A deep learning algorithm for computing mean field control problems via forward-backward score dynamics

Mo Zhou[1], Stanley Osher[1], Wuchen Li[2*]

[1]Department of Mathematics, University of California, Los Angeles, 90095, CA, USA.
[2]Department of Mathematics, University of South Carolina, Columbia, 29208, SC, USA.

*Corresponding author(s). E-mail(s): wuchen@mailbox.sc.edu;
Contributing authors: mozhou366@math.ucla.edu; sjo@math.ucla.edu;

Abstract

We propose a deep learning approach to compute mean field control problems with individual noises. The problem consists of the Fokker-Planck (FP) equation and the Hamilton-Jacobi-Bellman (HJB) equation. Using the differential of the entropy, namely the score function, we first formulate the deterministic forward-backward characteristics for the mean field control system, which is different from the classical forward-backward stochastic differential equations (FBSDEs). We further apply the neural network approximation to fit the proposed deterministic characteristic lines. Numerical examples, including the control problem with entropy potential energy, the linear quadratic regulator, and the systemic risks, demonstrate the effectiveness of the proposed method.

Keywords: Mean field control; Forward-backward score dynamics; Deep learning algorithms.

1 Introduction

Mean Field Control (MFC) has emerged as a powerful framework in the realm of control theory, offering a versatile approach to model and optimize large-scale systems

with a multitude of interacting components among agents. This paradigm finds applications across various domains, such as economics and finance [1], herd behavior [2], robotics [3], and data science [4]. Its significance lies in its ability to address complex systems with a large number of interacting agents by considering their mean field behaviors, thereby reducing the computational burden associated with analyzing each individual agent.

Along with the MFC problem, two important equations characterize the optimality condition of the system. The first one is the Fokker-Planck (FP) equation, which describes the evolution of the density of the system. The second one is the Hamilton–Jacobi-Bellman (HJB) equation, which describes the optimal expected cost w.r.t. certain initialization. Solving the coupled FP–HJB system below is crucial for solving the MFC problem

$$\begin{cases} \partial_t \rho(t,x) + \nabla_x \cdot (\rho(t,x) \mathrm{D}_p H(t,x,\nabla_x \phi(t,x))) = \dfrac{1}{\beta} \Delta_x \rho(t,x), \\[2mm] \partial_t \phi(t,x) + H(t,x,\nabla_x \phi(t,x)) + \dfrac{1}{\beta} \Delta_x \phi(t,x) = f(t,x,\rho(t,x)). \end{cases}$$

Here $H(t,x,p)\colon \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the Hamiltonian of the system, $x \in \mathbb{R}^d$ and $p \in \mathbb{R}^d$ are the space and adjoint variables respectively. D_p denotes the gradient of H w.r.t. adjoint variable p. And $1/\beta > 0$ is a diffusion coefficient parameter, which characterizes the intensity of diffusion. The function $f\colon \mathbb{R} \times \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$ involves the density ρ, which represents the interaction behaviors of agents. The density dependency in Hamilton-Jacobi equations distinguishes the MFC problem from the standard stochastic control. The details for the system are introduced in Section 2.

One traditional way to solve the MFC problem is through the FBSDE system, which utilizes the stochastic characterization of the problem. The viscous state dynamic, also known as the stochastic characteristic curve, is often studied together with the backward adjoint equation that describes the cost [5] or shadow price [6]. This gives the classic FBSDE system [7]. The theoretical analysis for this method is elegant. However, these approaches heavily rely on the sampling of Brownian motions (BM), which may give large numerical errors from random trajectories.

In recent years, the score function has emerged as a powerful tool in many problems, such as the generative model and the MFC. The score function is the gradient of the logarithm of the density function $\nabla \log \rho(t,x)$. Based on this score function, the probability flow [8] is defined to characterize the stochastic state dynamic with a deterministic ODE, potentially alleviating the issue of sampling the Brownian motion.

In this paper, we propose a new framework to solve the MFC problem based on the score function. This is to formulate deterministic characteristics lines for the Fokker-Planck equations and Hamilton-Jacobi equations. In detail, we denote x_t as the probability flow of the state, which satisfies an ODE whose velocity is subtracted by the score function. We also define $y_t = \phi(t, x_t)$ as an analog for the adjoint process. These two dynamics form a forward-backward score ODE system, which is summarized

below.

$$\begin{cases} \partial_t x_t = \mathrm{D}_p H(t, x_t, z_t) - \dfrac{1}{\beta} \nabla_x \log \rho(t, x_t), \\ \partial_t y_t = f_t - \dfrac{1}{\beta} h_t - H(t, x_t, z_t) + z_t^\top \mathrm{D}_p H(t, x_t, z_t) - \dfrac{1}{\beta} z_t^\top \nabla_x \log \rho(t, x_t), \end{cases} \tag{1}$$

with given initial condition x_0 and terminal condition y_T. Here, z_t, h_t, and f_t are short for $\nabla_x \phi(t, x_t)$, $\Delta_x \phi(t, x_t)$, and $f(t, x_t, \rho(t, x_t))$ respectively. A detailed description of system (1) is given in Section 2. Clearly, when $\beta = \infty$, equation (1) is the classical deterministic characteristic for the HJB without viscosity.

In the algorithm, we parameterize ϕ as a neural network and construct a loss function to match the adjoint process y_t with its neural network parametrization. Meanwhile, we also formulate a density estimation method to approximate the gradient of the logarithm of the density function. This density estimation distinguishes our method from those that require sampling of Brownian motions. The validity of the proposed method is justified through several numerical examples including mean field control problems with an entropy potential energy, linear quadratic control, and system risks.

Many studies have been conducted on MFC problems. Some of the works focus on the McKean–Vlasov equation for the system [9–12]. Others focus on controlling the physical and social systems, arising from conservation laws [13, 14] and reaction-diffusion equations [15]. In recent years, there has been a growing trend to use machine learning approaches to compute the MFG/MFC problem, such as [16–19]. There is also research studying convergence properties of these machine learning methods [20, 21]. There are also other algorithms to compute the MFC problem such as the Picard iterations method [22]. Many of the works mentioned above rely heavily on the theory of FBSDE [23]. This FBSDE is usually computed through the shooting method, where one matches the terminal condition with its approximation. However, such methods rely heavily on the sampling of Brownian motion, which only has an accuracy of order one-half. In numerical examples, our deterministic score dynamic provides an alternative for this problem. We leave the theoretical comparison for algorithms using forward-backward score dynamics and FB SDEs in future works.

The rest of this paper is organized as follows. Section 2 provides a theoretical background for the MFC problem and introduces our forward-backward ODE score system. In Section 3, we present a detailed description of our numerical method, including the construction of the loss function, accompanied by theoretical justification, the estimation for the density function, and the numerical discretization. Finally, Section 4 shows numerical examples that validate the effectiveness of our proposed algorithm, including the control problem with an entropy potential energy, the linear quadratic regulator, and the systemic risks.

2 Formulation of the score-base mean field control problem

In this section, we briefly review the MFC problem and formulate the deterministic forward-backward ODE system, with the score functions to represent the viscosity for the density function.

We consider the mean field control problem described by the cost functional:

$$\inf_v \int_0^T \int_{\mathbb{R}^d} \left(L(t,x,v(t,x))\rho(t,x) + F(t,x,\rho(t,x))\right) \, \mathrm{d}x \, \mathrm{d}t + \int_{\mathbb{R}^d} V(x)\rho(T,x) \, \mathrm{d}x \quad (2)$$

where $L \colon \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is the Lagrangian of the system, $v \colon \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}^d$ is the velocity field, $F \colon \mathbb{R} \times \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}$ is the running cost that involves the density, and $V \colon \mathbb{R}^d \to \mathbb{R}$ is the terminal energy. Here, the density $\rho(t,x)$ satisfies the Fokker-Planck equation

$$\partial_t \rho(t,x) + \nabla_x \cdot (\rho(t,x)v(t,x)) = \frac{1}{\beta}\Delta_x \rho(t,x), \quad \rho(0,x) = \rho_0(x). \quad (3)$$

The objective is to minimize the cost functional (2) over all feasible velocity fields of control.

We give the following assumption for the system.

Assumption 1. *The Lagrangian $L(t,x,v)$ is strongly convex in v.*

This assumption guarantees the well-posedness of the MFC problem, which is necessary in our setting. In more general control problems, where the drift is nonlinear in the control field and the volatility is not constant, well-posedness may be guaranteed even if the Lagrangian is not convex. The well-posedness is determined by the concavity (or convexity, depending on the definition,) of the generalized Hamiltonian [6, 24].

Under Assumption 1, the Hamiltonian, defined as the Legendre transform of the Lagrangian, attains a well-defined expression

$$H(t,x,p) = \sup_{v \in \mathbb{R}^d} v^\top p - L(t,x,v). \quad (4)$$

Applying standard results in convex analysis, we know that the maximum in (4) is realized at

$$v^* = \mathrm{D}_p H(t,x,p), \quad (5)$$

where D_p refers to the gradient w.r.t. p (cf. [25] chapter 2).

Next, we analyze the mean field control (MFC) problem by introducing a Lagrange multiplier, $\phi(t, x)$, resulting in the following expression

$$
\inf_{v} \sup_{\phi} \int_0^T \int_{\mathbb{R}^d} \left[L(t, x, v(t, x)) \rho(t, x) + F(t, x, \rho(t, x)) \right.
$$

$$
\left. + \phi(t, x) \left(\partial_t \rho(t, x) + \nabla_x \cdot (\rho(t, x) v(t, x)) - \frac{1}{\beta} \Delta_x \rho(t, x) \right) \right] \mathrm{d}x \, \mathrm{d}t \quad (6)
$$

$$
+ \int_{\mathbb{R}^d} V(x) \rho(T, x) \, \mathrm{d}x
$$

Solving the critical point system for the above inf-sup problem yields the following proposition.

Proposition 1. *Under Assumption 1, there exists a function* $\phi \colon [0, T] \times \mathbb{R}^d \to \mathbb{R}$, *such that the optimal velocity field satisfies*

$$
v(t, x) = \mathrm{D}_p H(t, x, \nabla_x \phi(t, x)),
$$

and the solution of the MFC is described by the FP–HJB system

$$
\begin{cases}
\partial_t \rho(t, x) + \nabla_x \cdot (\rho(t, x) \mathrm{D}_p H(t, x, \nabla_x \phi(t, x))) = \dfrac{1}{\beta} \Delta_x \rho(t, x), \\[2mm]
\partial_t \phi(t, x) + H(t, x, \nabla_x \phi(t, x)) + \dfrac{1}{\beta} \Delta_x \phi(t, x) = f(t, x, \rho(t, x)), \\[2mm]
\rho(0, x) = \rho_0(x), \quad \phi(T, x) = -V(x),
\end{cases} \quad (7)
$$

where $f(t, x, \rho(t, x)) = \dfrac{\partial F}{\partial \rho}(t, x, \rho(t, x)).$

Proof. For illustration purposes, we take the functional derivative of the inf-sup problem (6) for all the variables and derive the critical point system for the problem. Taking the derivative of (6) w.r.t. ϕ directly results in the Fokker Planck equation (3). To obtain other equations for the critical point system, we perform integration by part and reformulate (6) as

$$
\inf_{v} \sup_{\phi} \int_0^T \int_{\mathbb{R}^d} \left[L(t, x, v(t, x)) \rho(t, x) + F(t, x, \rho(t, x)) - \partial_t \phi(t, x) \rho(t, x) \right.
$$

$$
\left. - \nabla_x \phi(t, x)^\top v(t, x) \rho(t, x) - \frac{1}{\beta} \Delta_x \phi(t, x) \rho(t, x) \right] \mathrm{d}x \, \mathrm{d}t \quad (8)
$$

$$
+ \int_{\mathbb{R}^d} (V(x) \rho(T, x) + \phi(T, x) \rho(T, x) - \phi(0, x) \rho(0, x)) \, \mathrm{d}x.
$$

Taking derivative of (8) w.r.t. ρ, we obtain

$$
L(t, x, v(t, x)) - \partial_t \phi(t, x) - \nabla_x \phi(t, x)^\top v(t, x) - \frac{1}{\beta} \Delta_x \phi(t, x) + \frac{\partial F}{\partial \rho}(t, x, \rho(t, x)) = 0,
$$

5

which simplifies to

$$\partial_t\phi(t,x) + \nabla_x\phi(t,x)^\top v(t,x) - L(t,x,v(t,x)) + \frac{1}{\beta}\Delta_x\phi(t,x) = f(t,x,\rho(t,x)). \quad (9)$$

Taking derivative of (8) w.r.t. v, we arrive at

$$\left(\nabla_v L(t,x,v(t,x)) - \nabla_x\phi(t,x)\right)\rho(t,x) = 0,$$

which implies

$$v(t,x) = \mathrm{D}_p H(t,x,\nabla_x\phi(t,x)) \quad (10)$$

according to (5). Finally, taking derivative of (8) w.r.t. $\rho(T,\cdot)$, we arrive at the terminal condition $\phi(T,x) = -V(x)$. If we plug (10) into the Fokker–Planck equation and (9), we recover the FP–HJB system (7). $\qquad\square$

Remark 1. *The HJB equation in* (7) *is slightly different from the traditional HJB equation, which is defined as the optimal expected cost given a starting point. In our case, there is a sign shift and part of the density information is not involved in the HJB equation. But* $\nabla_x\phi$ *still decides the optimal velocity. For a more comprehensive model, the Lagrange multiplier* ϕ *should also depend on the density* ρ, *and the value function will be the solution to the master equation [26].*

As an illustrative example, consider the case where $L(t,x,v) = \frac{1}{2}|v|^2$. This choice leads to $H(t,x,p) = \frac{1}{2}p^2$, resulting in the following simplified FP–HJB system

$$\begin{cases} \partial_t\rho(t,x) + \nabla_x\cdot(\rho(t,x)\nabla_x\phi(t,x)) = \dfrac{1}{\beta}\Delta_x\rho(t,x), \\[2mm] \partial_t\phi(t,x) + \dfrac{1}{2}|\nabla_x\phi(t,x)|^2 + \dfrac{1}{\beta}\Delta_x\phi(t,x) = f(t,x,\rho(t,x)), \\[2mm] \rho(0,x) = \rho_0(x), \quad \phi(T,x) = -V(x). \end{cases} \quad (11)$$

When $f = 0$, the problem reduces to an optimal control one and the HJB equation becomes the viscous Burgers equation.

Traditional methods such as backward stochastic differential equations (BSDEs) often involve sampling for Brownian motion, leading to potential errors. In this work, we innovatively propose an alternative ODE system based on the score function. The probability flow [8] associated with the Fokker-Planck equation (3) is defined as

$$\partial_t x_t = v(t,x_t) - \frac{1}{\beta}\nabla_x\log\rho(t,x_t), \quad (12)$$

where the density ρ is involved. The term $\nabla_x\log\rho(t,x_t)$ is known as the score function, commonly used in generative models [27]. This probability flow constitutes an ODE system, with the only source of randomness being the initial condition x_0. If x_0 follows the distribution ρ_0, then the density for x_t in (12) precisely corresponds to the solution of the FP equation (3).

According to Proposition 1, the probability flow under optimal velocity becomes the following ODE system

$$\partial_t x_t = \mathrm{D}_p H(t, x_t, \nabla_x \phi(t, x_t)) - \frac{1}{\beta} \nabla_x \log \rho(t, x_t). \tag{13}$$

This deterministic system liberates us from the need for Brownian motion sampling. We next introduce our forward-backward ODE system.

Proposition 2. *Let ρ, ϕ be the solution to the FP–HJB system (7). Let x_t be the probability flow defined by (13). Let $y_t = \phi(t, x_t)$. Then, x_t and y_t satisfies the forward-backward ODE system*

$$\begin{cases} \partial_t x_t = \mathrm{D}_p H(t, x_t, z_t) - \dfrac{1}{\beta} \nabla_x \log \rho(t, x_t), \\ \partial_t y_t = f_t - \dfrac{1}{\beta} h_t - H(t, x_t, z_t) + z_t^\top \mathrm{D}_p H(t, x_t, z_t) - \dfrac{1}{\beta} z_t^\top \nabla_x \log \rho(t, x_t), \end{cases} \tag{14}$$

with initial condition $x_0 \sim \rho_0$ and terminal condition $y_T = -V(x_T)$. Here, f_t is short for $f(t, x_t, \rho(t, x_t))$, $z_t = \nabla_x \phi(t, x_t)$, and $h_t = \Delta_x \phi(t, x_t)$.

Proof. The dynamic for x_t comes from equation (13). For $y_t = \phi(t, x_t)$, we have

$$\partial_t y_t = \frac{\mathrm{d}}{\mathrm{d}t} \phi(t, x_t) = \partial_t \phi(t, x_t) + \nabla_x \phi(t, x_t)^\top \partial_t x_t$$

$$= f(t, x_t, \rho(t, x_t)) - \frac{1}{\beta} \Delta_x \phi(t, x_t) - H(t, x_t, \nabla_x \phi(t, x_t))$$

$$+ \nabla_x \phi(t, x_t)^\top \left(\mathrm{D}_p H(t, x_t, \nabla_x \phi(t, x_t)) - \frac{1}{\beta} \nabla_x \log \rho(t, x_t) \right),$$

where we used the HJB equation and the probability flow is the second equality. This recovers the second ODE in (14). □

Remark 2. *This forward-backward ODE system can be viewed as a deterministic analog of the FBSDE, which is commonly studied in MFC problems. To present the FBSDE in our scenario, we first give the stochastic version of our MFC*

$$\inf_v \mathbb{E} \left[\int_0^T \left(L(t, X_t, v(t, X_t)) + F(t, X_t, \rho(t, X_t)) / \rho(t, X_t) \right) \mathrm{d}t + V(X_T) \right]$$

subject to

$$\mathrm{d}X_t = v(t, X_t)\, \mathrm{d}t + \sqrt{2/\beta}\, \mathrm{d}W_t \qquad X_0 \sim \rho_0. \tag{15}$$

Here, $\rho(t, x)$ is the density function for X_t, W_t is a standard Wiener process in $\mathbb{R}^d$, and the expectation is taken over the whole trajectory $X_t \sim \rho(t, \cdot)$. The optimal stochastic state dynamic (15), together with the adjoint dynamic for value, forms the FBSDE for

our MFC problem

$$\begin{cases} \mathrm{d}X_t = \mathrm{D}_p H(t, X_t, Z_t) \, \mathrm{d}t + \sqrt{2/\beta} \mathrm{d}W_t, & X_0 \sim \rho_0, \\ \mathrm{d}Y_t = (L(t, X_t, \mathrm{D}_p H(t, X_t, Z_t)) + f(t, X_t, \rho(t, X_t))) \, \mathrm{d}t + \sqrt{2/\beta} Z_t^\top \, \mathrm{d}W_t, & Y_T = -V(X_T). \end{cases}$$

(16)

Here, Z_t is the auxiliary variable for the BSDE. The unique solution to the BSDE above is $Y_t = \phi(t, X_t)$ and $Z_t = \nabla_x \phi(t, X_t)$. In this case, (X_t, Y_t, Z_t) are based on the solutions of stochastic dynamics (16). In comparison, (x_t, y_t, z_t) are solutions from equation system (14). They interact with each other through the density of x_t.

3 Numerical method

In this section, we apply a neural network parametrization to ϕ and construct a loss function to match the forward-backward ODE system. Then, we introduce the density estimation function based on the kernel method.

3.1 Construction of loss function

In practice, we can parametrize ϕ as a neural network $\phi_{\mathcal{N}}$ and represent z_t and h_t through auto-differentiation. Then we construct a loss function that try to match the dynamic y_t from (14) and $\phi_{\mathcal{N}}(t, x_t)$

$$\mathcal{L} = \mathbb{E} \int_0^T |\phi_{\mathcal{N}}(t, x_t) - y_t|^2 \, \mathrm{d}t, \tag{17}$$

where the expectation is taken over $x_0 \sim \rho_0$. x_t and y_t are computed through the ODE system (14), with $y_0 = \phi_{\mathcal{N}}(0, x_0)$, $z_t = \nabla_x \phi_{\mathcal{N}}(t, x_t)$, and $h_t = \Delta_x \phi_{\mathcal{N}}(t, x_t)$.

Proposition 3. *Let $\phi_{\mathcal{N}}$ be a function such that $\phi_{\mathcal{N}}(T, \cdot) = -V(\cdot)$. Let ρ be the solution to the Fokker-Planck equation*

$$\partial_t \rho(t, x) + \nabla_x \cdot (\rho(t, x) \, \mathrm{D}_p H(t, x, \nabla_x \phi_{\mathcal{N}}(t, x))) = \frac{1}{\beta} \Delta_x \rho(t, x),$$

with initial condition $\rho(0, \cdot) = \rho_0(\cdot)$. If the loss $\mathcal{L}$ defined by (17) is 0, then ρ and $\phi_{\mathcal{N}}$ is the solution to the FP–HJB system (7).

Proof. According to the ODE system (14), we obtain

$$\begin{aligned} \phi_{\mathcal{N}}(t, x_t) =& \phi_{\mathcal{N}}(0, x_0) + \int_0^t \left(\partial_s \phi_{\mathcal{N}}(s, x_s) + \nabla_x \phi_{\mathcal{N}}(s, x_s)^\top \partial_s x_s \right) \mathrm{d}s \\ =& y_0 + \int_0^t \left[\partial_s \phi_{\mathcal{N}}(s, x_s) + z_s^\top \left(\mathrm{D}_p H(s, x_s, z_s) - \frac{1}{\beta} \nabla_x \log \rho(s, x_s) \right) \right] \mathrm{d}s. \end{aligned}$$

8

Also, y_t satisfies

$$
\begin{aligned}
y_t =& y_0 + \int_0^t \partial_s y_s \, \mathrm{d}s \\
=& y_0 + \int_0^t \left[f_s - \frac{1}{\beta}\Delta_x \phi_{\mathcal{N}}(s, x_s) - H(s, x_s, \nabla_x \phi_{\mathcal{N}}(s, x_s)) \right. \\
& \left. + z_s^{\top} \left(\mathrm{D}_p H(s, x_s, \nabla_x \phi_{\mathcal{N}}(s, x_s)) - \frac{1}{\beta}\nabla_x \log \rho(s, x_s) \right) \right] \mathrm{d}s.
\end{aligned}
$$

Subtracting the above two equations, we obtain

$$
\phi_{\mathcal{N}}(t, x_t) - y_t = \int_0^t \left(\partial_s \phi_{\mathcal{N}}(s, x_s) + \frac{1}{\beta}\Delta_x \phi_{\mathcal{N}}(s, x_s) + H(s, x_s, \nabla_x \phi_{\mathcal{N}}(s, x_s)) - f_s \right) \mathrm{d}s.
$$

So

$$
\mathcal{L} = \mathbb{E} \int_0^T \left[\int_0^t \left(\partial_s \phi_{\mathcal{N}}(s, x_s) + \frac{1}{\beta}\Delta_x \phi_{\mathcal{N}}(s, x_s) + H(s, x_s, \nabla_x \phi_{\mathcal{N}}(s, x_s)) - f_s \right) \mathrm{d}s \right]^2 \mathrm{d}t.
$$

Note that the integrand above is exactly the residual of the HJB equation, so $\mathcal{L} = 0$ implies that $\phi_{\mathcal{N}}$ is the solution to the HJB equation. Therefore, ρ and $\phi_{\mathcal{N}}$ is the solution to the FP–HJB system (7). $\qquad\square$

3.2 Density estimation

Computing the density information is usually necessary and unavoidable in the mean-field control problem. In this work, we consider the classic kernel density estimate (KDE) with a Gaussian Kernel.

Let

$$
\rho_{ref} = \frac{1}{N} \sum_{i=1}^{N} \delta_{x_i}
$$

be the reference measure from sampling. We define the kernel density

$$
\widehat{\rho}(t, x) = K(\cdot, \sigma_K) * \rho_{ref}(x) = \int K(x - y, \sigma_K)\, \rho_{ref}(y)\, \mathrm{d}y,
$$

where $K(z, \sigma_K) = (2\pi\sigma_K^2)^{-d/2} \exp(-|z|^2/(2\sigma_K^2))$ is the Gaussian kernel with $\sigma_K > 0$. The bandwidth σ_K is a hyperparameter.

We remark that we have also tested other kernels such as the Wasserstein proximal kernel proposed in [28]. Its performance is similar to the Gaussian kernel. Thus, we just report Gaussian KDE in this work.

3.3 Numerical implementation of the score-based MFC solver

In this section, we present the numerical implementation of the algorithm.

The function $\phi(t, x)$ is parametrized by a neural network $\phi_{\mathcal{N}}(t, x)$. As for the terminal condition $\phi(t, x) = -V(x)$, we give a hard parametrization through

$$\phi_{\mathcal{N}}(t, x) = \frac{T - t}{T} \mathcal{N}(t, x) - \frac{t}{T} V(x), \tag{18}$$

where $\mathcal{N}(t, x)$ is a standard fully connected deep neural network. If an explicit terminal condition is absent (see the example in Section 4.3 below), we set $\phi_{\mathcal{N}}(t, x) = \mathcal{N}(t, x)$ and include an additional loss term to enforce the terminal condition.

We discretize the ODE system (14) using forward Euler scheme with $N_T + 1$ nodes $\{t_j = j\Delta t\}_{j=0}^{N_T}$ and step size $\Delta t = T/N_T$,

$$\begin{cases} x_{t_{j+1}} = x_{t_j} + \left(\mathrm{D}_p H(t, x_{t_j}, z_{t_j}) - \frac{1}{\beta} \nabla_x \log \widehat{\rho}(t, x_{t_j}) \right) \Delta t \\ y_{t_{j+1}} = y_{t_j} + \left(f_{t_j} - \frac{1}{\beta} h_{t_j} - H(t, x_{t_j}, z_{t_j}) + z_{t_j}^\top \mathrm{D}_p H(t, x_{t_j}, z_{t_j}) - \frac{1}{\beta} z_{t_j}^\top \nabla_x \log \widehat{\rho}(t, x_{t_j}) \right) \Delta t \end{cases} \tag{19}$$

with $z_{t_j} = \nabla_x \phi_{\mathcal{N}}(t, x_{t_j})$ and $h_{t_j} = \Delta_x \phi_{\mathcal{N}}(t, x_{t_j})$ obtained from the auto-differentiation, and $\widehat{\rho}(t, x)$ obtained from the KDE. Given N sampled trajectories $\{x_t^{(i)}\}_{i=1}^N$, this KDE is given by

$$\widehat{\rho}(t, x) = \frac{1}{N} \sum_{i=1}^N K(x - x_t^{(i)}, \sigma_K). \tag{20}$$

The loss function (17) is then discretized into

$$\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^{N_T} \left[\phi_{\mathcal{N}}(t_j, x_{t_j}^{(i)}) - y_{t_j}^{(i)} \right]^2 \Delta t,$$

and we minimize this loss using the Adam optimization scheme. The algorithm is summarized in the pseudocode Algorithm 1. The detailed parameters are deferred to the appendix.

This algorithm can be generalized to the broader MFG setting, where the dependence of the HJB equation on the density ρ is more intricate, as illustrated in Section 4.3.

We will also compare our algorithm with the traditional BSDE method. In the BSDE approach, one sample the trajectories of the stochastic processes (16) using Euler–Maruyama scheme instead of sampling (14) with scheme (19). The other part has the same implementation as our score method. We also remark that in the BSDE method, a shooting loss that only try to match the terminal condition could be constructed. The performance for such shooting method is similar to matching the whole trajectory.

Algorithm 1 Forward-backward score solver for the MFC problem

---

Input: MFC problem, σ_K, number of training steps k_{end}, parameters for Adam optimizer, batch size N, number of time intervals N_T
Output: the solution to the HJB equation
  initialization: parameter for the network $\phi_\mathcal{N}$
  for $k = 1$ **to** k_{end} **do**
      Sample N points $\{x_0^{(i)}\}_{i=1}^N$ from the initial distribution ρ_0
      Compute $y_0^{(i)} = \phi_\mathcal{N}(0, x_0^{(i)})$
      Loss $\mathcal{L} = 0$
      for $j = 0$ **to** $N_T - 1$ **do**
         compute $\widehat{\rho}(t, x)$ through the KDE method (20)
         compute $\{z_{t_j}^{(i)}\}_{i=1}^N$ and $\{h_{t_j}^{(i)}\}_{i=1}^N$ through auto-differentiation
         compute $\{x_{t_{j+1}}^{(i)}\}_{i=1}^N$ and $\{y_{t_{j+1}}^{(i)}\}_{i=1}^N$ through the Euler scheme (19)
         update loss $\mathcal{L} = \mathcal{L} + \frac{1}{N} \sum_{i=1}^N \left(\phi_\mathcal{N}(t_{j+1}, x_{t_{j+1}}^{(i)}) - y_{t_{j+1}}^{(i)} \right)^2 \Delta t$
      end for
      update the parameters for $\phi_\mathcal{N}$ through Adam method to minimize the loss $\mathcal{L}$
  end for

---

4 Numerical examples

In this section, we substantiate the efficacy of our algorithm through the presentation of three numerical examples. The first case under consideration has previously been examined in [4], which has a potential energy. The second example pertains to a linear quadratic mean field control problem, while the third example delves into the realm of systemic risk in finance. Specifically, it addresses the intricate dynamics associated with banks' interactions through borrowing and lending, as elucidated in [5]. The details for the numerical implementation are deferred to the appendix. We also compare our score method with the traditional BSDE method.

4.1 An MFC example with entropy potential energy

We first consider an example that is also studied in [4]. The objective is

$$\inf_v \int_0^T \int_{\mathbb{R}^d} \left(\frac{1}{2} |v(t,x)|^2 + \gamma \log(\rho(t,x)) - \gamma \right) \rho(t,x) \, dx \, dt + \int_{\mathbb{R}^d} V(x) \rho(T,x) \, dx$$

subject to the dynamic (12) with initial distribution $\rho_0 \sim N(0, \frac{1}{\alpha} I_d)$. Here γ and α are two positive constants. The FP-HJB equation pair is

$$\begin{cases} \partial_t \rho(t,x) + \nabla_x \cdot \left(\rho(t,x) \nabla_x \phi(t,x) \right) = \Delta_x \rho(t,x), \\ \partial_t \phi(t,x) + \Delta_x \phi(t,x) + \frac{1}{2} |\nabla_x \phi(t,x)|^2 - \frac{1}{2} |x|^2 - \gamma \log \rho(t,x) = 0, \\ \rho(0,x) = \rho_0(x), \quad \phi(T,x) = -V(x). \end{cases} \quad (21)$$

With proper terminal cost and choice of γ and α that satisfy $\alpha^2 + \gamma\alpha = 1$ (i.e. $\alpha = \dfrac{-\gamma + \sqrt{\gamma^2 + 4}}{2}$), an analytical solution to the problem is obtained, given by

$$\phi(t, x) = \left(d\alpha + \frac{\gamma d}{2} \log(\frac{\alpha}{2\pi}) \right) t - \frac{\alpha |x|^2}{2}$$

and

$$\rho(t, x) = \left(\frac{\alpha}{2\pi} \right)^{d/2} \exp\left(-\frac{\alpha |x|^2}{2} \right).$$

This solution is utilized to assess the error and validate our algorithm. The outcomes of our numerical experiments are depicted in Figure 1. The results are presented separately for 1 and 2 dimensions.

In the first row, representing the 1 dimensional case, the initial figure in the first column displays the training curve for the neural network. It illustrates the L^2 relative error of ϕ, as well as its gradient and Laplacian, presented in logarithmic scale. Notably, approximating ϕ solely with L^2 norm is insufficient due to the essential information provided by $\nabla_x\phi$ for determining the optimal policy (cf. (5) with $p = \nabla_x\phi(t, x)$). Hence, we also report the errors for the derivatives. The shadow in the figure represents the standard deviation observed during multiple test runs. The final errors for these tests are 1.59%, 0.52%, 0.51% respectively. As a comparison, the final errors using the BSDE method are 2.77%, 2.63%, 2.36%, which is higher than our score method. The second figure in the first row illustrates a comparison between the initial value $\phi(0, \cdot)$ and its neural network approximation. Recall that we enforce a rigorous implementation of the terminal condition $\phi(T, \cdot)$ within the network. The network adeptly captures the value function. The third and fourth figures present comparison plots for the terminal density $\rho(T, \cdot)$ and score function $\nabla_x \log(\rho(T, x))$ with the corresponding KDE approximation from the trajectory (13). We also report that the Wasserstein-2 distance between our estimated samples and the true density is 6.41e−2 for our score method and 8.23e−2 for the BSDE method.

Moving to the second row, which corresponds to the 2 dimensional scenario, the initial figure in the first column represents the training curve. The final errors are 1.82%, 0.54%, 0.53%. As a comparison, the final errors using the BSDE method are 1.92%, 2.32%, 2.14%, which is higher than our score method. Given the challenge of directly visualizing high-dimensional input functions, a density plot of $\phi(0, \cdot)$ is shown in the second figure. Specifically, this plot illustrates the density function of $\phi(0, x_0)$ and its neural network approximation, where x_0 is uniformly distributed in the box $[-2/\sqrt{\alpha}, 2/\sqrt{\alpha}] \times [-2/\sqrt{\alpha}, 2/\sqrt{\alpha}]$. Note that $2/\sqrt{\alpha}$ is twice the standard deviation of the initial distribution. Similar visualization techniques for high-dimensional functions have been employed in previous studies [29]. The third and fourth figures in this row compare the contour plots of the density $\rho(T, \cdot)$ and its KDE approximation. The final Wasserstein-2 error for the samples is 9.18e−2 for our score method and 5.87e−2 for the BSDE method.
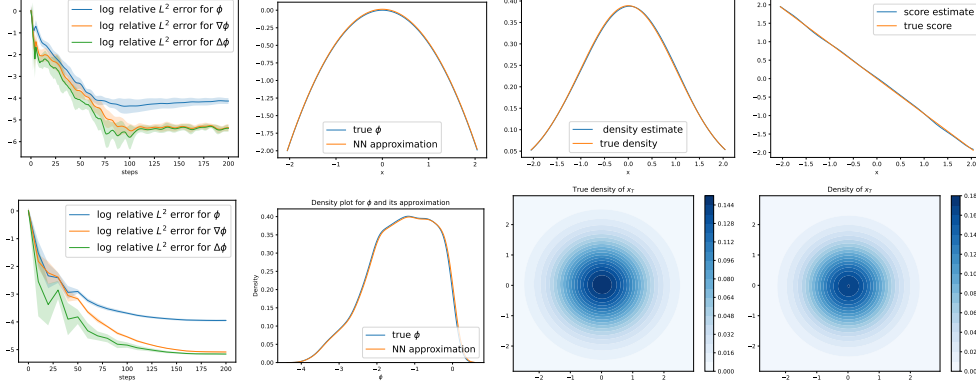
Fig. 1 Numerical results for the MFC problem with log density running cost. The first row illustrates the results in 1 dimension, including the training curve, plot of $\phi(0,\cdot)$, density plot, and score plot. The second row presents the results in 2 dimensions, including the training curve, density plot for $\phi(0,x_0)$, and a contour plot comparison for the density $\rho(T,x_T)$.

4.2 An LQ example

In the subsequent example, we explore a scenario involving linear dynamics and quadratic cost, commonly referred to as the linear quadratic (LQ) problem. The objective is defined by

$$\inf_v \int_0^T \int_{\mathbb{R}^d} \left(\frac{1}{2} |v(t,x)|^2 + \gamma \left(\frac{1}{2}\rho(t,x) - \rho^*(t,x) \right) \right) \rho(t,x)\, dx\, dt + \int_{\mathbb{R}^d} V(x)\rho(T,x)\, dx$$

subject to the dynamic (12) with initial distribution $\rho_0 \sim N(0, 2(T+1)I_d/\beta)$. In this example, the term with coefficient γ is intentionally introduced to increase the difficulty of the problem. The FP-HJB system governing this problem is

$$\begin{cases} \partial_t \rho(t,x) + \nabla_x \cdot (\rho(t,x)\nabla_x\phi(t,x)) = \dfrac{1}{\beta}\Delta_x\rho(t,x), \\[2mm] \partial_t \phi(t,x) + \dfrac{1}{2}|\nabla_x\phi(t,x)|^2 + \dfrac{1}{\beta}\Delta_x\phi(t,x) = \gamma\left(\rho(t,x) - \rho^*(t,x)\right), \\[2mm] \rho(0,x) = \rho_0(x), \quad \phi(T,x) = -V(x) = -|x|^2/2. \end{cases}$$

Here, ρ^* is the true solution with the optimal trajectory, given by

$$\rho^*(t,x) = (4\pi(T-t+1)/\beta)^{-\frac{d}{2}} \exp\left(-\frac{\beta|x|^2}{4(T-t+1)} \right).$$

The solution to the HJB equation is

$$\phi(t,x) = \frac{1}{\beta} d \log \frac{1}{T-t+1} - \frac{|x|^2}{2(T-t+1)}.$$

13

The numerical outcomes are presented in Figure 2, with the first and second rows showing the results for 1 and 2 dimensions respectively.

In the 1 dimensional scenario depicted in the first row, the first, second, and fourth figures mirror the format of Figure 1, illustrating the learning curves, $\phi(0, \cdot)$ plot, and the score $\nabla_x \log(\rho(T, \cdot))$ plot. The final errors for ϕ and their derivatives are 2.00%, 2.76%, 2.75%. The errors for the BSDE method are 2.14%, 2.84%, 2.83%. The final Wasserstein-2 error for the samples is 5.01e−2 for our score method and 4.81e−2 for the BSDE method. In this example, our score method has a similar performance compared with the BSDE method. The third figure displays the variance change for x_t, demonstrating the capability of our score-based dynamics to capture density changes. The green curve represents the true variance under the optimal trajectory, showing a decreasing trend. In contrast, the blue curve represents the variance of x_t when utilizing an untrained neural network, failing to capture the decreasing variance. After training, the variance for x_t aligns with the true variance, as depicted by the orange curve. Note that we have the exact initial condition, so the error of variance at $t = 0$ could be significantly reduced if we increase the number of samples. Here, we have an error of about 0.01 at $t = 0$ due to a limited number of samples.

For the 2 dimensional case presented in the second row, the four figures adopt a format similar to those in Figure 1. The final errors for ϕ and its derivatives are 1.55%, 2.64%, 2.65%. The errors for the BSDE method are 1.68%, 2.64%, 2.62%. The final Wasserstein-2 error for the samples is 6.25e−2 for our score method and 3.55e−2 for the BSDE method. In order to visualize the dynamic of the particles, we present the temporal evolution of particles in figure 3. The first line depicts the dynamic state (12) with the optimal velocity field and true density function. The second line shows the state dynamic with the velocity field and the density approximated by the neural network and KDE estimate respectively. The third line displays the stochastic dynamic (15) with the optimal velocity field. Our approach effectively captures the diminished variance in density, distinguishing itself from the stochastic dynamic, which exhibits a comparatively less structured behavior.

4.3 Systemic risk

This example, as explored in [30], operates within one-dimensional space, where x_t represents the logarithm of a bank's cash reserves. The model captures the interbank borrowing and lending activities through the drift. The objective is to minimize

$$J[\alpha] = \mathbb{E}\left[\int_0^T \left(\frac{1}{2}\alpha_t^2 - q\alpha_t(\overline{x}_t - x_t) + \frac{\varepsilon}{2}(\overline{x}_t - x_t)^2\right) \mathrm{d}t + \frac{c}{2}(\overline{x}_T - x_T)^2\right] \tag{22}$$

subject to the dynamic

$$\mathrm{d}x_t = (a(\overline{x}_t - x_t) + \alpha_t)\,\mathrm{d}t + \sigma\mathrm{d}W_t,$$

where $\overline{x}_t = \mathbb{E}x_t$ denotes the mean cash reserves.
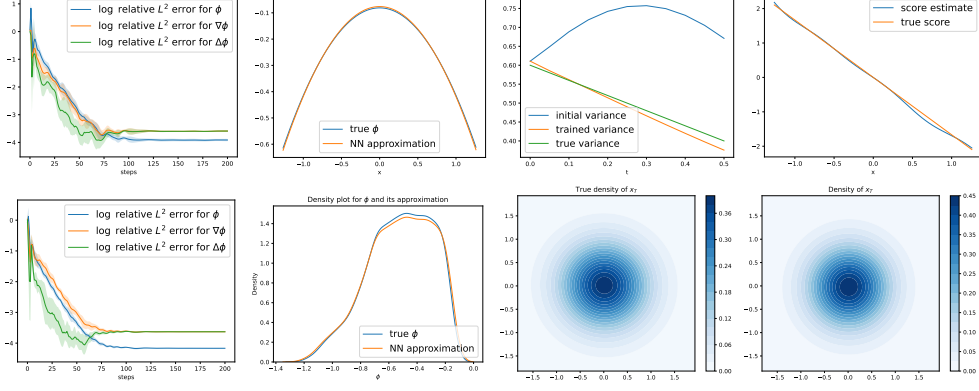
Fig. 2 Numerical results for the LQ MFC problem. The first row presents results in 1 dimension, including the training curve, plot of $\phi(0, \cdot)$, variance plot, and score plot. The second row is results for 2 dimensions, including the training curve, density plot for $\phi(0, x_0)$, and a contour plot comparison for the density $\rho(T, x_T)$.
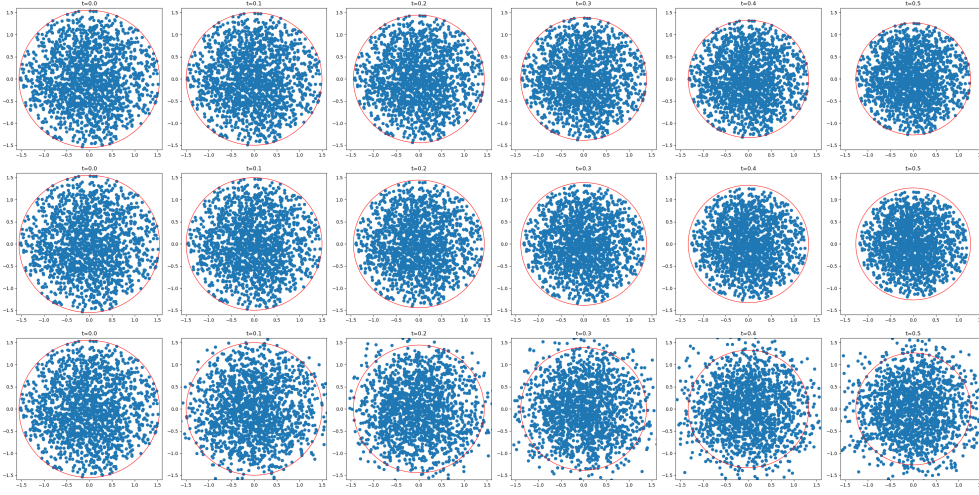

Fig. 3 Particle trajectories for the true score dynamic (first row), approximated score dynamic (second row), and stochastic dynamic (FBSDE) with true velocity (third row). The score dynamic demonstrates a more structured behavior.

The FP-HJB system governing this scenario is represented by

$$
\begin{cases}
\partial_t \phi + \dfrac{1}{2} \left| \nabla_x \phi \right|^2 + (a + q)(\overline{x}_t - x) \cdot \nabla_x \phi - \dfrac{\varepsilon - q^2}{2}(\overline{x}_t - x)^2 + \dfrac{1}{2}\sigma^2 \Delta \phi = 0, \\[2mm]
\partial_t \rho + \nabla_x \cdot \left[\rho \left((a + q)(\overline{x}_t - x) + \nabla_x \phi \right) \right] = \dfrac{1}{2}\sigma^2 \Delta \rho, \\[2mm]
\rho(0, x) = \rho_0(x), \quad \phi(T, x) = -\dfrac{c}{2}(\overline{x}_T - x)^2.
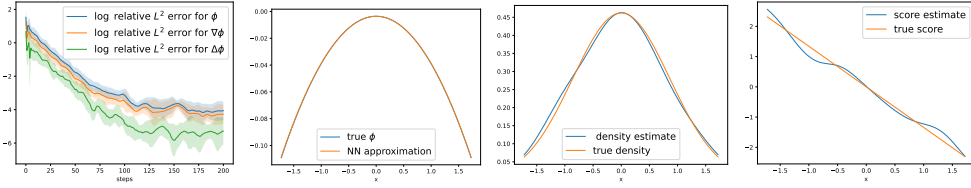\end{cases}
$$

15

Fig. 4 Numerical results for the systemic risk example in 1 dimension, including the training curve, plot of $\phi(0, \cdot)$, density plot, and score plot.

where $\overline{x}_t = \int_{\mathbb{R}} x \rho(t, x) \mathrm{d}x$. The solution of the HJB equation is expressed by $\phi(t, x) = -\eta_t(\overline{x}_t - x)^2 - \chi_t$, where η_t is the solution to the following Riccati equation

$$\partial_t \eta_t = 2(a + q)\eta_t + \eta_t^2 + q^2 - \varepsilon, \qquad \eta_T = c,$$

and $\chi_t = -\frac{1}{2}\sigma^2 \int_t^T \eta_s \mathrm{d}s$.

This example is a generalization of the problem described in section 2, because the density ρ is involved in the Lagrangian (running cost), the terminal cost, and the velocity field. Unlike the preceding examples, this scenario lacks an explicit expression for the terminal condition $\phi(T, \cdot)$. Consequently, a direct parametrization as in (18) is not possible. To address this, an additional loss term

$$T \cdot \mathbb{E}\left[\left(\phi(T, x_T) + \frac{c}{2}(\overline{x}_T - x_T)\right)^2\right]$$

is introduced during training to enforce the terminal condition. In the numerical test, $\overline{x}_t$ is approximated using the empirical mean of the sampled points.

The numerical results are presented in Figure 4, adopting a layout similar to the first row of Figure 1. The final errors for ϕ and its derivatives are 2.18%, 1.76%, 0.95%. The errors for BSDE method are 2.82%, 2.48%, 1.06%. The final Wasserstein-2 error for the samples is 4.10e−2 for our score method and 4.88e−2 for the BSDE method. So our score method performs better than the BSDE method in this example.

5 Conclusion and future directions

In this paper, we introduce a deep learning method to approximate the MFC problems. We first introduce forward-backward score dynamics, which formulates deterministic characteristics for the system of Fokker-Planck equations and Hamilton-Jacobi equations. In the algorithm, we apply the density estimation method to approximate the score function and use the neural network function to approximate the adjoint variable. We design a least square type loss function to fit the proposed deterministic characteristics. Numerical examples, including mean field control problems with entropy potential energies, linear quadratic control problems, and system risks, demonstrate the effectiveness of the proposed methods.

There are several promising directions for future research. From a theoretical perspective, an intriguing direction is to derive higher-order equations for our ODE system. Such equations could serve as a deterministic analog of the higher-order adjoint

16

equation in FBSDE. From the numerical aspect, the accuracy of the density estimate with the Gaussian kernel is below our expectation. Further numerical studies on density estimation and a more comprehensive comparison against classical forward-backward SDE methods are important for a thorough understanding of our proposed methodology. Finally, it is also important to generalize the algorithm to MFC problems where the diffusion is under control. In this scenario, the HJB equation is fully nonlinear, resulting in a more complicated system.

Data availability statement There is no data involved in this paper. Our algorithm is based on the packages in Python.

References

[1] Carmonaa, R., Laurièrea, M.: Deep learning for mean field games and mean field control with applications to finance. Machine Learning and Data Sciences for Financial Markets: A Guide to Contemporary Practices, 369 (2023)

[2] Bauso, D., Tembine, H., Basar, T.: Opinion dynamics in social networks through mean-field games. SIAM Journal on Control and Optimization **54**(6), 3225–3257 (2016)

[3] Elamvazhuthi, K., Berman, S.: Mean-field models in swarm robotics: A survey. Bioinspiration Biomimetics **15**(1), 015001 (2019)

[4] Lin, A.T., Fung, S.W., Li, W., Nurbekyan, L., Osher, S.J.: Apac-net: Alternating the population and agent control via two neural networks to solve high-dimensional stochastic mean field games. arXiv preprint arXiv:2002.10113 (2020)

[5] Carmona, R., Delarue, F., *et al.*: Probabilistic Theory of Mean Field Games with Applications I-II. Springer (2018)

[6] Yong, J., Zhou, X.Y.: Stochastic Controls: Hamiltonian Systems and HJB Equations vol. 43. Springer (1999)

[7] Zhou, X.Y.: Hamiltonian systems, HJB equations, and stochastic controls. In: Proceedings of the 36th IEEE Conference on Decision and Control, vol. 4, pp. 3436–3441 (1997). IEEE

[8] Boffi, N.M., Vanden-Eijnden, E.: Probability flow solution of the fokker–planck equation. Machine Learning: Science and Technology **4**(3), 035012 (2023)

[9] Carmona, R., Delarue, F.: Forward—backward stochastic differential equations

and controlled mckean—vlasov dynamics. The Annals of Probability **43**(5), 2647–2700 (2015)

[10] Raynal, P.-E.C., Frikha, N.: From the backward kolmogorov pde on the wasserstein space to propagation of chaos for mckean-vlasov sdes. Journal de Mathématiques Pures et Appliquées **156**, 1–124 (2021)

[11] Germain, M., Mikael, J., Warin, X.: Numerical resolution of mckean-vlasov fbsdes using neural networks. Methodology and Computing in Applied Probability **24**(4), 2557–2586 (2022)

[12] Han, J., Hu, R., Long, J.: Learning high-dimensional mckean-vlasov forward-backward stochastic differential equations with general distribution dependence. arXiv preprint arXiv:2204.11924 (2022)

[13] Li, W., Liu, S., Osher, S.: Controlling conservation laws ii: Compressible navier–stokes equations. Journal of Computational Physics **463**, 111264 (2022)

[14] Li, W., Liu, S., Osher, S.: Controlling conservation laws i: Entropy–entropy flux. Journal of Computational Physics **480**, 112019 (2023)

[15] Li, W., Lee, W., Osher, S.: Computational mean-field information dynamics associated with reaction-diffusion equations. Journal of Computational Physics **466**, 111409 (2022)

[16] Angiuli, A., Fouque, J.-P., Laurière, M.: Unified reinforcement q-learning for mean field game and control problems. Mathematics of Control, Signals, and Systems **34**(2), 217–271 (2022)

[17] Beck, C., E, W., Jentzen, A.: Machine learning approximation algorithms for high-dimensional fully nonlinear partial differential equations and second-order backward stochastic differential equations. Journal of Nonlinear Science **29**, 1563–1619 (2019)

[18] Ruthotto, L., Osher, S.J., Li, W., Nurbekyan, L., Fung, S.W.: 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 (2020)

[19] Zhou, M., Han, J., Lu, J.: Actor-critic method for high dimensional static hamilton–jacobi–bellman partial differential equations based on neural networks. SIAM Journal on Scientific Computing **43**(6), 4043–4066 (2021)

[20] Carmona, R., Laurière, M.: 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 (2021)

[21] Carmona, R., Laurière, M.: Convergence analysis of machine learning algorithms for the numerical solution of mean field control and games: Ii—the finite horizon case. The Annals of Applied Probability **32**(6), 4065–4105 (2022)

[22] Ji, S., Peng, S., Peng, Y., Zhang, X.: Three algorithms for solving high-dimensional fully coupled fbsdes through deep learning. IEEE Intelligent Systems **35**(3), 71–84 (2020)

[23] Ma, J., Yong, J.: Forward-backward Stochastic Differential Equations and Their Applications vol. 1702. Springer (1999)

[24] Zhou, M., Lu, J.: A policy gradient framework for stochastic optimal control problems with global convergence guarantee. arXiv preprint arXiv:2302.05816 (2023)

[25] Villani, C.: Topics in Optimal Transportation vol. 58. American Mathematical Soc. (2021)

[26] Cardaliaguet, P., Delarue, F., Lasry, J.-M., Lions, P.-L.: The Master Equation and the Convergence Problem in Mean Field Games:(ams-201). Princeton University Press (2019)

[27] Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.: Score-based generative modeling through stochastic differential equations. In: International Conference on Learning Representations (2020)

[28] Tan, H.Y., Osher, S., Li, W.: Noise-free sampling algorithms via regularized wasserstein proximals. arXiv preprint arXiv:2308.14945 (2023)

[29] Han, J., Lu, J., Zhou, M.: Solving high-dimensional eigenvalue problems using deep neural networks: A diffusion monte carlo like approach. Journal of Computational Physics **423**, 109792 (2020)

[30] Carmona, R.A., Fouque, J.P., Sun, L.H.: Mean field games and systemic risk. Communications in Mathematical Sciences **13**(4), 911–933 (2015)

[31] Flamary, R., Courty, N., Gramfort, A., Alaya, M.Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N.T.H., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D.J., Tavenard, R., Tong, A., Vayer, T.: Pot: Python optimal transport. Journal of Machine Learning Research **22**(78), 1–8 (2021)

Appendix A   Details for the numerical implementation

For all the examples in Section 4, we employ a fully connected network with 2 layers and 30 nodes in each layer, with the square of ReLU as the activation function. For

the first two examples, we implement a hard parametrization of the terminal condition (cf. (18)). For all the examples, we set $N_t = 10$, $k_{end} = 200$. The number of validation samples is $1000d$ for all the examples. The default parameters for the PyTorch Adam optimizer are utilized for model training. Detailed parameters for all examples are provided in table A1. We use the same network structure and same optimization scheme for our score method and the BSDE method.

	T	Learning rate	σ_K	batch size	others
potential energy 1d	0.5	0.02	0.35	200	$\gamma = 0.1$
potential energy 2d	0.5	0.1	0.4	1000	$\gamma = 0.1$
LQ 1d	0.5	0.1	0.35	200	$\beta = 5, \gamma = 0.1$
LQ 2d	0.5	0.1	0.4	1000	$\beta = 5, \gamma = 0.1$
Systemic risk	0.1	0.02	0.3	400	$\sigma = 1, q = 0.5, q = \varepsilon = 0.1$

Table A1 Parameters for the examples.

We elucidate the computation of the Wasserstein-2 distance between the samples $\{x_T^{(i)}\}_{i=1}^N$ and the density $\rho(T, \cdot)$. Directly computing the Wasserstein distance between them is non-trivial and computationally expensive. Alternatively, we compute the mean and covariance of the samples $\{x_T^{(i)}\}_{i=1}^N$, subsequently determining the Wasserstein-2 distance between two Gaussian distributions: one with the computed mean and covariance, and the other representing the true density. In comparison, we also calculate the systemic error, defined as the Wasserstein-2 distance between the true density and the estimated Gaussian. The mean and covariance for this estimation are derived from data directly sampled from the true density. The Wasserstein-2 error and the corresponding systemic error are listed in table A2.

	score method	BSDE method	systemic error
potential energy 1d	6.41e−2	8.23e−2	4.20e−2
potential energy 2d	9.18e−2	5.87e−2	4.28e−2
LQ 1d	5.01e−2	4.81e−2	2.59e−2
LQ 2d	6.25e−2	3.55e−2	2.64e−2
Systemic risk	4.10e−2	4.88e−2	3.08e−2

Table A2 Wasserstein-2 error and the systemic errors for all the examples. Each error is the average of multiple runs.

Here, we remark that we also tested the Wasserstein-2 errors between $\{x_T^{(i)}\}_{i=1}^N$ and data sampled from the true distribution, with the help of the package [31]. However, we observed that the systemic error, defined as the Wasserstein-2 distance between two sets of independent samples from the true distribution, was significantly large due to the limited sample size.