Bridging Agent Dynamics and Population Behaviors: Scalable Learning for Mean Field Games on Graph via Neural Operators

Xu Chen[1], **Shuo Liu**[2], **Sharon Di**[1]

[1]Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, NY, 10025
[2]Department of Computer Science, Columbia University, New York, NY, 10027
xc2412@columbia.edu, sl4921@columbia.edu, sharon.di@columbia.edu

Abstract

Mean field games (MFGs) are developed to model the decision-making processes of a large number of interacting agents in multi-agent systems. This paper studies mean field games on graphs ($\mathcal{G}$-MFGs). The equilibria of $\mathcal{G}$-MFGs, namely, mean field equilibra (MFE) are challenging to solve for their high-dimensional action space, because each agent has to make decisions when they are at junction nodes or on edges. Furthermore, when the initial population state varies on graphs, we have to recompute MFE, which could be computationally challenging and memory-demanding. To improve the scalability and avoid repeatedly solving $\mathcal{G}$-MFGs every time when its initial state changes, this paper proposes physics-informed graph neural operators (PIGNO). The PIGNO utilizes a graph neural operator to generate population dynamics, given initial population distributions. To better train the neural operator, it leverages the physics knowledge to propagate population state transitions on graphs. A learning algorithm is developed and its performance is evaluated on autonomous driving games on road networks. Our results demonstrate that the PIGNO is scalable and generalizable when tested on unseen initial conditions.

Introduction

Multi-agent systems (MAS) are prevalent in engineering and robotics applications. With a large number of interacting agents in the MAS, solving agents' optimal control could be computationally intractable and not scalable. To solve this challenge, MFGs are (Lasry and Lions 2007; Huang, Malhamé, and Caines 2006) developed to model strategic interactions among many agents who make dynamically optimal decisions, while a population distribution is propagated to represent the state of interacting agents. Since its inception, MFGs have been widely applied to social networks, (Yang et al. 2018), swarm robotics (Elamvazhuthi and Berman 2019) and intelligent transportation (Calderone and Sastry 2017; Huang et al. 2021; Cabannes et al. 2022).

MFGs are micro-macro games that bridge agent dynamics and population behaviors with two coupled processes: individuals' dynamics solved by optimal control (i.e., agent dynamic), and system evolution arising from individual choices (i.e., population behaviors).

In this work, we focus on a class of MFGs (Guéant 2015), namely, mean field games on graphs ($\mathcal{G}$-MFG) where the state space of the agent population is a graph, and agents select a sequence of nodal and edge transitions with a minimum individual cost. Solving these $\mathcal{G}$-MFGs, however, poses the following challenges: (1) With a graph-based state space, the action space expands significantly encompassing both nodes and edges, resulting in a high-dimensional search space. More specifically, the decision-making of a representative agent in $\mathcal{G}$-MFG consists of not only en-route choices at nodes, but also continuous velocity control on edges subject to congestion effects. (2) Existing work mainly assumes that the initial population distribution is fixed. The change of initial population states leads to re-computation of mean field equilibra (MFE), a task that requires computational and memory resources and hinders the practicality of deploying MFG solutions.

To address these challenges, this paper proposes a new learning tool for $\mathcal{G}$-MFGs, namely, physics-informed graph neural operator (PIGNO). The key element is a graph neural operator (GNO), which can generate population dynamics given the initial population distribution. To enhance the training process, the GNO incorporates physics knowledge regarding how agent and population dynamics propagate over the spatiotemporal domain.

Related Work

Researchers have explored various machine learning methods, such as reinforcement learning (RL) (Guo et al. 2019; Subramanian and Mahajan 2019; Perrin et al. 2022; Lauriere et al. 2022), and Physics-Informed Neural Networks (PINN) (Ruthotto et al. 2020; Carmona and Laurière 2021; Germain, Mikael, and Warin 2022; Chen, Liu, and Di 2023a). However, it can be time-consuming and memory-demanding for these learning tools to adapt to changes in initial population density. Specifically, each unique initial condition may require the assignment and retraining of a dedicated neural network to obtain the corresponding MFE. To enhance the scalability of the learning framework for MFGs, Chen et al. (Chen et al. 2023) introduced a physics-informed neural operator (PINO) framework. This framework utilizes a Fourier neural operator (FNO) to establish a functional mapping between mean field equilibrium and boundary conditions. However, the FNO fails to solve $\mathcal{G}$-MFGs because

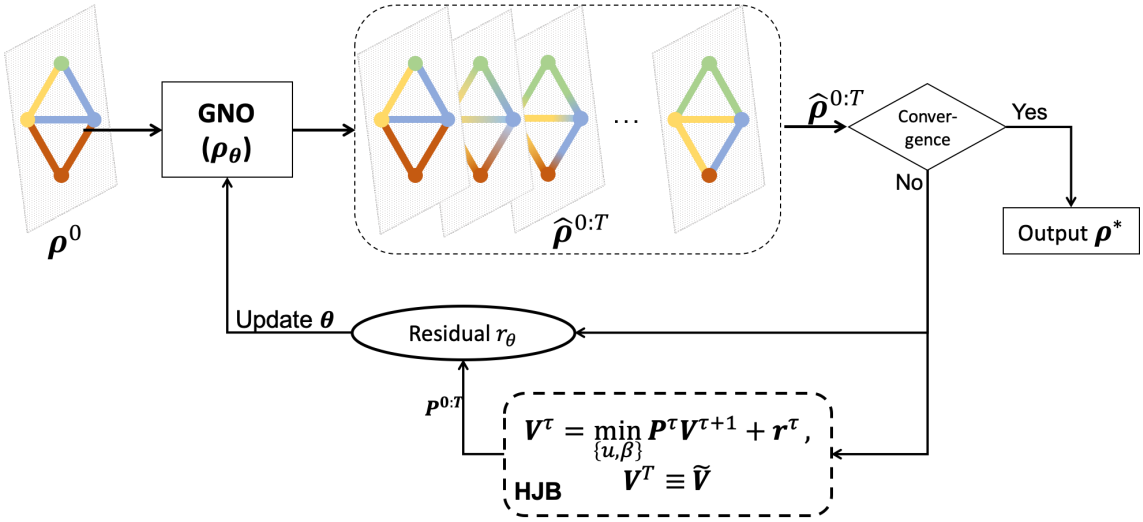


Figure 1: PIGNO for $\mathcal{G}$-MFGs

it cannot directly project information over a graph into a high-dimensional space and generate population dynamics in the graph state space. Therefore, in this paper, we propose a graph neural operator (GNO) that learns mappings between graph-based function spaces to solve $\mathcal{G}$-MFGs. The GNO leverages message passing neural networks (MPNNs) to handle state space and propagate state information efficiently by aggregating the neighbourhood messages.

Our contributions include: (1) We propose a scalable learning framework leveraging PIGNO to solve $\mathcal{G}$-MFGs with various initial population states; (2) We develop a learning algorithm and apply it to autonomous driving games on road networks to evaluate the algorithm performance.

Background

Mean Field Games on Graphs ($\mathcal{G}$-MFG)

Mean field games on graphs ($\mathcal{G}$-MFG) model population dynamics and a generic agent's optimal control on both nodes and edges. A $\mathcal{G}$-MFG consists of a forward FPK and backward HJB equations, which are defined on a graph $\mathcal{G} = \{\mathcal{N}, \mathcal{L}\}$ as follows: A $\mathcal{G}$-MFG with discrete time graph states (Chen, Liu, and Di 2023b) is:

$$[\mathcal{G}\text{-MFG}]:$$

$$(FPK)\ \boldsymbol{\rho}^{\tau+1} = [P^\tau]^T \boldsymbol{\rho}^\tau,\ \boldsymbol{\rho}^0 \equiv \tilde{\boldsymbol{\rho}} \tag{1a}$$

$$(HJB)\ \boldsymbol{V}^\tau = \min_{\boldsymbol{u},\boldsymbol{\beta}} P^\tau \boldsymbol{V}^{\tau+1} + \boldsymbol{r}^\tau,\ \boldsymbol{V}^T \equiv \tilde{\boldsymbol{V}} \tag{1b}$$

$\boldsymbol{\rho}^\tau = [\rho_{ij}^\tau]^T$ is the population density on each edge $(i,j) \in \mathcal{L}$ at time step τ. $\tilde{\boldsymbol{\rho}}$ denotes the initial population density over the graph. The Fokker–Planck (FPK) equation captures the evolution of population state on the graph. The Hamilton–Jacobi–Bellman (HJB) equation captures the optimal control of a generic agent, including the velocity control on edges and route choice on nodes. $\boldsymbol{V}^\tau = [V_{ij}^\tau]^T$ is the value function at each edge. $\tilde{\boldsymbol{V}}$ denotes the terminal cost. $\boldsymbol{u}^\tau = [u_{ij}^\tau]^T$ denotes the exit rate at each edge, which represents the agent's velocity control. $\boldsymbol{\beta}^\tau = [\beta_{ij}^\tau]^T$ is the probability of choosing node j as the next-go-to node at node i, i.e., route choice. $\boldsymbol{r}^\tau = [r_{ij}^\tau]^T$ is the cost incurred by the agent at time step τ. The transition matrix P^τ is determined by $\boldsymbol{u}^\tau$ and $\boldsymbol{\beta}^\tau$. The MFE is denoted by $SOL([\mathcal{G}\text{-MFG}]) = \{\boldsymbol{\rho}^*, \boldsymbol{V}^*, \boldsymbol{u}^*, \boldsymbol{\beta}^*\}$, satisfying Equ. 1.

Graph Neural Operator (GNO)

The graph neural operators (GNOs) are generalized neural networks that can learn functional mappings between high-dimensional spaces (Li et al. 2020). GNO utilizes an MPNN to update space representation according to messages from neighbourhood. In this paper, we adopt a GNO to establish mappings between initial population state $\boldsymbol{\rho}^0$ and population $\boldsymbol{\rho}^*$ at MFE. We leverage the physics knowledge (i.e., FPK and HJB equations) to train the GNO for solving MFE with various initial population densities, eliminating the need to recompute MFE.

Scalable Learning Framework

In this section, we propose a physics-informed graph neural operator (PIGNO) to learn $\mathcal{G}$-MFGs. Figure 1 illustrates the workflow of two couple modules: GNO for population behaviors and HJB for agent dynamics. The GNO and the HJB modules internally depend on each other. In the GNO module, we estimate population density $\boldsymbol{\rho}^{0:T}$ over the graph and update the GNO using a residual defined by the physical rule that captures population dynamics triggered by the transition matrix defined in the FPK equation. In the HJB module, the transition matrix $P^{0:T}$ is obtained given the population density $\boldsymbol{\rho}^{0:T}$. We adopt backward induction (Perrin et al. 2021) to solve the HJB equation since the dynamics of the agents and the cost functions are known in the MFG system. We now delve into the details of the proposed PIGNO.

PIGNO for Population Behaviors

The PIGNO learns a mapping between the initial population distribution and the population distribution over the graph

from time 0 to T. The input of PIGNO is the initial population density $\boldsymbol{\rho}^0$ along with the transition information to propagate population dynamics. The output of PIGNO is the population dynamics over the spatiotemporal domain, denoted by $\hat{\boldsymbol{\rho}} \equiv \hat{\boldsymbol{\rho}}^{0:T}$. The PIGNO is instantiated as the following MPNN: $\forall (i,j) \in \mathcal{L}, \tau = 0, 1, ..., T$

$$\hat{\rho}_{ij}^{\tau} = \rho_\theta(\rho_{ij}^0, \sum_{\forall (k,l) \in \mathcal{L}_{ij}^{\tau}} \kappa_\phi(\rho_{ij}^0, \rho_{kl}^0, e_{ij,kl}^{\tau})) \tag{2}$$

where, $\hat{\rho}_{ij}^{\tau}$ is the population density of edge (i,j) at time τ, $\mathcal{L}_{ij}^{\tau}$ is the set of neighbourhood edges of edge (i,j) at time τ, κ_ϕ is the graph kernel function for ρ_θ, $e_{ij,kl}^{\tau}$ denotes the cumulative message used to propagate population dynamics from time 0 to time τ. $e_{ij,kl}^{\tau}$ indicates the ratio of population entering from edge (k,l) to edge (i,j) till time τ, which is determined by the ratio of population exiting the edge (k,l) (i.e., the velocity control u) and the ratio of population choosing the edge (i,j) as the next-go-to edge (i.e., the route choice β). The MPNN utilizes the initial population distribution and the message to propagate population dynamics in the $\mathcal{G}$-MFG system. The neighbourhood message is transformed by a kernel function κ_ϕ and aggregated as an additional feature to estimate population density.

The PIGNO adopts a physics-informed training scheme, which combines both model-driven and data-driven methods. The training of PIGNO is guided by the residual (marked in red in Figure 1) determined by physical rules of population dynamics. Mathematically, the residual r_θ is:

$$r_\theta = \frac{1}{|\boldsymbol{\rho}^{\mathcal{D}}|} \sum_{\boldsymbol{\rho}^0 \in \boldsymbol{\rho}^{\mathcal{D}}} L_{\boldsymbol{\rho}^0}, \tag{3}$$

where, the set $\boldsymbol{\rho}^{\mathcal{D}}$ contains various initial densities over the graph. $L_{\boldsymbol{\rho}^0}$ is calculated as:

$$L_{\boldsymbol{\rho}^0} = \alpha_0 \cdot \frac{1}{T} \sum_{\tau=0}^{T-1} ||\hat{\boldsymbol{\rho}}^{\tau+1} - [P^{\tau}]^T \hat{\boldsymbol{\rho}}^{\tau}|| + \alpha_1 \cdot ||\hat{\boldsymbol{\rho}}^0 - \boldsymbol{\rho}^0||^2, \tag{4}$$

The first term in $L_{\boldsymbol{\rho}^0}$ evaluates the physical discrepancy based on Equ. 1a. It integrates the residual of the FPK equation, ensuring that the model adheres to established laws of motion. When predicted $\boldsymbol{\rho}$ becomes closer to $\boldsymbol{\rho}^*$ satisfying the FPK equation, the residual gets closer to 0. The second term quantifies the discrepancy between the estimations and the ground truth of the initial density. The observed data comes from the initial distribution of population $\boldsymbol{\rho}^0$. The training of ρ_θ based on observed data follows the traditional supervised learning scheme. α_0 and α_1 are the weight coefficients.

Solution Approach

In this section, we present our learning algorithm (Alg. 1). We first initialize the PIGNO ρ_θ, parameterized by θ. During the ith iteration of the training process, we first sample a batch of initial population densities $\boldsymbol{\rho}^0$. We use each $\boldsymbol{\rho}^0$ to generate the population density over the entire domain $\hat{\boldsymbol{\rho}}^{0:T}$. Given $\hat{\boldsymbol{\rho}}^{0:T}$, we obtain the spatiotemporal transition P for all nodes on the graph by solving the HJB equation. We then update the parameter θ of the neural operator according to the residual. At the end of each iteration, we check the convergence according to:

$$\frac{\sum_{\boldsymbol{\rho}^0 \in \boldsymbol{\rho}^D} |\hat{\boldsymbol{\rho}}^{0:T(i)} - \hat{\boldsymbol{\rho}}^{0:T(i-1)}|}{|\boldsymbol{\rho}^D|} < \epsilon \tag{5}$$

Algorithm 1: PIGNO-MFG

```
Initialize: PIGNO ρ_θ parameterized by θ;
for i ← 0 to I do
    Sample a batch of initial population densities ρ^0 from the set ρ^D of density distribution;
    Generate ρ̂^{0:T(i)} using the PIGNO ρ_θ corresponding to each ρ^0 in the batch;
    for each ρ̂^{0:T(i)} generated by ρ_θ do
        Obtain P^{0:T-1(i)} by solving the HJB.
    end for
    Obtain residual r_{θ(i)} according to Equ. 3;
    Update the PIGNO ρ_θ;
    Check convergence (Equ. 5).
end for
Output MFE
```

Numerical Experiments

In this section, we employ our algorithm to facilitate autonomous driving navigation in traffic networks. As illustrated in Figure 3, a substantial number of autonomous vehicles (AVs) move to destination node 4, with an objective to minimize total costs subject to the congestion effect. We use a representative agent as an example to elaborate on the speed control and density dynamics of the population in this scenario. At node 1, the representative agent first selects edge l_{12}. The agent then drives along edge l_{12} and selects continuous-time-space driving velocities on the edge. The agent selects her next-to-go edge at node 2, following this pattern until she reaches her destination at node 4. These choices regarding her route and speed actively will influence the evolution of population density across the network. The mathematical formulation of this autonomous driving game can be found in (Chen, Liu, and Di 2023b).

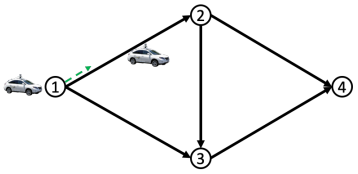
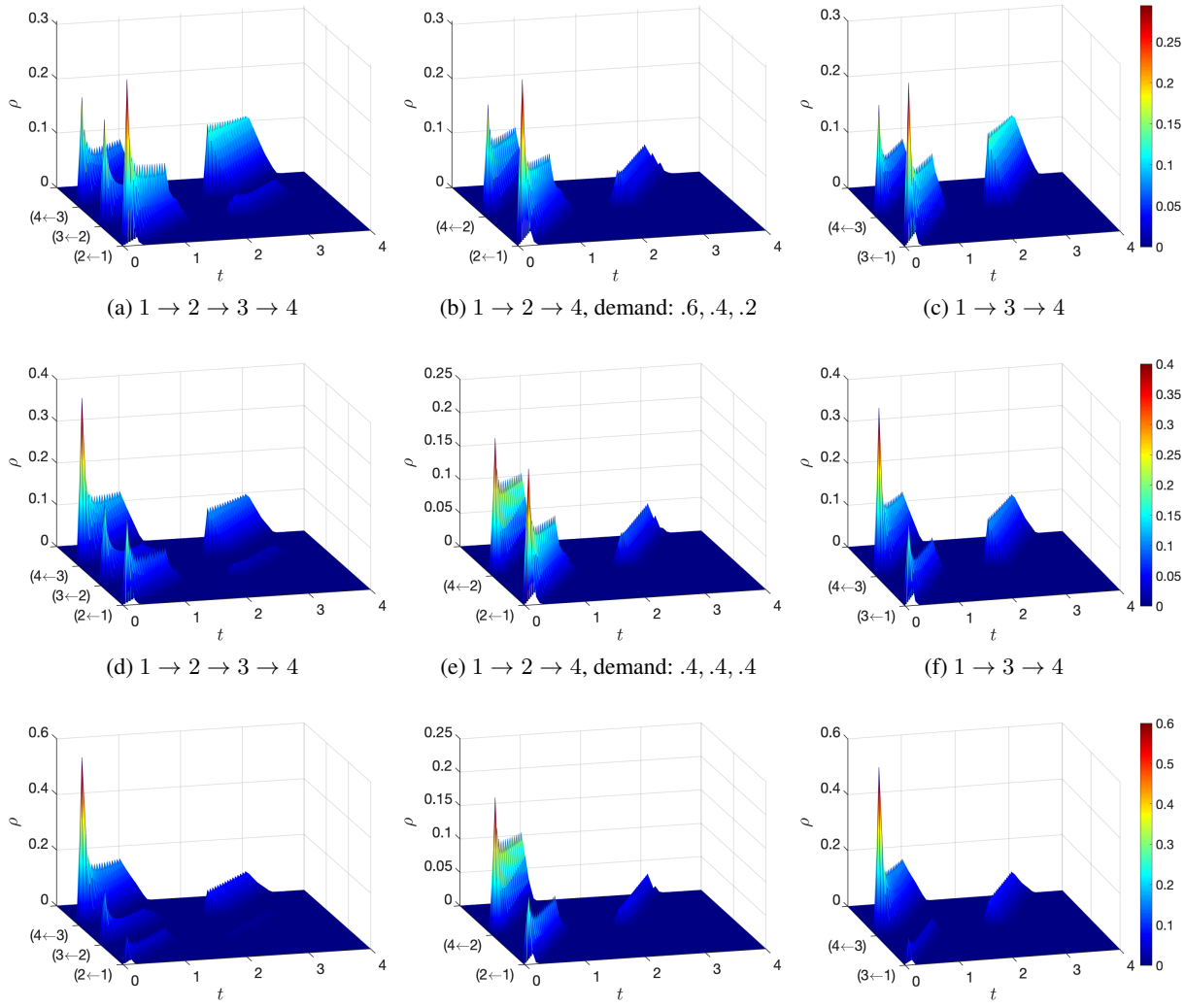


Figure 3: Autonomous driving game on the road network

We construct the initial population state over the network as follows: We assume at time 0, the traffic network is empty. Vehicles enter the road network at origin nodes $1, 2, 3$ and move toward the destination 4. Travel demands at each origin satisfy $d_i \sim \text{Uniform}[0,1], i = 1, 2, 3$. Therefore, each initial population distribution over the network consists of



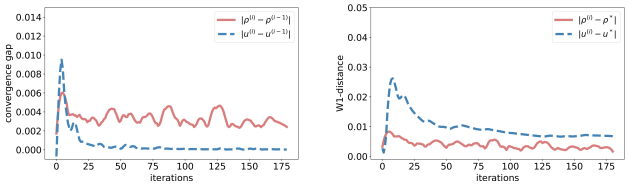
(a) $1 \to 2 \to 3 \to 4$ (b) $1 \to 2 \to 4$, demand: .6, .4, .2 (c) $1 \to 3 \to 4$

(d) $1 \to 2 \to 3 \to 4$ (e) $1 \to 2 \to 4$, demand: .4, .4, .4 (f) $1 \to 3 \to 4$

(g) $1 \to 2 \to 3 \to 4$ (h) $1 \to 2 \to 4$, demand: .2, .4, .6 (i) $1 \to 3 \to 4$

Figure 2: Population density $\boldsymbol{\rho}^*$ along each path on the road network with various travel demands $[d_1, d_2, d_3]$

travel demands at origins (i.e. $[d_1, d_2, d_3]$), which are sampled from three independent uniform distributions.



(a) Convergence gap (b) W1-distance

Figure 4: Algorithm performance

Figure. 4 demonstrates the convergence performance of the algorithm in solving $\mathcal{G}$-MFG. The x-axis represents the iteration index during training, the y-axis displays the convergence gap and the 1-Wasserstein distance, which measures the closeness between our results and the MFE obtained by numerical methods (Chen, Liu, and Di 2023b). The results demonstrate that our algorithm is able to converge stably after 50 iterations.

Figure 2 demonstrates the population density at MFE along three paths on the road network, i.e., $(1 \to 2 \to 3 \to 4)$, $(1 \to 2 \to 4)$, and $(1 \to 3 \to 4)$. The x-axis is the spatial position on the path and the y-axis represents the time. The z-axis represents the population density $\boldsymbol{\rho}^*$ at MFE. The running cost functional form follows a non-separable cost structure with a crossing term of the agent

action and the population density. We visualize the population density at MFE in $\mathcal{G}$-MFG with three initial population states, which are constructed by travel demands $[d_1, d_2, d_3]$: $[.6, .4, .2]$ (See Figure 2a, 2b, 2c), $[.4, .4, .4]$ (See Figure 2d, 2e, 2f) and $[.2, .4, .6]$ (See Figure 2g, 2h, 2i).

Conclusion

In this paper, we propose a scalable learning framework $\mathcal{G}$-MFGs. We apply PIGNO to estimate the population dynamics. We demonstrate the efficiency of this method in autonomous driving games. Our contribution lies in the scalability of PIGNO to handle various initial population densities without recomputing MFEs. Our framework offers a memory, data-efficient approach for solving $\mathcal{G}$-MFGs.

References

Cabannes, T.; Laurière, M.; Perolat, J.; Marinier, R.; Girgin, S.; Perrin, S.; Pietquin, O.; Bayen, A. M.; Goubault, E.; and Elie, R. 2022. Solving N-Player Dynamic Routing Games with Congestion: A Mean-Field Approach. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '22.

Calderone, D.; and Sastry, S. S. 2017. Markov Decision Process Routing Games. In *Proceedings of the 8th International Conference on Cyber-Physical Systems*, ICCPS '17. Association for Computing Machinery.

Carmona, R.; and Laurière, M. 2021. Convergence Analysis of Machine Learning Algorithms for the Numerical Solution of Mean Field Control and Games I: The Ergodic Case. *SIAM Journal on Numerical Analysis*, 59(3): 1455–1485.

Chen, X.; Fu, Y.; Liu, S.; and Di, X. 2023. Physics-Informed Neural Operator for Coupled Forward-Backward Partial Differential Equations. In *1st Workshop on the Synergy of Scientific and Machine Learning Modeling @ ICML2023*.

Chen, X.; Liu, S.; and Di, X. 2023a. A Hybrid Framework of Reinforcement Learning and Physics-Informed Deep Learning for Spatiotemporal Mean Field Games. In *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '23.

Chen, X.; Liu, S.; and Di, X. 2023b. Learning Dual Mean Field Games on Graphs. In *Proceedings of the 26th European Conference on Artificial Intelligence*, ECAI '23.

Elamvazhuthi, K.; and Berman, S. 2019. Mean-field models in swarm robotics: a survey. *Bioinspiration Biomimetics*, 15(1): 015001.

Germain, M.; Mikael, J.; and Warin, X. 2022. Numerical resolution of McKean-Vlasov FBSDEs using neural networks. *Methodology and Computing in Applied Probability*, 1–30.

Guéant, O. 2015. Existence and uniqueness result for mean field games with congestion effect on graphs. *Applied Mathematics and Optimization*, 72(2): 291–303.

Guo, X.; Hu, A.; Xu, R.; and Zhang, J. 2019. Learning Mean-Field Games. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Huang, K.; Chen, X.; Di, X.; and Du, Q. 2021. Dynamic driving and routing games for autonomous vehicles on networks: A mean field game approach. *Transportation Research Part C: Emerging Technologies*, 128: 103189.

Huang, M.; Malhamé, R. P.; and Caines, P. E. 2006. Large population stochastic dynamic games: closed-loop McKean-Vlasov systems and the Nash certainty equivalence principle. *Communications in Information & Systems*, 6(3): 221–252.

Lasry, J.-M.; and Lions, P.-L. 2007. Mean field games. *Japanese journal of mathematics*, 2(1): 229–260.

Lauriere, M.; Perrin, S.; Girgin, S.; Muller, P.; Jain, A.; Cabannes, T.; Piliouras, G.; Perolat, J.; Elie, R.; Pietquin, O.; and Geist, M. 2022. Scalable Deep Reinforcement Learning Algorithms for Mean Field Games. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 12078–12095. PMLR.

Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2020. Multipole Graph Neural Operator for Parametric Partial Differential Equations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20. Curran Associates Inc.

Perrin, S.; LauriÃšre, M.; PÃ©rolat, J.; Ãlie, R.; Geist, M.; and Pietquin, O. 2022. Generalization in Mean Field Games by Learning Master Policies. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36: 9413–9421.

Perrin, S.; Laurière, M.; Pérolat, J.; Geist, M.; Élie, R.; and Pietquin, O. 2021. Mean Field Games Flock! The Reinforcement Learning Way. *Accepted at the 30th International Joint Conference on Artificial Intelligence (IJCAI-21). arXiv preprint arXiv:2105.07933*.

Ruthotto, L.; Osher, S. J.; Li, W.; Nurbekyan, L.; and Fung, S. W. 2020. A machine learning framework for solving high-dimensional mean field game and mean field control problems. *Proceedings of the National Academy of Sciences*, 117(17): 9183–9193.

Subramanian, J.; and Mahajan, A. 2019. Reinforcement Learning in Stationary Mean-Field Games. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, 251–259.

Yang, J.; Ye, X.; Trivedi, R.; Xu, H.; and Zha, H. 2018. Deep Mean Field Games for Learning Optimal Behavior Policy of Large Populations. In *International Conference on Learning Representations*.