Research Article

Variational Conditional Normalizing Flows for Computing Second-Order Mean Field Control Problems

Jiaxi Zhao[1], Mo Zhou[2], Xinzhe Zuo[2], Wuchen Li[3]

1. Department of Mathematics, National University of Singapore, Singapore; 2. Department of Mathematics, University of California, Los Angeles, United States; 3. Department of Mathematics, University of South Carolina, United States

Mean field control (MFC) problems have vast applications in artificial intelligence, engineering, and economics, while solving MFC problems accurately and efficiently in high-dimensional spaces remains challenging. This work introduces variational conditional normalizing flow (VCNF), a neural network-based variational algorithm for solving general MFC problems based on flow maps. Formulating MFC problems as optimal control of Fokker–Planck (FP) equations with suitable constraints and cost functionals, we use VCNF to model the Lagrangian formulation of the MFC problems. In particular, VCNF builds upon conditional normalizing flows and neural spline flows, allowing efficient calculations of the inverse push-forward maps and score functions in MFC problems. We demonstrate the effectiveness of VCNF through extensive numerical examples, including optimal transport, regularized Wasserstein proximal operators, and flow matching problems for FP equations.

1. Introduction

In recent years, generative artificial intelligence (AI) has attracted much attention in the scientific computing community across various fields. Typical applications include text generation[1][2], image generation[3][4][5], and protein folding[6][7]. At the core of these applications are nonlinear generative models, such as normalizing flows[8] and neural ordinary differential equations[9]. These models aim to learn a transport map from a reference distribution to a target distribution, enabling sample generation that approximates the target. The optimization problems in generative models can be interpreted as variational problems in probability space, also known as the mean field control (MFC) problems[10][11].

 doi.org/10.32388/COTXYL

MFC problems incorporate interactions among multiple agents in the optimal control problem, providing a method for modeling large-scale collective behaviors in artificial intelligence[12], engineering[13][14], and economics[15][16]. In literature, the second-order MFC problems can be expressed as a coupled system of Hamilton–Jacobi–Bellman (HJB) and Fokker–Planck (FP) equations, which jointly characterize the optimal velocity field governing the agents' density evolution. The second-order MFC emphasizes Kolmogorov forward and backward operators from the diffusion processes in the coupled systems. In particular, optimal transport (OT), i.e., Wasserstein distances between probability distributions, with their dynamical formulation[17], can be viewed as a special class of first order MFC problem with fixed initial and terminal time constraints on distributions and without diffusion.

Traditional methods for solving OT and MFC problems are well-studied in low-dimensional spaces[18][17][19]. Recently, machine learning-based approaches have emerged as promising tools for tackling high-dimensional MFCs, pioneered by[20], which solves high-dimensional deterministic MFCs by approximating the value function using deep neural networks. In Lagrangian coordinates, solving MFC problems often involves solving the Monge–Ampére equation, which arises from the change of variable formula for probability densities. Classical numerical PDE algorithms[17][21] perform well in low dimensions. In comparison, modern generative models tackle the Monge–Ampére equation from a variational perspective, optimizing over a parametrized family of transformations by minimizing certain objective functions, such as the Kullback–Leibler (KL) divergence between the generated and target distribution. A typical example is the neural spline flow[8], which constructs transport maps based on monotone rational-quadratic splines. This network structure enables efficient computation of both the inverse transport map and the determinant of the Jacobian of the transport map. Such a formulation is particularly useful in approximating the score function, defined as the gradient of the log-density function. Additionally, the score function approximates the Kolmogorov forward operator associated with diffusion processes. Given the potential of generative models and score functions, a natural question arises: *Can we apply generative models to solve second-order MFC problems in high dimensional spaces?*

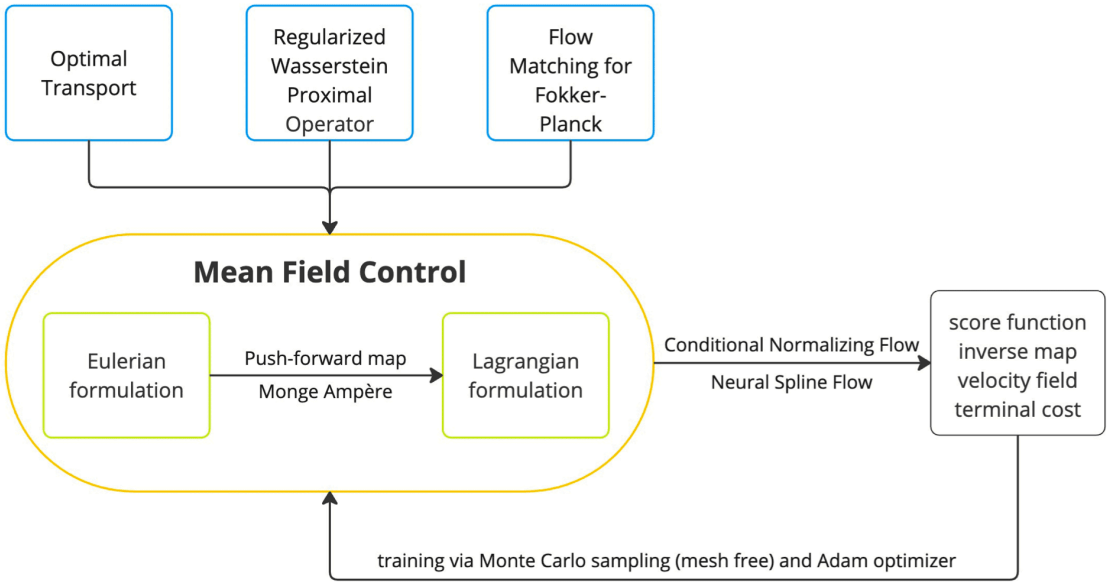

Figure 1. VCNF framework for solving generalized MFC problems.

In this paper, we introduce the variational conditional normalizing flow (VCNF), an extension of conditional normalizing flows[22] designed for solving general MFC problems. Conditioning on time t, VCNF models a family of distributions $p(x,t)$ that evolve continuously over time to solve the MFC problems. Unlike standard normalizing flows with fixed time stamps, VCNF allows for evaluation and sampling at arbitrary time. The time derivative can be computed using finite difference or automatic differentiation. By leveraging numerical approximations of score functions, VCNF can effectively handle second-order MFC problems with diffusive density evolution. Specifically, we demonstrate this capability by solving the FP equation within the flow matching framework. We show in the numerical experiments that VCNF is highly robust and efficient in solving various MFC problems. In particular, we are able to achieve around 10^{-3} relative error, which is commonly expected for neural network-based models in scientific computing[23], with little parameter tuning. In addition, VCNF effectively captures the evolution of multimodal distributions and accurately models the transitions between unimodal and multimodal distributions under different initial conditions, demonstrating its flexibility and reliability. A schematic of our framework is demonstrated in Figure 1, details of which will be discussed in subsections 2.2, 2.3, and 3.1.

This paper is organized as follows. In section 2, we review some background on MFC problems, including both Eulerian and Lagrangian formulations. Several important examples are covered. In section 3, we

present the construction of the model, the details for evaluating various quantities, and the algorithm for solving MFC problems. In section 4, we provide extensive numerical experiments to show the robustness and efficiency of VCNF. In section 5, we conclude this work and discuss potential future directions.

2. Mean field control problems

In this section, we review second-order MFC problems in Eulerian and Lagrangian coordinates. We also introduce several important examples, including OT, RWPO, and flow matching for FP equation.

2.1. Mean field control problem in Eulerian coordinates

We consider the MFC problem where one aims to minimize the variational objective

$$\inf_{\mathbf{v},p} \int_0^T \int_{\mathbb{R}^d} L(\mathbf{x},t,\mathbf{v}(\mathbf{x},t),p(\mathbf{x},t))p(\mathbf{x},t)d\mathbf{x}dt + \int_{\mathbb{R}^d} G(\mathbf{x},p(\mathbf{x},T))p(\mathbf{x},T)d\mathbf{x}, \tag{2.1}$$

subject to the FP equation with initial condition

$$\partial_t p(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot (p(\mathbf{x},t)\mathbf{v}(\mathbf{x},t)) = \gamma \Delta_{\mathbf{x}} p(\mathbf{x},t), \quad p(\cdot,0) = p_0. \tag{2.2}$$

The objective functional involves a running cost $L : \mathbb{R}^d \times [0,T] \times \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}$ and a terminal cost $G : \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}$. The state dynamic is characterized by the evolution of the density function (2). In the MFC problem, a central planner controls the drift of the dynamic $\mathbf{v}(\mathbf{x},t)$, which has a direct impact on the density $p(\mathbf{x},t)$. The goal of the MFC problem is to minimize the cost (1) w.r.t. $\mathbf{v}$ and p subject to the constraint (2). The diffusion term $\Delta_{\mathbf{x}} p(\mathbf{x},t)$ indicates that this MFC problem is of second-order.

The MFC problem also has a stochastic formulation:

$$\inf_{\mathbf{v}} \mathbb{E}\left[\int_0^T L(\mathbf{x}_t,t,\mathbf{v}(\mathbf{x}_t,t),p(\mathbf{x}_t,t)dt + G(\mathbf{x}_T,p(\mathbf{x}_T,T)\right],$$

subject to the state dynamic

$$d\mathbf{x}_t = \mathbf{v}(\mathbf{x}_t,t)dt + \sqrt{2\gamma}dW_t, \quad \mathbf{x}_0 \sim p_0.$$

$p(\cdot,t)$ is the density function of $\mathbf{x}_t$ and satisfies the FP equation (2). W_t is a d-dimensional standard Brownian motion. $\mathbb{E}$ is the expectation over all realizations of the state dynamic.

We remark that mean field games (MFGs) [24][25] differ from MFCs primarily due to the absence of a central planner in MFGs. Consequently, a change of control by a single agent does not affect the population distribution. However, for a large class of MFG problems, one can find the corresponding

variational formulations (also known as potential MFGs) and transform them into MFC problems [20][26][27].

Numerical methods for solving MFC problems generally fall into two broad categories. The first category focuses on the variational formulation of MFC, optimizing the objective functional directly. In this framework, probabilistic modeling techniques, such as neural ODE [9] and normalizing flow [28] are commonly used to get the population distribution. These methods offer advantages in flexibility and scalability, making them well-suited for high-dimensional problems. The second category leverages the HJB formulation of the MFC, where the optimal solution is characterized by a coupled FP-HJB system. Common approaches in this category parameterize the solution to the HJB equation and solve the FP-HJB system [29][30][31][32][33]. Hybrid approaches also exist, which employ mixed loss functions to enhance robustness and accuracy [34][35].

For completeness, we present the HJB-FP for MFC problem. Interested readers could refer to Chapter 4 [36] for more details. We define the Hamiltonian $H : \mathbb{R}^d \times [0,T] \times \mathbb{R}^d \times \mathbb{R}_+ \to \mathbb{R}$ as the Legendre transform of L w.r.t. $\mathbf{v}$,

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

Then, the optimal control, i.e., optimal velocity, is given by

$$\mathbf{v}(\mathbf{x}, t) = \mathrm{argmax}_{\mathbf{v} \in \mathbb{R}^d} \nabla_{\mathbf{x}} \phi(\mathbf{x}, t)^\top \mathbf{v} - L(\mathbf{x}, t, \mathbf{v}, p(\mathbf{x}, t)),$$

where $p(\mathbf{x}, t)$ and $\phi(\mathbf{x}, t)$ satisfy the coupled FP-HJB system

$$\begin{cases} \partial_t p(\mathbf{x}, t) + \nabla_{\mathbf{x}} \cdot (p(\mathbf{x}, t) \mathrm{D}_{\mathbf{w}} H(\mathbf{x}, t, \nabla_{\mathbf{x}} \phi(\mathbf{x}, t), p(\mathbf{x}, t))) = \gamma \Delta_{\mathbf{x}} p(\mathbf{x}, t)\,, \\ \partial_t \phi(\mathbf{x}, t) + H(\mathbf{x}, t, \nabla_{\mathbf{x}} \phi(\mathbf{x}, t), p(\mathbf{x}, t)) \\ \qquad\qquad - \dfrac{\partial L(\mathbf{x}, t, \mathbf{v}(\mathbf{x}, t), p(\mathbf{x}, t))}{\partial p} \cdot p(\mathbf{x}, t) = -\gamma \Delta_{\mathbf{x}} \phi(\mathbf{x}, t)\,, \\ p(\cdot, 0) = p_0\,, \qquad \phi(\mathbf{x}, T) = -G(\mathbf{x}, p(\mathbf{x}, T)) - \dfrac{\partial G(\mathbf{x}, p(\mathbf{x}, T))}{\partial p} p(\mathbf{x}, T)\,. \end{cases}$$

2.2. Mean field control problem in Lagrangian coordinates

We can also describe the MFC problem in Lagrangian coordinates, where we focus on the push-forward map f instead of the velocity field $\mathbf{v}$. Let $q(\mathbf{z})$ be a reference measure. In this paper, we use the standard Gaussian distribution $\mathcal{N}(0, I_d)$ as the reference measure, and we will not distinguish a probability measure with its density function. Let $f : \mathbb{R}^d \times [0,T] \to \mathbb{R}^d$ be a flow of invertible push forward map, which gives the density function at $t \in [0,T]$ by

$$p(\cdot,t) = f(\cdot,t)_{\#}q(\cdot)\,.$$

The associated Monge–Ampére equation is

$$p(\mathbf{x},t) = p(f(\mathbf{z},t),t) = q(\mathbf{z})\left|\det\frac{\partial f(\mathbf{z},t)}{\partial \mathbf{z}}\right|^{-1}.$$

In this case, the density function p satisfies the transport equation

$$\partial_t p(\mathbf{x},t) + \nabla_{\mathbf{x}}\cdot\left[p(\mathbf{x},t)\,\partial_t f(f^{-1}(\mathbf{x},t),t)\right] = \partial_t p(\mathbf{x},t) + \nabla_{\mathbf{x}}\cdot\left[p(\mathbf{x},t)\,\partial_t f(\mathbf{z},t)\right] = 0\,,$$

where $f^{-1}(\cdot,t)$ is the inverse map of $f(\cdot,t)$. If we assume that $p(\mathbf{x},t)$ satisfies the FP equation (2.2), then we can apply Nelson's transformation[37]. Since

$$\nabla p(\mathbf{x},t) = p(\mathbf{x},t)\cdot\frac{\nabla p(\mathbf{x},t)}{p(\mathbf{x},t)} = p(\mathbf{x},t)\nabla\log p(\mathbf{x},t)\,,$$

where $\nabla\log p(\mathbf{x},t)$ is often called the score function, the FP equation can be rewritten as

$$\partial_t p(\mathbf{x},t) + \nabla_{\mathbf{x}}\cdot\left[p(\mathbf{x},t)\left(\mathbf{v}(\mathbf{x},t) - \gamma\nabla_{\mathbf{x}}\log p(\mathbf{x},t)\right)\right] = 0\,. \tag{2.3}$$

We can compute the velocity field $\mathbf{v}$ through

$$\mathbf{v}(\mathbf{x},t) = \partial_t f(f^{-1}(\mathbf{x},t),t) + \gamma\nabla_{\mathbf{x}}\log p(\mathbf{x},t) = \partial_t f(\mathbf{z},t) + \gamma\nabla_{\mathbf{x}}\log p(f(\mathbf{z},t),t)\,.$$

Therefore, the MFC problem in Lagrangian coordinates is

$$\begin{aligned}\inf_{f,p}\int_0^T\int_{\mathbb{R}^d} & L\left(f(\mathbf{z},t),t,\partial_t f(\mathbf{z},t) + \gamma\nabla_{\mathbf{x}}\log p(f(\mathbf{z},t),t),p(f(\mathbf{z},t),t)\right)q(\mathbf{z})\,\mathrm{d}\mathbf{z}\,\mathrm{d}t\\ &+\int_{\mathbb{R}^d} G(f(\mathbf{z},t),p(f(\mathbf{z},t),T))q(\mathbf{z})\,\mathrm{d}\mathbf{z}\\ \text{s.t.}\quad & f(\cdot,t)_{\#}q = p(\cdot,t)\,,\ f(\cdot,0)_{\#}q = p_0\,.\end{aligned} \tag{2.4}$$

In VCNF, the density $p(f(\mathbf{z},t),t)$ is obtained through the logarithm of the Monge–Ampére equation

$$\log p(\mathbf{x},t) = \log p(f(\mathbf{z},t),t) = \log q(\mathbf{z}) - \log\left|\det\frac{\partial f(\mathbf{z},t)}{\partial \mathbf{z}}\right|. \tag{2.5}$$

The temporal derivative of the push forward map $f(\mathbf{z},t)$ and the score function $\nabla_{\mathbf{x}}\log p(f(\mathbf{z},t),t)$ are approximated through numerical differentiation (see subsection 3.2).

2.3. Examples of mean field control problems

We provide three important examples for MFC problems in this section, including OT, RWPO, and flow matching for FP equations.

Optimal transportation. The OT problem can be viewed as a special MFC problem where the terminal cost is replaced by a terminal constraint. By the Benamou–Brenier formulation[17], computing the Wasserstein-2 distance between two probability distributions is equivalent to the following variational problem

$$\begin{aligned} &\inf_{v,p} \int_0^1 \int_{\mathbb{R}^d} \frac{1}{2} \|v(\mathbf{x},t)\|^2 p(x,t)\,\mathrm{d}x\,\mathrm{d}t\,, \\ &\text{s.t.} \quad \partial_t p(\mathbf{x},t) + \nabla \cdot (p(\mathbf{x},t) v(\mathbf{x},t)) = 0\,, \quad p(\cdot,0) = p_0\,, \quad p(\cdot,1) = p_1\,. \end{aligned} \tag{2.6}$$

The minimal cost to (2.6) is $\frac{1}{2} W_2(p_0,p_1)^2$, where $W_2(\cdot,\cdot)$ denotes the Wasserstein-2 distance between two measures. In Lagrangian coordinates, the OT problem reads

$$\begin{aligned} &\inf_{f,p} \int_0^1 \int_{\mathbb{R}^d} \frac{1}{2} \|\partial_t f(\mathbf{z},t)\|^2 q(\mathbf{z})\,\mathrm{d}\mathbf{z}\,\mathrm{d}t\,, \\ &\text{s.t.} \quad f(\cdot,0)_{\#} q = p_0\,, \quad f(\cdot,1)_{\#} q = p_1\,. \end{aligned}$$

Regularized Wasserstein proximal operator. Closely related to OT, the RWPO computes a regularized OT problem with a diffusion term and a terminal cost[38]. The objective is

$$\begin{aligned} &\inf_{\mathbf{v},p} \int_0^T \int_{\mathbb{R}^d} \frac{1}{2} \|\mathbf{v}(\mathbf{x},t)\|^2 p(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t + \int_{\mathbb{R}^d} V(\mathbf{x}) p(\mathbf{x},T)\,\mathrm{d}\mathbf{x}\,, \\ &\text{s.t.} \quad \partial_t p(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot (p(\mathbf{x},t)\mathbf{v}(\mathbf{x},t)) = \frac{1}{\beta} \Delta_{\mathbf{x}} p(\mathbf{x},t)\,, \quad p(\cdot,0) = p_0\,. \end{aligned} \tag{2.7}$$

For derivation of this variational formulation, see Appendix B.1. Using Nelson's transformation introduced in Subsection 2.2, the RWPO problem in Lagrangian coordinate is

$$\begin{aligned} &\inf_{f,p} \int_0^T \int_{\mathbb{R}^d} \frac{1}{2} \left\| \partial_t f(\mathbf{z},t) + \frac{1}{\beta} \nabla_{\mathbf{x}} \log p(f(\mathbf{z},t),t) \right\|^2 q(\mathbf{z})\,\mathrm{d}\mathbf{z}\,\mathrm{d}t + \int_{\mathbb{R}^d} V(f(\mathbf{z},T))\, q(\mathbf{z})\,\mathrm{d}\mathbf{z}\,, \\ &\text{s.t.} \quad f(\cdot,t)_{\#} q = p(\cdot,t)\,, \quad f(\cdot,0)_{\#} q = p_0\,. \end{aligned} \tag{2.8}$$

Flow matching for Fokker–Planck equation. Another important application of MFC is the flow matching problem for the FP equation

$$\partial_t p(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot (p(\mathbf{x},t)\mathbf{b}(\mathbf{x},t)) = \gamma \Delta_{\mathbf{x}} p(\mathbf{x},t)\,.$$

For example,[39][40] use NFs to solve the steady state of the (fractional) FP equation. Similarly, CNFs are used to approximate the time-dependent FP equation[41].

In our setting, we aim to learn the push forward map $f : \mathbb{R}^d \times [0,T] \to \mathbb{R}^d$ such that $f(\cdot,t)_{\#} q = p(\cdot,t)$ represents the solution to the FP equation at t. The objective functional is

$$\begin{aligned} &\inf_{f,p} \int_0^T \int_{\mathbb{R}^d} \|\partial_t f(f(\mathbf{z},t),t) - b(f(\mathbf{z},t),t) + \gamma \nabla_{\mathbf{x}} \log p(f(\mathbf{z},t),t)\|^2 q(\mathbf{z})\,\mathrm{d}\mathbf{z}\,\mathrm{d}t\,, \\ &\text{s.t.} \quad f(\cdot,t)_{\#} q = p(\cdot,t)\,, \quad f(\cdot,0)_{\#} q = p_0\,. \end{aligned} \tag{2.9}$$

3. Variational conditional normalizing flows

In this section, we describe the architecture of the VCNF model based on invertible spline transformations. We then explain its applications to solving MFC problems.

3.1. Conditional normalizing flow via neural spline flow

As discussed in Subsection 2.2, the Lagrangian formulation of MFC problems requires parameterizations of a family of invertible maps in high dimensions. Following the idea in[8], we build multi-dimensional invertible transformations through one dimensional invertible transformations $\phi(\cdot,\theta):\mathbb{R}\to\mathbb{R}$ parameterized by θ based on spline functions (see details in Appendix A.1). Comparing to models[42] with affine transformation, this class of models are powerful in fitting complex distributions[8].

The overall transformation is built on compositions of several invertible auto-regressive layers:

$$f_\psi(\cdot,t)=f_\psi^{(l)}(\cdot,t)\circ\cdots\circ f_\psi^{(1)}(\cdot,t). \tag{3.1}$$

where ψ denotes the parameters of the whole model. $\{f_\psi^{(i)}(\cdot,t)\}_{i=1}^{l}$ are auto-regressive layers with different coupling orders. A typical auto-regressive layer with coupling order $(1,2,\dots,d)$ consists of d coupling transformations $\mathbf{x}^{(k)}\mapsto\mathbf{x}^{(k+1)}$ composed sequentially (for $k=0,\dots,d-1$), each of which consists of the following steps:

1. Compute $\theta_k=\mathrm{NN}^{(k)}(\mathbf{x}_{1:k}^{(k)},t)\in\mathbb{R}^{3K-1}$ where $\mathrm{NN}^{(k)}$ is the conditioning network.
2. Compute $x_{k+1}^{(k+1)}=\phi(x_{k+1}^{(k)},\theta_k)$, the detailed parametrization of $\phi(x_{k+1}^{(k)},\theta_k)$ is explained in Appendix 1.
3. Set $x_i^{(k+1)}=x_i^{(k)}$ for $i=1,\dots,k,k+2,\dots,d$ and return $\mathbf{x}^{(k+1)}$.

Note that θ_k, the parameters of one dimensional invertible transformations, are not the *inputs* but the *outputs* of the conditioning networks $\mathrm{NN}^{(k)}:\mathbb{R}^{k+1}\to\mathbb{R}^{3K-1}$, which are fully-connected feedforward neural networks for $k=0,\dots,d-1$. The K is the number of the bins for the spline functions. For example, an auto-regressive layer of the CNF in 2 dimensions with coupling order $(1,2)$ can be written as

$$\begin{pmatrix}x_1^{(0)}\\x_2^{(0)}\end{pmatrix}\xrightarrow[x_2^{(1)}=x_2^{(0)}]{x_1^{(1)}=\phi(x_1^{(0)},\mathrm{NN}_1^{(0)}(t))}\begin{pmatrix}x_1^{(1)}\\x_2^{(1)}\end{pmatrix}\xrightarrow[x_2^{(2)}=\phi(x_2^{(1)},\,\mathrm{NN}_2^{(1)}(x_1^{(1)},t))]{x_1^{(2)}=x_1^{(1)}}\begin{pmatrix}x_1^{(2)}\\x_2^{(2)}\end{pmatrix}.$$

Specifically, the 2D transformation is decomposed into two steps: in the first step, the first coordinate performs invertible transformations parametrized by the output of the conditioning network $\mathrm{NN}^{(0)}$ and the second step contains an invertible transformation of the second coordinate.

As illustrated in (3.2), for different times t_1 and t_2, the outputs of the conditioning networks and their corresponding invertible transformations are different. Therefore the distributions generated at different times are not the same. Moreover, as the coupling is directed, another auto-regressive layer implementing the other coupling direction (from x_2 to x_1) is needed to guarantee the expressivity. In this work, we apply a network structure that has two auto-regressive layers with reverse coupling orders $(1, 2, \ldots, n)$ and $(n, n-1, \ldots, 1)$, so that any two dimensions are coupled. Our model is simpler than the network in[26] with eight auto-regressive layers, while producing competitive results. Within each auto-regressive layer, we use 5 bins for each spline transformation layer; the conditioner network contains 2 hidden layers with width 16. Consequently, the total number of parameters of the network is 1200 for all of our 2D problems. The prior q is chosen to be the standard Gaussian. For all experiments, we use the Adam[43] optimizer with a learning rate of 10^{-3}.

Our VCNF can also approximate key ingredients of MFC problems efficiently. The velocity field, which is the temporal derivative of the push forward map $\partial_t f(\mathbf{z}, t)$, can be calculated by auto-differentiation or numerical differentiation schemes. Throughout the paper, we use the following central difference schemes to approximate the temporal derivative of the push forward map $f(\mathbf{z}, t)$ and the spatial derivative for the log-density $\log p(\mathbf{x}, t)$:

$$\begin{aligned} \partial_t f(\mathbf{x}, t) &\approx \frac{f(\mathbf{x}, t + \Delta t/2) - f(\mathbf{x}, t - \Delta t/2)}{\Delta t} =: \mathrm{D}_t^{\Delta t} f(\mathbf{x}, t)\,, \\ \partial_{\mathbf{x}_i} \log p(\mathbf{x}, t) &\approx \frac{\log p(\mathbf{x} + \Delta x \mathbf{e}_i/2, t) - \log p(\mathbf{x} - \Delta x \mathbf{e}_i/2, t)}{\Delta x} =: \left(\mathrm{D}_{\mathbf{x}}^{\Delta x} \log p(\mathbf{x}, t)\right)_i, \forall i \end{aligned} \tag{3.3}$$

where $\mathbf{e}_i$ is the standard basis in $\mathbb{R}^n$, and $\log p$ is evaluated via (2.5). Applying chain rule to (3.1), one obtains $\log\left|\det \frac{\partial f_\psi}{\partial \mathbf{z}}\right| = \sum_{i=1}^{l} \log\left|\det \frac{\partial f_\psi^{(i)}}{\partial \mathbf{z}^{(i)}}\right|$ where $f_\psi^{(i)}$ is the i-th auto-regressive layer and $\mathbf{z}^{(i)} = f_\psi^{(i-1)} \circ \cdots \circ f_\psi^{(1)}(\mathbf{z}, t)$ (with convention $\mathbf{z}^{(1)} = \mathbf{z}$) is the spatial input of $f_\psi^{(i)}$.

3.2. Numerical approximation for mean field control problems

In this section, we explain the numerical approximation to the MFC problems. We use the dynamical formulation of OT as an illustrative example.

As per (2.6), we need to calculate the kinetic energy associated with the temporal normalizing flow and enforce the constraints. From the Lagrangian perspective, the continuity equation is inherently satisfied by the flow model. However, the initial and terminal conditions at $t = 0, 1$ must be enforced explicitly. To address this, we introduce two extra terms of KL divergence to penalize the deviation of the VCNF model at $t = 0$ and 1. The objective functional we used is written in time-continuous form as

$$\int_0^1 \int \frac{1}{2} \|v(\mathbf{x}, t)\|^2 p(x, t) \mathrm{d}x \, \mathrm{d}t + \lambda \left(\mathrm{D}_{\mathrm{KL}}(p_0 || p(\cdot, 0)) + \mathrm{D}_{\mathrm{KL}}(p_1 || p(\cdot, 1)) \right) ,$$

where $\lambda > 0$ is a weight parameter. While one can use primal-dual algorithms to update the dual variable λ, we find in experiments that using a fixed λ yields better performance. Note that when implementing the KL divergence numerically, the term $\int \log(p_0(\mathbf{x})) p_0(\mathbf{x}) \mathrm{d}x$ in $\mathrm{D}_{\mathrm{KL}}(p_0 || p(\cdot, 0)) = \int \log(p_0(\mathbf{x})) p_0(\mathbf{x}) \mathrm{d}x - \int \log(p(\mathbf{x}, 0)) p_0(\mathbf{x}) \mathrm{d}x$ can be omitted as it does not depend on the trainable parameters.

Next, as the kinetic energy contains double integration over both the spatial and temporal dimensions, we employ a Monte Carlo sampling method. The integration over t is simulated through uniform sampling from $[0, 1]$. As for the integration over $\mathbf{x}$, we first sample $\{\mathbf{z}_j\}_{j=1}^{N_k}$ from latent distribution $q(\cdot)$ and then compute the transformed samples using the normalizing flow model $\mathbf{x} = f(\mathbf{z}, t)$, as described in (2.4).

Lastly, while the velocity field can be computed via auto-differentiation, we find that the finite difference scheme yields satisfactory results while being computationally more efficient.

Combining all components, we compute the loss function in a mini-batch setting through the Monte Carlo sampling method, ensuring a balance between computational efficiency and approximation accuracy. We first obtain i.i.d. samples $\{\mathbf{z}_j^{(i)}\}_{i=1,\, j=1}^{N_t \quad N_k}$ and $\{\mathbf{z}_j^{(b,0)}, \mathbf{z}_j^{(b,1)}\}_{j=1}^{N_b}$ from $q(\mathbf{z})$, and then approximate the loss function as

$$\frac{1}{N_t N_k} \sum_{i=1}^{N_t} \sum_{j=1}^{N_k} \frac{1}{2} \left\| \mathrm{D}_t^{\Delta t} f(\mathbf{z}_j^{(i)}, t^{(i)}) \right\|_2^2 - \frac{\lambda}{N_b} \sum_{j=1}^{N_b} \left(\log p(\mathbf{x}_j^{(0)}, 0) + \log p(\mathbf{x}_j^{(1)}, 1) \right) , \tag{3.4}$$

where $\mathbf{x}_j^{(0)} = f(\mathbf{z}_j^{(b,0)}, 0)$ and $\mathbf{x}_j^{(1)} = f(\mathbf{z}_j^{(b,1)}, 1)$. Here, N_k is the batch size of the samples from the normalizing flow models, while N_b is the batch size of the samples from the initial and target distributions. In our experiments, we set $N_t = 20, N_k = 64, N_b = 2048$.

3.3. Algorithm

With the above discussions, we present Algorithm 3.1 to solve the MFC problem using VCNF. We illustrate our algorithm using the OT problem. Other related MFC problems can be treated similarly by modifying the loss function in step 6.

Algorithm 3.1 Solving MFC with VCNF

1: **Input**: time horizon T; time batch size N_t; spatial batch size N_k, N_b; number of iterations N; penalty strength λ; problem dependent parameters such as γ.
2: **for** $k = 1, 2, \ldots, N$ **do**
3: Sample $t^{(1)}, ..., t^{(N_t)}$ uniformly from $[0, T]$.
4: For each $t^{(i)}$, generate N_k samples $\{\mathbf{z}_j^{(i)}\}_{j=1}^{N_k}$ from reference measure $q(\mathbf{z})$.
5: For $t = 0, 1$, feed the samples $\{\mathbf{z}_j^{(b,0)}, \mathbf{z}_j^{(b,1)}\}_{j=1}^{N_b}$ through the VCNF to obtain samples $\{\mathbf{x}_j^{(0)}, \mathbf{x}_j^{(1)}\}_{j=1}^{N_b}$ from $p(\cdot|t=0), p(\cdot|t=1)$.
6: Evaluate the loss function through (3.4)
7: Update the parameters in VCNF using Adam algorithm
8: **end for**
9: **Output**: Optimized VCNF for solving the MFC problem

4. Numerical experiments

In this section, we perform numerical experiments to test the effectiveness of our VCNF. All the experiments were conducted on a single NVIDIA A100 GPU with 40GB memory and CUDA 12.2. with driver 535.129.03. For all experiments, we use the Adam[43] optimizer with a learning rate of 10^{-3} for 30000 optimization steps. The code reproducing all the experiments can be found in[44].

4.1. Optimal transport

We present the numerical results for OT problems in this subsection. We first test the OT problems between two Gaussian distributions, where a closed-form solution is used to compute the errors. We then test more general distributions and present qualitative results.

2D OT: Gaussian to Gaussian. Given two Gaussian distributions $\mathcal{N}(\mu_0, \Sigma_0)$ and $\mathcal{N}(\mu_1, \Sigma_1)$, the optimal transport map is given by

$$T(x) = \mu_1 + A(\mathbf{x} - \mu_0), \quad A = \Sigma_0^{-1/2} \left(\Sigma_0^{1/2} \Sigma_1 \Sigma_0^{1/2} \right)^{1/2} \Sigma_0^{-1/2}.$$

The corresponding optimal flow map is $f(\cdot, t) = (1-t)\mathrm{Id} + tT(\cdot)$. The squared Wasserstein-2 distance between $\mathcal{N}(\mu_0, \Sigma_0)$ and $\mathcal{N}(\mu_1, \Sigma_1)$ is

$$\|\mu_0 - \mu_1\|_2^2 + \mathrm{Tr}\left(\Sigma_0 + \Sigma_1 - 2\left(\Sigma_0^{1/2}\Sigma_1\Sigma_0^{1/2}\right)^{1/2}\right).$$

We calculate the accuracy of our method using the above formulations for several different initial and target distributions listed in Table 1. The benchmarks are squared Wasserstein-2 distances between the initial and target distributions multiplied by $\frac{1}{2}$. The approximated value of squared Wasserstein-2 distances for different penalty λ and the relative errors (in bracket) are presented in the table. We leave the errors for the transportation maps in the appendix.

	Benchmark	$\lambda = 100$	$\lambda = 200$	$\lambda = 500$	$\lambda = 1000$	$\lambda = 2000$	$\lambda = 5000$
case 1	36.000	34.594 (3.90%)	35.296 (1.96%)	35.868 (0.37%)	35.882 (0.33%)	36.341 (0.95%)	36.568 (1.58%)
case 2	36.125	35.045 (2.99%)	35.606 (1.43%)	36.281 (0.43%)	36.474 (0.97%)	36.799 (1.87%)	37.030 (2.51%)
case 3	36.860	35.133 (4.69%)	35.572 (3.49%)	36.226 (1.72%)	35.898 (2.61%)	36.892 (0.09%)	37.871 (2.75%)
case 4	36.931	33.560 (9.13%)	35.370 (4.23%)	36.555 (1.02%)	36.700 (0.63%)	37.090 (0.43%)	37.380 (1.22%)
case 5	0.125	0.1254 (0.32%)	0.1264 (1.12%)	0.1284 (2.72%)	0.1289 (3.12%)	0.1323 (5.84%)	0.1347 (7.76%)
case 6	0.860	0.8345 (2.91%)	0.8715 (1.40%)	0.9068 (5.47%)	0.9351 (8.72%)	0.9571 (11.28%)	0.9973 (15.93%)
case 7	0.931	0.8907 (4.33%)	0.9232 (0.84%)	0.9564 (2.73%)	0.9733 (4.54%)	1.0300 (10.63%)	1.0779 (15.78%)

Table 1. Accuracy of VCNF on 2D Gaussian-to-Gaussian in OT problem. The source and target distributions are Gaussian distributions with different mean and covariances. Mean: in cases 1-4, the source is centered at $(-3, -3)$, and the target is centered at $(3, 3)$; in cases 5-7, both the source and the target are centered at $(0, 0)$. Covariances: the source covariance for case 1 is [[1,0], [0,1]]; the source covariance for case 2 and 5 is [[1,0], [0,0.25]]; the source covariance for case 3 and 6 as [[4,1.5],[1.5,3]]; the source covariance for case 4 and 7: [[5,1], [1,0.5]]. The covariances for the target distributions are identity matrices for all cases. The N_t is fixed to be 20.

Figure 2 shows a visualization of the optimal transportation process. The red dots show the particle trajectories from the source distribution to the target distribution. The densities are also visualized at

each time stamp. Greater dots represent earlier time stamps and this convention is kept for later visualization.

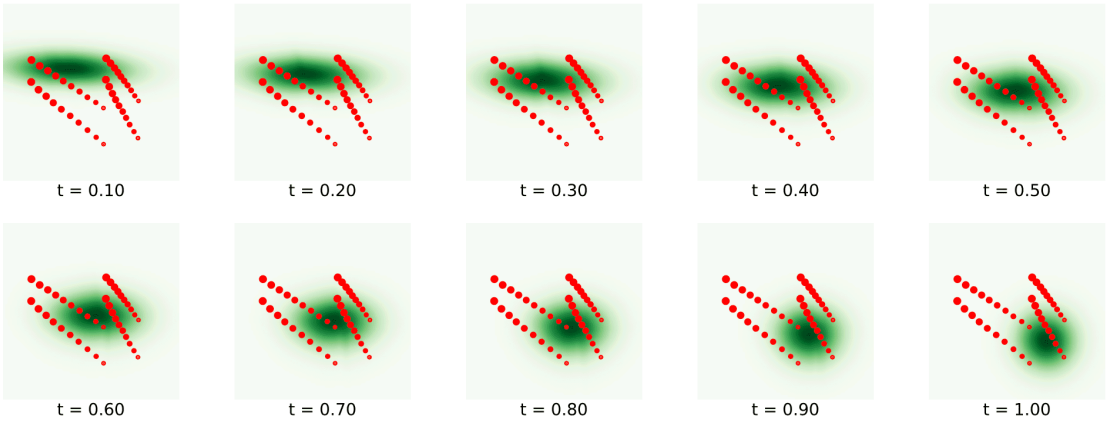

Figure 2. 2D Gaussian-to-Gaussian in OT problem. The source and target distributions are given by $\mathcal{N}\left(\begin{pmatrix}-3\\3\end{pmatrix}, \begin{pmatrix}5 & 1\\1 & 0.5\end{pmatrix}\right)$ and $\mathcal{N}\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\right)$ respectively. We visualize both the density evolution and several trajectories of the particles. In all the figures, the density heatmap is plotted for the domain $[-6, 3] \times [-3, 6]$. The four red curves represent the trajectories of four starting points $(-5, 3.5), (-5, 2.5), (-1, 3.5), (-1, 2.5)$ moving from upper left to lower right.

2D OT: Gaussian mixture to Gaussian. VCNF can also solve complex transport problems from a Gaussian mixture distribution to a Gaussian distribution as shown in Figure 3. We compute the transport map from a mixture of eight Gaussian distributions with centers $(5, 0), (3, 4), (0, 5), (-3, 4), (-5, 0), (-3, -4), (0, -5), (3, -4)$ to a single Gaussian distribution. All eight trajectories in red dots and the density plot at each time stamp demonstrate the effectiveness of our VCNF model.

doi.org/10.32388/COTXYL

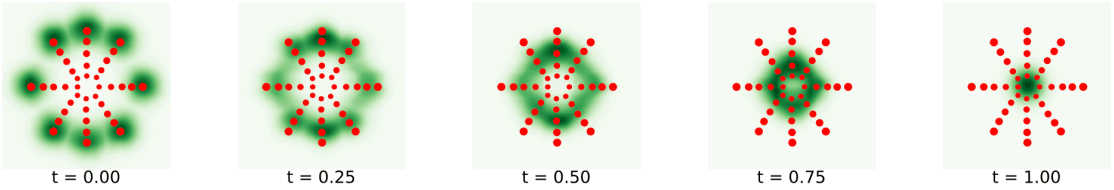

Figure 3. 2D Gaussian mixture-to-Gaussian in OT problem. All trajectories start from the centers of the Gaussian mixture components, i.e. $(5,0),(3,4),(0,5),(-3,4),(-5,0),(-3,-4),(0,-5),(3,-4)$ and the plot domain is $[-7.5,7.5]\times[-7.5,7.5]$. The terminal Gaussian distribution is centered at the origin and all Gaussian distributions have covariance matrix $\mathbf{I}$.

4.2. Regularized Wasserstein proximal operator

In this subsection, we compute the RWPO problem (2.7) using VCNF. We take the initial distribution $\rho_0 \sim \mathcal{N}(0, 2(T+1)I_d/\beta)$ and the potential function $V(\mathbf{x}) = \|x\|^2/2$. To estimate the velocity field and the score function in (2.8), we apply the finite difference scheme (3.3). The reason we avoid using auto-differentiation for the score function is that, the number of forward calculations is proportional to the dimension of the state space, which is not efficient in high dimensions. We also remark that gradient estimators based on control variates can mitigate this issue[45].

Denote by ψ all the trainable parameters. The objective functional (2.8) is approximately

$$\begin{aligned} L(\psi) &= \int_0^T \int_{\mathbb{R}^d} \frac{1}{2} \left\| \partial_t f_\psi(\mathbf{z},t) + \frac{1}{\beta}\nabla_{\mathbf{x}} \log p(f_\psi(\mathbf{z},t)) \right\|^2 q(\mathbf{z})\,\mathrm{d}\mathbf{z}\,\mathrm{d}t \\ &\quad + \lambda \mathrm{D}_{\mathrm{KL}}(p_0 || p(\cdot,0)) + \mathbb{E}_{\mathbf{x}\sim f_\psi(\cdot,1)_\# p}[V(\mathbf{x})] \\ &\approx \frac{T}{2N_tN_k} \sum_{i=1}^{N_t}\sum_{j=1}^{N_k} \left\| \mathrm{D}_t^{\Delta t} f(\mathbf{z}_j^{(i)}, t^{(i)}) + \frac{1}{\beta}\mathrm{D}_{\mathbf{x}}^{\Delta \mathbf{x}} \log p(f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}), t^{(i)}) \right\|^2 \\ &\quad + C + \frac{\lambda}{N_b}\sum_{i=1}^{N_b}(-\log p(\mathbf{x}_i^{(0)}, 0)) + \frac{1}{N_1}\sum_{j=1}^{N_1} V(f_\psi(\mathbf{z}_j^{(1)}, 1)), \end{aligned} \tag{4.1}$$

where N_b, N_1, N_k stands for the batch size of the initial condition loss, potential loss, and kinetic energy loss of the objective function, respectively. The constant C represents the entropy of p_0 which is irrelavant to the optimization.

doi.org/10.32388/COTXYL

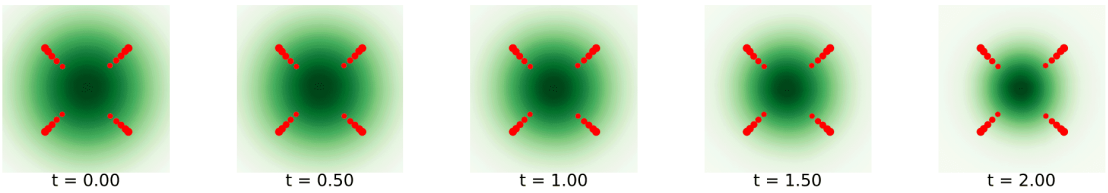

Figure 4. RWPO with initial condition of Gaussian distribution with covariance $\Sigma = 4\mathbf{I}$ and quadratic potential $V(\mathbf{x}) = \|x\|^2/2$. We choose $T = 2, \beta = 1, \lambda = 200$ with t batch size 20. The density on the domain $[-4, 4] \times [-4, 4]$ is visualized with trajectories starting from $(-3, -3), (-3, 3), (3, 3), (3, -3)$ that contract to the middle.

We perform numerical experiments with different diffusive constants β and report the accuracy in Table 2. We observe that the VCNF model gives reasonable results for a wide range of the penalty parameter λ.

	Benchmark	$\lambda = 100$	$\lambda = 200$	$\lambda = 500$	$\lambda = 1000$	$\lambda = 2000$
$\beta = 1, T = 1$	3.386	3.360 (0.79%)	3.380 (0.18%)	3.3984 (0.37%)	3.404 (0.56%)	3.460 (2.18%)
$\beta = 0.5, T = 1$	6.773	7.254 (7.10%)	6.856 (1.23%)	7.570 (11.77%)	7.016 (3.60%)	6.872 (1.46%)
$\beta = 1, T = 2$	4.197	4.197 (0.01%)	4.184 (0.31%)	4.217 (0.46%)	4.241 (1.05%)	4.284 (2.06%)
$\beta = 0.5, T = 2$	8.394	8.944 (6.54%)	8.409 (0.18%)	9.025 (7.51%)	10.207 (21.59%)	9.559 (13.88%)

Table 2. Accuracy of VCNF on solving the RWPO with a quadratic potential. T is the time period and β is the inverse diffusion coefficient. The N_t is fixed to be 20. It can be observed from the table that as T increases and β decreases, the accuracy of our model deteriorates. This may be caused by the density being subject to greater evolutions.

Regularized Wasserstein proximal operator with double well potential. Next, we consider the experiments in[46]. We can set V to be a double-well potential, given by $V(\mathbf{x}) = ((x_1 - a)^2 + (x_2 + a)^2)((x_1 + a)^2 + (x_2 - a)^2)/4$. The explicit solution is stated in B.1. The

doi.org/10.32388/COTXYL

value of a is taken to be $0.5, 1$. Our solution's visualization and accuracy summary are presented in Figure 5 and Table 3.

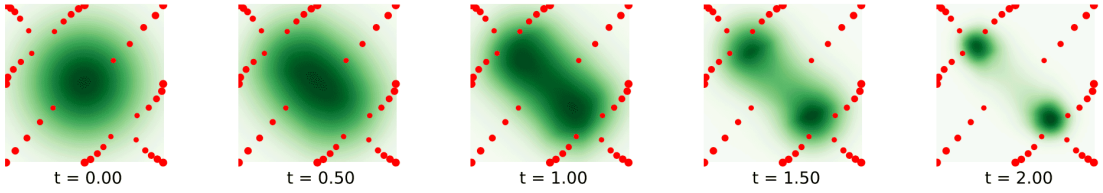

Figure 5. RWPO with initial condition of Gaussian distribution with covariance $\Sigma = 4/5\mathbf{I}$ and double-well potential $V(\mathbf{x}) = ((x_1 - 1)^2 + (x_2 + 1)^2)((x_1 + 1)^2 + (x_2 - 1)^2)/4$. We choose $T = 2, \beta = 5, \lambda = 100$, and $N_t = 10$. The density on domain $[-2, 2] \times [-2, 2]$ is visualized and red points denote the trajectories starting from the boundary points $(-2, -2), (-2, 0), (-2, 2), (0, 2), (2, 2), (2, 0), (2, -2), (0, -2)$ and then contracting to two wells.

Density	$\lambda = 100$	$\lambda = 200$	$\lambda = 500$	$\lambda = 1000$	$\lambda = 2000$	$\lambda = 5000$
$\beta = 5, a = 1$	2.621%	2.948%	3.067%	3.281%	3.307%	4.545%
$\beta = 5, a = 0.5$	4.129%	4.262%	4.309%	4.357%	4.933%	4.755%
$\beta = 10, a = 1$	2.766%	3.437%	4.146%	5.566%	6.927%	5.238%
$\beta = 10, a = 0.5$	4.008%	3.858%	5.066%	4.916%	4.472%	5.287%

Table 3. Accuracy of VCNF on solving RWPO with Gaussian initial condition and double-well potential. Throughout the experiments, we change a (the double-well potential) and β (inverse diffusion coefficient) to test the accuracy and robustness of VCNF. T is fixed to be 2 and $N_t = 10$.

4.3. Flow matching for Fokker–Planck equation

Lastly, we compute the flow matching problem for the FP equation using VCNF. With the help of the score function, one can rewrite the FP equation (2.3) as a continuity equation. Therefore, solving the FP equation is equivalent to matching the velocity field of VCNF to the drift field in the FP equation subtracted by the score function. Note that the score function can be efficiently evaluated using VCNF.

Recently, solving FP equations under Lagrangian coordinate with neural network is also investigated in[47], with numerical analysis on the convergence order of the proposed method.

Fokker–Planck equations from Ornstein–Uhlenbeck (OU) processes. We first consider the FP equation for the OU process $\partial_t p(\mathbf{x},t) = \nabla_{\mathbf{x}} \cdot (a\mathbf{x}\, p(\mathbf{x},t)) + \gamma \Delta_{\mathbf{x}} p(\mathbf{x},t)\,, t \in [0,T]$. By rewriting it as a continuity equation with velocity given by $-a\mathbf{x} - \gamma \nabla_{\mathbf{x}} \log p(\mathbf{x},t)$, we can solve it by flow matching with the loss function

$$\int_0^T \int \|v(\mathbf{x},t) + a\mathbf{x} + \gamma \nabla_{\mathbf{x}} \log p(\mathbf{x},t)\|^2 p(x,t)\,\mathrm{d}x\,\mathrm{d}t + \lambda \mathrm{D}_{\mathrm{KL}}(p_0||p(\cdot,0))\,.$$

We remark that this loss function is different from the physics-informed loss function[40], which is derived from the FP equation via auto-differentiation. Instead, our method only requires first-order derivatives, which is computationally more efficient. This loss is also similar to the flow matching loss function in[48]. We approximate the integration via Monte Carlo sampling method:

$$\begin{aligned} L(\psi) \approx \frac{1}{N_t N_k} \sum_{i=1}^{N_t} \sum_{j=1}^{N_k} \left\| \partial_t f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}) + a f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}) + \gamma \nabla_{\mathbf{x}} \log p(f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}), t^{(i)}) \right\|^2 \\ + \lambda \sum_{i=1}^{N_b} (-\log p(\mathbf{x}_i^{(0)}, 0))\,. \end{aligned} \tag{4.2}$$

The score function is estimated via finite differences (3.3). We report the root mean squared error (RMSE) of the solution of FP equation using our model at time $T = 1$ over a grid of size $N = 500 \times 500$ on $[-5,5] \times [-5,5]$. The RMSE error between the computed density $p(\cdot,t) = f_\psi(\cdot,t)_{\#} q(\cdot)$ and the true solution p^* at $t = 1$ is approximated by Monte Carlo sampling:

$$\mathrm{RMSE}(p^*, p) \approx \sqrt{\frac{1}{N} \sum_{i=1}^{N} p^*(\mathbf{x}_i, t=1) - p(\mathbf{x}_i, t=1)^2},$$

where $\mathbf{x}_i = f_\psi(\mathbf{z}_i, 1)$ and $\mathbf{z}_i$ are i.i.d. sampled from $q(\cdot)$. The results are concluded in Table 4. We observe that our model is robust and is not sensitive to the penalty parameter λ.

Density	$\lambda = 100$	$\lambda = 200$	$\lambda = 500$	$\lambda = 1000$	$\lambda = 2000$	$\lambda = 5000$
$a = 1$	8.274×10^{-4}	5.966×10^{-4}	5.840×10^{-4}	6.776×10^{-4}	6.521×10^{-4}	1.074×10^{-3}
$a = 0.5$	3.750×10^{-4}	1.482×10^{-4}	2.477×10^{-4}	4.138×10^{-4}	4.354×10^{-4}	2.812×10^{-3}

Table 4. Accuracy of VCNF on solving the FP equation via the flow matching framework. We vary the drift parameter a. The initial distribution is Gaussian with zero mean and covariance $4\mathbf{I}$ and $\gamma = 0.5$. This table records the RMSE of the solution obtained by VCNF calculated on a 500×500 grid of $[-5, 5] \times [-5, 5]$.

Fokker–Planck equation with non-gradient velocity field. In the previous example, the velocity field a can be viewed as the gradient field of some potential function, which gives the FP equation a gradient flow interpretation in Wasserstein-2 metric space. Moreover, the invariant measure satisfies the detailed balance condition at equilibrium. However, FP equations with non-gradient velocity fields are also of importance[49][50]. Many FP equations admit non-equilibrium stationary states without detailed balance, this is also closely related to the recent success in generative modeling using diffusion models[51]. In this section, we consider FP equations with a non-gradient velocity field and a "smiling" invariant measure:

$$\pi(\mathbf{x}) \propto \exp(-U(\mathbf{x})) = \exp\left(-\frac{1}{4}(x_1^2 + x_2^2 - 4)^2 - (x_2 + 1)^2\right).$$

The non-gradient velocity field is given by a combination of the original gradient vector field and a small Hamiltonian vector field perturbation of size $\delta \in \mathbb{R}$

$$\mathbf{v}(\mathbf{x}) = -\nabla U(\mathbf{x}) - \delta \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \nabla U(\mathbf{x}) = -\begin{pmatrix} (x_1 + \delta x_2)(x_1^2 + x_2^2 - 4) + 2\delta(x_2 + 1) \\ (x_2 - \delta x_1)(x_1^2 + x_2^2 - 4) + 2(x_2 + 1) \end{pmatrix}.$$

This Hamiltonian vector field perturbation will leave the original invariant measure unchanged while breaking the detailed balance condition for the stochastic process[50]. Our results are demonstrated in Figure 6. Similar to (4.2), the objective function $L(\psi)$ is approximately given by

$$\begin{aligned} &\frac{1}{N_t N_k} \sum_{i=1}^{N_t} \sum_{j=1}^{N_k} \left\| \partial_t f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}) + \gamma \nabla_{\mathbf{x}} \log p(f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}), t^{(i)}) - \mathbf{v}(f_\psi(\mathbf{z}_j^{(i)}, t^{(i)})) \right\|^2 \\ &+ \lambda \sum_{i=1}^{N_b} (-\log p(\mathbf{x}_i^{(0)}, 0)), \end{aligned} \tag{4.3}$$

with $a(f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}))$ substituted by $-\mathbf{v}(f_\psi(\mathbf{z}_j^{(i)}, t^{(i)}))$.

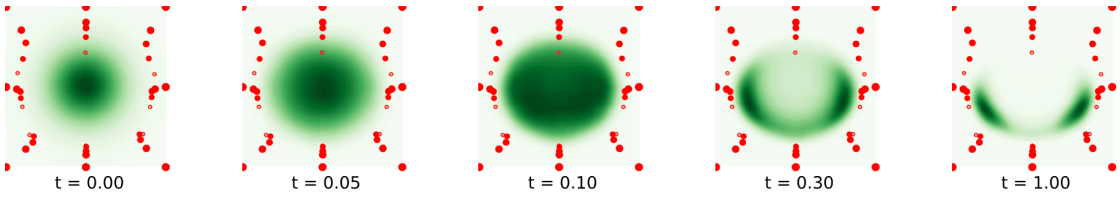

Figure 6. FP equation with standard Gaussian initial distribution and "smiling" invariant distribution. The "smiling" distribution $\pi(\mathbf{x})$ has potential function $V(\mathbf{x}) = \frac{1}{4}(x_1^2 + x_2^2 - 4)^2 + (x_2 + 1)^2$. The δ is fixed to be 0.5. Notice that we choose non-uniform time step for better visualization of the trajectories. The domain of visualization is $[[-3, 3] \times [-3, 3]$ with all trajectories starting at the boundary points $(-3, -3), (-3, 0), (-3, 3), (0, 3), (3, 3), (3, 0), (3, -3)$. All the trajectories converge to the lower hemicycle $x_1^2 + x_2^2 = 4$.

4.4. Scalability of the algorithm

We also test the scalability of the VCNF on solving high dimensional problems. For OT, we can solve for transport maps between Gaussian distributions over 20D within 17 mins to achieve 1% relative error using 30000 steps. For the FP equation, our model can handle 10D problems in 34 mins for 30000 steps and achieve a 10^{-5} absolute L_2 error calculated by Monte Carlo sampling with 10^8 samples. For the RWPO, our algorithm takes 53 mins to run 30000 steps and achieves a relative error of 0.44%. We also find that choosing the smallest temporal batch size, i.e., $N_t = 1$ throughout all the high-dimensional experiments is extremely advantageous as it significantly reduces the computational cost while maintaining a good accuracy.

5. Discussion

In this paper we developed VCNF, which is a neural network-based framework for solving MFC problems using methods in generative models. In particular, our neural network structure makes use of both CNFs and neural spline flows. By leveraging the conditional generative model, one can capture the probability distribution's temporal evolution at any given time, which naturally fits into the trajectory-wise formulation of the MFC problems. Moreover, the CNF structure allows us to evaluate various quantities of interests, such as the velocity, kinetic energy, score function, through numerical differentiation and

Monte–Carlo sampling. By taking a Lagrangian perspective of MFC problems, the objective functional can be efficiently estimated and optimized by Monte–Carlo sampling. We demonstrate the effectiveness and accuracy of the VCNF in solving various problems, including OT, RWPOs, and controlling non-gradient vector field FP equations. Specifically, we solve the initial value FP equation under the framework of flow matching technique from generative models. Our model is robust to handle evolution problems between distributions with different modalities, e.g., from a Gaussian mixture distribution to a Gaussian distribution. Moreover, our method can be efficiently scaled to high dimensions with reasonable computational cost.

In future work, we shall explore using VCNF on flow matching problems to simulate general FP equations, including those with nonlinear drift vector fields. Another interesting direction is to apply the proposed framework to real image data sets for solving time-reversible diffusion models.

Appendix A. Implementation details

A.1. Details of the network architecture

We briefly introduce the conditional normalizing flow beyond subsection 3.1 with consistent notations. We refer interested readers to[8] for more details. We first discuss how the conditioning network is used to parametrize the monotonic rational-quadratic spline:

- A conditioning neural network $\mathrm{NN}^{(k)}$ takes $x_{1:k}$ and t as input and outputs a vector θ_k of length $3K-1$.
- Vector θ_k is partitioned as $\theta_k = [\theta_k^w, \theta_k^h, \theta_k^d]$, where θ_k^w, θ_k^h have length K, and θ_k^d has length $K-1$.
- Vectors θ_k^w, θ_k^h are passed through an elementwise softmax function separately and then multiplied by $2B$. Two outputs $l^{(x)} = (l_1^{(x)}, \dots, l_K^{(x)}), (l_1^{(y)}, \dots, l_K^{(y)})$ are both vectors with K positive components that sum up to $2B$. Therefore, they are used as the width of K bins spanning $[-B, B]$ in x and y axes, see Figure 1 of[8]. Equivalently, the nodes of the spline function are given by

$$x_k = \sum_{i=1}^{k} l_i^{(x)} - B, \quad y_k = \sum_{i=1}^{k} l_i^{(y)} - B, \quad x_0 = y_0 = -B, \quad x_K = y_K = B. \tag{A.1}$$

- The vector θ_k^d is passed through a softplus function and is interpreted as the values of the derivatives $\{\delta^{(k)}\}_{k=1}^{K-1}$ at the internal knots.

The above method constructs a monotonic, continuously-differentiable, rational-quadratic spline which passes through the knots, with the given derivatives at the knots. Defining $s^{(k)} = \frac{y^{(k+1)} - y^{(k)}}{x^{(k+1)} - x^{(k)}}, \xi(x) = \frac{x - x^{(k)}}{x^{(k+1)} - x^{(k)}} \in [0, 1]$, the expression for the rational-quadratic spline over the k-th bin is given by

$$\frac{\alpha^{(k)}(\xi)}{\beta^{(k)}(\xi)} = y^{(k)} + \frac{(y^{(k+1)} - y^{(k)})[s^{(k)}\xi^2 + \delta^{(k)}\xi(1-\xi)]}{s^{(k)} + [\delta^{(k+1)} + \delta^{(k)} - 2s^{(k)}]\xi(1-\xi)} \tag{A.2}$$

for $\xi \in [0, 1]$. In practice, we use $K = 5$ bins for each spline transformation layer.

Appendix B. Mathematical formulation of the numerical experiments

In this section, we provide the details of the mathematical formulations of the numerical experiments discussed in section 4.

B.1. Regularized Wasserstein proximal operators

Let the initial distribution be $p_0 \sim \mathcal{N}(0, 2(T+1)I_d/\beta)$. The FP-HJB system governing the RWPO problem is

$$\begin{cases} \partial_t p(\mathbf{x}, t) + \nabla_{\mathbf{x}} \cdot (p(\mathbf{x}, t) \nabla_{\mathbf{x}} \phi(\mathbf{x}, t)) = \dfrac{1}{\beta} \Delta_{\mathbf{x}} p(\mathbf{x}, t)\,, \\ \partial_t \phi(\mathbf{x}, t) + \dfrac{1}{2} \|\nabla_{\mathbf{x}} \phi(\mathbf{x}, t)\|^2 + \dfrac{1}{\beta} \Delta_{\mathbf{x}} \phi(\mathbf{x}, t) = 0\,, \\ p(0, \mathbf{x}) = p_0(\mathbf{x})\,, \quad \phi(\mathbf{x}, t) = -V(\mathbf{x}) = -\|\mathbf{x}\|^2/2\,. \end{cases}$$

The true solution p^* is given by

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

The solution to the HJB equation is

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

The optimal velocity is $v^* = \nabla \phi$. Given the close form solution, we can obtain the exact value of the objective functional as

$$
\begin{aligned}
&\int_0^T \int_{\mathbb{R}^d} \frac{1}{2}\|v^*(\mathbf{x},t)\|^2 p^*(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t + \int_{\mathbb{R}^d} V(\mathbf{x}) p^*(\mathbf{x},t)\,\mathrm{d}\mathbf{x} \\
&= \int_0^T \int_{\mathbb{R}^d} \frac{\|\mathbf{x}\|^2}{2(T-t+1)^2} p^*(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t + \int_{\mathbb{R}^d} \frac{1}{2}\|\mathbf{x}\|^2 p^*(\mathbf{x},t)\,\mathrm{d}\mathbf{x} \\
&= \frac{d}{\beta}\left(\int_0^T \frac{\mathrm{d}t}{(T-t+1)} + 1\right) = \frac{d}{\beta}\left(\log(T+1)+1\right).
\end{aligned}
\tag{B.1}
$$

To solve the RWPO problem using our framework, note that the continuity equation contains a diffusion term, which is equivalent to substracting a score function from the original velocity field

$$
\partial_t p(\mathbf{x},t) + \nabla_{\mathbf{x}} \cdot \left(p(\mathbf{x},t)(\mathbf{v}(\mathbf{x},t) - \frac{1}{\beta}\nabla \log p(\mathbf{x},t)) \right) = 0\,. \tag{B.2}
$$

Consequently, the loss function can be directly modified to include the score function

$$
\inf_{\mathbf{v}} \int_0^T \int_{\mathbb{R}^d} \frac{1}{2}\left\| \partial_t f(\mathbf{z},t) + \frac{1}{\beta}\nabla \log p(\mathbf{x},t) \right\|^2 p(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t + \int_{\mathbb{R}^d} V(\mathbf{x}) p(\mathbf{x},T)\,\mathrm{d}\mathbf{x}\,. \tag{B.3}
$$

For generalized potential function $V(\mathbf{x})$, one has the following kernel solution.

Proposition B.1. (Kernel solutions). *The solution to the FP-HJB system for RWPO problem is given by the following kernel formulation.*

$$
p(\mathbf{x},t) = \frac{1}{\left(4\pi\frac{1}{\beta}\frac{t(T-t)}{T}\right)^{\frac{d}{2}}} \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \frac{e^{-\frac{\beta}{2}\left(V(\mathbf{y}') + \frac{\|\mathbf{x}-\mathbf{y}'\|^2}{2(T-t)} + \frac{\|\mathbf{x}-\mathbf{y}\|^2}{2t}\right)}}{\int_{\mathbb{R}^d} e^{-\frac{\beta}{2}\left(V(\tilde{\mathbf{y}}) + \frac{\|\mathbf{y}-\tilde{\mathbf{y}}\|^2}{2T}\right)}\mathrm{d}\tilde{\mathbf{y}}} p(0,\mathbf{y})\mathrm{d}\mathbf{y}\mathrm{d}\mathbf{y}',
$$

and

$$
\phi(\mathbf{x},t) = \frac{2}{\beta}\log\Big(\int_{\mathbb{R}^d} \frac{1}{(\frac{4\pi}{\beta}(T-t))^{\frac{d}{2}}} e^{-\frac{\beta}{2}(V(\mathbf{y}) + \frac{\|\mathbf{x}-\mathbf{y}\|^2}{2(T-t)})}\mathrm{d}\mathbf{y} \Big)\,.
$$

In particular,

$$
p(\mathbf{x},T) = \int_{\mathbb{R}^d} K(\mathbf{x},\mathbf{y},\beta,T,V) p(0,\mathbf{y})\mathrm{d}\mathbf{y}\,,
$$

where $K\colon \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}_+ \times \mathbb{R}_+ \times C^1(\mathbb{R}^d) \to \mathbb{R}$ *is the kernel function*

$$
K(\mathbf{x},\mathbf{y},\beta,T,V) := \frac{e^{-\frac{\beta}{2}\left(V(\mathbf{x}) + \frac{\|\mathbf{x}-\mathbf{y}\|^2}{2T}\right)}}{\int_{\mathbb{R}^d} e^{-\frac{\beta}{2}\left(V(\widetilde{\mathbf{y}}) + \frac{\|\widetilde{\mathbf{y}}-\mathbf{y}\|^2}{2T}\right)}\mathrm{d}\widetilde{\mathbf{y}}}.
$$

The optimal cost is given by $-\int_{\mathbb{R}^d} \phi(\mathbf{x},0) p(\mathbf{x},0)\,\mathrm{d}\mathbf{x}$.

B.2. Fokker–Planck equation

We first consider the FP equation corresponding to the Ornstein–Uhlenbeck (OU) process with constant drift $a > 0$ and constant diffusion $\gamma > 0$

$$\begin{aligned} \mathrm{d}\mathbf{x}_t &= -a\mathbf{x}_t \mathrm{d}t + \sqrt{2\gamma}\,\mathrm{d}\mathbf{B}_t\,, \\ \partial_t p(\mathbf{x}, t) &= \nabla_{\mathbf{x}} \cdot (a\mathbf{x}p(\mathbf{x}, t)) + \gamma \Delta_{\mathbf{x}} p(\mathbf{x}, t)\,. \end{aligned} \tag{B.4}$$

Using Itô's calculus, we have

$$\mathrm{d}\|\mathbf{x}_t\|^2 = 2(\gamma - a\|\mathbf{x}_t\|^2)\,\mathrm{d}t + 2\sqrt{2\gamma}\mathbf{x}_t\,\mathrm{d}\mathbf{B}_t\,. \tag{B.5}$$

Taking expectations on both sides, we obtain that the second moment of $\mathbf{x}_t$ satisfies the following ODE

$$\frac{\mathrm{d}\mathbb{E}\|\mathbf{x}_t\|^2}{\mathrm{d}t} = 2(\gamma - a\mathbb{E}\|\mathbf{x}_t\|^2)\,. \tag{B.6}$$

The analytic solution of (B.6) is given by $\mathbb{E}\|\mathbf{x}_t\|^2 = \frac{\gamma}{a} + (\mathbb{E}\|\mathbf{x}_0\|^2 - \frac{\gamma}{a})e^{-2at}$, which is used to calculate the accuracy of VCNF.

Acknowledgements

J. Zhao, M. Zhou, X. Zuo, and W. Li are supported by AFOSR YIP award No. FA9550-23-1-0087. W. Li is also supported by NSF DMS-2245097, and NSF RTG: 2038080.

Notes

AMS subject classifications: 65K10, 68T07, 49M41

References

1. ^*Achiam J, Adler S, Agarwal S, Ahmad L, Akkaya I, Aleman FL, Almeida D, Altenschmidt J, Altman S, Anadkat S, et al. (2023). "Gpt-4 technical report". arXiv preprint arXiv:2303.08774.*
2. ^*Ray PP (2023). "ChatGPT: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope". Internet of Things and Cyber-Physical Systems. **3**: 121–154.*
3. ^*Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2022). "High-resolution image synthesis with latent diffusion models". Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10684–10695.*

4. ^ Song Y, Sohl-Dickstein J, Kingma DP, Kumar A, Ermon S, Poole B (2020). "Score-based generative modeling through stochastic differential equations". arXiv preprint arXiv:2011.13456. Available from: https://arxiv.org/abs/2011.13456.
5. ^ Ho J, Salimans T, Gritsenko A, Chan W, Norouzi M, Fleet DJ (2022). "Video diffusion models". Advances in Neural Information Processing Systems. **35**: 8633–8646.
6. ^ Jumper J, Evans R, Pritzel A, Green T, Figurnov M, Ronneberger O, Tunyasuvunakool K, Bates R, Žídek A, Potapenko A, et al. Highly accurate protein structure prediction with AlphaFold. Nature. **596** (7873): 583–589. 2021.
7. ^ Abramson J, Adler J, Dunger J, Evans R, Green T, Pritzel A, Ronneberger O, Willmore L, Ballard AJ, Bambrick J, et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. Nature. 2024:1-3.
8. a, b, c, d, e, f Durkan C, Bekasov A, Murray I, Papamakarios G. "Neural spline flows". In: Proceedings of the 33rd International Conference on Neural Information Processing Systems. Red Hook, NY, USA: Curran Associates Inc.; 2019. p. 675.
9. a, b Chen RTQ, Rubanova Y, Bettencourt J, Duvenaud DK (2018). "Neural ordinary differential equations". Advances in Neural Information Processing Systems. **31**.
10. ^ Fornasier M, Solombrino F (2014). "Mean-field optimal control". ESAIM: Control, Optimisation and Calculus of Variations. **20** (4): 1123–1152.
11. ^ E W, Han J, Li Q (2018). "A mean-field optimal control formulation of deep learning". Research in the Mathematical Sciences. **6**. S2CID 49563185.
12. ^ Liu Z, Wu B, Lin H. "A mean field game approach to swarming robots control". In: 2018 Annual American Control Conference (ACC). IEEE; 2018. p. 4293–4298.
13. ^ Lee W, Liu S, Tembine H, Li W, Osher S (2021). "Controlling propagation of epidemics via mean-field control". SIAM Journal on Applied Mathematics. **81** (1): 190–207.
14. ^ Lee W, Liu S, Li W, Osher S (2022). "Mean field control problems for vaccine distribution". Research in the Mathematical Sciences. **9** (3): 51.
15. ^ Achdou Y, Buera FJ, Lasry J-M, Lions P-L, Moll B (2014). "Partial differential equation models in macroeconomics". Philos Trans R Soc Lond Ser A Math Phys Eng Sci. **372** (2028): 20130397, 19. doi:10.1098/rsta.2013.0397. MR 3268061.
16. ^ Cardaliaguet P, Lehalle C-A (2018). "Mean field game of controls and an application to trade crowding". Math. Financ. Econ.. **12** (3): 335–363. doi:10.1007/s11579-017-0206-z. MR 3805247.

17. ^ a b c d Benamou J-D, Brenier Y (2000). "A computational fluid mechanics solution to the Monge--Kantorovich mass transfer problem". Numerische Mathematik. **84** (3): 375–393.
18. ^ Benamou J-D, Carlier G, Santambrogio F (2017). "Variational mean field games". Active Particles, Volume 1: Advances in Theory, Models, and Applications. Springer. pp. 141–171.
19. ^ Yu J, Lai R, Li W, Osher S (2024). "A fast proximal gradient method and convergence analysis for dynamic mean field planning". Mathematics of Computation. **93** (346): 603–642.
20. ^ a b Ruthotto L, Osher SJ, Li W, Nurbekyan L, Fung SW (2020). "A machine learning framework for solving high-dimensional mean field game and mean field control problems". Proceedings of the National Academy of Sciences. **117** (17): 9183-9193. doi:10.1073/pnas.1922204117.
21. ^ Froese BD (2012). "A numerical method for the elliptic Monge–Ampere equation with transport boundary conditions". SIAM Journal on Scientific Computing. **34** (3): A1432–A1459.
22. ^ Winkler C, Worrall D, Hoogeboom E, Welling M (2019). "Learning likelihoods with conditional normalizing flows". arXiv preprint arXiv:1912.00042. Available from: https://arxiv.org/abs/1912.00042.
23. ^ Raissi M, Perdikaris P, Karniadakis GE (2019). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". Journal of Computational Physics. **378**: 686–707.
24. ^ Lasry J-M, Lions P-L (2007). "Mean field games". Japanese journal of mathematics. **2** (1): 229–260.
25. ^ Cardaliaguet P (2010). "Notes on mean field games". Technical report.
26. ^ a b Huang H, Yu J, Chen J, Lai R (2023). "Bridging mean-field games and normalizing flows with trajectory regularization". Journal of Computational Physics. **487**: 112155. doi:10.1016/j.jcp.2023.112155. Available from: https://www.sciencedirect.com/science/article/pii/S0021999123002504.
27. ^ Zhang BJ, Katsoulakis MA (2023). "A mean-field games laboratory for generative modeling". arXiv preprint arXiv:2304.13534. Available from: https://arxiv.org/abs/2304.13534.
28. ^ Kobyzev I, Prince SJ, Brubaker MA (2020). "Normalizing flows: An introduction and review of current methods". IEEE Transactions on Pattern Analysis and Machine Intelligence. **43** (11): 3964–3979.
29. ^ Briceno-Arias LM, Kalise D, Silva FJ (2018). "Proximal methods for stationary mean field games with local couplings". SIAM Journal on Control and Optimization. **56** (2): 801–836.
30. ^ Liu S, Jacobs M, Li W, Nurbekyan L, Osher SJ (2021). "Computational Methods for First-Order Nonlocal Mean Field Games with Applications". SIAM Journal on Numerical Analysis. **59** (5): 2639–2668. doi:10.1137/20M1334668.

31. ^Zhou M, Osher S, Li W (2024). "A deep learning algorithm for computing mean field control problems via forward-backward score dynamics". arXiv preprint arXiv:2401.09547. Available from: https://arxiv.org/abs/2401.09547.

32. ^Darbon J, Langlois GP, Meng T (2020). "Overcoming the curse of dimensionality for some Hamilton–Jacobi partial differential equations via neural network architectures". Research in the Mathematical Sciences. **7** (3): 20.

33. ^Zhou M, Han J, Lu J (2021). "Actor-critic method for high dimensional static Hamilton--Jacobi--Bellman partial differential equations based on neural networks". SIAM Journal on Scientific Computing. **43** (6): A4043–A4066.

34. ^Lin AT, Fung SW, Li W, Nurbekyan L, Osher SJ (2021). "Alternating the population and control neural networks to solve high-dimensional stochastic mean-field games". Proceedings of the National Academy of Sciences. **118** (31): e2024713118. doi:10.1073/pnas.2024713118. Link to article.

35. ^Zhou M, Lu J (2024). "Solving time-continuous stochastic optimal control problems: Algorithm design and convergence analysis of actor-critic flow". arXiv preprint arXiv:2402.17208. Available from: https://arxiv.org/abs/2402.17208.

36. ^Bensoussan A, Frehse J, Yam P, et al. Mean field games and mean field type control theory. **101**. Springer; 2013.

37. ^Nelson E (1966). "Derivation of the Schrf6dinger equation from Newtonian mechanics". Physical Review. **150** (4): 1079.

38. ^Li W, Liu S, Osher S (2023). "A kernel formula for regularized Wasserstein proximal operators". Research in the Mathematical Sciences. **10** (4): 43.

39. ^Tang K, Wan X, Liao Q (2022). "Adaptive deep density approximation for Fokker–Planck equations". Journal of Computational Physics. **457**: 111080.

40. a, bZeng L, Wan X, Zhou T (2023). "Adaptive Deep Density Approximation for Fractional Fokker–Planck Equations". Journal of Scientific Computing. **97** (3): 68.

41. ^Feng X, Zeng L, Zhou T (2021). "Solving time dependent Fokker--Planck equations via temporal normalizing flow". arXiv preprint arXiv:2112.14012. Available from: https://arxiv.org/abs/2112.14012.

42. ^Dinh L, Sohl-Dickstein J, Bengio S (2016). "Density estimation using real nvp". arXiv preprint arXiv:1605.08803. Available from: https://arxiv.org/abs/1605.08803.

43. a, bKingma DP, Ba J (2015). "Adam: A Method for Stochastic Optimization". In: Bengio Y, LeCun Y, editors. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Confe

rence Track Proceedings. Available from: http://arxiv.org/abs/1412.6980.

44. ^*Zhao J. Code repository for variational conditional normalizing flows for computing second-order mean fi eld control problems. GitHub repository. 2025. Available from: https://github.com/jiaxi98/cnf_ot.*

45. ^*Grathwohl W, Choi D, Wu Y, Roeder G, Duvenaud DK (2017). "Backpropagation through the Void: Optimizin g control variates for black-box gradient estimation". ArXiv. **abs/1711.00123**. S2CID 3535369.*

46. ^*Zhou M, Osher S, Li W (2024). "Score-based neural ordinary differential equations for computing mean fiel d control problems". arXiv preprint arXiv:2409.16471. Available from: https://arxiv.org/abs/2409.16471.*

47. ^*Zuo X, Zhao J, Liu S, Osher S, Li W (2024). "Numerical Analysis on Neural Network Projected Schemes for A pproximating One Dimensional Wasserstein Gradient Flows". arXiv preprint arXiv:2402.16821. Available fro m: https://arxiv.org/abs/2402.16821.*

48. ^*Lipman Y, Chen RT, Ben-Hamu H, Nickel M, Le M (2022). "Flow matching for generative modeling". arXiv preprint arXiv:2210.02747. Available from: https://arxiv.org/abs/2210.02747.*

49. ^*Onsager L. "Reciprocal relations in irreversible processes. I." Physical review. **1931**;37(4):405.*

50. a, b*Gao Y, Liu JG (2021). "Random walk approximation for irreversible drift-diffusion process on manifold: e rgodicity, unconditional stability and convergence". arXiv preprint arXiv:2106.01344. Available from: https:// arxiv.org/abs/2106.01344.*

51. ^*Sohl-Dickstein J, Weiss E, Maheswaranathan N, Ganguli S. "Deep unsupervised learning using nonequilibri um thermodynamics". In: International conference on machine learning. PMLR; 2015. p. 2256-2265.*

Declarations

Funding: J. Zhao, M. Zhou, X. Zuo, and W. Li are supported by AFOSR YIP award No. FA9550-23-1-0087. W. Li is also supported by NSF DMS-2245097, and NSF RTG: 2038080.

Potential competing interests: No potential competing interests to declare.