only have access to the aggregated measurements (2). Therefore, we define an auxiliary output sequence $\mu_i(t)$ for $i = 1, \ldots, M$ and $M \in \mathbb{N}$ such that for $i = 1, 2$, μ_i is defined as

$$\mu_1(t) = \frac{1}{|K_i|} \sum_{j=1}^{|K_i|} x(t, \beta_j), \;\; \mu_2(t) = \frac{1}{|K_i|} \sum_{j=1}^{|K_i|} (x(t, \beta_j) - \mu_1(t))^2, \quad (5)$$

where $\mu_i \in \mathbb{R}^n$, $\mu = (\mu_1, \ldots, \mu_M)'$ and the exponent is taken component-wise. Similarly, for any $i \in \{1, \ldots, M\}$, μ_i is defined as a monomial of degree i. With the outputs μ_i defined as in (5), instead of steering the system (1) using (2), we reformulate this problem in terms of these outputs and define the learning objective in terms of the output sequence. To formalize this, we introduce a transformation $\mathbb{T} : \mathbb{R}^n \times K \to (\mu_1, \ldots, \mu_M)'$, where the states of the ensembles are mapped to an output sequence. These outputs mapped from the states of (1) can be viewed as central moments of a distribution and the outputs computed using the aggregated measurements as in (5) can be viewed as sample moments. For instance, to steer the ensemble system (1) to a neighborhood of a given final state (x_f), first, the states are mapped to the output space, so that we have $\mu_f = \mathbb{T}(x_f) = (\mu_{1f}, \mu_{2f})'$ for $M = 2$. Then the control objective is defined in terms of the desired output sequence such that: (i) $\left|\mu_{1f} - \mu_1(t_i)\right| < c$, and (ii) $\left|\mu_{2f} - \mu_2(t_i)\right| < d$, where c and d quantify the error tolerance allowed in terms of the auxiliary output.

To check the feasibility of the proposed approach, we present some preliminary results using two numerical examples in the next section. As a starting point, in these experiments, we discretized the entire output space and the action space, resulting in a finite Markov decision process. A Q-table is set up for each example and updated according to Algorithm 1. The parameter ε is the decay rate for exploration in each step [12], and the learning is episodic with the maximum number of episodes denoted as E. In each episode, the system is initialized with the given initial condition and the learning is performed for a fixed number of steps S depending on the final time T. In the proposed learning framework, we update the control inputs at the sample-instants t_i and hold this control input until the next sampling instant using a zero-order hold-like mechanism, resulting in a piecewise constant control. The Q-table is updated based on the collected reward after each episode.

Algorithm 1 Aggregated Q-Learning

Input: $\mu(t_0)$, T, μ_f, U, K, α, γ, E, S, c, ε.
Output: $Q(\mu, u)$ (Q-table)
Initialization : $Q(\mu, u)$, for all $\mu, u \in U$
for *episode* $= 1$ to E **do**
  for *step* $= 1$ to S **do**
    $u(t) \leftarrow \varepsilon - greedy$: Choose an action u that gives the maximum $Q(\mu(t_i), u)$ value with probability ε
    Apply $u(t)$ and acquire the next measurement $Y(t_{i+1})$
    Compute: $\mu(t_{i+1})$ from $Y(t_{i+1})$ and receive reward $r(t_{i+1})$
    Update Q-value:
    $Q(\mu(t_i), u(t_i)) = Q(\mu(t_i), u(t_i)) + \alpha[r(t_i) + \gamma \max_u Q(\mu(t_{i+1}), u) - Q(\mu(t_i), u(t_i))]$
    if $||\mu(t_i) - \mu_f|| < c$ **then**
      break
    end if
  end for
end for
return $Q(\mu, u)$

TABLE I

PARAMETERS USED IN EXAMPLE 1 AND EXAMPLE 2.

Symbols	*Ex.*1	*Ex.*2
Learning rate (α)	0.99	0.99
Discount factor (γ)	0.9	0.9
Maximum number of episode (E)	5000	1000
Maximum step in each episode (S)	1000	400
Range of perturbation constant (β)	$[0.95, 1.05]$	-
Epsilon(ε)	0.999999	0.99999

B. Numerical Examples

In this section, we evaluate the performance of the proposed learning scheme using two numerical experiments. In both of the examples, the control sequence learned using the aggregated Q-learning algorithm successfully steered the corresponding ensemble systems as in (1) from a given initial state to the desired state.

Example 1: Consider a nonlinear ensemble system of the form (1) with the states $q = (x, y, \theta)$ and the system dynamics

$$\dot{q}(t,\beta) = \beta\left[g_1(q(t,\beta))u(t) + g_2(q(t,\beta))v(t)\right], \qquad (6)$$

where $g_1(q) = \begin{bmatrix}\cos\theta & \sin\theta & 0\end{bmatrix}^T$ and $g_2(q) = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$. The state space of the system is given by $\mathbb{R}^2 \times \mathbb{S}^1$ [17] and the two control inputs u, v are influencing the dynamics of this system to generate forward/backward displacement and angular shift, respectively. The admissible controls are defined as $u \in \{+1, 0, -1\}$ and $v \in \{+\pi/2, 0, -\pi/2\}$. We consider five systems with parameters β defined as in Table I. The control objective is to steer the entire ensemble from the initial state $q(0,\beta) = (-2,-2,0)$ to the neighborhood of the target state $q_f = (3,2,0)$ simultaneously using the same control input.

In this example, we defined the output sequences for states x and y with $M = 2$ as in (5), and another sequence with $M = 1$ for the state θ. In other words, $\mu(t) = (\mu_1^x(t), \mu_1^y(t), \mu_1^\theta(t), \mu_2^x(t), \mu_2^y(t))' \in \mathbb{R}^5$ and $\mu_f = (3,2,0,0,0)'$. The output space was fully discretized into $21 \times 21 \times 11 \times 11 \times 11$ grids, wherein each discretization dimension corresponds to the entries of the $\mu(t)$ vector. In an episode, if the agents reached the goal or spent more than 1000 steps without reaching the goal, the episode was terminated and a new episode was initiated. During the process of learning, if all the systems in the population reached the goal with tolerance $c = 0.01$ and $d = 0.01$, an immediate reward of $+100$ was returned. Otherwise, a reward of $-\|(\mu(t) - \mu_f)\|^2$ was returned, where $\mu(t)$ and μ_f denotes output at time t and the desired outputs, respectively. A uniform sampling time of 0.5s was selected. The parameters used in the learning procedure are listed in Table I. The resulting snapshots of the trajectories of the system (6) and the corresponding control sequence are presented in Fig. 2. The performance of the learning algorithm using both the episodic reward and the averaged rewards are presented in Fig. 3.

Example 2: The design of electrical or magnetic stimulations to simultaneously generate neural spikes is of great interest in experimental neuroscience. This can potentially be used to design external stimulation for precise manipulation of the spikes to gain insights into the neural coding mechanism or to relieve an epileptic symptom in patients [18]. In practice, limitations on the amplitude of the control signal need to be taken into consideration to avoid generating

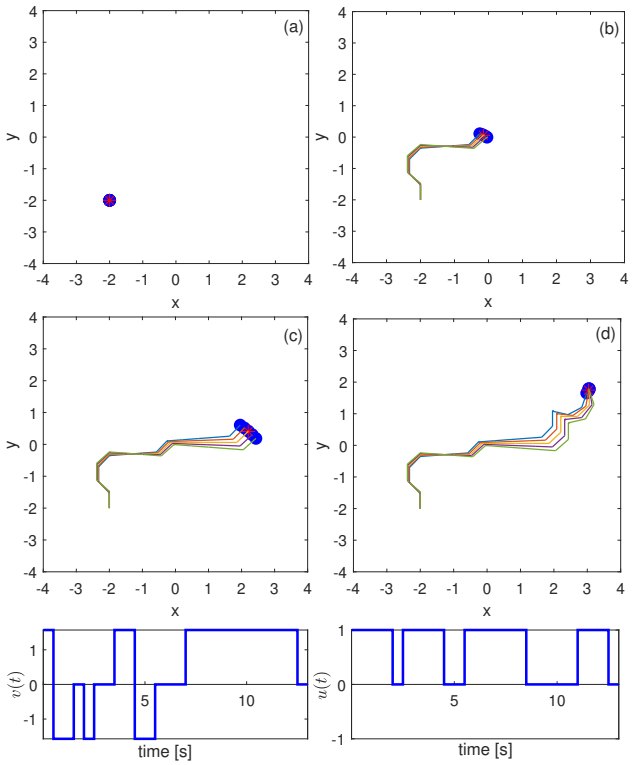

Fig. 2. Example 1: The trajectory snapshots of the controlled ensemble system. The blue dots in the figure represent each individual agent, while the red dot represent the averaged x, y-coordinate of the five agents. Each plot from (a) to (d) represents a snapshot at $t = 0, 5, 8$ and 13, respectively. In figure (d), all five agents are within the neighborhood of the target state $(3,2)'$. The piecewise constant control sequences learned using the aggregated Q-learning are shown in (e) and (f). The control $v(t)$ changes the orientation of each system in the population, while the control $u(t)$ corresponds to a unit displacement in the forward/backward direction.

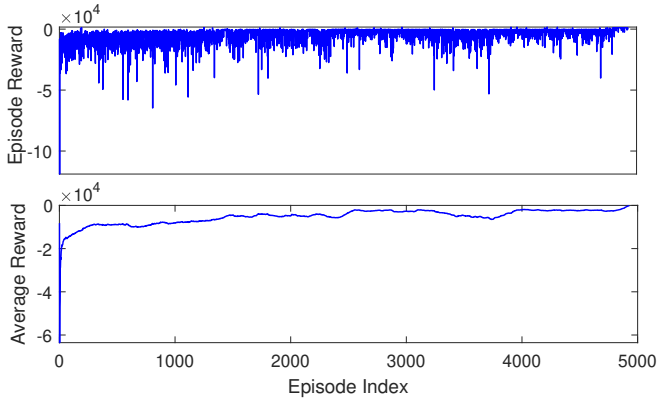

Fig. 3. Example 1: Performance of the learning algorithm represented using the episodic reward and the average reward computed over the number of episodes.

harmful stimulations [19]. Therefore, the task of inducing simultaneous neural spikes with a bounded control sequence is of great importance. We begin our example of neurocontrol by examining a two-neuron system case. In particular, we consider the phase-model of a neural oscillator with the phase dynamics given by $\dot{\theta}(t) = f(\theta) + Z(\theta)u(t)$. In this dynamics, θ is the phase of an uncoupled neural oscillators and $u(t)$ represents external current stimulus [11], [20]. The functions $f(\theta)$ and $Z(\theta)$ represent the baseline dynamics and phase response curve (PRC) of the neuron, respectively. By designing a suitable control sequence $u(t)$, the spike time can be advanced or delayed in a desirable manner.

In particular, we consider the phases of two neural oscillators $\theta = (\theta_1, \theta_2)'$ with sinusoidal PRCs and constant drift, i.e., $f(\theta) = (\omega_1, \omega_2)'$ and $Z(\theta) = (z_1 \sin\theta_1, z_2 \sin\theta_2)'$, where ω_1 and ω_2 are the natural frequencies of the two neurons, and z_1 and z_2 are model-dependent constants. Our objective is to steer the ensemble from the initial state $\theta(0)$ to θ_f. In this example, we aim to learn the control sequence that drives the system from $\theta(0) = (0,0)'$ to the target phase $\theta_f = (3\pi/2, 3\pi/2)'$. The parameters used in this example were $\omega_1 = 1, \omega_2 = 2$, and $z_1 = z_2 = 1$. The trajectory and control sequence are shown in Fig. 4. The time discretization in this example was set as $\triangle t = 0.05$ and we limit the time window for the phases to reach $3\pi/2$ at $T = 20$s. During the learning process, a reward of 100 was given when the target phase was reached; a new episode was initiated once the target state is reached or the number of steps in the episode reached an upper bound as in Example 1. The learning parameters used in this Example are listed in Table I and the resulting phase and control trajectories are displayed in Fig. 4.

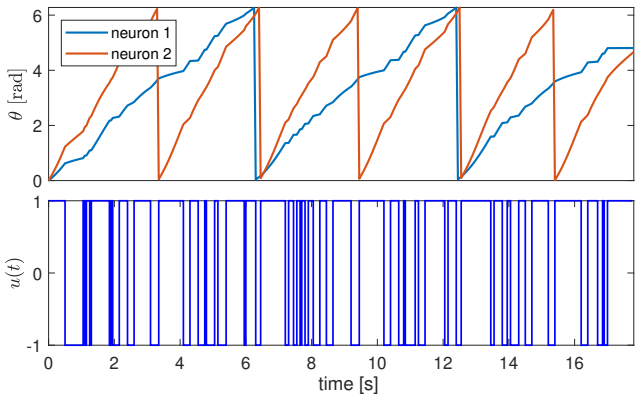

Fig. 4. Example 2: The trajectories and piecewise constant control sequence that drives the two neurons from $(0,0)'$ to $(3\pi/2, 3\pi/2)'$. (Top) Phase trajectories wrapped to $[0, 2\pi]$ for the two neurons. (Bottom) The control sequence learned using aggregated Q-Learning. The available control inputs are $\{+1, -1\}$.

C. Discussions: Defining Learning Objectives

Note that the finite output sequence defined in (5) can be generalized using the notion of central moments to define an infinite sequence that describe a *measure*. In particular, in this context, the states are to be viewed as random variables and the definition in (5) describe the sample moments with respect to the Dirac (or the occupation) measure. This leads to a moment-based control framework for ensemble control systems. Though there have been some results on controlling moments of homogeneous population of dynamical systems [21]–[24], the proposed approach, when generalized, will lead to a fully data-driven moment-based control framework for inhomogeneous ensembles.

In this work, we consider cases where all agents are steered to a common target. In this case, designing the reward function based on the first- and second- order outputs ($M = 2$) is straightforward. However, when the objective is to form a specific spike pattern, the reward function needs to be designed carefully. An intuitive example is making the spike interval of a two-neuron system to be half of the spike period, or equivalently, steering the neural system to a target state $x_f = (\pi, 2\pi)^T$ within time T. Under such condition, the reward should be defined using μ_f of the target state, which is given by $\mu_f = (\mu_1, \mu_2) = (3\pi/2, 4.9348)$.

D. Performance consideration and Sequence control

It is worth mentioning that the performance of the controlled system using the proposed output sequence is insensitive to the size of the network. Here, we provide an illustrative example using a population of neurons described using the phase-models as in Example 2.

To illustrate this idea, using the proposed output sequence, we defined a control policy $u(t) = -\frac{\kappa}{N}(\mu_2^2(t) + \mu_2^4(t))$ to induce simultaneous spikes in an ensemble of neurons, where $\kappa > 0$, N is the number of neurons considered, and the subscript integer represents the order of the output sequence. We use the distribution of system-specific parameters $z_i \in [1.0, 1.2]$ and $\omega_i \in [1, 1.5]$. More specifically, for a 100-neuron system, $(\omega_1, \omega_2, \ldots, \omega_{100}) = (1, 1.0051, 1.0101, \ldots, 1.5)$, $(z_1, z_2, \ldots, z_{100}) = (1, 1.002, 1.004, \ldots, 1.2)$.

In the following analyses, we look at how the spike time deviates under two circumstances: (i) when the size of the population changes, and (ii) when the size of the set K_i of aggregated measurements as defined in (2) is varied for each sampling instant. For case (i) with 100 neurons, the phase trajectories (wrapped to 2π) and the corresponding control input are recorded in Fig. 5. In Fig. 6 (a), a single control law is used to simultaneously steer the phases of 10-100 neurons to induce simultaneous spiking. We can observe that standard deviation of the spike times remain less than 0.05 over different population size for the same control. Also note that the dimensions of μ is not dependent on the number of neurons.

In the second case, we check the control performance of the proposed aggregated measurement-based control policy when the set of measured neurons is limited. By evolving the entire 100-neuron ensemble with only a limited number (10 to 100) of tractable neurons, we were able to simultaneously induce spiking with a satisfactory standard deviation, as shown in Fig. 6 (b). Note that in both the cases, the control input recorded in Fig. 5 appears to be a resetting control,

and finding a bounded control input as in Examples 1 and 2 that achieves the control objective is desired.

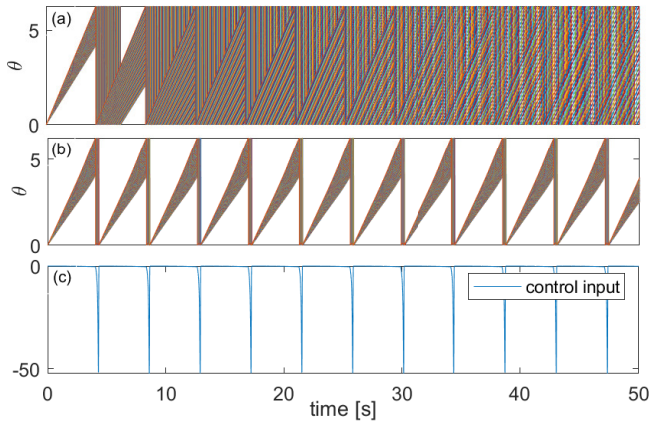

Fig. 5. (a) The phase trajectories of a 100-neuron system without applying a control. (b) The phase trajectories of a 100-neuron system that simultaneously spikes under a given control sequence. (c) The control input designed using the outputs μ_2, μ_4 that is applied to the entire population to generate simultaneous spiking.

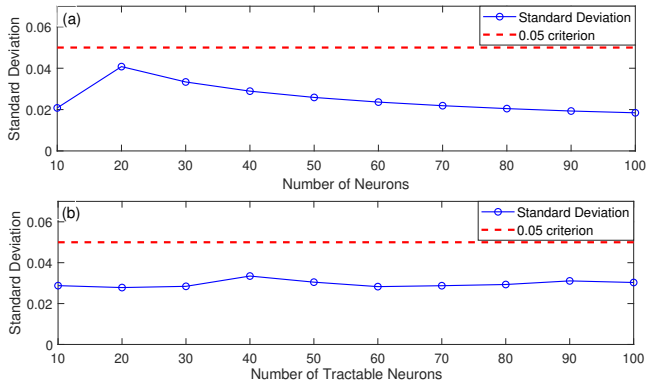

Fig. 6. (a) Standard deviation in the spike time vs the number of neurons in the ensemble. (b) Standard deviation in the spike time vs the number of tractable neurons in the ensemble. Standard deviation in both cases is less than 0.05, as the missing information is compensated by the outputs, which is representative of the entire population.

IV. Conclusions

In this paper, an aggregated Q-Learning scheme is proposed to learn a control sequence for steering a population of dynamical systems. By introducing the notion of aggregated measurements, we first define an output sequence that is representative of the entire population. We then directly learn a control sequence to manipulate the outputs of the ensemble from an initial state to the desired target state. The proposed algorithm circumvents the requirement to derive an analytic solution to obtain a control input but learns the control sequence directly from the aggregated measurement data. We demonstrated the feasibility of the proposed method using two examples. Using numerical analysis, we demonstrated that the proposed approach is scalable, efficient, and can use population-level aggregated measurements to design parameter independent feedback control for an inhomogeneous ensemble system with application to neural oscillators.

References

[1] P. Dayan and L. F. Abbott, *Theoretical neuroscience*. Cambridge, MA: MIT Press, 2001, vol. 806.

[2] G. Hong and C. M. Lieber, "Novel electrode technologies for neural recordings," *Nature Reviews Neuroscience*, p. 1, 2019.

[3] J. T. Ritt and S. Ching, "Neurocontrol: Methods, models and technologies for manipulating dynamics in the brain," in *American Control Conference (ACC), 2015*. IEEE, 2015, pp. 3765–3780.

[4] J.-S. Li, I. Dasanayake, and J. Ruths, "Control and synchronization of neuron ensembles," *IEEE Transactions on Automatic Control*, vol. 58, no. 8, pp. 1919–1930, 2013.

[5] A. Nandi, H. Schättler, J. T. Ritt, and S. Ching, "Fundamental limits of forced asynchronous spiking with integrate and fire dynamics," *The Journal of Mathematical Neuroscience*, vol. 7, no. 1, p. 11, 2017.

[6] Y. Ahmadian, A. M. Packer, R. Yuste, and L. Paninski, "Designing optimal stimuli to control neuronal spike timing," *Journal of Neurophysiology*, vol. 106, no. 2, p. 1038, 2011.

[7] B. Mitchell and L. Petzold, "Control of neural systems at multiple scales using model-free, deep reinforcement learning," *Scientific Reports*, vol. 8, no. 1, p. 10721, 2018.

[8] S. Liu, N. M. Sock, and S. Ching, "Learning-based approaches for controlling neural spiking," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 2827–2832.

[9] V. Narayanan, J. T. Ritt, J. Li, and S. Ching, "A learning framework for controlling spiking neural networks," in *2019 American Control Conference (ACC)*, July 2019, pp. 211–216.

[10] A. A. Faisal, L. P. J. Selen, and D. M. Wolpert, "Noise in the nervous system," *Nature Reviews Neuroscience*, vol. 9, pp. 292–303, 2008.

[11] E. M. Izhikevich, *Dynamical systems in neuroscience*. MIT press, 2007.

[12] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement learning: An introduction*. MIT press, 1998.

[13] K. A. Morgansen and R. W. Brockett, "Optimal regulation and reinforcement learning for the nonholonomic integrator," in *2000 American Control Conference (ACC)*. IEEE, 2000, pp. 462–466.

[14] W. Jiang, V. Narayanan, and J. Li, "Model learning and knowledge sharing for cooperative multiagent systems in stochastic environment," *IEEE Transactions on Cybernetics*, pp. 1–11, 2020.

[15] C. J. C. H. Watkins and P. Dayan, "Q-learning," in *Machine Learning*, 1992, pp. 279–292.

[16] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine*, vol. 9, no. 3, 2009.

[17] A. Becker and T. Bretl, "Approximate steering of a unicycle under bounded model perturbation using ensemble control," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 580–591, 2012.

[18] W. Truccolo, J. A. Donoghue, L. R. Hochberg, E. N. Eskandar, J. R. Madsen, W. S. Anderson, E. N. Brown, E. Halgren, and S. S. Cash, "Single-neuron dynamics in human focal epilepsy," *Nature Neuroscience*, vol. 14, pp. 635–641, 2011.

[19] J. K. Krauss, J. Yianni, T. J. Loher, and T. Z. Aziz, "Deep brain stimulation for dystonia," *Journal of Clinical Neurophysiology*, vol. 21, no. 1, pp. 18–30, 2004.

[20] E. Brown, J. Moehlis, and P. Holmes, "On the phase reduction and response dynamics of neural oscillator populations," *Neural Computation*, vol. 16, no. 4, pp. 673–715, 2004.

[21] R. Brockett, "Notes on the control of the liouville equation," in *Control of partial differential equations*. Springer, 2012, pp. 101–129.

[22] G. Dirr, U. Helmke, and M. Schönlein, "Controlling mean and variance in ensembles of linear systems," *IFAC-PapersOnLine*, vol. 49, no. 18, pp. 1018–1023, 2016.

[23] S. Zeng and F. Allgöwer, "On the moment dynamics of discrete measures," in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 4901–4906.

[24] S. Zeng, H. Ishii, and F. Allgöwer, "On the state estimation problem for discrete ensembles from discrete-time output snapshots," in *2015 American Control Conference (ACC)*, July 2015, pp. 4844–4849.