Title

Alternating the population and control neural networks to solve high-dimensional stochastic mean-field games

Journal

Proceedings of the National Academy of Sciences of the United States of America, 118(31)

ISSN

0027-8424

Authors

Lin, Alex Tong
Fung, Samy Wu
Li, Wuchen
et al.

Publication Date

2021-08-03

DOI

10.1073/pnas.2024713118

Peer reviewed

Alternating the population and control neural networks to solve high-dimensional stochastic mean-field games

Alex Tong Lin[a,1,2], Samy Wu Fung[a,b,1,2], Wuchen Li[c], Levon Nurbekyan[a], and Stanley J. Osher[a,2]

[a]Department of Mathematics, University of California, Los Angeles, CA 90095; [b]Department of Applied Mathematics and Statistics, Colorado School of Mines, Golden, CO 80401; and [c]Department of Mathematics, University of South Carolina, Columbia, SC 29208

Contributed by Stanley J. Osher, March 4, 2021 (sent for review November 30, 2020; reviewed by Hailiang Liu and Renyuan Xu)

We present APAC-Net, an alternating population and agent control neural network for solving stochastic mean-field games (MFGs). Our algorithm is geared toward high-dimensional instances of MFGs that are not approachable with existing solution methods. We achieve this in two steps. First, we take advantage of the underlying variational primal-dual structure that MFGs exhibit and phrase it as a convex–concave saddle-point problem. Second, we parameterize the value and density functions by two neural networks, respectively. By phrasing the problem in this manner, solving the MFG can be interpreted as a special case of training a generative adversarial network (GAN). We show the potential of our method on up to 100-dimensional MFG problems.

mean-field games | generative adversarial networks | Hamilton–Jacobi–Bellman | optimal control | optimal transport

Mean field games (MFGs) are a class of problems that model large populations of interacting agents. They have been widely used in economics (1–4), finance (2, 5–7), industrial engineering (8–10), swarm robotics (11, 12), epidemic modeling (13, 14), and data science (15–17). In MFGs, a continuum population of small rational agents play a noncooperative differential game on a time horizon $[0, T]$. At the optimum, the agents reach a Nash equilibrium, where they can no longer unilaterally improve their objectives. Given the initial distribution of agents $\rho_0 \in \mathcal{P}(\mathbb{R}^n)$, where $\mathcal{P}(\mathbb{R}^n)$ is the space of all probability densities, the solution to MFGs are obtained by solving the system of partial differential equations (PDEs),

$$\begin{aligned} &-\partial_t \phi - \nu\Delta\phi + H(x, \nabla\phi) = f(x, \rho) && \text{(HJB)} \\ &\partial_t \rho - \nu\Delta\rho - \operatorname{div}(\rho \nabla_p H(x, \nabla\phi)) = 0 && \text{(FP)} \\ &\rho(x, 0) = \rho_0, \quad \phi(x, T) = g(x, \rho(\cdot, T)), \end{aligned} \qquad [1.1]$$

which couples a Hamilton–Jacobi–Bellman (HJB) equation and a Fokker–Planck (FP) equation. Here, $\phi\colon \mathbb{R}^n \times [0, T] \to \mathbb{R}$ is the value function, i.e., the policy that guides the agents; $H\colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is the Hamiltonian, which describes the physics of the environment; $\rho(\cdot, t) \in \mathcal{P}(\mathbb{R}^n)$ is the distribution of agents at time t; $f\colon \mathbb{R}^n \times \mathcal{P}(\mathbb{R}^n) \to \mathbb{R}$ denotes the interaction between the agents and the population; and $g\colon \mathbb{R}^n \times \mathcal{P}(\mathbb{R}^n) \to \mathbb{R}$ is the terminal condition, which guides the agents to the final distribution. Under standard assumptions, i.e., convexity of H in the second variable, and monotonicity of f and g—namely, that

$$\int_{\mathbb{R}^n} (f(y, \rho_1) - f(y, \rho_2)) d(\rho_1 - \rho_2)(y) > 0 \text{ for all } \rho_1 \neq \rho_2,$$

and similarly for g—then the solution to Eq. **1.1** exists and is unique. See refs. 18–20 for more details. Although there are a plethora of fast solvers for the solution of Eq. **1.1** in two and three dimensions (20–25), numerical methods for solving Eq. **1.1** in high dimensions are practically nonexistent due to the need for grid-based spatial discretization. These grid-based methods are prone to the curse of dimensionality, i.e., their computational complexity grows exponentially with spatial dimension (26). Thus, grid-based methods cannot be tractably used on, e.g., modeling an energy-efficient heating, ventilation, and air-conditioning system in a complex building, where the dimensions can be as high as 1,000 (27).

Our Contribution. We present APAC-Net, an alternating population and agent control neural network approach geared toward high-dimensional MFGs in the deterministic and stochastic case. To this end, we phrase the MFG problem as a saddle-point problem (19, 22, 28) and parameterize the value function and the density function. This formulation provides some alleviation to the curse of dimensionality by avoiding the use of spatial grids or uniformly sampling in high dimensions. While spatial grids for MFGs are also avoided in ref. 29, their work is limited to the deterministic setting ($\nu = 0$). APAC-Net models high-dimensional MFGs in the stochastic setting ($\nu > 0$). It does this by drawing from a natural connection between MFGs and generative adversarial neural networks (GANs) (30) (*Section 3*), a powerful class of generative models that have shown remarkable success on various types of datasets (30–35).

1. Variational Primal-Dual Formulation of MFGs

We derive the mathematical formulation of MFGs for our framework; in particular, we arrive at a primal-dual convex–concave formulation tailored for our alternating networks approach. An

Significance

Mean-field games (MFGs) is an emerging field that models large populations of agents. They play a central role in many disciplines, such as economics, data science, and engineering. Since many applications come in the form of high-dimensional stochastic MFGs, numerical methods that use spatial grids are prone to the curse of dimensionality. To this end, we exploit the variational structure of potential MFGs and reformulate it as a generative adversarial network (GAN) training problem. This reformulation allays a bit the curse of dimensionality when solving high-dimensional MFGs in the stochastic setting, by avoiding spatial grids or uniform sampling in high dimensions, and instead utilizes the structure of the MFG and its connection with GANs.

Author contributions: A.T.L., S.W.F., W.L., L.N., and S.J.O. designed research; A.T.L., S.W.F., and L.N. performed research; and A.T.L., S.W.F., W.L., L.N., and S.J.O. wrote the paper.

Reviewers: H.L., Iowa State University; and R.X., University of Southern California.

The authors declare no competing interest.

[1]A.T.L and S.W.F contributed equally to this work.

[2]To whom correspondence may be addressed. Email: atlin@math.ucla.edu, swufung@mines.edu, or sjo@math.ucla.edu.

Published August 2, 2021.

MFG system Eq. **1.1** is called potential, if there exist functionals $\mathcal{F}, \mathcal{G}$ such that

$$\delta_\rho \mathcal{F} = f(x,\rho) \quad \text{and} \quad \delta_\rho \mathcal{G} = g(x,\rho), \tag{2.1}$$

where

$$\begin{aligned} \langle \delta_\rho \mathcal{F}(\rho), \mu \rangle &= \lim_{h\to 0} \frac{\mathcal{F}(\rho + h\mu) - \mathcal{F}(\rho)}{h}, \ \forall\, \mu, \\ \langle \delta_\rho \mathcal{G}(\rho), \mu \rangle &= \lim_{h\to 0} \frac{\mathcal{G}(\rho + h\mu) - \mathcal{G}(\rho)}{h}, \ \forall\, \mu. \end{aligned} \tag{2.2}$$

That is, there exist functionals $\mathcal{F}, \mathcal{G}$ such that their variational derivatives with respect to ρ are the interaction and terminal costs f and g from Eq. **1.1**. A critical feature of potential MFGs is that the solution to Eq. **1.1** can be formulated as the solution to a convex–concave saddle-point optimization problem. To this end, we begin by stating Eq. **1.1** as a variational problem (19, 22) akin to the Benamou–Brenier formulation for the Optimal Transport (OT) problem:

$$\begin{aligned} &\inf_{\rho, v} \int_0^T \left\{ \int_\Omega \rho(x,t) L(x, v(x,t))\, dx + \mathcal{F}(\rho(\cdot,t)) \right\} dt + \mathcal{G}(\rho(\cdot,T)) \\ &\text{s.t. } \partial_t \rho - \nu \Delta \rho + \nabla \cdot (\rho v) = 0, \quad \rho(x,0) = \rho_0(x), \end{aligned} \tag{2.3}$$

where $L\colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is the Lagrangian function corresponding to the Legendre transform of the Hamiltonian H, $\mathcal{F}, \mathcal{G}\colon \mathcal{P}(\mathbb{R}^n) \to \mathbb{R}$ are mean-field interaction terms, and $v\colon \mathbb{R}^n \times [0,T] \to \mathbb{R}^n$ is the velocity field. Next, setting ϕ as a Lagrange multiplier, we insert the PDE constraint into the objective to get

$$\begin{aligned} &\sup_{\phi} \inf_{\rho(x,0)=\rho_0(x), v} \\ &\int_0^T \left\{ \int_\Omega \rho(x,t) L(x, v(x,t))\, dx + \mathcal{F}(\rho(\cdot,t)) \right\} dt + \mathcal{G}(\rho(\cdot,T)) \\ &- \int_0^T \int_\Omega \phi(x,t) \left(\partial_t \rho - \nu \Delta \rho + \nabla \cdot (\rho(x,t) v(x,t)) \right) dx\, dt. \end{aligned} \tag{2.4}$$

Finally, integrating by parts and minimizing with respect to v to obtain the Hamiltonian via $H(x,p) = \inf_v \{-p \cdot v + L(x,v)\}$, we obtain

$$\begin{aligned} &\inf_{\rho(x,0)=\rho_0(x)} \sup_{\phi} \\ &\int_0^T \left\{ \int_\Omega (\partial_t \phi + \nu \Delta \phi - H(x, \nabla \phi)) \rho(x,t)\, dx + \mathcal{F}(\rho(\cdot,t)) \right\} dt \\ &+ \int_\Omega \phi(x,0) \rho_0(x) dx - \int_\Omega \phi(x,T) \rho(x,T) dx + \mathcal{G}(\rho(\cdot,T)). \end{aligned} \tag{2.5}$$

Here, our approach follows that of refs. 22, 36, and 37. This formula can also be obtained in the context of HJB equations in density spaces (23) or by integrating the HJB and the FP equations in Eq. **1.1** with respect to ρ and ϕ, respectively (28). In ref. 28, it was observed that all MFG systems admit an infinite-dimensional two-player general-sum game formulation, and the potential MFGs are the ones that correspond to zero-sum games. In this interpretation, Player 1 represents the *mean field* or the *population as a whole*, and their strategy is the population density ρ. Furthermore, Player 2 represents the *generic agent*, and their strategy is the value function ϕ. The aim of Player 2 is to provide a strategy that yields the best response of a generic agent against the population. This interpretation is in accord with the intuition behind GANs, as the key observation is that under mild assumptions on $\mathcal{F}$ and $\mathcal{G}$, each spatial integral is really an expectation from ρ. The formulation Eq. **2.5** is the cornerstone of our method.

2. Connections to GANs

GANs. In GANs (30), we have a discriminator and generator, and the goal is to obtain a generator that is able to produce samples from a desired distribution. The generator does this by taking samples from a known distribution $\mathcal{N}$ and transforming them into samples from the desired distribution. Meanwhile, the purpose of the discriminator is to aid the optimization of the generator. Given a generator network G_θ and a discriminator network D_ω, the original GAN objective is to find an equilibrium to the minimax problem

$$\inf_{G_\theta} \sup_{D_\omega} \mathbb{E}_{x\sim\rho_0} [\log D_\omega(x)] + \mathbb{E}_{z\sim\mathcal{N}} [\log(1 - D_\omega(G_\theta(z)))]. \tag{3.1}$$

Here, the discriminator acts as a classifier that attempts to distinguish real images from fake/generated images, and the goal of the generator is to produce samples that "fool" the discriminator.

Wasserstein GANs. In Wasserstein GANs (31), the motivation is drawn from OT theory, where now the objective function is changed to the Wasserstein-1 (W1) distance in the Kantorovich–Rubenstein dual formulation

$$\inf_{G_\theta} \sup_{D_\omega} \mathbb{E}_{x\sim\rho_0} [D_\omega(x)] - \mathbb{E}_{z\sim\mathcal{N}} [D_\omega(G_\theta(z))], \ \text{s.t. } \|\nabla D_\omega\| \le 1, \tag{3.2}$$

and the discriminator is required to be 1-Lipschitz. In this setting, the goal of the discriminator is to compute the W1 distance between the distribution of ρ_0 and $G_\theta(z)$. In practice, using the W1 distance helps prevent the generator from suffering "mode collapse," a situation where the generator produces samples from only one mode of the distribution ρ_0; for instance, if ρ_0 is the distribution of images of handwritten digits, then mode collapse entails producing only, say, the 0 digit. Originally, weight-clipping was to enforce the Lipschitz condition of the discriminator network (31), but an improved method using a penalty on the gradient was used in ref. 32.

GANs ↔ MFGs. A Wasserstein GAN can be seen as a particular instance of a deterministic MFG (19, 22, 38). Specifically, consider the MFG Eq. **2.5** in the following setting. Let $\nu = 0$, $\mathcal{G}$ be a hard constraint with target measure ρ_T (as in OT), let $\mathcal{F} = 0$, and let H be the Hamiltonian defined by

$$H(x,p) = \mathbb{1}_{\|p\| \le 1} = \begin{cases} 0 & \|p\| \le 1 \\ \infty & \text{otherwise} \end{cases}, \tag{3.3}$$

where we note that this Hamiltonian arises when the Lagrangian is given by $L(x,v) = \|v\|_2$. Then, Eq. **2.5** reduces to,

$$\begin{aligned} &\sup_{\phi} \int_\Omega \phi(x) \rho_0(x)\, dx - \int_\Omega \phi(x) \rho_T(x)\, dx \\ &\quad \text{s.t. } \|\nabla \phi(x)\| \le 1, \end{aligned} \tag{3.4}$$

where we note that the optimization in ρ leads to $\partial_t \phi - H(x, \nabla\phi) = 0$. And since $H(p) = \mathbb{1}_{\|p\|\le 1}$, we have that $\partial_t \phi = 0$, and $\phi(x,t) = \phi(x)$ for all t. We observe that the above is precisely the W1 distance in the Kantorovich–Rubenstein duality (39).

3. APAC-Net

Rather than discretizing the domain and solving for the function values at grid-points, APAC-Net avoids them by parameterizing

the function and solving for the function itself. We make a small commentary that this rhymes with history, in moving from the Riemann to the Lebesgue integral: The former focused on gridpoints of the domain, whereas the latter focused on the function values.

The training process for our MFG is similar to that of GANs. We initialize the neural networks $N_\omega(x,t)$ and $N_\theta(z,t)$. We then let

$$\begin{aligned}\phi_\omega(x,t) &= (1-t)N_\omega(x,t) + tg(x),\\ G_\theta(z,t) &= (1-t)z + tN_\theta(z,t),\end{aligned} \qquad \textbf{[4.1]}$$

where $z \sim \rho_0$ are samples drawn from the initial distribution. Thus, we set $\rho(\cdot,t) = G_\theta(\cdot,t)\#\rho_0$, i.e., the push-forward of ρ_0. We make a small comment that this idea of sampling to solve a PDE is similar in spirit to that of the Feynman–Kac approach. In this setting, we train $G_\theta(\cdot,t)$ to produce samples from $\rho(\cdot,t)$. We note that ϕ_ω and G_θ automatically satisfy the terminal and initial condition, respectively. In particular, G_θ produces samples from ρ_0 at $t=0$.

Our strategy for training consists of alternately training G_θ (the population) and ϕ_ω (the value function for an individual agent). Intuitively, this means we are *alternating the population and agent control neural networks* (APAC-Net) in order to find the equilibrium. Specifically, we train ϕ_ω by first sampling a batch $\{z_b\}_{b=1}^B$ from the given initial density ρ_0 and $\{t_b\}_{b=1}^B$ uniformly from $[0,1]$; so, we are really sampling from the products of the densities ρ_0 and Unif$([0,1])$. Next, we compute the push-forward $x_b = G_\theta(z_b,t_b)$ for $b=1,\dots,B$. We then compute the loss,

$$\begin{aligned}\text{loss}_\phi = \frac{1}{B}\sum_{b=1}^B \phi_\omega(x_b,0) + \frac{1}{B}\sum_{b=1}^B \partial_t\phi_\omega(x_b,t_b)\\ + \nu\Delta\phi_\omega(x_b,t_b) - H(\nabla_x\phi_\omega(x_b,t_b)),\end{aligned} \qquad \textbf{[4.2]}$$

where we can optionally add a regularization term

$$\begin{aligned}\text{penalty} = \lambda\frac{1}{B}\sum_{b=1}^B \|\partial_t\phi_\omega(x_b,t_b) + \nu\Delta\phi_\omega(x_b,t_b)\\ - H(\nabla_x\phi_\omega(x_b,t_b)) + f(x_b,t_b)\|,\end{aligned} \qquad \textbf{[4.3]}$$

to penalize deviations from the HJB equations (29, 40). This extra regularization term has also been found effective in, e.g., Wasserstein GANs (3), where the norm of the gradient (i.e., the HJB equations) is penalized. Finally, we back-propagate the loss to the weights of ϕ_w. To train the generator, we again sample $\{z_b\}_{b=1}^B$ and $\{t_b\}_{b=1}^B$ as before and compute

$$\begin{aligned}\text{loss}_G = \frac{1}{B}\sum_{b=1}^B \partial_t\phi_\omega(G_\theta(z_b),t_b) + \nu\Delta\phi_\omega(G_\theta(z_b),t_b)\\ - H(\nabla_x\phi_\omega(G_\theta(z_b),t_b)) + f(G_\theta(z_b),t_b).\end{aligned} \qquad \textbf{[4.4]}$$

Finally, we back-propagate this loss with respect to the weights of G_θ (see Algorithm [Alg.] 1).

Algorithm 1: APAC-Net

Require: ν diffusion parameter, G terminal cost, H Hamiltonian, f interaction term.
Require: Initialize neural networks N_ω and N_θ, batch size B
Require: Set ϕ_ω and G_θ as in Eq. **4.1**
while not converged **do**
  train ϕ_ω:
  Sample batch $\{(z_b,t_b)\}_{b=1}^B$ where $z_b \sim \rho_0$ and $t_b \sim$ Unif$(0,T)$
  $x_b \leftarrow G_\theta(z_b,t_b)$ for $b=1,\dots,B$.
  $\ell_0 \leftarrow \frac{1}{B}\sum_{b=1}^B \phi_\omega(x_b,0)$
  $\ell_t \leftarrow \frac{1}{B}\sum_{b=1}^B \partial_t\phi_\omega(x_b,t_b) + \nu\Delta\phi_\omega(x_b,t_b)$
    $-H(\nabla_x\phi_\omega(x_b,t_b))$
  $\ell_{\text{HJB}} \leftarrow \lambda\frac{1}{B}\sum_{b=1}^B \|\partial_t\phi_\omega(x_b,t_b) + \nu\Delta\phi_\omega(x_b,t_b)$
    $-H(\nabla_x\phi_\omega(x_b,t_b)) + f(x_b,t_b)\|$
  Back-propagate the loss $\ell_{\text{total}} = \ell_0 + \ell_t + \ell_{\text{HJB}}$ to ω weights.
  train G_θ:
  Sample batch $\{(z_b,t_b)\}_{b=1}^B$ where $z_b \sim \rho_0$ and $t_b \sim$ Unif$(0,T)$
  $\ell_t \leftarrow \frac{1}{B}\sum_{b=1}^B \partial_t\phi_\omega(G_\theta(z_b,t_b),t_b) + \nu\Delta\phi_\omega(G_\theta(z_b,t_b),t_b)$
    $-H(\nabla_x\phi_\omega(G_\theta(z_b,t_b),t_b)) + f(G_\theta(z_b,t_b),t_b)$
  Back-propagate the loss $\ell_{\text{total}} = \ell_t$ to θ weights.

4. Related Works

High-Dimensional MFGs and Optimal Control. To the best of our knowledge, the first work to solve MFGs efficiently in high dimensions ($d=100$) was done in ref. 29. Their work consisted of using Lagrangian coordinates and parameterizing the value function using a neural network. Finally, to estimate the densities, the instantaneous change of variables formula (41). This combination allowed them to successfully avoid using spatial grids when solving deterministic MFG problems ($\nu=0$) with quadratic Hamiltonians. Besides only computing MFGs with $\nu=0$, another limitation is that for nonquadratic Hamiltonians, the instantaneous change of variables formula may lead to high computational costs when estimating the density. APAC-Net circumvents this limitation by rephrasing the MFG as a saddle-point problem Eq. **2.5** and using a GAN-based approach to train two neural networks instead. For problems involving high-dimensional optimal control and differential games, spatial grids were also avoided (20, 23, 24, 42, 43). However, these methods are based on generating individual trajectories per agent and cannot be directly applied to MFGs without spatial discretization of the density, thus limiting their use in high dimensions.

Reinforcement Learning. Our work bears connections with multiagent reinforcement learning (RL), where neither the Lagrangian L nor the dynamics (constraint) in Eq. **2.3** are known. Here, a key difference is that multiagent RL generally considers a finite number of players. Ref. 44 proposes a primal-dual distributed method for multiagent RL. Refs. 16 and 45 propose a Q-Learning approach to solve these multiagent RL problems. Ref. 17 studies the convergence of policy gradient methods on mean-field RL problems, i.e., problems where the agents try instead to learn the control, which is socially optimal for the entire population. Ref. 46 uses an inverse RL approach to learn the MFG model along with its reward function. Ref. 47 proposes an actor–critic method for finding the Nash equilibrium in linear-quadratic MFGs and establishing linear convergence.

Generative Modeling with OT. There is a class of works that focus on using OT, a class of MFGs, to solve problems arising in data science and, in particular, GANs. Ref. 48 presents a tractable method to train large-scale generative models using the Sinkhorn distance, which consist of loss functions that interpolate between Wasserstein (OT) distance and Maximum Mean Discrepancy (MMD). Ref. 49 proposes a mini-batch MMD-based distance to improve training GANs. Ref. 50 proposes a class of regularized Wasserstein GAN problems with theoretical guarantees. Ref. 51 uses a trained discriminator from GANs to further improve the quality of generated samples. Ref. 52 phrases the adversarial problem as a matching problem in order to avoid solving a minimax problem. Finally, ref. 53 provides an excellent survey

on recent numerical methods for OT and their applications to GANs.

GAN-Based Approach for MFGs. Our work is most similar to ref. 38, where a connection between MFGs and GANs is also made. However, APAC-Net differs from ref. 38 in two fundamental ways. First, instead of choosing the value function to be the generator, we set the *density* function as the generator. This choice is motivated by the fact that the generator outputs samples from a desired distribution. It is also aligned with other generative modeling techniques arising in continuous normalizing flows (40, 54–56). Second, rather than setting the generator/discriminator losses as the residual errors of Eq. **1.1**, we follow the works of refs. 19, 22, 23, and 28 and utilize the underlying variational primal-dual structure of MFGs (see Eq. **2.5**); this allows us to arrive at the Kantorovich–Rubenstein dual formulation of Wasserstein GANs (39).

5. Numerical Experiments

We demonstrate the potential of APAC-Net on a series of high-dimensional MFG problems. We note that the below examples would not have been possible with previous grid-based methods, as the number of grid-points required to even start the problem grows exponentially with the dimension. We also illustrate the behavior of the MFG solutions for different values of ν and use an analytical solution to illustrate the accuracy of APAC-Net. We also provide additional high-dimensional results in Appendix B.

Experimental Setup. We assume without loss of generality $T=1$, unless otherwise stated. In all experiments, our neural networks have three hidden layers, with 100 hidden units per layer. We use a residual neural network (ResNet) for both networks, with skip connection weight 0.5. For ϕ_ω, we use the Tanh activation function, and for G_θ, we use the ReLU activation function. For training, we use ADAM with $\beta=(0.5,0.9)$, learning rate 4×10^{-4} for ϕ_ω, learning rate 1×10^{-4} for G_θ, weight decay of 10^{-4} for both networks, batch size 50, and $\lambda=1$ (the HJB penalty parameter) in Alg. 1.

The Hamiltonians in our experiments have the form

$$H(x,p,t)=c\|p\|_2+f(x,\rho(x,t)), \qquad \textbf{[6.1]}$$

where $f(x,\rho(x,t))$ varies with the environment (either avoiding obstacles or avoiding congestion, etc.), and c is a constant (that represents maximal speed). Furthermore, we choose as terminal cost

$$\mathcal{G}(\rho(\cdot,T))=\int_\Omega \|x-x_T\|_2\rho(x,T)\,dx, \qquad \textbf{[6.2]}$$

which is the distance between the population and a target destination. To allow for verification of the high-dimensional solutions, we set the obstacle and congestion costs to only affect the first two dimensions. In Figs. 1–4 and 6, time is represented by color. Specifically, blue denotes starting time, red denotes final time, and the intermediate colors denote intermediate times. We also plot the HJB residual error—that is, ℓ_{HJB} in Alg. 1—on 4,096 fixed sampled points, which helps us monitor the convergence of APAC-Net. As in standard machine-learning methods,

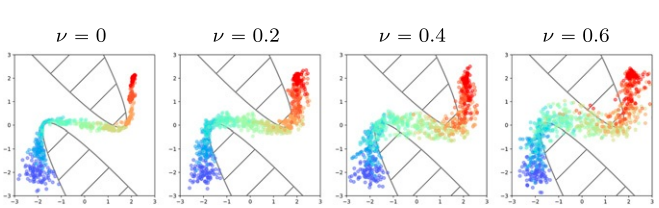

Fig. 1. Comparison of 2D solutions for different values of ν. The agents start at the blue points ($t=0$) and end at the red points ($t=1$).

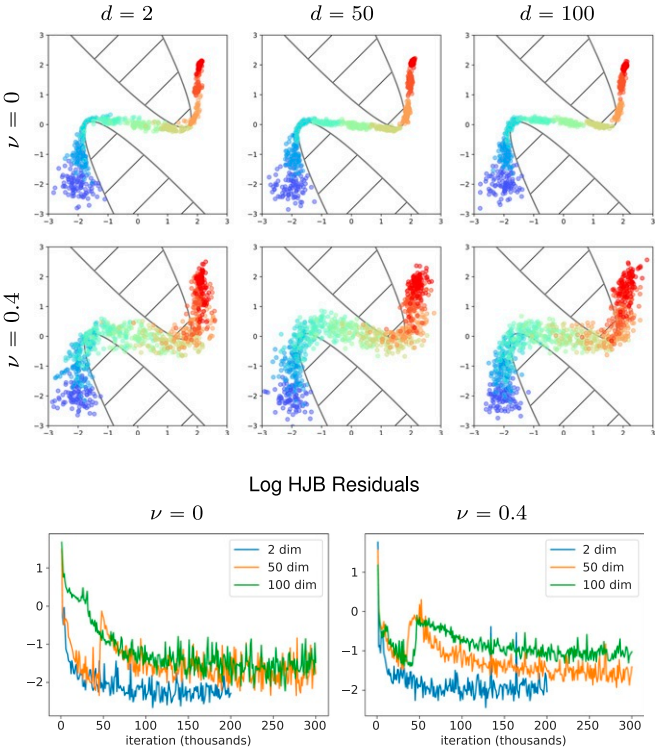

Fig. 2. Computation of the obstacle problem in dimensions (dim) 2, 50, and 100 with stochasticity parameter $\nu=0$ and 0.4. For dimensions 50 and 100, we plot the first two dimensions.

all of the plots in this section are generated by using *validation* data—i.e., data not used in training—in order to gauge generalizability of APAC-Net. Further details, as well as additional experiments, can be found in the appendix.

Effect of Stochasticity Parameter ν. We investigate the effect of the stochasticity parameter ν on the behavior of the MFG solutions. In Fig. 1, we show the solutions for two-dimensional (2D) MFGs using $\nu=0, 0.2, 0.4$, and 0.6. As ν increases, the density of agents widens along the paths due to the added diffusion term in the HJB and FP equations in Eq. **1.1**. The hatched markings are obstacles and are described in Eqs. **6.4** and **6.5**. These results are consistent with those in ref. 57.

Obstacles. We compute the solution to an MFG where the agents are required to avoid obstacles. In this case, we let

$$\begin{aligned} f(x_1,x_2,\dots,x_d)=\gamma_{\text{obst}}&(\max\{f_1(x_1,x_2),0\}\\ &+\max\{f_2(x_1,x_2),0\}), \end{aligned} \qquad \textbf{[6.3]}$$

with $\gamma_{\text{obst}}=5$, and denoting $R=\begin{pmatrix}\cos(\theta)&-\sin(\theta)\\ \sin(\theta)&\cos(\theta)\end{pmatrix}$ with $\theta=\pi/5$, $Q=\begin{pmatrix}50\\00\end{pmatrix}$, and $b=(0,2)$, then

$$\begin{aligned} f_1(x_1,x_2)&=-v^\top Qv-b\cdot v-1,\\ &\text{with } v=((x_1,x_2)-(-2,0.5))R. \end{aligned} \qquad \textbf{[6.4]}$$

Similarly, we let

$$\begin{aligned} f_2(x_1,x_2)&=-w^\top Qw+b\cdot w-1,\\ &\text{with } w=((x_1,x_2)-(2,-0.5))R. \end{aligned} \qquad \textbf{[6.5]}$$

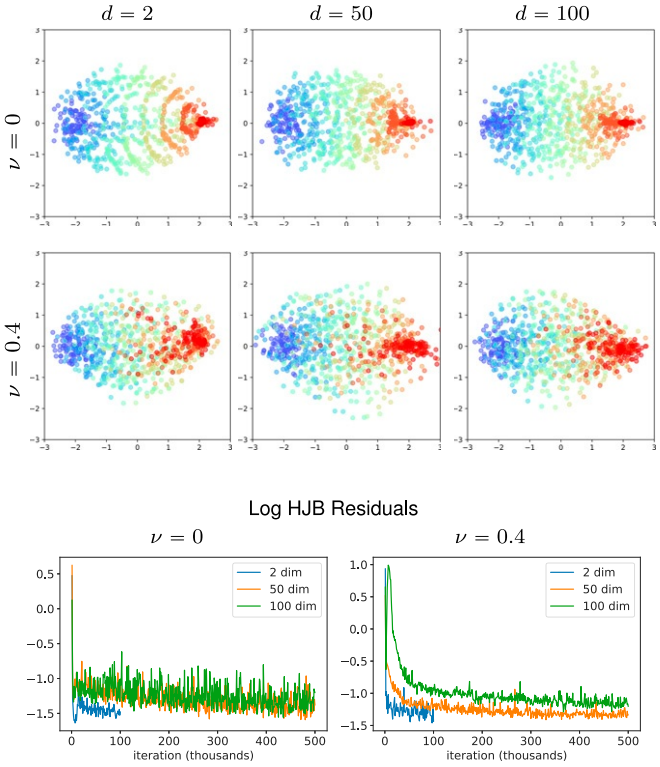

Fig. 3. Computation of the congestion problem in dimensions (dim) 2, 50, and 100 with stochasticity parameter $\nu = 0$ and 0.4. For dimensions 50 and 100, we plot the first two dimensions.

The obstacles f_1 and f_2 are shown in hatched markings in Fig. 2. Our initial density ρ_0 is a Gaussian centered at $(-2, -2, 0, \ldots, 0)$ with SD $1/\sqrt{10} \approx 0.32$, and the terminal function is $g(x) = \|(x_1, x_2) - (2, 2)\|_2$. We chose $c = 8$ in Eq. **6.1**. The numerical results are shown in Fig. 2. Observe that results are similar across dimensions, verifying our high-dimensional computation. We run the 2D problem for 200,000 (200 k) iterations and the 50- and 100-dimensional problems for 300 k iterations.

Congestion. We choose the interaction term to penalize congestion, so that the agents are encouraged to spread out. In particular, we have

$$\mathcal{F}(\rho(\cdot, t)) = \int_\Omega \int_\Omega \frac{1}{\|(x_1, x_2) - (y_1, y_2)\|^2 + 1} \, d\rho(x, t) \, d\rho(y, t), \qquad \textbf{[6.6]}$$

which is the (bounded) inverse squared distance, averaged over pairs of agents. Computationally, we sample from ρ twice and then calculate the integrand. Here, our initial density ρ_0 is a Gaussian centered at $(-2, 0, -2, \ldots, -2)$ with SD $1/\sqrt{10} \approx 0.32$, the terminal function is $\mathcal{G}(x) = \|(x_1, x_2) - (2, 0)\|_2$, and we chose $c = 5$ in Eq. **6.1**. Results are shown in Fig. 3, where we see qualitatively similar results across dimensions. We run the 2D problem for 100 k iterations and the 50- and 100-dimensional problems for 500 k iterations.

Congestion with Bottleneck Obstacle. We combine the congestion problem with a bottleneck obstacle. The congestion penalization is the same as Eq. **6.6**, and the obstacle represents a bottleneck—thus, agents are encouraged to spread out, but must squeeze together to avoid the obstacle. The initial density, terminal functions, c in Eq. **6.1**, and the expression penalizing congestion are the same as in the congestion experiment above. The obstacle is chosen to be

$$f(v) = \gamma_{\text{obst}} \max\left\{-v^\top \begin{pmatrix} 5 & 0 \\ 0 & -1 \end{pmatrix} v - 0.1, 0\right\}, \quad \text{with } v = (x_1, x_2) \qquad \textbf{[6.7]}$$

with $\gamma_{\text{obst}} = 5$. As intuitively expected, the agents spread out before and after the bottleneck, but squeeze together in order to avoid the obstacle (Fig. 4). We run the 2D problem for 100 k iterations and the 50- and 100-dimensional problems for 500 k iterations. We observe similar results across dimensions.

Analytic Comparison. We verify our method by comparing it to an analytic solution for dimensions 2, 50, and 100 with congestion ($\gamma = 0.1$) and without congestion ($\gamma = 0$). For

$$f = \gamma \ln(\rho), \quad H(x, p) = \frac{\|p\|^2}{2} - \frac{\beta \|x\|^2}{2}, \qquad g(x) = \frac{\alpha |x|^2}{2} - \left(\nu d\alpha + \frac{\gamma d}{2} \ln \frac{\alpha}{2\pi\nu}\right), \qquad \textbf{[6.8]}$$

and $\nu = \beta = 1$ in Eq. **1.1**, the explicit formula for ϕ is given by

$$\phi(x, t) = \frac{\alpha |x|^2}{2} - \left(d\alpha + \frac{\gamma d}{2} \ln \frac{\alpha}{2\pi}\right) t, \qquad \rho(x, t) = \left(\frac{\alpha}{2\pi}\right)^{\frac{d}{2}} e^{-\frac{\alpha |x|^2}{2}}, \qquad \textbf{[6.9]}$$

where $\alpha = \frac{-\gamma + \sqrt{\gamma^2 + 4}}{2}$. For the $\gamma = 0.1$ case, we use Kernel Density Estimation (58, 59) to estimate ρ from samples of the generator. The derivation of the analytic solution can be found in Appendix A.

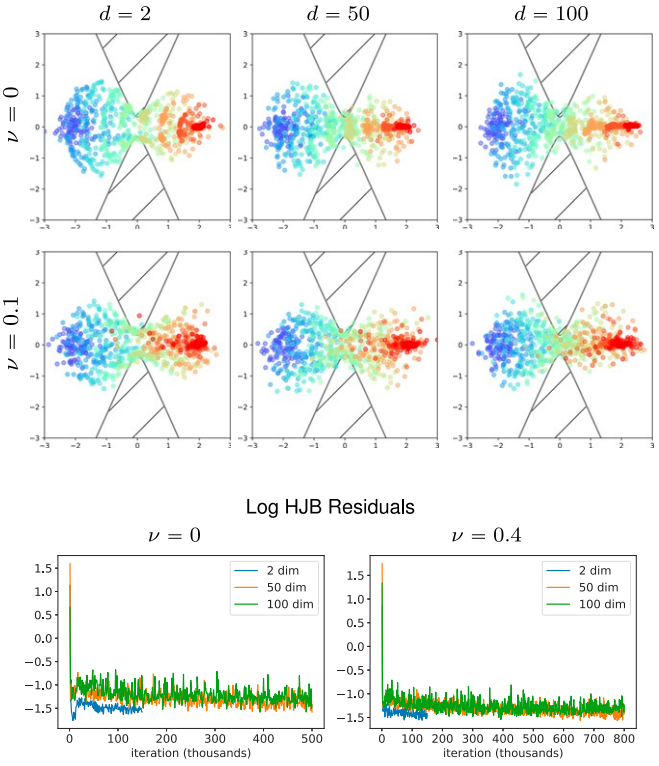

Fig. 4. Computation of the congestion problem with a bottleneck in dimensions (dim) 2, 50, and 100 with stochasticity parameter $\nu = 0$ and 0.1. For dimensions 50 and 100, we plot the first two dimensions.

For $d=2$, we compute the relative error on a grid of size $32\times 32\times 16$, where we discretize the spatial domain $\Omega=[-2,2]^2$ with 32×32 points and the time domain $[0,1]$ with 16 points. Note here that the grid-points are *validation* points—i.e., points that were not used in training. For $d=50$ and $d=100$, we use 4,096 sampled points for validation since we cannot build a grid. We run a total of 30 k and 60 k iterations for $\gamma=0$ and $\gamma=0.1$, respectively. We validate every 1 k iterations. Fig. 5 shows that our learned model approaches the true solution across all dimensions for both values of γ, indicating that APAC-Net generalizes well.

A Realistic Example: The Quadcopter. In this experiment, we examine a realistic scenario where the dynamics are that of a Quadcopter (also known as Quadrotor craft), which is an aerial vehicle with four rotary wings (similar to many of the consumer drones seen today). The dynamics of the quadrotor craft are given as,

$$\begin{cases} \ddot{x} &= \frac{u}{m}l(\sin(\phi)\sin(\psi)+\cos(\phi)\cos(\psi)\sin(\theta)\rho)\\ \ddot{y} &= \frac{u}{m}l(-\cos(\psi)\sin(\phi)+\cos(\phi)\sin(\theta)\sin(\psi)\rho)\\ \ddot{z} &= \frac{u}{m}\cos(\theta)\cos(\phi)-g\\ \ddot{\psi} &= \tilde{\tau}_\psi\\ \ddot{\theta} &= \tilde{\tau}_\theta\\ \ddot{\phi} &= \tilde{\tau}_\phi \end{cases},$$

where x, y, and z are the usual Euclidean spatial coordinates, and ϕ, θ, and ψ are the angular coordinates of roll, pitch, and yaw, respectively. The constant m is the mass, to which we put 0.5 (kilogram), and g is the gravitation acceleration constant on Earth, to which we put 9.81 (meters per second squared). The variables u, $\tilde{\tau}_\psi$, $\tilde{\tau}_\theta$, $\tilde{\tau}_\phi$ are the controls representing thrust and angular acceleration. In order to fit a control framework, the above second-order system is turned into a first-order system:

$$\begin{cases} \dot{x}_1 &= x_2\\ \dot{x}_2 &= \frac{u}{m}l(\sin(\phi_1)\sin(\psi_1)+\cos(\phi_1)\cos(\psi_1)\sin(\theta_1)\rho)\\ \dot{y}_1 &= y_2\\ \dot{y}_2 &= \frac{u}{m}l(-\cos(\psi_1)\sin(\phi_1)+\cos(\phi_1)\sin(\theta_1)\sin(\psi_1)\rho)\\ \dot{z}_1 &= z_2\\ \dot{z}_2 &= \frac{u}{m}\cos(\theta_1)\cos(\phi_1)-g\\ \dot{\psi}_1 &= \psi_2\\ \dot{\psi}_2 &= \tilde{\tau}_\psi\\ \dot{\theta}_1 &= \theta_2\\ \dot{\theta}_2 &= \tilde{\tau}_\theta\\ \dot{\phi}_1 &= \phi_2\\ \dot{\phi}_2 &= \tilde{\tau}_\phi \end{cases},$$

which we will compactly denote as $\dot{\mathbf{x}}=\mathbf{h}(\mathbf{x},\mathbf{u})$, where $\mathbf{h}$ is the right-hand side, $\mathbf{x}$ is the state, and $\mathbf{u}$ is the control. As can be observed, the above is a 12-dimensional system that is highly nonlinear and high-dimensional. In the stochastic case, we add a noise term to the dynamics: $d\mathbf{x}=\mathbf{h}(\mathbf{x},\mathbf{u})\,dt+\sigma\,d\mathbf{W}_t$, where $\mathbf{W}$ denotes a Wiener process (standard Brownian motion). The interpretation here is that we are modeling the situation when the quadcopter suffers from noisy measurements.

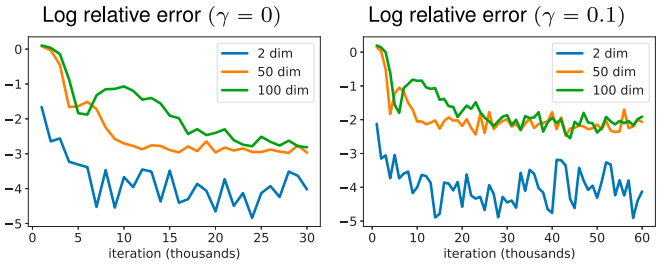

Fig. 5. Log relative errors in 2, 50, and 100 dimensions (dim) and for $\gamma=0, 0.1$. Here, $\gamma=0$ means no interaction. For the $d=2$ case, we compute the validation on a 32×32 grid over 16 uniformly spaced timesteps with the true ϕ from Eq. **6.9**. For the $d=50$ and 100 cases, we compute on a sample of 4,096 sample points, sampled from the initial density.

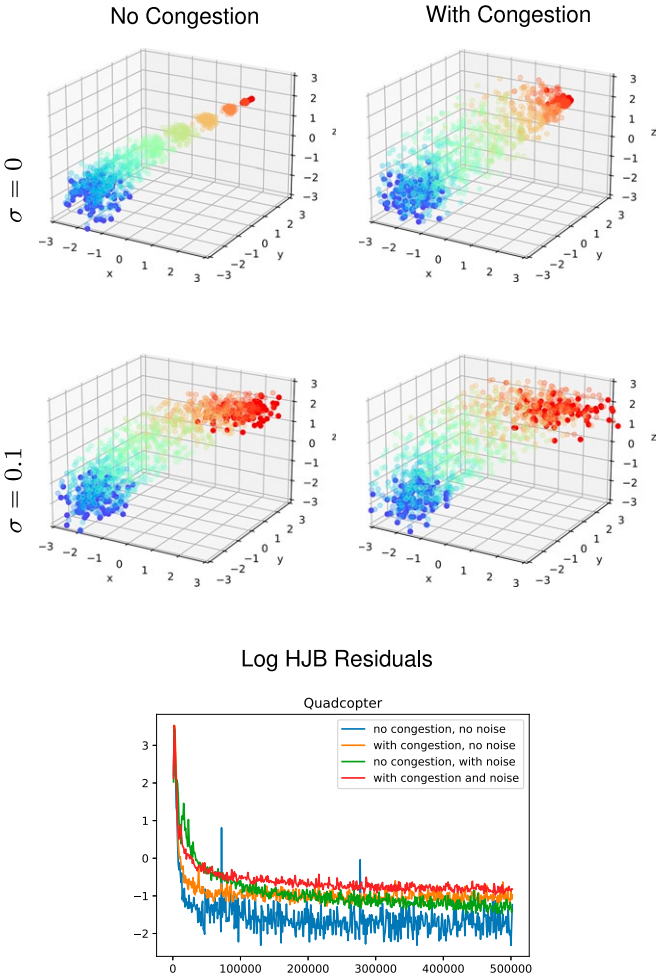

Fig. 6. Computation of the control of the fully nonlinear multiagent quadcopter (12 dimensions). This experiment represents a real-world example of control, where we move the agents from one point to another, with the constraint of having a velocity of zero at the destination. We compute examples with and without noise ($\sigma>0$) and with and without congestion. The HJB residuals (Eq. **4.3**) are plotted in the graph.

In our experiments, we set our Lagrangian cost function to be $L(\mathbf{u})=\frac{1}{2}\|\mathbf{u}\|_2^2$, and, thus, our Hamiltonian becomes $H(\mathbf{p})=\frac{1}{2}\|\mathbf{p}\|_2^2$. Our initial density ρ_0 is a Gaussian in the spatial coordinates (x,y,z) centered at $(-2,-2,-2)$ with SD 0.5, and we set all other initial coordinates to zero (i.e., initial velocity, initial angular position, and initial angular velocity are all set to zero). We set our terminal cost to be a simple norm difference between the agent's current position and the position $(2,2,2)$, and we also want the agents to have zero velocity, i.e.,

$$\mathcal{G}(\rho(\cdot,T))=\int_\Omega \|(x,y,z,\dot{x},\dot{y},\dot{z})-(2,2,2,0,0,0)\|_2\rho(x,T)dx.$$

In our experiments, we set the final time to be $T=4$.

We also add a congestion term to our experiments where the congestion is in the spatial positions, so as to encourage

agents to spread out (and thus making midair collisions less likely):

$$\mathcal{F}(\rho(x,y,z)) = \gamma_{\text{cong}} \frac{1}{(2\pi)^{\frac{3}{2}}} \int_\Omega \int_\Omega e^{\left(-\frac{1}{2}\|(x,y,z)-(\hat{x},\hat{y},\hat{z})\|_2^2\right)} d\rho(x,y,z) \times d\rho(\hat{x},\hat{y},\hat{z}),$$

where in our experiments we put $\gamma_{\text{cong}} = 20$.

For hyperparameters, we use the same setup as before, except we raise the batch size to 150. Our results are shown in Fig. 6. We see that with congestion, the agents spread out more as expected. Furthermore, in the presence of noise, the agents' sensors are noisy, and so at terminal time, the agents do not get as close to the terminal point of $(2, 2, 2)$. This noise also adds an envelope of uncertainty, so we do see that the agents are not as streamlined as in the noiseless cases.

We note that most previous attempts at modeling the behavior of quadcopters relied on linearization techniques, but here, we solve the fully nonlinear problem with neural networks. And our approach solves the quadcopter problem in a deterministic and stochastic MFGs context, where we consider many quadcopters and their interactions with each other. This multiagent modeling of the quadcopter would be practically infeasible for grid-based methods, as such a grid would need to discretize a high-dimensional space.

6. Conclusion

We present APAC-Net, an alternating population-agent control neural network approach for tackling high-dimensional stochastic MFGs. To this end, our algorithm avoids the use of spatial grids by parameterizing the controls, ϕ and ρ, using two neural networks, respectively. Consequently, our method is geared toward high-dimensional instances of these problems that are beyond reach with existing grid-based methods. APAC-Net therefore sets the stage for solving realistic high-dimensional MFGs arising in, e.g., economics (1–4), swarm robotics (11, 12), and, perhaps most important/relevant, epidemic modeling (13, 14). Our method also has natural connections with Wasserstein GANs, where ρ acts as a generative network and ϕ acts as a discriminative network. Unlike GANs, however, APAC-Net incorporates the structure of MFGs via Eqs. **2.5** and **4.1**, which absolves the network from learning an entire MFG solution from the ground up. Our experiments show that our method is able to solve 100-dimensional MFGs.

Since our method was presented solely in the setting of potential MFGs, a natural extension is the nonpotential MFG setting, where the MFG can no longer be written in a variational form. Instead, one would have to formulate the MFG as a monotone inclusion problem (60). Moreover, convergence properties of the training process of APAC-Net may be investigated following the techniques presented in, e.g., ref. 61. Finally, a practical direction involves examining guidelines on the design of more effective network architectures, e.g., PDE-based networks (62, 63), neural ordinary differential equations (41), or sorting networks (64).

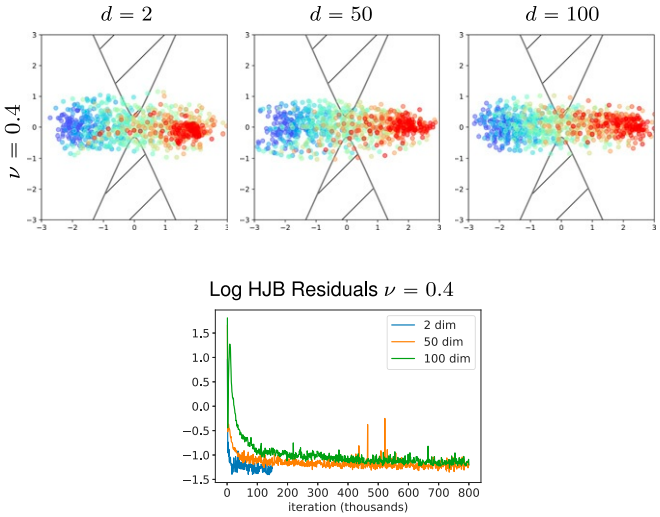

Fig. 7. Computation of the congestion problem with a bottleneck in dimensions (dim) 2, 50, and 100 with stochasticity parameter $\nu = 0.4$. For dimensions 50 and 100, we plot the first two dimensions.

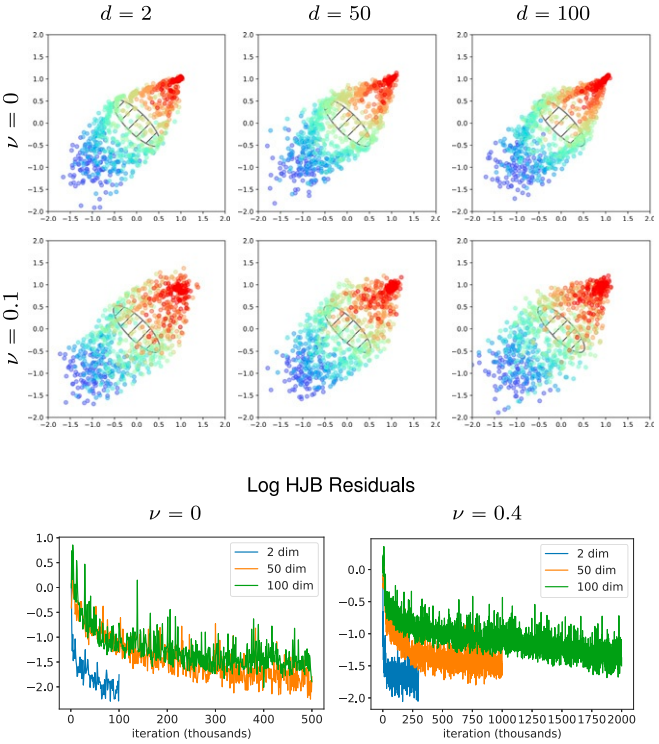

Fig. 8. Computation of the obstacle problem where the obstacle is symmetric. We plot the results for dimensions (dim) 2 and 100, and for $\nu = 0$ and $\nu = 0.1$.

Appendix A. Derivation of Analytic Solution

We derive explicit formulas used to test our approximate solutions in *Section 6*. Assume that $\nu, \beta > 0, \gamma \geq 0$ and

$$H(x,p,t) = \frac{|p|^2}{2} - \frac{\beta|x|^2}{2}, \quad f(x,\rho) = \gamma \ln \rho. \qquad \textbf{[A.1]}$$

Then, Eq. **1.1** becomes

$$\begin{aligned} -\partial_t \phi - \nu\Delta\phi + \frac{|\nabla\phi|^2}{2} - \frac{\beta|x|^2}{2} &= \gamma \ln \rho, \\ \partial_t \rho - \nu\Delta\rho - \operatorname{div}(\rho\nabla\phi) &= 0, \\ \rho(x,0) = \rho_0, \quad \phi(x,T) &= \Psi(x). \end{aligned} \qquad \textbf{[A.2]}$$

We find solutions to this system by searching for stationary solutions first:

$$\begin{aligned} -\nu\Delta\phi + \frac{|\nabla\phi|^2}{2} - \frac{\beta|x|^2}{2} &= \gamma \ln \rho + \bar{H}, \\ -\nu\Delta\rho - \operatorname{div}(\rho\nabla\phi) &= 0, \end{aligned} \qquad \textbf{[A.3]}$$

and then writing

$$\phi(x,t)=\phi(x)-t\bar{H},\quad \rho(x,t)=\rho(x). \tag{A.4}$$

The second equation in Eq. **A.3** yields $\rho=ce^{-\frac{\phi}{\nu}}$, where c is chosen so that $\int\rho=1$. Plugging this in the first equation in Eq. **A.3**, we obtain

$$-\nu\Delta\phi+\frac{|\nabla\phi|^2}{2}-\frac{\beta|x|^2}{2}=\gamma\ln c-\frac{\gamma}{\nu}\phi+\bar{H}. \tag{A.5}$$

Now, we make an ansatz that $\phi(x)=\frac{\alpha|x|^2}{2}$. Then, we have that $\Delta\phi=d\alpha,\ \nabla\phi=\alpha x$, and obtain

$$-\nu d\alpha+\frac{\alpha^2|x|^2}{2}-\frac{\beta|x|^2}{2}=\gamma\ln c-\frac{\gamma\alpha|x|^2}{2\nu}+\bar{H}. \tag{A.6}$$

Therefore, we have that

$$\alpha^2+\frac{\gamma\alpha}{\nu}=\beta,\quad \bar{H}=-\nu d\alpha-\gamma\ln c. \tag{A.7}$$

From the first equation, we obtain that

$$\alpha=\frac{-\gamma+\sqrt{\gamma^2+4\nu^2\beta}}{2\nu}. \tag{A.8}$$

On the other hand, we have that

$$\int\rho=c\int e^{-\frac{\alpha|x|^2}{2\nu}}dx=c\left(\frac{2\pi\nu}{\alpha}\right)^{\frac{d}{2}}=1, \tag{A.9}$$

so

$$c=\left(\frac{\alpha}{2\pi\nu}\right)^{\frac{d}{2}}\quad\text{and}\quad \bar{H}=-\nu d\alpha-\frac{\gamma d}{2}\ln\frac{\alpha}{2\pi\nu}. \tag{A.10}$$

Summarizing, we get that for any $\nu,\beta>0,\gamma\geq0$, the following is a solution for Eq. **A.2**:

$$\begin{aligned}\phi(x,t)&=\frac{\alpha|x|^2}{2}-\left(\nu d\alpha+\frac{\gamma d}{2}\ln\frac{\alpha}{2\pi\nu}\right)t,\\ \rho(x,t)&=\left(\frac{\alpha}{2\pi\nu}\right)^{\frac{d}{2}}e^{-\frac{\alpha|x|^2}{2\nu}},\end{aligned} \tag{A.11}$$

where α is given by Eq. **A.8**, and

$$\begin{aligned}g(x)&=\frac{\alpha|x|^2}{2}-\left(\nu d\alpha+\frac{\gamma d}{2}\ln\frac{\alpha}{2\pi\nu}\right)T,\\ \rho_0(x)&=\left(\frac{\alpha}{2\pi\nu}\right)^{\frac{d}{2}}e^{-\frac{\alpha|x|^2}{2\nu}}.\end{aligned} \tag{A.12}$$

Choosing $\beta=\nu=1$, Eq. **A.11** gives the analytic solution used in *Section 6*.

Appendix B: Details on Numerical Results and More Experiments

Congestion. Here, we elaborate on how we compute the congestion term,

$$\mathcal{F}(\rho(x,t))=\int_\Omega\int_\Omega\frac{1}{\|(x_1,x_2)-(y_1,y_2)\|^2+1}\,d\rho(x,t)\,d\rho(y,t). \tag{B.1}$$

We do this by first using the batch $\{z_b\}_{b=1}^B$, which was used for training (and sampled from ρ_0), and then compute another batch $\{y_b\}_{b=1}^B$, again sampled from ρ_0. Letting $\{t_b\}_{b=1}^B$ be a batch of time points uniformly sampled in $[0,1]$, we estimate the interaction cost with,

$$\mathcal{F}(\rho(x,t))\approx\sum_{i=1}^B\frac{1}{\|G_\theta(z_b,t_b)-G_\theta(y_b,t_b)\|^2+1}. \tag{B.2}$$

Congestion with Bottleneck Obstacle and Higher Stochasticity. When choosing a stochasticity parameter $\nu>0.1$, the stochasticity dominates the dynamics, and the obstacles do not interact as much with the obstacle. We plot these results in Fig. 7, where for two dimensions, we trained for 150 k iterations, and for 50 and 100 dimensions, we trained for 800 k iterations. All environment and training parameters are the same as in the *Congestion with Bottleneck Obstacle*, except that now $\nu=0.4$.

Analytic Comparison. Here, we mention specifically how we performed Kernel Density Estimation. Namely, in order to estimate the density ρ, we take a batch of samples $\{z_b\}_{b=1}^B$ (during training, this is the training batch). Then, at uniformly spaced time points $\{t_b\}_{b=1}\subseteq[0,1]$, we estimate the density with the formula,

$$\rho(z_b,t_b)\approx\frac{1}{B}\frac{1}{(\sigma h\sqrt{2\pi})^d}\sum_{i=1}^B\sum_{j=1}^B\exp\left(\frac{\|z_i-z_j\|^2}{(h\sigma)^2}\right), \tag{B.3}$$

where we choose $\sigma=\sqrt{\frac{\gamma}{\nu}}$, d is the dimension, and $h=B^{-\frac{1}{d+4}}$, in accordance with Scott's rule for multivariate Kernel Density Estimation (65).

Density Splitting Via Symmetric Obstacle. Here, we compute an example where we have a symmetric obstacle, and, thus, the generator will learn to split the density. Agents will go left or right of the obstacle, depending on their starting position. Here, we chose the obstacle as,

$$\begin{aligned}&f(x_1,x_2,\ldots,x_d)=\alpha_{\text{obst}}\max\left\{-v^\top Qv+0.1,0\right\},\\ &Q=\begin{pmatrix}1 & 0.8\\ 0.8 & 1\end{pmatrix},\ v=(x_1,x_2),\end{aligned} \tag{B.4}$$

and we choose $\gamma_{\text{obst}}=20$. The environment and training parameters are the same as in the obstacles example in the main text, except that we choose the HJB penalty λ in Alg. 1 to be 0.1. Qualitatively, we see that the solution agrees with our intuition: The agents will go left or right depending on their starting position. Note that the results are similar across dimensions, verifying our computation. For the $2d$, $\nu=0$ case, we trained for 100 k iterations; for the $2d$, $\nu=0.1$ case, we trained for 300 k iterations; for the $50d$ and $100d$, $\nu=0$ case, we trained for 500 k iterations; for the $50d$ $\nu=0.1$ case, we trained for 1,000 k iterations; and the for the $100d$, $\nu=0.1$ case, we trained for 2,000 k iterations. In Fig. 8, we plot the agents moving through the symmetric obstacles, as well as the HJB residuals.

Data Availability. To promote access and progress, we provide our PyTorch implementation at GitHub (https://github.com/atlin23/apac-net).

ACKNOWLEDGMENTS This work was supported by Air Force Office of Scientific Research (AFOSR) Multidisciplinary University Research Initiative Grant FA9550-18-1-0502, AFOSR Grant FA9550-18-1-0167, and Office of Naval Research Grant N00014-18-1-2527.

1. Y. Achdou, F. J. Buera, J.-M. Lasry, P.-L. Lions, B. Moll, Partial differential equation models in macroeconomics. *Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci.* **372**, 20130397 (2014).
2. Y. Achdou, J. Han, J.-M. Lasry, P.-L. Lions, B. Moll. "Income and wealth distribution in macroeconomics: A continuous-time approach." (NBER Working Paper 23732, National Bureau of Economic Research, Cambridge, MA, 2017). https://www.nber.org/papers/w23732. Accessed 4 July 2021.
3. O. Guéant, J.-M. Lasry, P.-L. Lions, "Mean field games and applications" in *Paris-Princeton Lectures on Mathematical Finance 2010*, R. Carmona et al., Eds. (Lecture Notes in Mathematics, Springer, Berlin, Germany, 2011), vol. 2003, pp. 205–266.
4. D. A. Gomes, L. Nurbekyan, E. A. Pimentel, *Economic Models and Mean-Field Games Theory* (IMPA Mathematical Publications, Rio de Janeiro, Brazil, 2015).
5. D. Firoozi, P. E. Caines, "An optimal execution problem in finance targeting the market trading speed: An MFG formulation" in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)* (IEEE, Piscataway, NJ, 2017), pp. 7–14.
6. P. Cardaliaguet, C.-A. Lehalle, Mean field game of controls and an application to trade crowding. *Math. Financ. Econ.* **12**, 335–363 (2018).
7. P. Casgrain, S. Jaimungal, Algorithmic trading in competitive markets with mean field games. *SIAM News* **52**, 1–2 (2019).
8. A. De Paola, V. Trovato, D. Angeli, G. Strbac, A mean field game approach for distributed control of thermostatic loads acting in simultaneous energy-frequency response markets. *IEEE Trans. Smart Grid* **10**, 5987–5999 (2019).
9. A. C. Kizilkale, R. Salhab, R. P. Malhamé, An integral control formulation of mean field game based large scale coordination of loads in smart grids. *Automatica* **100**, 312–322 (2019).
10. D. A. Gomes, J. Saúde, A mean-field game approach to price formation in electricity markets. arXiv [Preprint] (2018). https://arxiv.org/abs/1807.07088 (Accessed 4 July 2021).
11. Z. Liu, B. Wu, H. Lin, "A mean field game approach to swarming robots control" in *2018 Annual American Control Conference (ACC)* (IEEE, Piscataway, NJ, 2018), pp. 4293–4298.
12. K. Elamvazhuthi, S. Berman, Mean-field models in swarm robotics: A survey. *Bioinspiration Biomimetics* **15**, 015001 (2019).
13. W. Lee, S. Liu, H. Tembine, S. Osher, Controlling propagation of epidemics via mean-field games. *SIAM J. Appl. Math.* **81**, 190–207 (2021).
14. S. L. Chang, M. Piraveenan, P. Pattison, M. Prokopenko, Game theoretic modelling of infectious disease dynamics and intervention methods: A review. *J. Biol. Dynam.* **14**, 57–89 (2020).
15. E. Weinan, J. Han, Q. Li, A mean-field optimal control formulation of deep learning. *Res. Math. Sci.* **6**, 10 (2019).
16. X. Guo, A. Hu, R. Xu, J. Zhang, "Learning mean-field games" in *NeurIPS 2019: The 33rd Annual Conference on Neural Information Processing Systems*, H. Wallach et al., Eds. (Advances in Neural Information Processing Systems, 2019), vol. 32, pp. 4967–4977.
17. R. Carmona, M. Laurière, Z. Tan, Linear-quadratic mean-field reinforcement learning: Convergence of policy gradient methods. arXiv [Preprint] (2019). https://arxiv.org/abs/1910.04295 (Accessed 4 July 2021).
18. J.-M. Lasry, P.-L. Lions, Jeux à champ moyen. II–Horizon fini et contrôle optimal. *Compt. Rendus Math.* **343**, 679–684 (2006).
19. J.-M. Lasry, P.-L. Lions, Mean field games. *Jpn. J. Math.* **2**, 229–260 (2007).
20. Y. T. Chow, J. Darbon, S. Osher, W. Yin, Algorithm for overcoming the curse of dimensionality for time-dependent non-convex Hamilton–Jacobi equations arising from optimal control and differential games problems. *J. Sci. Comput.* **73**, 617–643 (2017).
21. Y. Achdou, I. Capuzzo-Dolcetta, Mean field games: Numerical methods. *SIAM J. Numer. Anal.* **48**, 1136–1162 (2010).
22. J.-D. Benamou, G. Carlier, F. Santambrogio, "Variational mean field games" in *Active Particles*, N. Bellomo, P. Degond, E. Tadmor, Eds. (Modeling and Simulation in Science, Engineering, and Technology, Springer, Cham, Switzerland, 2017), vol. 1, pp. 141–171.
23. Y. T. Chow, W. Li, S. Osher, W. Yin, Algorithm for Hamilton–Jacobi equations in density space via a generalized Hopf formula. *J. Sci. Comput.* **80**, 1195–1239 (2019).
24. Y. T. Chow, J. Darbon, S. Osher, W. Yin, Algorithm for overcoming the curse of dimensionality for certain non-convex Hamilton–Jacobi equations, projections and differential games. *Annal. Math. Sci. Appl.* **3**, 369–403 (2018).
25. M. Jacobs, F. Léger, W. Li, S. Osher, Solving large-scale optimization problems with a convergence rate independent of grid size. *SIAM J. Numer. Anal.* **57**, 1100–1123 (2019).
26. R. Bellman, Dynamic programming. *Science* **153**, 34–37 (1966).
27. S. Li, S. E. Li, K. Deng, "Mean-field control for improving energy efficiency" in *Automotive Air Conditioning*, Q. Zhang, S. E. Li, K. Deng, Eds. (Springer, Cham, Switzerland, 2016), pp. 125–143.
28. M. Cirant, L. Nurbekyan, The variational structure and time-periodic solutions for mean-field games systems. *Minimax Theory Appl* **3**, 227–260 (2018).
29. L. Ruthotto, S. J. Osher, W. Li, L. Nurbekyan, S. W. Fung, A machine learning framework for solving high-dimensional mean field game and mean field control problems. *Proc. Natl. Acad. Sci. U.S.A.* **117**, 9183–9193 (2020).
30. I. Goodfellow *et al.*, "Generative adversarial nets" in *NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger, Eds. (MIT Press, Cambridge, MA, 2014), vol. 2, pp. 2672–2680.
31. M. Arjovsky, S. Chintala, L. Bottou, "Wasserstein generative adversarial networks" in *ICML'17: Proceedings of the 34th International Conference on Machine Learning*, D. Precup, Y. W. Teh, Eds. (JMLR, 2017), vol. 70, pp. 214–223.
32. I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A. C. Courville, "Improved training of Wasserstein GANs" in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, U. von Luxburg, I. Guyon, S. Bengio, H. Wallach, R. Fergus, Eds. (Curran Associates, Inc., Red Hook, NY, 2017), pp. 5767–5777.
33. A. T. Lin, W. Li, S. Osher, G. Montúfar, Wasserstein proximal of GANs. UCLA CAM [Preprint] (2018). ftp://ftp.math.ucla.edu/pub/camreport/cam18-53.pdf (Accessed 4 July 2021).
34. Y. Dukler, W. Li, A. Lin, G. Montufar, "Wasserstein of Wasserstein loss for learning generative models" in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri, R. Salakhutdinov, Eds. (Proceedings of Machine Learning Research, PMLR, 2019), vol. 97, pp. 1716–1725.
35. K. F. C. Chu, K. Minami, Smoothness and stability in GANs. arXiv [Preprint] (2020). https://arxiv.org/abs/2002.04185 (Accessed 4 July 2021).
36. P. Cardaliaguet, P. J. Graber, Mean field games systems of first order. *ESAIM Control, Optim. Calc. Var.* **21**, 690–722 (2015).
37. P. Cardaliaguet, P. J. Graber, A. Porretta, D. Tonon, Second order mean field games with degenerate diffusion and local coupling. *Nonlin. Diff. Equ. Appl.* **22**, 1287–1317 (2015).
38. H. Cao, X. Guo, M. Laurière, Connecting GANs and MFGs. arXiv [Preprint] (2020). https://arxiv.org/abs/2002.04112 (Accessed 4 July 2021).
39. C. Villani, *Topics in Optimal Transportation* (Graduate Studies in Mathematics, American Mathematical Society, Providence, RI, 2003), vol. 58.
40. D. Onken, S. W. Fung, X. Li, L. Ruthotto, OT-Flow: Fast and accurate continuous normalizing flows via optimal transport. arXiv [Preprint] (2020). https://arxiv.org/abs/2006.00104 (Accessed 4 July 2021).
41. T. Q. Chen, Y. Rubanova, J. Bettencourt, D. K. Duvenaud, "Neural ordinary differential equations" in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, Eds. (Curran Associates, Red Hook, NY, 2018), pp. 6572–6583.
42. A. T. Lin, Y. T. Chow, S. J. Osher, A splitting method for overcoming the curse of dimensionality in Hamilton–Jacobi equations arising from nonlinear optimal control and differential games with applications to trajectory generation. *Commun. Math. Sci.* **16**, pp. 1933–1973 (2018).
43. J. Darbon, S. Osher, Algorithms for overcoming the curse of dimensionality for certain Hamilton–Jacobi equations arising in control theory and elsewhere. *Res. Math. Sci.* **3**, 19 (2016).
44. H.-T. Wai, Z. Yang, Z. Wang, M. Hong, "Multi-agent reinforcement learning via double averaging primal-dual optimization" in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, Eds. (Curran Associates, Red Hook, NY, 2018), pp. 9649–9660.
45. X. Guo, A. Hu, R. Xu, J. Zhang, A general framework for learning mean-field games. arXiv [Preprint] (2020). https://arxiv.org/abs/2003.06069 (Accessed 4 July 2021).
46. J. Yang, X. Ye, R. Trivedi, H. Xu, H. Zha, "Deep mean field games for learning optimal behavior policy of large populations" in *6th International Conference on Learning Representations* (ICLR, 2018).
47. Z. Fu, Z. Yang, Y. Chen, Z. Wang, "Actor-critic provably finds Nash equilibria of linear-quadratic mean-field games" in *8th International Conference on Learning Representations* (ICLR, 2020).
48. A. Genevay, G. Peyre, M. Cuturi, "Learning generative models with Sinkhorn divergences" in *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, A. Storkey, F. Perez-Cruz, Eds. (Proceedings of Machine Learning Research, PMLR, 2018), vol. 84, pp. 1608–1617.
49. T. Salimans, H. Zhang, A. Radford, D. Metaxas, "Improving GANs using optimal transport" in *6th International Conference on Learning Representations* (ICLR, 2018).
50. M. Sanjabi, J. Ba, M. Razaviyayn, J. D. Lee, "On the convergence and robustness of training GANs with regularized optimal transport" in *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, Eds. (Curran Associates, Red Hook, NY, 2018), pp. 7091–7101.
51. A. Tanaka, "Discriminator optimal transport" in *NeurIPS 2019: The 33rd Annual Conference on Neural Information Processing Systems*, H. Wallach et al., Eds. (Advances in Neural Information Processing Systems, 2019), vol. 32, pp. 6813–6823.
52. J. Lin, K. Lensink, E. Haber, Fluid flow mass transport for generative networks. arXiv [Preprint] (2019). https://arxiv.org/abs/1910.01694 (Accessed 4 July 2021).
53. G. Peyré *et al.*, Computational optimal transport. *Found. Trend. Math. Learn.* **11**, 355–607 (2019).
54. C. Finlay, B.-H. Jacobsen, L. Nurbekyan, A. M. Oberman, How to train your neural ODE. arXiv [Preprint] (2020). https://arxiv.org/abs/2002.02798 (Accessed 4 July 2021).
55. W. Grathwohl, R. T. Q. Chen, J. Betterncourt, I. Sutskever, D. Duvenaud, "FFJORD: Free-form continuous dynamics for scalable reversible generative models" in *International Conference on Learning Representations 2019* (ICLR, 2019).
56. D. Onken, L. Ruthotto, Discretize-optimize vs. optimize-discretize for time-series regression and continuous normalizing flows. arXiv [Preprint] (2020). https://arxiv.org/abs/2005.13420 (Accessed 4 July 2021).

57. C. Parkinson, D. Arnold, A. L. Bertozzi, S. Osher, A model for optimal human navigation with stochastic effects. arXiv [Preprint] (2020). https://arxiv.org/abs/2005.03615 (Accessed 4 July 2021).
58. M. Rosenblatt, Remarks on some nonparametric estimates of a density function. *Ann. Math. Statist.* **27**, 832–837 (1956).
59. E. Parzen, On estimation of a probability density function and mode. *Ann. Math. Statist.* **33**, 1065–1076 (1962).
60. S. Liu, L. Nurbekyan, Splitting methods for a class of non-potential mean field games. arXiv [Preprint] (2020). https://arxiv.org/abs/2007.00099 (Accessed 4 July 2021).
61. H. Cao, X. Guo, Approximation and convergence of GANs training: An SDE approach. arXiv [Preprint] (2020). https://arxiv.org/abs/2006.02047 (Accessed 4 July 2021).
62. E. Haber, L. Ruthotto, Stable architectures for deep neural networks. *Inverse Probl.* **34**, 014004 (2017).
63. L. Ruthotto, E. Haber, Deep neural networks motivated by partial differential equations. *J. Math. Imag. Vis.* **62**, 1–13 (2019).
64. C. Anil, J. Lucas, R. Grosse, "Sorting out Lipschitz function approximation" in *International Conference on Machine Learning*, K. Chaudhuri, R. Salakhutdinov, Eds. (Proceedings of Machine Learning Research, PMLR, 2019), pp. 291–301.
65. D. W. Scott, S. R. Sain, Multidimensional density estimation. *Handb. Stat.* **24**, 229–261 (2005).