UNIVERSITY OF CALIFORNIA
SANTA CRUZ

**MEASURE-VALUED PROXIMAL RECURSIONS
FOR LEARNING AND CONTROL**

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL AND COMPUTER ENGINEERING

by

Iman Nodozi

June 2024

The dissertation of Iman Nodozi
is approved:

_____

Professor Dejan Milutinovic, Chair

_____

Associate Professor Abhishek Halder

_____

Assistant Professor Yu Zhang

_____

Peter Biehl
Vice Provost and Dean of Graduate Studies

Contents

Part II: Stochastic Modeling

Part III: Stochastic Learning

List of Figures

x

List of Tables

List of Notations

In this Dissertation, boldfaced capital letters are used to denote matrices (e.g., $\boldsymbol{A}$), while boldfaced lowercase letters represent vectors (e.g., $\boldsymbol{X}$). The following table enumerates all the symbols and their corresponding definitions as employed throughout this Dissertation.

Notation	Definition
$\mathbb{R}$	Set of Real numbers
$\mathbb{R}^n$	Euclidean space
$\mathcal{B}(\mathcal{X})$	Borel σ-field over $\mathcal{X} \subseteq \mathbb{R}^n$
$\langle \cdot, \cdot \rangle$	Euclidean inner product
$\langle \boldsymbol{A}, \boldsymbol{B} \rangle$	Frobenius inner product between matrices
$\nabla_{\boldsymbol{x}}$	Gradient operator
$\nabla_{\boldsymbol{x}} \cdot$	Divergence operator
$\Delta_{\boldsymbol{x}}$	Laplacian operator
$\mathbf{Hess}(.)$	Hessian operator
$\log(\cdot)$	Element-wise log
$\exp(\cdot)$	Element-wise exponential
$\mathrm{diag}(\cdot)$	Diagonal matrix of appropriate dimensions
$\boldsymbol{I}_n$	$n \times n$ identity matrix
$\mathbf{1}$	Column vectors containing all ones
$\mathbf{0}$	Column vectors containing all zeros
$\mathbb{P}$	Probability measure
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian PDF with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
$\mathbb{E}_{\rho}[\cdot]$	Expectation operator w.r.t. the PDF ρ

Notation	Definition
$\mathcal{P}\left(\mathbb{R}^n\right)$	The space of all probability density functions supported over the subsets of $\mathbb{R}^n$
$\mathcal{P}_2\left(\mathbb{R}^n\right)$	The collection of probability density functions with finite second moments
$\otimes$	Kronecker product
$\sim$	Follows the probability distribution
$\mu^{\boldsymbol{u}}$	The joint measure depends on the choice of control $\boldsymbol{u}$
$\operatorname{ess\,sup}$	Essential supremum
$\|\cdot\|_2$	2 norms
$\|\cdot\|_\infty$	∞ norms
$[\![n]\!]$	The set of natural numbers from 1 to n, $\{1, 2, \cdots, n\}$

Abstract

Measure-valued Proximal Recursions for Learning and Control

Iman Nodozi

In this dissertation, we investigate convex optimization problems over the space of probability measures, highlighting applications in stochastic control, stochastic modeling, and stochastic learning. The theory and algorithms we develop can be applied to problems such as sampling from unnormalized priors, policy iteration in reinforcement learning, simulating mean-field dynamics, Wasserstein GANs, optimal distribution steering a.k.a. Schrödinger bridge, and its zero-noise limit: optimal mass transport. We propose proximal recursions for solving these measure-valued optimization problems, offering novel algorithms that extend the concept of gradient steps to the space of probability measures.

We propose new algorithms for solving generalized Schrödinger bridge problems where the drift and/or diffusion coefficient could be nonlinear in state as well as affine or non-affine in control. We illustrate our results on both model-based and model-free numerical case studies.

Furthermore, we demonstrate that our measure-valued proximal recursions are also useful in stochastic modeling, specifically offering insights for a controlled mean field model. We illustrate these ideas by deriving a controlled mean field dynamics model for chiplet dynamics in micro assembly applications. This model extends finite population dynamics to a continuum, formulating a nonlocal, nonlinear PDE that encapsulates stochastic forces and nonlinear interactions between chiplets and elec-

trodes. The deduced mean field evolution, was found to be a Wasserstein gradient flow of a Lyapunov-like energy functional.

We then turn to applying our measure-valued proximal recursions in stochastic learning. Here we propose two algorithms: one centralized and another distributed. The proposed centralized algorithm solves mean field learning dynamics in a neural network in over-parameterized limit. The proposed distributed algorithm generalizes the well-known Euclidean alternating direction method of multipliers (ADMM) to the space of probability measures. Numerical examples are given to illustrate the performance of the proposed algorithms w.r.t. the state-of-the-art.

Acknowledgements

1 | Introduction

1.1 Overview

In this Dissertation, we develop theory and algorithms for solving convex optimization problems over the space of probability measures. These problems take the form

$$\underset{\mu \in \mathcal{P}_2(\mathbb{R}^d)}{\arg\inf} F(\mu) \tag{1.1}$$

where the functional $F(\cdot)$ is convex, and $\mathcal{P}_2(\mathbb{R}^d)$ denotes the set of all probability measures supported on $\mathbb{R}^d$ with finite second moments. The perspective that motivates this research is that problems of the form (1.1) appear in different guises across stochastic learning and control. We give some examples below.

Sampling. Sampling from a given unnormalized prior is a central problem in statistics and machine learning. A popular approach is to perform the so-called Langevin Monte Carlo and its variants [6–10], where the idea is to evolve a point cloud subject to a dynamics given by suitably designed Itô stochastic differential equation (SDE) such that the stationary measure associated with this dynamics coincides with the desired prior. The generator corresponding to the measure-valued dynamics turns out to be a gradient flow of certain energy-like convex functional $F(\cdot)$ w.r.t. suitable distance on the manifold $\mathcal{P}_2(\mathbb{R}^d)$. In other words, sampling via such Langevin-type algorithms can be equivalently seen as solving a variational problem of the form (1.1).

Policy iteration in reinforcement learning. The task of computing optimal policy in reinforcement learning can be transcribed into a problem of the form (1.1) by lifting the parameterized policy space to the space of probability measures supported on the

policy space. This viewpoint has been put forth by several recent works [11–14]. The key idea is that even though optimizing a parameterized policy over a large but finite dimensional parameter space is a nonconvex problem, optimizing a measure supported on the said joint parameter space is a convex problem in the measure-valued decision variable. Thus, it is of interest to design gradient descent algorithms to directly solve (1.1) and to use the resulting measure to realize the policy.

Mean field neural network learning. Starting in 2018, several works [15–18] pointed out that the learning dynamics for two layer neural networks under stochastic gradient descent in the infinite width (i.e., over-parametrization) limit leads to a PDE that is gradient flow of a functional $F(\mu)$ w.r.t. a metric on $\mathcal{P}_2(\mathbb{R}^d)$, i.e., it precisely solves the infinite dimensional convex problem (1.1). Recent works have extended this viewpoint beyond regularized empirical risk minimization, e.g., in unsupervised learning settings such as GAN [19], for second order learning dynamics [20], and also for different network architecture [21].

Stochastic prediction. The stochastic prediction problem in continuous time asks to evolve the transient joint probability measures or probability density functions (PDFs) over some state space subject to given sample path $\boldsymbol{x}(t)$ dynamics specified by an Itô SDE modeling the underlying physics, and joint stochastic uncertainties over the space of initial conditions and parameters specified by respective joint PDFs. Mathematically, this amounts to solving a linear or nonlinear Fokker-Planck-Kolmogorov's (FPK) forward partial differential equation (PDE) initial value problem (IVP) of the form

$$\frac{\partial \rho}{\partial t} = \mathcal{L}\rho, \quad \rho(\boldsymbol{x}, t = 0) = \rho_0(\boldsymbol{x}) \text{ (given)}, \tag{1.2}$$

where $\rho(\boldsymbol{x}, t)$ denotes the transient joint state PDF of the stochastic state $\boldsymbol{x}(t)$ at time t, and $\mathcal{L}$ denotes the second order FPK operator. That the flow generated by this deterministic PDE could be equivalently cast in a form (1.1) has been known for some time [22, 23]. However, designing proximal algorithms to solve (1.1) for the purpose of stochastic prediction, is a relatively recent endeavor [5, 24].

Stochastic estimation. Unlike the stochastic prediction problem that amounts to computing the prior subject to a given stochastic dynamics, the stochastic estimation problem amounts to computing the posterior or the conditional joint state PDF given the history of noisy measurements up until time t. In other words, in a continuous time stochastic estimation setting, in addition to a prior stochastic process model, one also has a stochastic measurement model as well as time-varying noisy sensor data. The conditional joint state PDF solves a stochastic PDE known as the Kushner-Stratonovich PDE [25–27]. A common computational approach for solving the stochastic estimation problem is the so-called particle filtering algorithms where one first estimates the prior joint PDF via Monte Carlo methods, and then maps the prior to the posterior via Bayesian update. Recent works [28–30] have shown that each of these two steps may be realized by solving a problem of the form (1.1), i.e., the transient conditional can be realized by composing the minimizers of two sequential measure-valued convex optimization problems.

Stochastic control for finite horizon distribution steering. The need to solve equations of the form (1.2) also appears in designing feedback controllers that steers the controlled state of a stochastic dynamical system from a prescribed initial joint PDF to another over a finite horizon while minimizing the total control effort in doing so. This is an atypical stochastic control problem with growing interests in systems-

control literature [31, 32]. Optimal control problems for probability and/or population density functions have also been investigated before in robotics literature [33–37]. Solving this class of problems requires solving for the so-called Schrödinger factors, which solve a system of uncontrolled forward-backward Kolmogorov PDEs. Having an algorithmic handle to solve (1.1) then enables computing the optimal feedback controls; see e.g., [38, 39].

Motivated by these examples, this Dissertation endeavors to advance the field by designing different algorithms tailored for efficiently solving problems encapsulated by (1.1). The remainder of this Dissertation is structured as follows.

1.2   Organization

This Dissertation is divided into three main parts. Each part is designed to not only stand on its own by presenting comprehensive results in its specific area of focus but also to contribute to the Dissertation's overarching themes.

1. **Part I** focuses on stochastic control, specifically on generalized Schrödinger bridge problems. It begins with a discussion on optimal mass transport and the Schrödinger bridge, offering essential background in **Section 2.1** for understanding optimal mass transport and its generalized version. **Section 2.2** then introduces the Schrödinger bridge and its generalized version, laying the foundation for the subsequent analysis.

    In this part, we explore control-affine and control non-affine Schrödinger bridge problems across chapters, **Chapter 3** and **Chapter 4**.

    Chapter 3 focuses on control-affine generalized Schrödinger bridge problems.

In this chapter, we formulate and solve the problem of finite horizon minimum control effort steering of the state probability distribution between prescribed endpoint joints applied to a finite population of networked noisy nonuniform Kuramoto oscillators. Within this chapter:

- **Section 3.1** outlines the problem formulation specifically for control-affine Schrödinger problems.

- **Section 3.2** discusses the existence and uniqueness of solutions to these problems, providing a theoretical foundation for the proposed methods.

- **Section 3.3** dives into the recovery of optimal solutions through Schrödinger factors. These factors address a nonlinearly boundary-coupled system of linear partial differential equations (PDEs).

- **Section 3.4** presents a case study: optimal steering of distributions for the Kuramoto oscillators. We extend our analysis to both first and second-order Kuramoto oscillators.

Chapter 4 considers control non-affine generalized Schrödinger bridge problems. Across this chapter, we distinguish between two primary methodologies: model-based control non-affine GSBP and model-free control non-affine GSBP. On the application side, we show that the minimum effort control of colloidal self-assembly (where the controlled drift and diffusion coefficients for colloidal self-assembly are typically non-affine in control) can be naturally formulated in the order-parameter space as a generalized Schrödinger bridge problem. Within this chapter:

- **Section 4.1** outlines the problem formulation specifically for control non-

affine Schrödinger problems.

- In **Section 4.2** we consider a specific model for the sample path dynamics of colloidal SA.

- In **Section 4.2.2**, we detail our proposed stochastic optimal control problem formulation for the model-based non-affine control system for colloidal SA.

- In **Section 4.2.3**, we derive the first order conditions for optimality in the form of a system of coupled PDEs.

- We then learn the solutions for this system of equations by training a physics-informed neural network (PINN) [40, 41], as detailed in **Section 4.2.4**.

- In **Section 4.3**, we propose a data-driven learning and control framework, named 'neural Schrödinger bridge', to solve such generalized Schrödinger bridge problems by innovating on recent advances in neural networks.

- In **Section 4.3.1**, we define the neural Schrödinger bridge problem, utilizing neural network representations for the drift and diffusion coefficients. This approach leverages learning from molecular dynamics (MD) simulation data to inform the controlled neural stochastic differential equations (SDEs).

- In **Section 4.3.2**, we gave an overview of the proposed learning and control framework for solving the neural Schrödinger bridge problem.

- **Section 4.3.3** discusses solving the GSBP optimality conditions using PINN with Sinkhorn losses.

6

- A detailed numerical case study is presented in **Section 4.3.4**, focusing on a colloidal self-assembly (SA) system within an isothermal-isobaric (NPT) ensemble.

2. **Part II** focuses on stochastic modeling, specifically on controlled mean field models. We ground our development by deducing a controlled mean field model for chiplet population dynamics in **Chapter 5**. This models the dynamics of a continuum of chiplet population in the form of a nonlocal, nonlinear partial differential equation. Within this chapter:

- We derive a controlled mean field dynamics (**Section 5.1**) for the macroscopic motion of the chiplet population. The derived model is non-affine in control, and rather non-standard compared to the existing nonlocal dynamics models available in the literature.

- We establish that the derived mean field dynamics model can be understood as the Wasserstein gradient flow (**Section 5.2**) of a free energy functional over the manifold of chiplet population density functions.

3. **Part III** considers measure-valued proximal recursions for stochastic learning, and spans **Chapters 6** and **7**. In **Chapters 6**, we design a proximal algorithm for mean field learning. The mean field limit here applies to shallow overparameterized neural networks, i.e., networks with single hidden layer having infinite width. Within this chapter:

- In **Section 6.1**, we provide the necessary background for the empirical risk minimization and for the corresponding mean field limit.

- The proposed proximal algorithm (including its derivation, convergence guarantee, and implementation) is detailed in **Section 6.2**.

- We then report numerical case studies in **Section 6.3** and **Section 6.4** for binary and multi-class classifications, respectively.

- Numerical results for a synthetic one-dimensional case study of learning a sinusoid using the proposed proximal algorithm is provided in **Section 6.5**.

Finally, in **Chapter 7**, we propose a distributed algorithm to solve measure-valued optimization problems with convex additive objectives. Within this chapter:

- Our main idea is presented in **Section 7.2**.

- In **Section 7.3**, we propose our two-layer ADMM algorithm.

- In **Section 7.4**, we detail the proposed computational framework.

- In **Section 7.5**, we present sufficient conditions that guarantee the convergence of the inner layer ADMM.

- In **Section 7.6**, we provide numerical examples to demonstrate the proposed distributed computation framework.

The summary of our research and the future work directions are provided in **Chapter 8**. All proofs are deferred to the Appendix.

1.3 Preliminaries

In this section, we introduce and discuss in depth several foundational concepts and mathematical tools that are essential for understanding the work presented in this Dissertation.

1.3.1 Wasserstein space

For $\mu \in \mathcal{P}_2(\mathcal{X})$, and for any measurable map T defined on $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$, let $T_{\#}\mu$ denote the pushforward a.k.a. transport of the probability measure μ via T.

For $\mathcal{X}, \mathcal{Y} \subseteq \mathbb{R}^d$, the *squared 2-Wasserstein distance* between a pair of probability measures $\mu_x \in \mathcal{P}_2(\mathcal{X}), \mu_y \in \mathcal{P}_2(\mathcal{Y})$, is defined as

$$W^2(\mu_x, \mu_y) := \inf_{\pi \in \Pi(\mu_x, \mu_y)} \int_{\mathcal{X} \times \mathcal{Y}} c(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\pi(\boldsymbol{x}, \boldsymbol{y}), \tag{1.3}$$

where $\Pi(\mu_x, \mu_y)$ is the set of joint probability measures or couplings over the product space $\mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^{2d}$, having $\boldsymbol{x}$ marginal μ_x, and $\boldsymbol{y}$ marginal μ_y. Throughout, we use the ground cost $c(\boldsymbol{x}, \boldsymbol{y}) := \|\boldsymbol{x} - \boldsymbol{y}\|_2^2$ (the squared Euclidean distance) for $\boldsymbol{x} \in \mathcal{X}, \boldsymbol{y} \in \mathcal{Y}$. To lighten nomenclature, we henceforth refer to (1.3) as the "squared Wasserstein distance" dropping the prefix 2.

It is well-known [42, Ch. 7] that the Wasserstein distance W defines a metric on $\mathcal{P}_2(\mathcal{X})$. The minimizer of the linear program (1.3), denoted as π^{opt}, is referred to as the *optimal transportation plan*. If $\mu \in \mathcal{P}_{2,\mathrm{ac}}(\mathcal{X})$, then π^{opt} is supported on the graph of the *optimal transport map* T^{opt} pushing μ_x to μ_y. We can rewrite (1.3) as

$$W^2(\mu_x, \mu_y) = \inf_{\text{Measurable } T : T_{\#}\mu_x = \mu_y} \int_{\mathcal{X}} c(\boldsymbol{x}, T(\boldsymbol{x})) \, \mathrm{d}\mu_x, \tag{1.4}$$

and for the ground cost $c(\boldsymbol{x}, \boldsymbol{y}) := \|\boldsymbol{x} - \boldsymbol{y}\|_2^2$, the $\arg\inf$ for (1.4) is precisely T^{opt} that is unique a.e. [43]. We refer to $(\mathcal{P}_2(\mathcal{X}), W)$ as the *Wasserstein space* since it allows

to define a Riemannian-like geometry. In particular, letting $L^2(\mu)$ denote the space of functions from $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ to $(\mathcal{Y}, \mathcal{B}(\mathcal{Y}))$, which are square integrable w.r.t. $\mu \in \mathcal{P}_2(\mathcal{X})$, we define the tangent space of $(\mathcal{P}_2(\mathcal{X}), W)$ at $\mu \in \mathcal{P}_2(\mathcal{X})$ as

$$\mathcal{T}_\mu \mathcal{P}_2(\mathcal{X}) := \overline{\{\nabla \phi \mid \phi \in C_c^\infty(\mathcal{X})\}},$$

where the overline denotes closure w.r.t. $L^2(\mu)$; see e.g., [44, Ch. 13].

A proper lsc functional $\Phi : \mathcal{P}_2(\mathcal{X}) \mapsto (-\infty, +\infty]$ is said to be *convex along generalized geodesics defined by the 2-Wasserstein distance* [45, Ch. 9], if for any $t \in [0, 1]$ and any $\mu_1, \mu_2 \in \mathcal{P}_2(\mathcal{X})$, $\mu_3 \in \mathcal{P}_{2,\mathrm{ac}}(\mathcal{X})$, we have

$$\Phi\left(\left(t T_{3\to1}^{\mathrm{opt}} + (1-t) T_{3\to2}^{\mathrm{opt}}\right)_\# \mu_3\right) \le t\Phi(\mu_1) + (1-t)\Phi(\mu_2),$$

where $T_{3\to1}^{\mathrm{opt}}$ and $T_{3\to2}^{\mathrm{opt}}$ are the optimal transport maps pushing μ_3 forward to μ_1, and μ_3 forward to μ_2, respectively. The measure-valued curve $t \mapsto \left(t T_{3\to1}^{\mathrm{opt}} + (1-t) T_{3\to2}^{\mathrm{opt}}\right)_\# \mu_3$ interpolates between $\mu_2(t = 0)$ and $\mu_1(t = 1)$.

Given proper lsc $\Phi : \mathcal{P}_2(\mathcal{X}) \mapsto (-\infty, +\infty]$, its strong Fréchet subdifferential $\mu \mapsto \partial\Phi(\mu)$ allows defining the *Wasserstein gradient flow* (WGF) of the functional Φ, see e.g., [45, Ch. 11], [44, Ch. 23], [46]. Additionally, when Φ is convex along generalized geodesics mentioned before, then the WGF can be characterized as the continuity equation

$$\frac{\partial \mu}{\partial t} + \nabla \cdot (\mu \boldsymbol{v}(\mu)) = 0, \ \boldsymbol{v}(\mu) \in \partial\Phi(\mu) \cap \mathcal{T}_\mu \mathcal{P}_2(\mathcal{X}) \ \Leftrightarrow \ \boldsymbol{v}(\mu) = \nabla \frac{\delta \Phi}{\delta \mu}, \qquad (1.5)$$

where ∇ is the d dimensional Euclidean gradient operator, and $\frac{\delta}{\delta\mu}$ denotes the functional derivative w.r.t. μ. More generally, for non-smooth Φ, one can define WGF via Evolution Variational Inequality (EVI) [45, Thm. 11.1.4], [47].

Following (1.5), we can formally define the Wasserstein gradient [42, Ch. 9.1], [45,

Ch. 8] as

$$\nabla^W \Phi(\mu) := -\nabla \cdot \left(\mu \nabla \frac{\delta \Phi}{\delta \mu} \right), \tag{1.6}$$

and express the WGF in the form

$$\frac{\partial \mu}{\partial t} = -\nabla^W \Phi(\mu). \tag{1.7}$$

In this work, we consider smooth Φ with singleton $\partial \Phi(\mu) = \{\nabla^W \Phi(\mu)\}$ [45, Ch. 10.4].

1.3.2   Sinkhorn regularization

For $\pi \in \Pi(\mu_x, \mu_y)$ and a reference probability measure π_0 supported over $\mathcal{X} \times \mathcal{Y}$, the notation $\pi \ll \pi_0$ means that π is absolutely continuous w.r.t. π_0. Given a strictly convex regularizer $R(\cdot)$, define the *regularized squared Wasserstein distance*

$$W_\varepsilon^2(\mu_x, \mu_y) := \inf_{\substack{\pi \in \Pi(\mu_x, \mu_y) \\ \pi \ll \pi_0}} \int_{\mathcal{X} \times \mathcal{Y}} c(\boldsymbol{x}, \boldsymbol{y}) \, \mathrm{d}\pi(\boldsymbol{x}, \boldsymbol{y}) + \varepsilon \int_{\mathcal{X} \times \mathcal{Y}} R\left(\frac{\mathrm{d}\pi}{\mathrm{d}\pi_0} \right) \mathrm{d}\pi_0(\boldsymbol{x}, \boldsymbol{y}) \tag{1.8}$$

where $\varepsilon > 0$ is a regularization parameter, and $\dfrac{\mathrm{d}\pi}{\mathrm{d}\pi_0}$ denotes the Radon-Nikodym derivative. Examples of π_0 include the product measure $\mu_x(\boldsymbol{x})\mu_y(\boldsymbol{y})$ [48] and the uniform measure [49]. In this work, we consider the entropic regularizer

$$R(x) := x \log x - x \quad \text{for } x \geq 0, \quad \text{with the convention } 0 \log 0 = 0. \tag{1.9}$$

It is known [50] that $W_\varepsilon^2 \to W^2$ in the limit $\varepsilon \downarrow 0$. Even though the Sinkhorn loss (1.8) does not define a metric over $\mathcal{M}$, its computation offers several advantages over that of (1.3). For instance, the entropic regularization makes the objective in (1.8) strictly convex, and its discrete implementation was proposed [49] as a fast numerical approximant of the OMT (1.3). The work in [49] considered the discrete version of (1.8) with an entropic regularizer R as above, and named it as the *Sinkhorn divergence*. This

entropy or Sinkhorn regularized squared Wasserstein distance has found widespread applications in the computation and analysis of variational problems involving the Wasserstein distance (see e.g., [50–53]), and will be useful in our development too.

1.3.3   Wasserstein barycenter

Given the measures $\mu_1, \ldots, \mu_n \in \mathcal{P}_2(\mathcal{X})$ and positive weights $w_1, \cdots, w_n$, the *Wasserstein barycenter* [54] is given by

$$\underset{\mu \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \sum_{i=1}^{n} w_i W^2(\mu, \mu_i). \tag{1.10}$$

In (1.10), replacing W^2 by W_ε^2 defined in (1.8) with R as in (1.9), results in the *Sinkhorn regularized Wasserstein barycenter*

$$\underset{\mu \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \sum_{i=1}^{n} w_i W_\varepsilon^2(\mu, \mu_i). \tag{1.11}$$

Notice that both (1.10) and (1.11) are in the form (1.1) with additive objectives wherein the summand functionals are convex.

1.3.4   Wasserstein gradient of a functional

The *Wasserstein gradient* of a functional $\Phi : \mathcal{P}_2(\mathbb{R}^d) \mapsto \mathbb{R}$, denoted as $\nabla^W \Phi$, evaluated at $\rho \in \mathcal{P}_2(\mathbb{R}^d)$, is given by [45, Ch. 8]

$$\nabla^W \Phi(\rho) := -\nabla \cdot \left(\rho \nabla \frac{\delta \Phi}{\delta \rho} \right) \tag{1.12}$$

where ∇ denotes the standard Euclidean gradient, and $\frac{\delta}{\delta \rho}$ denotes the functional derivative w.r.t. ρ.

To exemplify the definition (1.12), consider the functional $\Phi(\rho) = \int \rho \log \rho$ (negative entropy) for $\rho \in \mathcal{P}_2(\mathbb{R}^d)$. Then $\frac{\delta \Phi}{\delta \rho} = 1 + \log \rho$, $\nabla(1 + \log \rho) = \nabla \rho / \rho$, and we get

$\nabla^W \Phi(\rho) = -\nabla \cdot \nabla \rho = -\Delta \rho$, where $\Delta := \nabla \cdot \nabla$ denotes the Euclidean Laplacian operator.

1.3.5 Wasserstein proximal operator

We use the notation $\mathrm{prox}_{G(\cdot)}^W(\zeta)$ to denote the *Wasserstein proximal operator* of proper lsc $G : \mathcal{P}_2(\mathcal{X}) \mapsto (-\infty, +\infty]$, acting on $\zeta \in \mathcal{P}_2(\mathcal{X})$, given by

$$\mathrm{prox}_{G(\cdot)}^W(\zeta) := \underset{\mu \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \frac{1}{2} W^2(\mu, \zeta) + G(\mu). \qquad (1.13)$$

The Wasserstein proximal operator (1.13) can be seen as a generalization of the finite dimensional Euclidean proximal operator of proper lsc $g : \mathbb{R}^d \mapsto (-\infty, +\infty]$, given by

$$\mathrm{prox}_g^{\|\cdot\|_2}(\boldsymbol{z}) := \underset{\boldsymbol{x} \in \mathbb{R}^d}{\arg\inf} \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + g(\boldsymbol{x}). \qquad (1.14)$$

Wasserstein proximal operators of the form (1.13) go back to the seminal work of [22], and have been used in stochastic prediction [5], control [38], learning [12, 47, 55, 56], and in modeling of population dynamics [57].

1.3.6 Legendre-Fenchel conjugate

The Legendre-Fenchel conjugate of a real-valued function f is

$$f^*(\boldsymbol{y}) := \sup_{\boldsymbol{x} \in \mathrm{domain}(f)} \left(\langle \boldsymbol{y}, \boldsymbol{x} \rangle - f(\boldsymbol{x}) \right),$$

where $\langle \cdot, \cdot \rangle$ denotes the standard inner product. The function f^* is convex even if f is not. When $f(\boldsymbol{x}) = \langle \boldsymbol{a}, \boldsymbol{x} \rangle$, $\boldsymbol{a} \in \mathbb{R}^d \setminus \{\boldsymbol{0}\}$, then $f^*(\boldsymbol{y})$ is the indicator function of $\{\boldsymbol{a}\}$, i.e.,

$$f^*(\boldsymbol{y}) = \begin{cases} 0 & \text{if } \boldsymbol{y} = \boldsymbol{a}, \\ +\infty & \text{otherwise.} \end{cases} \qquad (1.15)$$

1.3.7 Euclidean ADMM

The constrained optimization problem $\min\limits_{\boldsymbol{x}\in\mathbb{R}^N} f(\boldsymbol{x})$ subject to $\boldsymbol{x}\in\mathcal{C}\subset\mathbb{R}^N$, where the function f and the set $\mathcal{C}$ are convex, can be re-written as $\min\limits_{\boldsymbol{x},\boldsymbol{z}\in\mathbb{R}^N} f(\boldsymbol{x}) + \mathbf{1}_{\mathcal{C}}(\boldsymbol{z})$ subject to $\boldsymbol{x}=\boldsymbol{z}$ where the indicator function $\mathbf{1}_{\mathcal{C}}(\boldsymbol{z}):=0$ if $\boldsymbol{z}\in\mathcal{C}$, and $\mathbf{1}_{\mathcal{C}}(\boldsymbol{z}):=+\infty$ if $\boldsymbol{z}\notin\mathcal{C}$. Denote the dual variable associated with the constraint $\boldsymbol{x}=\boldsymbol{z}$ as $\boldsymbol{\nu}\in\mathbb{R}^N$, and let $\widetilde{\boldsymbol{\nu}}:=\boldsymbol{\nu}/\tau$ be the scaled dual variable for some parameter $\tau>0$. The Euclidean augmented Lagrangian for this problem is $L_\tau(\boldsymbol{x},\boldsymbol{z},\widetilde{\boldsymbol{\nu}}):=f(\boldsymbol{x})+\mathbf{1}_{\mathcal{C}}(\boldsymbol{z})+\frac{\tau}{2}\|\boldsymbol{x}-\boldsymbol{z}+\widetilde{\boldsymbol{\nu}}\|_2^2$. Each iteration of the Euclidean ADMM algorithm in the so-called "scaled form" [58, Ch. 5], comprises of the following three steps:

$$\boldsymbol{x}^{k+1} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^N} f(\boldsymbol{x}) + \frac{\tau}{2}\|\boldsymbol{x}-\boldsymbol{z}^k+\widetilde{\boldsymbol{\nu}}^k\|_2^2 \overset{(1.14)}{=} \mathrm{prox}_{\frac{1}{\tau}f}^{\|\cdot\|_2}\left(\boldsymbol{z}^k-\widetilde{\boldsymbol{\nu}}^k\right), \tag{1.16a}$$

$$\boldsymbol{z}^{k+1} = \mathrm{proj}_{\mathcal{C}}\left(\boldsymbol{x}^{k+1}+\widetilde{\boldsymbol{\nu}}^k\right), \tag{1.16b}$$

$$\widetilde{\boldsymbol{\nu}}^{k+1} = \widetilde{\boldsymbol{\nu}}^k + \left(\boldsymbol{x}^{k+1}-\boldsymbol{z}^{k+1}\right), \tag{1.16c}$$

where the iteration index $k\in\mathbb{N}_0$ (the set of whole numbers $\{0,1,2,\cdots\}$), and $\mathrm{proj}_{\mathcal{C}}$ denotes the Euclidean projection onto $\mathcal{C}$. The steps (1.16a)-(1.16b) involve alternating minimization of the augmented Lagrangian L_τ, and the step (1.16c) involves dual ascent. Notice that in the scaled form Euclidean ADMM, the parameter τ *does not* appear in (1.16c) as the pre-factor of the term in parenthesis. For Euclidean ADMM convergence results, see e.g., [59], [60].

For a separable objective $f(\boldsymbol{x}_1,\cdots,\boldsymbol{x}_n) = \sum\limits_{i=1}^{n} f_i(\boldsymbol{x}_i)$, where $\boldsymbol{x}_i\in\mathbb{R}^N$ and f_i convex for all $i\in[n]$, it is immediate from (1.16) that the updates (1.16a) and (1.16c) can be parallelized across the index $i\in[n]$. The nature of computation in step (1.16b) depends on the constraint set $\mathcal{C}$, see e.g., [61, Ch. 5]. For instance, if $\mathcal{C}$ is the consensus constraint $\boldsymbol{x}_1=\cdots=\boldsymbol{x}_n=\boldsymbol{z}$, then (1.16b) requires an averaging of the local updates,

resulting in a "broadcast and gather" computation. In (7.3.2), we will encounter an instance of (1.16) that will admit parallelization.

1.3.8   Thompson metric

Consider $z, \widetilde{z} \in \mathcal{K}$, where $\mathcal{K}$ is a non-empty open convex cone. Further, suppose that $\mathcal{K}$ is a normal cone, i.e., there exists constant α such that $\|z\| \leq \alpha \|\widetilde{z}\|$ for $z \leq \widetilde{z}$. Thompson [62] proved that $\mathcal{K}$ is a complete metric space w.r.t. the so-called Thompson metric given by

$$d_{\mathrm{T}}(z, \widetilde{z}) := \max\{\log \gamma(z/\widetilde{z}), \log \gamma(\widetilde{z}/z)\},$$

where $\gamma(z/\widetilde{z}) := \inf\{c > 0 \mid z \leq c\widetilde{z}\}$. In particular, if $\mathcal{K} \equiv \mathbb{R}_{>0}^N$ (positive orthant of $\mathbb{R}^N$), then

$$d_{\mathrm{T}}(z, \widetilde{z}) = \log \max \left\{ \max_{i=1,\dots,N} \left(\frac{z_i}{\widetilde{z}_i} \right), \max_{i=1,\dots,N} \left(\frac{\widetilde{z}_i}{z_i} \right) \right\}. \tag{1.17}$$

1.3.9   Some other definitions

Following the definition of Frobenius inner product between matrices, we define the Frobenius inner product between the operator $\mathbf{Hess}$ and a matrix field $Q(x)$ where $x \in \mathcal{X} \subseteq \mathbb{R}^n$, as

$$\langle \mathbf{Hess}, Q(x) \rangle := \sum_{i,j} \frac{\partial^2}{\partial x_i \partial x_j} Q_{ij}(x). \tag{1.18}$$

Given probability measures μ_0, μ_1 on $\mathbb{R}^d$, the *total variation distance*

$$\mathrm{dist}_{\mathrm{TV}}(\mu_0, \mu_1) := \frac{1}{2} \sup_f \left| \int f \, \mathrm{d}(\mu_0 - \mu_1) \right|$$

where the supremum is over all measurable $f : \mathbb{R}^d \to \mathbb{R}$, $\|f\|_\infty \leq 1$. For $f : \mathbb{R}^d \to \mathbb{R}$, we define its *Lipschitz constant* $\|f\|_{\mathrm{Lip}} := \sup_{x \neq y} \frac{|f(x) - f(y)|}{\|x - y\|_2}$, and its *bounded Lipschitz*

constant $\|f\|_{\mathrm{BL}} := \max\{\|f\|_{\infty}, \|f\|_{\mathrm{Lip}}\}$. The *bounded Lipschitz distance* [63, Ch. 11.3] between probability measures μ_0, μ_1 is $\mathrm{dist}_{\mathrm{BL}}(\mu_0, \mu_1) := \sup_{\|f\|_{\mathrm{BL}} \leq 1} |\int f \, \mathrm{d}(\mu_0 - \mu_1)|$. Notice that $\mathrm{dist}_{\mathrm{BL}}(\mu_0, \mu_1) \leq 2 \, \mathrm{dist}_{\mathrm{TV}}(\mu_0, \mu_1)$.

For $\mathcal{X} \subseteq \mathbb{R}^d$, we use $C_b(\mathcal{X})$ to denote the space of all bounded continuous functions $\varphi : \mathcal{X} \mapsto \mathbb{R}$, and $C_b^k(\mathcal{X})$ comprises those which are also k times continuously differentiable (in the sense of mixed partial derivatives of order k). We say that a function sequence $\{g_n\}_{n \in \mathbb{N}}$ where $g_n \in L^1(\mathcal{X})$, converges weakly to a function $g \in L^1(\mathcal{X})$, if $\lim_{n \to \infty} \int_{\mathcal{X}} (g_n - g) \, \psi = 0$ for all $\psi \in C_b(\mathcal{X})$. We symbolically denote the weak convergence as $g_n \rightharpoonup g$.

1.4 Publications

Papers resulting from the research reported in this dissertation include [64–72]. Below is a list of them, categorized into journal papers, conference proceedings, and preprints.

Journal Papers

1. Alexis Teter, **Iman Nodozi,**, and Abhishek Halder. "Proximal Mean Field Learning in Shallow Neural Networks". *Transactions on Machine Learning Research*, 2023.
   URL:
   `https://openreview.net/forum?id=vyRBsqj5iG`

2. **Iman Nodozi**, Charlie Yan, Mira Khare, Abhishek Halder, and Ali Mesbah. "Neural Schrödinger Bridge with Sinkhorn Losses: Application to Data-driven

Minimum Effort Control of Colloidal Self-assembly". *IEEE Transactions on Control Systems Technology*, 2023, pp. 1-14.

URL:

`https://ieeexplore.ieee.org/abstract/document/10347388`

3. **Iman Nodozi**, Abhishek Halder, and Ion Matei. "A Controlled Mean Field Model for Chiplet Population Dynamics". *IEEE Control Systems Letters*, 2023, pp. 1825 - 1830, also in *62nd IEEE Conference on Decision and Control (CDC)*, Singapore, 2023.

URL:

`https://ieeexplore.ieee.org/abstract/document/10141980`

Conference Proceedings

1. Charlie Yan, **Iman Nodozi**, and Abhishek Halder. "Optimal Mass Transport over the Euler Equation". *Proceedings of the 62nd IEEE Conference on Decision and Control (CDC)*, pp. 6819-6826, Singapore, 2023.

   🎗 Invited paper in Session 'Optimal Transport'

   URL:

   `https://ieeexplore.ieee.org/document/10383425`

2. **Iman Nodozi**, Jared O'Leary, Abhishek Halder, and Ali Mesbah. "A Physics-informed Deep Learning Approach for Minimum Effort Stochastic Control of Colloidal Self-Assembly". *Proceedings of American Control Conference (ACC)*, pp. 609-615, San Diego, California, USA, 2023.

   🎗 Invited paper in Session 'Learning and Stochastic Optimal Control'

   URL:

```
https://ieeexplore.ieee.org/abstract/document/10156176
```

3. **Iman Nodozi**, and Abhishek Halder. "Schrödinger Meets Kuramoto via Feynman-Kac: Minimum Effort Distribution Steering for Noisy Nonuniform Kuramoto Oscillators". *Proceedings of the 61st IEEE Conference on Decision and Control (CDC)*, pp. 2953-2960, Cancún, Mexico, 2022.

   URL:

   ```
   https://ieeexplore.ieee.org/abstract/document/9993420
   ```

4. **Iman Nodozi**, and Abhishek Halder. "A Distributed Algorithm for Measure-valued Optimization with Additive Objective". *25th International Symposium on Mathematical Theory of Networks and Systems (MTNS)*, Bayreuth, Germany, 2022.

   🎗 Invited paper in Session 'Optimal transport: Theory and applications in networks and systems'

   URL:

   ```
   https://arxiv.org/pdf/2202.08930.pdf
   ```

Preprints

1. Alexis Teter, **Iman Nodozi**,, and Abhishek Halder. " Solution of the Probabilistic Lambert's Problem: Optimal Transport Approach".
   URL:

   ```
   https://arxiv.org/pdf/2402.01209.pdf
   ```

2. Alexis Teter, **Iman Nodozi**, and Abhishek Halder. "Solution of the Probabilistic Lambert Problem: Connections with Optimal Mass Transport, Schrödinger Bridge, and Reaction-Diffusion PDEs".

URL:

`https://arxiv.org/pdf/2401.07961.pdf`

3. **Iman Nodozi**, and Abhishek Halder. "Wasserstein Consensus ADMM".

URL:

`https://arxiv.org/pdf/2309.07351.pdf`

2 | Background on Optimal Mass Transport and Schrödinger Bridge

2.1 Optimal Mass Transport

2.1.1 Classical OMT

We start by summarizing the rudiments on classical OMT. Well-known references for this topic are [42, 44]; for a brief summary see e.g., [73].

Static Formulation. The *static formulation* of classical OMT goes back to Gaspard Monge in 1781, which concerns with finding a mass preserving transport map $\boldsymbol{\theta}$: $\mathbb{R}^n \mapsto \mathbb{R}^n$ pushing a given measure μ_0 to another μ_T while minimizing a transportation cost $\int_{\mathbb{R}^n} c(\boldsymbol{x}, \boldsymbol{\theta}(\boldsymbol{x})) \mathrm{d}\mu_0$ where c is some ground cost functional, i.e.,

$$c : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}_{\geq 0}.$$

In other words, Monge's formulation is a static optimization problem:

$$\underset{\text{measurable } \boldsymbol{\theta}:\mathbb{R}^n\mapsto\mathbb{R}^n}{\arg\inf} \quad \int_{\mathbb{R}^n} c(\boldsymbol{x}, \boldsymbol{\theta}(\boldsymbol{x})) \mathrm{d}\mu_0 \tag{2.1}$$

$$\text{subject to} \qquad \boldsymbol{x} \sim \mu_0, \quad \boldsymbol{\theta}(\boldsymbol{x}) \sim \mu_T. \tag{2.2}$$

Notice that this is an infinite dimensional nonlinear nonconvex problem over measurable maps $\boldsymbol{\theta}(\cdot)$.

A common choice for c is half of the squared Euclidean distance, but in general, the choice of the functional c plays an important role for guaranteeing the existence-uniqueness of the minimizer $\boldsymbol{\theta}^{\mathrm{opt}}(\cdot)$. Even when the existence-uniqueness of the *optimal transport map* $\boldsymbol{\theta}^{\mathrm{opt}}(\cdot)$ can be guaranteed, Monge's formulation is computationally less malleable as it requires solving a nonlinear nonconvex problem over all measur-

able pushforward mappings $\boldsymbol{\theta} : \mathbb{R}^n \mapsto \mathbb{R}^n$ taking μ_0 to μ_T. For $c(\boldsymbol{x}, \boldsymbol{y}) \equiv \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{y}\|_2^2$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, $\mathrm{d}\mu_0(\boldsymbol{x}) = \rho_0(\boldsymbol{x})\mathrm{d}\boldsymbol{x}, \mathrm{d}\mu_T(\boldsymbol{x}) = \rho_T(\boldsymbol{y})\mathrm{d}\boldsymbol{y}$, it is known [43] that $\boldsymbol{\theta}^{\mathrm{opt}}$ exists, is unique, and admits a representation $\boldsymbol{\theta}^{\mathrm{opt}} = \nabla\psi$ for some convex function ψ. Even then, the direct computation of ψ is numerically challenging because it reduces to solving a second order nonlinear elliptic Monge-Ampère PDE [42, p. 126]:

$$\det\left(\nabla^2\psi(\boldsymbol{x})\right)\rho_T(\nabla\psi(\boldsymbol{x})) = \rho_0(\boldsymbol{x})$$

where $\det$ and ∇^2 denote the determinant and the Hessian, respectively.

A more tractable reformulation of the static OMT is due to Leonid Kantorovich in 1942 [74], which instead of finding the optimal transport map $\boldsymbol{\theta}^{\mathrm{opt}}$, seeks to compute an *optimal coupling* π^{opt} between the given measures μ_0, μ_T that solves

$$\underset{\pi \in \Pi_2(\mu_0, \mu_T)}{\arg\inf} \int_{\mathbb{R}^n \times \mathbb{R}^n} c(\boldsymbol{x}, \boldsymbol{y})\mathrm{d}\pi(\boldsymbol{x}, \boldsymbol{y}) \tag{2.3}$$

where $\Pi_2(\mu_0, \mu_T)$ denotes the set of all joint probability measures π supported over the product space $\mathbb{R}^n \times \mathbb{R}^n$ with $\boldsymbol{x}$ marginal μ_0, and $\boldsymbol{y}$ marginal μ_T. Notice that (2.3) is an infinite dimensional linear program. The map $\boldsymbol{\theta}^{\mathrm{opt}}$ is precisely the support of the optimal coupling π^{opt}. In the other direction, we can recover π^{opt} from $\boldsymbol{\theta}^{\mathrm{opt}}$ as $\pi^{\mathrm{opt}} = \left(\mathrm{Id} \times \boldsymbol{\theta}^{\mathrm{opt}}\right) \sharp \mu_0$ where Id denotes the identity map, and $\sharp$ denotes the pushforward of a probability measure.

Dynamic Formulation. Let $\mathcal{P}_2(\mathbb{R}^n)$ denote the collection of probability density functions (PDFs) supported over $\mathbb{R}^n$ that have finite second moments, i.e.,

$$\mathcal{P}_2(\mathbb{R}^n) := \left\{\rho : \mathbb{R}^n \mapsto \mathbb{R}_{\geq 0} \mid \int_{\mathbb{R}^n} \rho\mathrm{d}\boldsymbol{x} = 1, \int_{\mathbb{R}^n} \|\boldsymbol{x}\|_2^2\rho\mathrm{d}\boldsymbol{x} < \infty\right\}.$$

The *dynamic formulation* of OMT due to Benamou and Brenier [75] appeared at the turn of the 21st century. When $c(\boldsymbol{x}, \boldsymbol{y}) \equiv \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{y}\|_2^2$ and μ_0, μ_T admit respective PDFs

ρ_0, ρ_T, the dynamic formulation is the following stochastic optimal control problem:

$$\underset{(\rho^{\boldsymbol{u}},\boldsymbol{u})\in\mathcal{P}_2(\mathbb{R}^n)\times\mathcal{U}}{\arg\inf} \int_0^T \int_{\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{u}\|_2^2\, \rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}},t)\mathrm{d}\boldsymbol{x}^{\boldsymbol{u}}\mathrm{d}t \tag{2.4a}$$

$$\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x}^{\boldsymbol{u}}} \cdot (\rho^{\boldsymbol{u}}\boldsymbol{u}) = 0, \tag{2.4b}$$

$$\rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}},t=0) = \rho_0, \quad \rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}},t=T) = \rho_T. \tag{2.4c}$$

The constraint (2.4b) is the *Liouville PDE* (see e.g., [76]) that governs the evolution of the state PDF $\rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}},t)$ under a feasible control policy $\boldsymbol{u} \in \mathcal{U}$. So (2.4) is a problem of optimally steering a given joint PDF ρ_0 to another ρ_T over time horizon $[0,T]$ using a vector of single integrators, i.e., with full control authority in $\mathcal{U}$. The solution $(\rho^{\mathrm{opt}}, \boldsymbol{u}^{\mathrm{opt}})$ for (2.4) satisfies

$$\rho^{\mathrm{opt}}(\boldsymbol{x}^{\boldsymbol{u}},t) = \boldsymbol{\theta}_t \,\sharp\, \rho_0, \quad \boldsymbol{\theta}_t := \left(1 - \frac{t}{T}\right)\mathrm{Id} + \frac{t}{T}\boldsymbol{\theta}^{\mathrm{opt}}, \tag{2.5a}$$

$$\boldsymbol{u}^{\mathrm{opt}}(\boldsymbol{x}^{\boldsymbol{u}},t) = \nabla_{\boldsymbol{x}^{\boldsymbol{u}}}\phi(\boldsymbol{x}^{\boldsymbol{u}},t), \quad \frac{\partial \phi}{\partial t} + \frac{1}{2}\|\nabla_{\boldsymbol{x}^{\boldsymbol{u}}}\phi\|_2^2 = 0. \tag{2.5b}$$

Thus, (2.5a) tells that the optimally controlled PDF is obtained as pushforward of the initial PDF via a map that is a linear interpolation between identity and the optimal transport map. Consequently, the PDF ρ^{opt} itself is a (nonlinear) McCann's displacement interpolant [77] between ρ_0 and ρ_T. The optimal control in (2.5b) is obtained as the gradient of the solution of a *Hamilton-Jacobi-Bellman (HJB) PDE*.

The $\psi(\boldsymbol{x})$ in static OMT and the $\phi(\boldsymbol{x},t)$ in dynamic OMT are related [42, Thm. 5.51] through the *Hopf-Lax representation formula*

$$\phi(\boldsymbol{x},t) = \inf_{\boldsymbol{y}\in\mathbb{R}^n}\left(\phi(\boldsymbol{y},0) + \frac{1}{2t}\|\boldsymbol{x}-\boldsymbol{y}\|_2^2\right), \ t \in (0,T], \tag{2.6a}$$

$$\phi(\boldsymbol{y},0) = \psi(\boldsymbol{y}) - \frac{1}{2}\|\boldsymbol{y}\|_2^2, \tag{2.6b}$$

i.e., $\phi(\boldsymbol{x},t)$ is the Moreau-Yosida proximal envelope [61, Ch. 3.1] of $\phi(\boldsymbol{y},0) = \psi(\boldsymbol{y}) - \frac{1}{2}\|\boldsymbol{y}\|_2^2$, and hence $\phi(\boldsymbol{x},t)$ is continuously differentiable w.r.t. $\boldsymbol{x} \in \mathbb{R}^n$.

Classical OMT allows defining a distance metric, called the *Wasserstein metric* W, on the manifold of probability measures or PDFs. In particular, when $c(\boldsymbol{x}, \boldsymbol{y}) \equiv \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{y}\|_2^2$, the infimum value achieved in (2.3) is the one half of the squared Wasserstein metric between μ_0 and μ_T, i.e.,

$$W^2(\mu_0, \mu_T) := \inf_{\pi \in \Pi_2(\mu_0, \mu_T)} \int_{\mathbb{R}^n \times \mathbb{R}^n} \|\boldsymbol{x} - \boldsymbol{y}\|_2^2 \, \mathrm{d}\pi(\boldsymbol{x}, \boldsymbol{y}), \tag{2.7}$$

which is also equal to the infimum value achieved in (2.4), provided μ_0, μ_T are absolutely continuous. The tuple $(\mathcal{P}_2(\mathbb{R}^n), W)$ defines a complete separable metric space, i.e., a polish space. This offers a natural way to metrize the topology of weak convergence of probability measures w.r.t. the metric W.

2.1.2   Generalized OMT

The dynamic version of the generalized OMT is the following stochastic optimal control problem:

$$\arg\inf_{(\rho^{\boldsymbol{u}}, \boldsymbol{u}) \in \mathcal{P}_2(\mathbb{R}^n) \times \mathcal{U}} \int_0^T \int_{\mathbb{R}^n} (q(\boldsymbol{x}^{\boldsymbol{u}}) + r(\boldsymbol{u})) \, \rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}, t) \mathrm{d}\boldsymbol{x}^{\boldsymbol{u}} \mathrm{d}t \tag{2.8a}$$

$$\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x}^{\boldsymbol{u}}} \cdot (\rho^{\boldsymbol{u}} \boldsymbol{f}(t, \boldsymbol{x}^{\boldsymbol{u}}, \boldsymbol{u})) = 0, \tag{2.8b}$$

$$\rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}, t = 0) = \rho_0, \quad \rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}, t = T) = \rho_T. \tag{2.8c}$$

We suppose that the cost functions $q(\cdot), r(\cdot)$ and the controlled vector field $\boldsymbol{f}$ are sufficiently smooth to make the problem (2.8) well-posed. In particular, we assume that $q + r$ is at least lower bounded.

Notice that the problem (2.8) reduces to the problem (2.4) when

$$q(\cdot) \equiv 0, \quad r(\cdot) \equiv \frac{1}{2}\|\cdot\|_2^2, \quad \boldsymbol{f} \equiv \boldsymbol{u}.$$

In other words, (2.8) generalizes (2.4) in two ways. One generalization comes from

considering more general (separable) cost-to-go in the objective. Another generaliza-
tion comes from considering more general (both time, state and control-dependent)
vector field in the Liouville PDE constraint.

Notice that the PDE (2.4b) is induced by the underlying controlled ODE: $\dot{\boldsymbol{x}}^{\boldsymbol{u}} =
\boldsymbol{u}$. In contrast, the PDE (2.8b) is induced by the underlying controlled ODE: $\dot{\boldsymbol{x}}^{\boldsymbol{u}} =
\boldsymbol{f}(t, \boldsymbol{x}^{\boldsymbol{u}}, \boldsymbol{u})$.

Just like the classical dynamic OMT (2.4) corresponds to the classical static OMT
(2.3), similar correspondence can be associated with (2.8). To do this, we slightly gen-
eralize the setting: we replace $\mathbb{R}^n$ in (2.3) with an n dimensional Riemannian manifold
$\mathcal{M}$. Consider an absolutely continuous curve $\boldsymbol{\gamma}(t) \in \mathcal{M}$, $t \in [0, T]$, and $(\boldsymbol{\gamma}, \dot{\boldsymbol{\gamma}}) \in \mathcal{TM}$
(tangent bundle). Then for $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{M}$, we think of $c(\boldsymbol{x}, \boldsymbol{y})$ in (2.3) to be derived from a
Lagrangian $L : [0, T] \times \mathcal{TM} \mapsto \mathbb{R}$, i.e., express c as an action integral

$$c(\boldsymbol{x}, \boldsymbol{y}) = \inf_{\boldsymbol{\gamma}(\cdot) \in \Gamma_{\boldsymbol{xy}}} \int_0^T L(t, \boldsymbol{\gamma}(t), \dot{\boldsymbol{\gamma}}(t)) \, \mathrm{d}t, \tag{2.9}$$

where

$$\Gamma_{\boldsymbol{xy}} := \{\boldsymbol{\gamma} : [0, T] \mapsto \mathbb{R}^n \mid \boldsymbol{\gamma}(\cdot) \text{ is absolutely continuous}, \boldsymbol{\gamma}(0) = \boldsymbol{x}, \boldsymbol{\gamma}(T) = \boldsymbol{y}\}.$$

In particular, the choice $\mathcal{M} \equiv \mathbb{R}^n$ and $L(t, \boldsymbol{\gamma}, \dot{\boldsymbol{\gamma}}) \equiv \frac{1}{2}\|\dot{\boldsymbol{\gamma}}\|_2^2$ results in $c(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2^2$,
i.e., the standard Euclidean OMT (2.3). When $\mathcal{M} \equiv \mathbb{R}^n$ and $\boldsymbol{f}(t, \boldsymbol{x}^{\boldsymbol{u}}, \boldsymbol{u}) \equiv \widetilde{\boldsymbol{f}}(t, \boldsymbol{x}^{\boldsymbol{u}}) +
\boldsymbol{B}(t)\boldsymbol{u}$, i.e., a control-affine vector field with $\boldsymbol{B}(t) \in \mathbb{R}^{n \times n}$ nonsingular for all $t \in [0, T]$,
then the Lagrangain L in (2.9) is

$$L(t, \boldsymbol{\gamma}, \dot{\boldsymbol{\gamma}}) = q(\boldsymbol{\gamma}) + r\left((\boldsymbol{B}(t))^{-1}\left(\dot{\boldsymbol{\gamma}} - \widetilde{\boldsymbol{f}}(t, \boldsymbol{\gamma})\right)\right).$$

We say that a Lagrangian L is *superlinear* (1-coercive) if

$$\lim_{\|\dot{\boldsymbol{\gamma}}\|_2 \to \infty} \frac{L}{\|\dot{\boldsymbol{\gamma}}\|_2} = +\infty. \tag{2.10}$$

For the identified Lagrangian L in (2.9), if the mapping $\dot{\boldsymbol{\gamma}} \mapsto L(\cdot, \cdot, \dot{\boldsymbol{\gamma}})$ is strictly convex as well as superlinear, then we say L is a *weak Tonelli Lagrangian* [44, p. 118], [78, Ch. 6.2]. It is known that if L in (2.9) is a weak Tonelli Lagrangian, then [78, Thm. 1.4.2] the existence and uniqueness of the minimizing pair $(\rho^{\mathrm{opt}}, \boldsymbol{u}^{\mathrm{opt}})$ for (2.8) is guaranteed.

2.2 Schrödinger Bridge Problem

From a control-theoretic viewpoint, both the SBP and the OMT are stochastic optimal control problems. Even then, it is helpful to think about the SBP as a further stochastic generalization of the OMT. We explain this next.

2.2.1 Classical SBP

The classical SBP concerns with the minimum effort additive control needed to move a given distribution to another over a presribed finite time horizon subject to the constraint that the uncontrolled sample paths evolve according to Brownian motion (i.e., standard Wiener process). This is a stochastic optimal control of the form

$$\underset{(\rho^{\boldsymbol{u}}, \boldsymbol{u}) \in \mathcal{P}_2(\mathbb{R}^n) \times \mathcal{U}}{\arg\inf} \int_0^T \int_{\mathbb{R}^n} \frac{1}{2} \|\boldsymbol{u}\|_2^2 \, \rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}, t) \mathrm{d}\boldsymbol{x}^{\boldsymbol{u}} \mathrm{d}t \tag{2.11a}$$

$$\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x}^{\boldsymbol{u}}} \left(\rho^{\boldsymbol{u}} \boldsymbol{u} \right) = \beta^{-1} \Delta_{\boldsymbol{x}^{\boldsymbol{u}}} \rho^{\boldsymbol{u}}, \tag{2.11b}$$

$$\rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}, t = 0) = \rho_0, \quad \rho^{\boldsymbol{u}}(\boldsymbol{x}^{\boldsymbol{u}}, t = T) = \rho_T. \tag{2.11c}$$

The constant β is referred to as the inverse temperature. Clearly, if $\beta^{-1} \downarrow 0$, then the classical SBP (2.11) reduces to the classical OMT (2.4).

Notice that unlike the first order Liouville PDE (2.4b), the PDE constraint (2.11b) involves a second order Laplacian term. Thus, (2.11b) is a Fokker-Planck-Kolmogorov (FPK) PDE. This can be motivated as follows. The macroscopic dynamics (2.11b) is

induced by the controlled sample path dynamics:

$$\mathrm{d}\boldsymbol{x^u} = \boldsymbol{u}\,\mathrm{d}t + \sqrt{2\beta^{-1}}\mathrm{d}\boldsymbol{w} \tag{2.12}$$

where $\boldsymbol{u} \in \mathbb{R}^n$ is the control, and $\boldsymbol{w}$ is the standard Wiener process in $\mathbb{R}^n$. In other words, the trajectory-level controlled dynamics for the classical SBP is "single integrator + Gaussian white noise". The controlled Itô stochastic differential equation (SDE) generalizes the controlled ODE

$$\frac{\mathrm{d}\boldsymbol{x^u}}{\mathrm{d}t} = \boldsymbol{u},$$

which was indeed the sample path dynamics associated with (2.4b).

The SBP originated in the works of Erwin Scrödinger [79–81] and as such predates both the mathematical theory of stochastic processes and feedback control. Scrödinger's original motivation behind this study was to seek a probabilistic interpretation of quantum mechanics.

2.2.2   Generalized SBP

The generalized SBP is a stochastic optimal control problem of the form:

$$\inf_{(\rho^{\boldsymbol{u}},\boldsymbol{u})\in\mathcal{P}_2(\mathbb{R}^n)\times\mathcal{U}} \int_0^T \int_{\mathbb{R}^n} \left(q(\boldsymbol{x}^u) + r(\boldsymbol{u})\right)\rho^{\boldsymbol{u}}(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}\mathrm{d}t \tag{2.13a}$$

$$\frac{\partial\rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x^u}}\cdot\left(\rho^{\boldsymbol{u}}\boldsymbol{f}\left(t,\boldsymbol{x^u},\boldsymbol{u}\right)\right) = \beta^{-1}\langle\mathbf{Hess}, \boldsymbol{G}\left(t,\boldsymbol{x^u},\boldsymbol{u}\right)\rangle\rho^{\boldsymbol{u}} \tag{2.13b}$$

$$\rho^{\boldsymbol{u}}\left(\boldsymbol{x^u}, t=0\right) = \rho_0, \quad \rho^{\boldsymbol{u}}\left(\boldsymbol{x^u}, t=T\right) = \rho_T, \tag{2.13c}$$

where $\langle\cdot,\cdot\rangle$ in (2.13b) is the Frobenius inner product, $\boldsymbol{G}\left(t,\boldsymbol{x^u},\boldsymbol{u}\right) \in \mathbb{R}^{n\times n}$ is the diffuison tensor. The diffusion tensor $\boldsymbol{G}\left(t,\boldsymbol{x^u},\boldsymbol{u}\right) := \left(\boldsymbol{g}\left(t,\boldsymbol{x^u},\boldsymbol{u}\right)\right)^{\top}\boldsymbol{g}\left(t,\boldsymbol{x^u},\boldsymbol{u}\right)$ for some diffusion coefficient $\boldsymbol{g}\left(t,\boldsymbol{x^u},\boldsymbol{u}\right) \in \mathbb{R}^{n\times p}$. Thus, the diffusion tensor, by definition, is symmetric positive semi-definite.

As in the case for the OMT, here too, the qualifier "generalized" refers to the presence of prior dynamics given by the controlled drift-diffusion coefficient pair $(\boldsymbol{f}, \boldsymbol{g})$ which was not considered in Schrödinger's original investigations [79, 80]. Specifically, the FPK PDE (2.13b) is induced by a controlled sample path dynamics

$$\mathrm{d}\boldsymbol{x^u} = \boldsymbol{f}\left(t, \boldsymbol{x^u}, \boldsymbol{u}\right)\mathrm{d}t + \sqrt{2\beta^{-1}}\,\boldsymbol{g}\left(t, \boldsymbol{x^u}, \boldsymbol{u}\right)\mathrm{d}\boldsymbol{w}, \qquad (2.14)$$

where $\boldsymbol{u} \in \mathbb{R}^m$ is the control, and $\boldsymbol{w}$ is a standard Wiener process in $\mathbb{R}$.

Let us now comment on how the classical SBP (2.11) can be seen as a special case of the gernealized SBP (2.13). By setting $\boldsymbol{f} \equiv \boldsymbol{u}$ and $\boldsymbol{g} \equiv \boldsymbol{I}_n$, we see that (2.14) reduces to (2.12), and consequently the macroscopic dynamics (2.13b) specializes to (2.11b). The reduction of (2.13) to (2.11) then follows by setting $q(\cdot) \equiv 0$, $r(\cdot) = \frac{1}{2}\|\boldsymbol{u}\|_2^2$.

In recent years, Schrödinger bridge problems and their connections to OMT have come to prominence in both control [38, 39, 82, 83] and machine learning [84–86] communities.

A different way to interpret classical SBP is to view it as a stochastic dynamic version of the optimal mass transport (OMT) problem. The dynamic OMT [75] is a special case of (2.13c) with $\boldsymbol{f} \equiv \boldsymbol{u}$, $\boldsymbol{g} \equiv \boldsymbol{0}$. For details on these connections from a stochastic control perspective, we refer the readers to [82]. In recent years, SBPs and their generalizations have come to prominence in both control [32, 38, 39, 82] and machine learning [84–87] communities. In particular, a data-driven maximum likelihood sampling solution of the classical SBP (i.e., with $\boldsymbol{f} \equiv \boldsymbol{u}$, $\boldsymbol{g} \equiv \boldsymbol{I}_n$) was proposed in [88] assuming availability of the samples from the endpoint measures μ_0, μ_T. Similar line of ideas were pursued in [89, 90].

While solution methods for the GSBP (2.14) in general are not available in the current literature, specialized algorithms for particular forms of $\boldsymbol{f}, \boldsymbol{g}$ have appeared.

In [39], this result was extended for the case when additional (deterministic) state constraints are present.

GSBPs with nonlinear drifts and full-state feedback linearizable structures were considered in [91, 92]. GSBP instances for both first- and second-order noisy nonuniform Kuramoto oscillator models were solved in [83] using Feynman-Kac path integral techniques. Closest to the GSBP (2.13) is the work in [68], which considered *control non-affine* drift and diffusion coefficients and showed that the conditions of optimality involves additional coupled PDEs compared to the control-affine case. However, the developments in [68] were still model based. Data-driven solution of control non-affine GSBPs at the level of generality (2.13), as pursued in this work, is novel w.r.t. the existing literature.

3 | Control-affine Generalized Schrödinger Bridge

3.1 Problem Formulation

In this chapter, we consider a control-affine nonlinear for the *minimum energy* generalized SBP. Specifically, we aim to solve (2.13) given that:

$$q(\cdot) \equiv 0, \quad r(\cdot) \equiv \frac{1}{2}\|\cdot\|_2^2,$$

where the associated drift and diffusion coefficient pair $(\boldsymbol{f}, \boldsymbol{g})$ in (2.14), are

$$\boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}) \equiv \widetilde{\boldsymbol{f}}(t, \boldsymbol{x}) + \boldsymbol{B}(t)\boldsymbol{u}, \qquad \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{u}) \equiv \boldsymbol{B}(t) \in \mathbb{R}^{n \times m}.$$

i.e., the drift coefficient $\boldsymbol{f}$ is control affine and the diffusion coefficient $\boldsymbol{g}$ is $C([0, T])$ matrix that is independent of state and input. In this case, the stochastic process noise enters through the input channels (e.g., modeling disturbance in forcing and/or actuation uncertainties). So, in this chapter, we focus on problems the form

$$
\begin{aligned}
\inf_{\boldsymbol{u} \in \mathcal{U}} \quad & \mathbb{E}_{\mu^{\boldsymbol{u}}} \left\{ \int_0^T \tfrac{1}{2} \|\boldsymbol{u}(\boldsymbol{x}, t)\|_2^2 \, \mathrm{d}t \right\} \\
\text{subject to} \quad & \mathrm{d}\boldsymbol{x} = \widetilde{\boldsymbol{f}}(\boldsymbol{x}, t)\mathrm{d}t + \boldsymbol{B}(t)\boldsymbol{u}(\boldsymbol{x}, t)\mathrm{d}t + \sqrt{2\epsilon}\boldsymbol{B}(t)\mathrm{d}\boldsymbol{w}(t), \qquad (3.1) \\
& \boldsymbol{x}(t = 0) \sim \mu_0(\boldsymbol{x}), \quad \boldsymbol{x}(t = T) \sim \mu_T(\boldsymbol{x}),
\end{aligned}
$$

where μ_0, μ_T denote the joint state probability measures at $t = 0$ and $t = T$, respectively. The vector field $\widetilde{\boldsymbol{f}}$ represents a prior nonlinear dynamics. In (3.1), the set of feasible controls $\mathcal{U}$ comprises of the finite energy inputs over the time horizon $[0, T]$, i.e., $\mathcal{U} := \{\boldsymbol{u} : \mathbb{R}^n \times [0, 1] \mapsto \mathbb{R}^m \mid \|\boldsymbol{u}\|_2 < \infty\}$. In words, problem (3.1) asks to synthesize feedback control that steers the initial joint state distribution μ_0 to the terminal joint state distribution μ_T over $[0, T]$ while minimizing the average control effort in doing so.

We make the following assumptions for $\widetilde{\boldsymbol{f}}(\boldsymbol{x},t)$ and $\boldsymbol{B}(t)$:

- **non-explosion and Lipschitz:** there exist constants c_1, c_2 such that $\|\widetilde{\boldsymbol{f}}(\boldsymbol{x},t)\|_2 +$ $\|\boldsymbol{B}(t)\|_2 \leq c_1 (1 + \|\boldsymbol{x}\|_2)$, and $\|\widetilde{\boldsymbol{f}}(\boldsymbol{x},t) - \widetilde{\boldsymbol{f}}(\boldsymbol{y},t)\|_2 \leq c_2 \|\boldsymbol{x} - \boldsymbol{y}\|_2$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, t \in [0,T]$.

- **uniformly lower bounded diffusion:** there exists a constant c_3 such that the diffusion tensor $\boldsymbol{D}(t) := \boldsymbol{B}(t)\boldsymbol{B}(t)^{\top}$ satisfies $\langle \boldsymbol{x}, \boldsymbol{D}(t)\boldsymbol{x} \rangle \geq c_3 \|\boldsymbol{x}\|_2^2$ for all $\boldsymbol{x} \in \mathbb{R}^n, t \in [0,T]$.

The first assumption rules out [93, p. 66] the possibility of finite time blow up of the sample path $\boldsymbol{x}(t)$, and ensures the existence-uniqueness for the same. The second assumption together with the first, ensures [94, Ch. 1] that for $T \geq t > s \geq 0$, the transition density $K(s, \boldsymbol{y}, t, \boldsymbol{x})$ associated with the uncontrolled nonlinear SDE

$$\mathrm{d}\boldsymbol{x}(t) = \widetilde{\boldsymbol{f}}(\boldsymbol{x},t)\mathrm{d}t + \sqrt{2\epsilon}\boldsymbol{B}(t)\mathrm{d}\boldsymbol{w}(t) \tag{3.2}$$

is strictly positive and everywhere continuous in $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$.

Assuming the absolute continuity of the joint probability measure $\mu^{\boldsymbol{u}}$ for all times, we write $\mathrm{d}\mu^{\boldsymbol{u}}(\boldsymbol{x},t) = \rho^{\boldsymbol{u}}(\boldsymbol{x},t)\mathrm{d}\boldsymbol{x}$ and hereafter consider the associated joint PDF $\rho^{\boldsymbol{u}}(\boldsymbol{x},t)$. Problem (3.1) can then be recast as

$$\inf_{(\rho^{\boldsymbol{u}},\boldsymbol{u})} \frac{1}{2} \int_0^T \int_{\mathbb{R}^n} \|\boldsymbol{u}(\boldsymbol{x},t)\|_2^2 \rho^{\boldsymbol{u}}(\boldsymbol{x},t)\mathrm{d}\boldsymbol{x}\mathrm{d}t$$

$$\text{subject to } \frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla \cdot (\rho^{\boldsymbol{u}}(\widetilde{\boldsymbol{f}} + \boldsymbol{B}(t)\boldsymbol{u})) = \epsilon \langle \boldsymbol{D}(t), \text{Hess}\,(\rho^{\boldsymbol{u}}) \rangle \tag{3.3}$$

$$\rho^{\boldsymbol{u}}(\boldsymbol{x},0) = \rho_0(\boldsymbol{x})(\text{given}), \quad \rho^{\boldsymbol{u}}(\boldsymbol{x},T) = \rho_T(\boldsymbol{x})(\text{given}).$$

Next, we briefly discuss the existence-uniqueness for the solution of (3.3).

3.2   Existence and Uniqueness

Consider the change of variable $(\rho, \boldsymbol{u}) \mapsto (\rho, \boldsymbol{m})$ given by $\boldsymbol{m} := \rho\boldsymbol{u}$, which results in the following reformulation of (3.3):

$$\inf_{(\rho,\boldsymbol{m})} \quad \frac{1}{2} \int_0^T \int_{\mathbb{R}^n} J(\rho, \boldsymbol{m}) \mathrm{d}\boldsymbol{x}\mathrm{d}t \tag{3.4a}$$

$$\text{Subject to } \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho(\widetilde{\boldsymbol{f}} + \boldsymbol{B}(t)\boldsymbol{m})) = \epsilon\langle \boldsymbol{D}(t), \mathbf{Hess}(\rho)\rangle \tag{3.4b}$$

$$\rho^{\boldsymbol{u}}(\boldsymbol{x}, 0) = \rho_0(\boldsymbol{x}, \quad \rho^{\boldsymbol{u}}(\boldsymbol{x}, T) = \rho_T(\boldsymbol{x}) \tag{3.4c}$$

where

$$J(\rho, \boldsymbol{m}) := \begin{cases} \|\boldsymbol{m}\|_2^2/\rho & \text{if} \quad \rho > 0, \\ 0 & \text{if} \quad (\boldsymbol{m}, \rho) = (\boldsymbol{0}, 0), \\ +\infty & \text{otherwise.} \end{cases}$$

Since $J(\rho, \boldsymbol{m})$ is the perspective function of the strictly convex map $\boldsymbol{m} \mapsto \|\boldsymbol{m}\|_2^2$, hence J is jointly strictly convex in $(\rho, \boldsymbol{m})$. The constraints (3.4b)-(3.4c) are linear in $(\rho, \boldsymbol{m})$. Thus, the problem (3.4) is convex. However, proving the existence of minimizing pair $(\rho^{\mathrm{opt}}, \boldsymbol{m}^{\mathrm{opt}})$, or equivalently $(\rho^{\mathrm{opt}}, \boldsymbol{u}^{\mathrm{opt}})$, is nontrivial. It turns out that the existence-uniqueness for the solution of (3.3) or (3.4), can be guaranteed [95, Theorem 3.2] for compactly supported ρ_0, ρ_1, provided the transition density for (3.2) is strictly positive and everywhere continuous in $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. This is where the aforesaid assumptions on $\widetilde{\boldsymbol{f}}(\boldsymbol{x}, t), \boldsymbol{B}(t)$ come into play. The details for two specific case studies, viz. the first and the second order noisy nonuniform Kuramoto oscillators, are given in the next sections.

3.3   Conditions for Optimality

We next show that the first order optimality conditions for (3.3) lead to a coupled system of nonlinear PDEs. Consider the Lagrangian associated with (3.3):

$$\mathcal{L}(\rho, \boldsymbol{u}, \psi) :=$$
$$\int_0^T \int_{\mathbb{R}^n} \left\{ \frac{1}{2} \|\boldsymbol{u}(\boldsymbol{x}, t)\|_2^2 \rho(\boldsymbol{x}, t) + \psi(\boldsymbol{x}, t) \times \left(\frac{\partial \rho}{\partial t} + \nabla \cdot ((\widetilde{\boldsymbol{f}} + \boldsymbol{B}(t)\boldsymbol{u})\rho(\boldsymbol{x}, t)) \right. \right. \quad (3.5)$$
$$\left. \left. -\epsilon \langle \boldsymbol{D}(t), \text{Hess}(\rho) \rangle \right) \right\} \mathrm{d}\boldsymbol{x} \mathrm{d}t$$

where $\psi(\boldsymbol{x}, t)$ is a $C^1\left(\mathbb{R}^n; \mathbb{R}_{>0}\right)$ Lagrange multiplier. Let

$$\mathcal{P}_{01}\left(\mathbb{R}^n\right) := \left\{ \rho(\boldsymbol{x}, t) \mid \rho \geq 0, \int_{\mathbb{R}^n} \rho \mathrm{d}\boldsymbol{x} = 1, \rho(\boldsymbol{x}, 0) = \rho_0, \rho(\boldsymbol{x}, T) = \rho_T \right\}. \quad (3.6)$$

Performing the unconstrained minimization of the Lagrangian $\mathcal{L}$ over $\mathcal{P}_{01}\left(\mathbb{R}^n\right) \times \mathcal{U}$ yields the following result.

Proposition 3.1.   *[38, Proposition 1] **(Optimal control and optimal state PDF)** The pair $\left(\rho^{\mathrm{opt}}(\boldsymbol{x}, t), \boldsymbol{u}^{\mathrm{opt}}(\boldsymbol{x}, t)\right)$ that solves (3.3), must satisfy the system of coupled PDEs*

$$\frac{\partial \psi}{\partial t} + \frac{1}{2} \left\| \boldsymbol{B}(t)^\top \nabla \psi \right\|_2^2 + \langle \nabla \psi, \widetilde{\boldsymbol{f}} \rangle = -\epsilon \langle \boldsymbol{D}(t), \text{Hess}(\psi) \rangle, \quad (3.7\text{a})$$

$$\frac{\partial}{\partial t} \rho^{\mathrm{opt}} + \nabla \cdot \left(\rho^{\mathrm{opt}} \left(\widetilde{\boldsymbol{f}} + \boldsymbol{B}(t)^\top \nabla \psi \right) \right) = \epsilon \langle \boldsymbol{D}(t), \text{Hess}\left(\rho^{\mathrm{opt}}\right) \rangle, \quad (3.7\text{b})$$

with boundary conditions

$$\rho^{\mathrm{opt}}(\boldsymbol{x}, 0) = \rho_0(\boldsymbol{x}), \quad \rho^{\mathrm{opt}}(\boldsymbol{x}, T) = \rho_T(\boldsymbol{x}), \quad (3.8)$$

and

$$\boldsymbol{u}^{\mathrm{opt}}(\boldsymbol{x}, t) = \boldsymbol{B}(t)^\top \nabla \psi(\boldsymbol{x}, t).$$

The proof of this result can be found in [38, Appendix A].

The PDE (3.7a) is the Hamilton-Jacobi-Bellman (HJB) equation while the PDE

(3.7b) is the controlled FPK equation. Computing the optimal pair $(\rho^{\mathrm{opt}}, \boldsymbol{u}^{\mathrm{opt}})$, or equivalently $(\rho^{\mathrm{opt}}, \psi)$ form (3.7)-(3.8) calls for solving a system of coupled nonlinear PDEs with atypical boundary conditions, and is challenging in general.

Interestingly, it is possible to transform the HJB-FPK system of coupled nonlinear PDEs (20)-(21) into a system of boundary-coupled linear PDEs, via the so-called Hopf-Cole transform [96], [97] given by

$$\varphi(\boldsymbol{x}, t) = \exp\left(\frac{\psi(\boldsymbol{x}, t)}{2\epsilon}\right) \tag{3.9a}$$

$$\hat{\varphi}(\boldsymbol{x}, t) = \rho^{\mathrm{opt}}(\boldsymbol{x}, t) \exp\left(-\frac{\psi(\boldsymbol{x}, t)}{2\epsilon}\right). \tag{3.9b}$$

The following result is of our interest.

Theorem 3.1. *[38, Theorem 2] (Hopf-Cole transform) Given* $\widetilde{\boldsymbol{f}}, \epsilon, \rho_0, \rho_1,$ *consider the Hopf-Cole transform* $(\rho^{\mathrm{opt}}, \psi) \mapsto (\varphi, \hat{\varphi})$ *defined via (3.9), applied to the system of nonlinear PDEs (3.7)-(3.8). Then the pair* $(\varphi, \hat{\varphi})$ *satisfies the system of linear PDEs*

$$\frac{\partial \varphi}{\partial t} = -\langle \nabla \varphi, \widetilde{\boldsymbol{f}} \rangle - \epsilon \langle \boldsymbol{D}(t), \mathrm{Hess}(\varphi) \rangle \tag{3.10a}$$

$$\frac{\partial \hat{\varphi}}{\partial t} = -\nabla \cdot (\hat{\varphi} \widetilde{\boldsymbol{f}}) + \epsilon \langle \boldsymbol{D}(t), \mathrm{Hess}(\hat{\varphi}) \rangle \tag{3.10b}$$

with boundary conditions

$$\varphi(\boldsymbol{x}, 0) \hat{\varphi}(\boldsymbol{x}, 0) = \rho_0(\boldsymbol{x}), \quad \varphi(\boldsymbol{x}, T) \hat{\varphi}(\boldsymbol{x}, T) = \rho_T(\boldsymbol{x}). \tag{3.11}$$

For the proof of the above, we refer the readers to [38, Appendix B].

The optimal controlled state PDF $\rho^{\mathrm{opt}}(\boldsymbol{x}, t)$ solving (3.7) is given by

$$\rho^{\mathrm{opt}}(\boldsymbol{x}, t) = \varphi(\boldsymbol{x}, t) \hat{\varphi}(\boldsymbol{x}, t). \tag{3.12}$$

The optimal control for the same is given by

$$\boldsymbol{u}^{\mathrm{opt}}(\boldsymbol{x}, t) = 2\epsilon \boldsymbol{B}(t)^{\top} \nabla \log \varphi. \tag{3.13}$$

We note that (3.10a) is the backward Kolmogorov equation in variable φ, and (3.10b) is the forward Kolmogorov or the FPK equation in variable $\hat{\varphi}$, associated with (3.2).

The essence of Theorem 3.1 is that instead of solving the system of coupled nonlinear PDEs (3.7), we can solve the system of linear PDEs (3.10) provided that we compute the pair $(\varphi_1, \hat{\varphi}_0)$ which serves as the endpoint data for

$$\frac{\partial \varphi}{\partial t} = -\langle \nabla\varphi, \widetilde{\boldsymbol{f}} \rangle - \epsilon \langle \boldsymbol{D}(t), \text{Hess}(\varphi) \rangle, \qquad \varphi(\boldsymbol{x}, T) = \varphi_T(\boldsymbol{x}), \tag{3.14a}$$

$$\frac{\partial \hat{\varphi}}{\partial t} = -\nabla \cdot (\hat{\varphi}\widetilde{\boldsymbol{f}}) + \epsilon \langle \boldsymbol{D}(t), \text{Hess}(\hat{\varphi}) \rangle, \qquad \hat{\varphi}(\boldsymbol{x}, 0) = \hat{\varphi}_0(\boldsymbol{x}). \tag{3.14b}$$

Denoting the forward and backward Kolmogorov operators in (3.14) as L_{FK} and L_{BK} respectively, we can write (3.14) succinctly as an infinite dimensional two point boundary value problem

$$\frac{\partial}{\partial t}\begin{pmatrix} \varphi \\ \hat{\varphi} \end{pmatrix} = \begin{pmatrix} L_{\text{BK}} & 0 \\ 0 & L_{\text{FK}} \end{pmatrix}\begin{pmatrix} \varphi \\ \hat{\varphi} \end{pmatrix}, \qquad \begin{pmatrix} \varphi(\boldsymbol{x}, T) \\ \hat{\varphi}(\boldsymbol{x}, 0) \end{pmatrix} = \begin{pmatrix} \varphi_T(\boldsymbol{x}) \\ \hat{\varphi}_0(\boldsymbol{x}) \end{pmatrix}. \tag{3.15}$$

We next consider a case study for distribution steering via feedback control, namely doing the same subject to controlled dynamics of a networked noisy nonuniform Kuramoto oscillators. We point out the related literature and engineering relevance of this problem.

3.4 Case Study: Kuramoto Oscillators

We consider the controlled sample path dynamics for a population of n *first order* Kumarmoto oscillators, given by the Itô SDEs

$$\text{d}\theta_i = \left(-\frac{\partial V}{\partial \theta_i} + v_i\right)\text{d}t + \sqrt{2}\sigma_i\,\text{d}w_i, \; i \in [n] \coloneqq \{1, 2, \cdots, n\},$$

where $V(\theta_1, \cdots, \theta_n)$ is a given smooth potential, the angular variable $\theta_i \in [0, 2\pi)$ is the state, v_i is the control input, $\sigma_i > 0$ is the noise strength, and w_i is the standard (scalar) Wiener process noise for the ith oscillator. Defining re-scaled input $u_i := \sigma_i v_i$, we write this dynamics in vector form:

$$\mathrm{d}\boldsymbol{\theta} = (-\nabla_{\boldsymbol{\theta}} V(\boldsymbol{\theta}) + \boldsymbol{S}\boldsymbol{u}) \, \mathrm{d}t + \sqrt{2}\boldsymbol{S} \, \mathrm{d}\boldsymbol{w} \qquad (3.16)$$

where $\boldsymbol{S} := \mathrm{diag}(\sigma_1, \ldots, \sigma_n) > \boldsymbol{0}$, $\boldsymbol{\theta} := (\theta_1, \cdots, \theta_n)^\top$, $\boldsymbol{u} := (u_1, \cdots, u_n)^\top$, and $\boldsymbol{w} := (w_1, \cdots, w_n)^\top$ is the standard Wiener process in n dimensions. For the first order Kuramoto model (3.16), the state space is the n-torus $\mathbb{T}^n \equiv [0, 2\pi)^n$, and the potential

$$V(\boldsymbol{\theta}) := \sum_{\substack{i<j \\ i,j \in [n]}} k_{ij}(1 - \cos(\theta_i - \theta_j - \varphi_{ij})) - \sum_{i=1}^n P_i \theta_i, \qquad (3.17)$$

wherein the parameters $P_i > 0$. For $i \neq j$, the coupling coefficients $k_{ij} = k_{ji} \geq 0$ (and not all $k_{ij} = 0$), and $k_{ii} \equiv 0$. Likewise, for $i \neq j$, the phase shift $\varphi_{ij} = \varphi_{ji} \in [0, \frac{\pi}{2})$, and $\varphi_{ii} \equiv 0$.

We also consider the controlled sample path dynamics for a population of n *second order* Kumarmoto oscillators, given by the second order Langevin equations

$$m_i \ddot{\theta}_i + \gamma_i \dot{\theta}_i = -\frac{\partial V}{\partial \theta_i} + \sqrt{2}\sigma_i \times \text{standard Gaussian white noise,}$$

where $i \in [n]$, and $V(\cdot)$ is given by (3.17). We rewrite this dynamics as the vector Itô SDE

$$\begin{pmatrix} \mathrm{d}\boldsymbol{\theta} \\ \mathrm{d}\boldsymbol{\omega} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\omega} \\ -\boldsymbol{M}^{-1}\nabla_{\boldsymbol{\theta}} V(\boldsymbol{\theta}) - \boldsymbol{M}^{-1}\boldsymbol{\Gamma}\boldsymbol{\omega} + \boldsymbol{M}^{-1}\boldsymbol{S}\boldsymbol{u} \end{pmatrix} \mathrm{d}t + \begin{pmatrix} \boldsymbol{0}_{n \times 1} \\ \sqrt{2}\boldsymbol{M}^{-1}\boldsymbol{S} \, \mathrm{d}\boldsymbol{w} \end{pmatrix} \qquad (3.18)$$

where $\boldsymbol{\omega} := (\dot{\theta}_1, \cdots, \dot{\theta}_n)^\top$, $\boldsymbol{M} := \mathrm{diag}(m_1, \ldots, m_n) > \boldsymbol{0}$, $\boldsymbol{\Gamma} := \mathrm{diag}(\gamma_1, \ldots, \gamma_n) > \boldsymbol{0}$, and $\boldsymbol{0}_{n \times 1}$ denotes the $n \times 1$ vector of zeros. For the second order Kuramoto model, the state space is the product of cylinders $\mathbb{T}^n \times \mathbb{R}^n$.

We address the following problem:

> synthesize feedback controller $\boldsymbol{u}$ that transfers the stochastic state of (3.16) or (3.18) from a prescribed initial to a prescribed terminal joint probability distribution over a given finite time horizon, say $t \in [0, T]$.

This fits in the research theme of designing state feedback for dynamically reshaping (as opposed to simply mitigating) uncertainties [31, 98] subject to networked Kuramoto oscillator dynamics. As such, both first and second order Kuramoto oscillator models are ubiquitous across physical, biological and engineering systems, see e.g., [99, 100].

Notice that while the uncontrolled dynamics in (3.16) has gradient drift, the same in (3.18) has mixed conservative-dissipative drift. A consequence is that unlike (3.16), the stochastic process induced by (3.18), is *not* reversible and its infinitesimal generator is hypoelliptic [101]. This makes the analysis and feedback synthesis for (3.18) even more challenging than (3.16). For the controlled dynamics (3.16), the measure $\mu^{\boldsymbol{u}}$ is supported over the state space $\mathbb{T}^n$. Likewise, for (3.18), the measure $\mu^{\boldsymbol{u}}$ is supported over $\mathbb{T}^n \times \mathbb{R}^n$.

3.4.1 Related literature and novelty of our work

While there exists a significant literature on the dynamics and control of Kuramoto oscillators in general [102–108], the *stochastic control* of Kuramoto oscillators remains under-investigated. Ref. [109] considered global asymptotic phase agreement and frequency synchronization in almost sure sense.

In the physics literature, several studies [110–112] analyze the distributional dynamics associated with the Kuramoto oscillators. However, these studies consider the *univariate* distributional dynamics arising from the *mean-field limit*, i.e., by abstracting

the dynamical interaction in the *infinite population* ($n \to \infty$) regime. In comparison, the perspective and approach taken in this paper are significantly different because we focus on the dynamics of *joint* probability distribution supported over the states of a *finite population* of oscillators. This is particularly relevant for engineering applications such as power systems, where a network of finitely many generators (often modeled as second order nonuniform Kuramoto oscillators) and loads (often modeled as first order nonuniform Kuramoto oscillators) interact together with their controlled stochastic dynamics, see e.g., [106]. Well-known techniques such as the Kron reduction [113] allow transcribing such networked system in the form (3.18) with all-to-all connection topology. Despite the engineering relevance, research on the multivariate distributional dynamics for a finite population of nonuniform Kuramoto oscillators is scant.

From a methodological standpoint, we recast the problem of minimum effort feedback steering of distributions subject to (3.16) or (3.18), as an instance of generalized Schrödinger bridge problem – a topic undergoing rapid development [32, 82] in the systems-control community. In [114], a similar approach was taken to realize feedback steering toward the invariant distribution of an uncontrolled oscillator dynamics. Building on our prior work [38], here we focus on finite horizon steering between two arbitrary compactly supported joint state probability distributions subject to (3.16) or (3.18). However, for our controlled Kuramoto dynamics, it will turn out that the algorithmic approach proposed in [38] will no longer apply and we will introduce new ideas for the same.

3.4.2 Optimal steering of distributions

We consider the stochastic optimal control problem over a prescribed time horizon $[0, T]$, given by

$$\inf_{\boldsymbol{u} \in \mathcal{U}} \mathbb{E}_{\mu^{\boldsymbol{u}}} \left[\int_0^T \|\boldsymbol{u}\|_2^2 \, \mathrm{d}t \right] \tag{3.19}$$

subject to *either*

$$(3.16), \ \boldsymbol{\theta}(t = 0) \sim \mu_0 \text{ (given)}, \ \boldsymbol{\theta}(t = T) \sim \mu_T \text{ (given)},$$

or

$$(3.18), \ \begin{pmatrix} \boldsymbol{\theta}(t = 0) \\ \boldsymbol{\omega}(t = 0) \end{pmatrix} \sim \mu_0 \text{ (given)}, \ \begin{pmatrix} \boldsymbol{\theta}(t = T) \\ \boldsymbol{\omega}(t = T) \end{pmatrix} \sim \mu_T \text{ (given)},$$

where μ_0, μ_T denote the joint state probability measures at $t = 0$ and $t = T$, respectively. In (3.19), the set of feasible controls $\mathcal{U}$ comprises of the finite energy inputs over the time horizon $[0, T]$.

Assuming the absolute continuity of the joint probability measure $\mu^{\boldsymbol{u}}$ for all times, we write $\mathrm{d}\mu^{\boldsymbol{u}}(\boldsymbol{x}, t) = \rho^{\boldsymbol{u}}(\boldsymbol{x}, t)\mathrm{d}\boldsymbol{x}$ and hereafter consider the associated joint PDF $\rho^{\boldsymbol{u}}(\boldsymbol{x}, t)$. Problem (3.19) can then be recast as

$$\inf_{(\rho^{\boldsymbol{u}}, \boldsymbol{u})} \int_0^T \int_{\mathcal{X}} \|\boldsymbol{u}(\boldsymbol{x}, t)\|_2^2 \, \rho^{\boldsymbol{u}}(\boldsymbol{x}, t) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t \tag{3.20}$$

subject to either

$$\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} = -\nabla_{\boldsymbol{\theta}} \cdot \left(\rho^{\boldsymbol{u}} (\boldsymbol{S}\boldsymbol{u} - \nabla_{\boldsymbol{\theta}} V) \right) + \langle \boldsymbol{D}, \mathbf{Hess}(\rho^{\boldsymbol{u}}) \rangle, \tag{3.21a}$$

$$\text{or } \frac{\partial \rho^{\boldsymbol{u}}}{\partial t} = \nabla_{\boldsymbol{\omega}} \cdot \left(\rho^{\boldsymbol{u}} \left(\boldsymbol{M}^{-1} \nabla_{\boldsymbol{\theta}} V(\boldsymbol{\theta}) + \boldsymbol{M}^{-1} \boldsymbol{\Gamma} \boldsymbol{\omega} - \boldsymbol{M}^{-1} \boldsymbol{S} \boldsymbol{u} \right.$$
$$\left. + \boldsymbol{M}^{-1} \boldsymbol{D} \boldsymbol{M}^{-1} \nabla_{\boldsymbol{\omega}} \log \rho^{\boldsymbol{u}} \right) - \langle \boldsymbol{\omega}, \nabla_{\boldsymbol{\theta}} \rho^{\boldsymbol{u}} \rangle, \tag{3.21b}$$

where the diffusion matrix $\boldsymbol{D} := \boldsymbol{S}\boldsymbol{S}^{\top}$, and $\rho^{\boldsymbol{u}}(\boldsymbol{x}, t = 0) = \rho_0$ (given), $\rho^{\boldsymbol{u}}(\boldsymbol{x}, t = T) = \rho_T$

(given).

For the first order Kuramoto oscillators, we have $\boldsymbol{x} := \boldsymbol{\theta}$, $\mathcal{X} := \mathbb{T}^n$, and for the second order Kuramoto oscillators, we have $\boldsymbol{x} := (\boldsymbol{\theta}, \boldsymbol{\omega})^\top$, $\mathcal{X} := \mathbb{T}^n \times \mathbb{R}^n$. The constraints (3.21a) and (3.21b) are the controlled Fokker-Planck or Kolmogorov's forward (FPK) PDEs corresponding to (3.16) and (3.18), respectively.

Endpoint PDFs

In this work, we suppose that the endpoint joint PDFs ρ_0, ρ_T are supported on compact subsets of $\mathcal{X}$. For instance, when $\mathcal{X} = \mathbb{T}^n$, one may model ρ_0, ρ_T as multivariate von Mises PDFs [115, 116] supported on $\mathbb{T}^n$:

$$\rho_k(\boldsymbol{\theta}) = \frac{1}{Z_k} \exp\left(\langle \boldsymbol{\kappa}_k, \cos(\boldsymbol{\theta} - \boldsymbol{m}_k) \rangle + \frac{1}{2} \langle \sin(\boldsymbol{\theta} - \boldsymbol{m}_k), \boldsymbol{\Lambda}_k \sin(\boldsymbol{\theta} - \boldsymbol{m}_k) \rangle \right), \quad k \in \{0, T\},$$

(3.22)

where the parameters are the mean vectors $\boldsymbol{m}_0, \boldsymbol{m}_T \in \mathbb{T}^n$, the concentration vectors $\boldsymbol{\kappa}_0, \boldsymbol{\kappa}_T \in \mathbb{R}^n_{\geq 0}$, and the symmetric matrices[1] $\boldsymbol{\Lambda}_0, \boldsymbol{\Lambda}_T \in \mathbb{S}^n$ having zero diagonal entries. In (3.22), $\sin(\cdot)$ and $\cos(\cdot)$ denote the elementwise sines and cosines, respectively. The normalization constants Z_0, Z_T in (3.22) depend on the respective concentration vector and symmetric matrix parameters.

The nonnegative entries of the concentration vectors $\boldsymbol{\kappa}_0, \boldsymbol{\kappa}_T$ admit a natural interpretation: zero concentration vectors represent uniform distribution over $\mathbb{T}^n$. Large positive entries promote a higher concentration around the corresponding mean components. When $\boldsymbol{\Lambda}$ is a zero matrix, then multivariate von Mises PDF can be written as the product of univariate von Mises PDFs, see e.g., [117, Ch. 3].

When $\mathcal{X} = \mathbb{T}^n \times \mathbb{R}^n$, we suppose that for $k \in \{0, T\}$, the $\boldsymbol{\omega}$ marginals of ρ_k have

---

[1]We use $\mathbb{S}^n$ to denote the set of $n \times n$ real symmetric matrices.

compact supports $\Omega_k \subset \mathbb{R}^n$, and thus the joints ρ_k are supported on compact subsets of $\mathcal{X}$.

3.4.3 Existence and uniqueness of solution

First order case

From (3.17), we observe that $V \in C^2(\mathbb{T}^n)$, which allows us [118, Ch. 1.2] to conclude that the transition probability kernels associated with (3.16) remain continuous for all $t \geq 0$. Furthermore, the endpoint PDFs having compact supports imply ρ_0, ρ_T are positive over their respective supports. Thus, following [38, Appendix E], the transition probability kernels associated with (3.16) also remain positive for all $t \in [0, T]$.

The continuity and positivity of the transition probability kernels associated with (3.16), together guarantee [81, Sec. 10], [119, Thm. 3.2] the existence-uniqueness for the solution of the variational problem (3.20) subject to (3.21a) and the endpoint PDF constraints.

Second order case

That the transition probability kernels remain positive, is ensured per the compactness assumption of the endpoint joint PDFs' supports together with the maximum principle for parabolic PDEs.

Showing that the transition probability kernels also remain continuous for all times, in this case, reduces to showing three conditions: (i) $V \in C^2(\mathbb{T}^n)$, (ii) $\inf V > -\infty$, and (iii) uniform boundedness of the Hessian: $\|\mathbf{Hess}(V)\|_2 \leq c$ for some $c > 0$ that does not depend on $\boldsymbol{\theta}$; see e.g., [120, Theorem 7], [121, Theorem 5]. The satisfaction of the conditions (i)-(ii) are immediate. The induced 2-norm of $\mathbf{Hess}(V)$ is upper

bounded by

$$\sum_{i<j} k_{ij} \cos\left(\theta_i - \theta_j - \varphi_{ij}\right) \le \sum_{i<j} k_{ij},$$

and since $k_{ij} \ge 0$ for all $i, j \in [n]$ and there exists $i, j \in [n]$ such that $k_{ij} > 0$, therefore, (iii) also holds.

As in the first order case, the continuity and positivity of the transition probability kernels, together guarantee the existence-uniqueness of the solution of (3.20) subject to (3.21b) and the endpoint PDF constraints.

In the following Section, we express the solutions of (3.20) in terms of the so-called Schrödinger factors for both first and second order controlled Kuramoto dynamics.

3.4.4 Optimal solutions and Schrödinger factors

First order case

Since $\boldsymbol{S}$ is not identity, the strengths of the process noise acting along the components of (3.16) are nonuniform. To account this anisotropic noise, we consider an invertible linear map $\boldsymbol{\theta} \mapsto \boldsymbol{\xi} := \boldsymbol{S}^{-1}\boldsymbol{\theta}$, which by Itô's lemma [93, Ch.4.2], results in the following SDE for the transformed state vector $\boldsymbol{\xi}$:

$$\mathrm{d}\boldsymbol{\xi} = \left(\boldsymbol{u} - \boldsymbol{\Upsilon}\nabla_{\boldsymbol{\xi}}\tilde{V}(\boldsymbol{\xi})\right)\mathrm{d}t + \sqrt{2}\,\mathrm{d}\boldsymbol{w} \tag{3.23}$$

where the matrix $\boldsymbol{\Upsilon} := \left(\prod_{i=1}^{n} \sigma_i^2\right)\boldsymbol{S}^{-2} = \operatorname{diag}\left(\prod_{j\neq i} \sigma_j^2\right) \succ \boldsymbol{0}$, and the potential

$$\tilde{V}(\boldsymbol{\xi}) := \left(\frac{1}{2}\sum_{i<j} k_{ij}\left(1 - \cos(\sigma_i\xi_i - \sigma_j\xi_j - \varphi_{ij})\right) - \sum_{i=1}^{n} \sigma_i P_i \xi_i\right) \Bigg/ \left(\prod_{i=1}^{n} \sigma_i^2\right).$$

In this new state coordinate, the problem (3.20) subject to (3.21a) and the endpoint PDF constraints, takes the form

$$\inf_{(\tilde{\rho}^{\boldsymbol{u}}, \boldsymbol{u})} \int_0^T \int_{\mathcal{X}} \|\boldsymbol{u}(\boldsymbol{\xi}, t)\|_2^2\, \tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, t)\, \mathrm{d}\boldsymbol{\xi}\, \mathrm{d}t \tag{3.24a}$$

$$\frac{\partial \tilde{\rho}^{\boldsymbol{u}}}{\partial t} = -\nabla_{\boldsymbol{\xi}} \cdot (\tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{u} - \boldsymbol{\Upsilon} \nabla_{\boldsymbol{\xi}} \tilde{V})) + \Delta_{\boldsymbol{\xi}} \tilde{\rho}^{\boldsymbol{u}}, \tag{3.24b}$$

$$\tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, 0) = \rho_0(\boldsymbol{S}\boldsymbol{\xi}) \left(\prod_{i=1}^{n} \sigma_i \right), \quad \tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, T) = \rho_T(\boldsymbol{S}\boldsymbol{\xi}) \left(\prod_{i=1}^{n} \sigma_i \right). \tag{3.24c}$$

Applying Proposition 3.1 and Theorem 3.1 to (3.24), we derive a boundary-coupled system of linear PDEs for the function pair $(\varphi(t, \boldsymbol{\xi}), \hat{\varphi}(t, \boldsymbol{\xi}))$, given by[2]

$$\frac{\partial \hat{\varphi}}{\partial t} = \nabla_{\boldsymbol{\xi}} \cdot (\hat{\varphi} \boldsymbol{\Upsilon} \nabla_{\boldsymbol{\xi}} \tilde{V}) + \Delta_{\boldsymbol{\xi}} \hat{\varphi}, \tag{3.25a}$$

$$\frac{\partial \varphi}{\partial t} = \langle \nabla_{\boldsymbol{\xi}} \varphi, \boldsymbol{\Upsilon} \nabla_{\boldsymbol{\xi}} \tilde{V} \rangle - \Delta_{\boldsymbol{\xi}} \varphi, \tag{3.25b}$$

$$\hat{\varphi}_0(\boldsymbol{\xi}) \varphi_0(\boldsymbol{\xi}) = \tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, 0) = \rho_0(\boldsymbol{S}\boldsymbol{\xi}) \left(\prod_{i=1}^{n} \sigma_i \right), \tag{3.25c}$$

$$\hat{\varphi}_T(\boldsymbol{\xi}) \varphi_T(\boldsymbol{\xi}) = \tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, T) = \rho_T(\boldsymbol{S}\boldsymbol{\xi}) \left(\prod_{i=1}^{n} \sigma_i \right), \tag{3.25d}$$

whose solution recovers the optimal decision variables $(\tilde{\rho}^{\mathrm{opt}}, \boldsymbol{u}^{\mathrm{opt}})$ for problem (3.24) via the mapping

$$\tilde{\rho}^{\mathrm{opt}}(\boldsymbol{\xi}, t) = \hat{\varphi}(\boldsymbol{\xi}, t) \varphi(\boldsymbol{\xi}, t), \quad \boldsymbol{u}^{\mathrm{opt}}(\boldsymbol{\xi}, t) = \nabla_{\boldsymbol{\xi}} \log \varphi(\boldsymbol{\xi}, t). \tag{3.26}$$

We refer to the function pair $(\varphi, \hat{\varphi})$ as the *Schrödinger factors*, so named since their product gives $\tilde{\rho}^{\mathrm{opt}}$ at all times, i.e., $(\varphi, \hat{\varphi})$ comprise a factorization of $\tilde{\rho}^{\mathrm{opt}}$. The optimally controlled joint state PDF ρ^{opt} for (3.20) is then obtained as

$$\rho^{\mathrm{opt}}(\boldsymbol{\theta}, t) = \tilde{\rho}^{\mathrm{opt}} \left(\boldsymbol{S}^{-1} \boldsymbol{\theta}, t \right) / \left(\prod_{i=1}^{n} \sigma_i \right).$$

The optimal control in original coordinates is $\boldsymbol{S} \nabla_{\boldsymbol{\theta}} \log \varphi(\boldsymbol{S}^{-1}\boldsymbol{\theta}, t)$.

Now the matter boils down to solving (3.25). Notice that (3.25a)-(3.25b) are the *uncontrolled* forward and backward Kolmogorov PDEs, respectively, associated with (3.23). It is tempting to apply further change of variables

$$t \mapsto s := 1 - t, \quad \varphi(\boldsymbol{\xi}, t) \mapsto p(\boldsymbol{\xi}, s)$$

---

[2]For notational ease, let $\hat{\varphi}_0 := \hat{\varphi}(\boldsymbol{\xi}, 0)$, $\hat{\varphi}_T := \hat{\varphi}(\boldsymbol{\xi}, T)$, $\varphi_0 := \varphi(\boldsymbol{\xi}, 0)$, and $\varphi_T := \varphi(\boldsymbol{\xi}, T)$.

proposed in [38, Theorem 3] to (3.25), for transforming (3.25a)-(3.25b) into forward-forward PDEs as in [38, equation (33)]. When possible, this strategy allows using a single FPK initial value problem (IVP) solver to set up a provably contractive fixed point recursion for computing the pair $(\hat{\varphi}_0, \varphi_1)$. In our case, the aforesaid mappings transform (3.25b) to

$$\frac{\partial p}{\partial s} = \nabla_{\boldsymbol{\xi}} \cdot \left(p \nabla_{\boldsymbol{\xi}} \tilde{V}\right) + \Delta_{\boldsymbol{\xi}} p + \underbrace{p \left\langle \nabla_{\boldsymbol{\xi}} \tilde{V}, (\boldsymbol{I}_n - \boldsymbol{\Upsilon}) \nabla_{\boldsymbol{\xi}} \tilde{V} \right\rangle + \left\langle \nabla_{\boldsymbol{\xi}} p, (\boldsymbol{I}_n - \boldsymbol{\Upsilon}) \nabla_{\boldsymbol{\xi}} \tilde{V} \right\rangle}_{\text{extra terms compared to [38, equation (33b)]}}, \quad (3.27)$$

which has some additional terms compared to [38, equation (33b)].

An interesting observation follows: (3.27) becomes the same forward Kolmogorov operator as in (3.25a) only if $\boldsymbol{\Upsilon}$ equals identity. Consequently, the Algorithm COMPUTEFACTORSSBP proposed in [38, Sec. V.D] that uses a single FPK IVP solver, cannot be applied to our case. We need two different solvers for (3.25a) and (3.25b). To solve (3.25a), we implement a modified form of the PROXRECUR algorithm given in [5, Sec. III.B] with the following distance functional which is a weighted version of the squared 2-Wasserstein distance between a pair of joint PDFs $\tilde{\varrho}, \tilde{\varrho}_{k-1}$, given by

$$W_{\boldsymbol{\Upsilon}}^2 \left(\tilde{\varrho}, \tilde{\varrho}_{k-1}\right) := \inf_{\pi \in \Pi(\tilde{\varrho}, \tilde{\varrho}_{k-1})} \int_{\mathbb{T}^{2n}} \left\langle \boldsymbol{\theta} - \bar{\boldsymbol{\theta}}, \boldsymbol{\Upsilon}(\boldsymbol{\theta} - \bar{\boldsymbol{\theta}}) \right\rangle \mathrm{d}\pi(\boldsymbol{\theta}, \bar{\boldsymbol{\theta}}), \quad (3.28)$$

where $\Pi\left(\tilde{\varrho}, \tilde{\varrho}_{k-1}\right)$ is the set of joint PDFs supported on $\mathbb{T}^{2n}$, having finite second moments, with given marginals $\tilde{\varrho}, \tilde{\varrho}_{k-1}$.

To solve (3.25b), we employ the Feynman-Kac formula [122] as detailed in Algorithm. 1.

Second order case

In the second order Kuramoto model (3.18), the anisotropy in process noise affects the last n components. Motivated by our treatment in the first order case, we now

consider the invertible linear map

$$\begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{\omega} \end{pmatrix} \mapsto \begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{pmatrix} := \left(\boldsymbol{I}_2 \otimes (\boldsymbol{M}\boldsymbol{S}^{-1}) \right) \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{\omega} \end{pmatrix} \tag{3.29}$$

which by Itô's Lemma [93, Ch.4.2], results in the following SDE for the transformed state vector $(\boldsymbol{\xi}, \boldsymbol{\eta})^\top$:

$$\begin{pmatrix} \mathrm{d}\boldsymbol{\xi} \\ \mathrm{d}\boldsymbol{\eta} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\eta} \\ \boldsymbol{u} - \widetilde{\boldsymbol{\Upsilon}} \nabla_{\boldsymbol{\xi}} U(\boldsymbol{\xi}) - \nabla_{\boldsymbol{\eta}} F(\boldsymbol{\eta}) \end{pmatrix} \mathrm{d}t + \begin{pmatrix} \boldsymbol{0}_{n \times n} \\ \boldsymbol{I}_n \end{pmatrix} \mathrm{d}\boldsymbol{w} \tag{3.30}$$

where the matrix $\widetilde{\boldsymbol{\Upsilon}} := \left(\prod_{i=1}^n \sigma_i^2 m_i^{-2} \right) \boldsymbol{M}\boldsymbol{S}^{-2}$, and the potentials

$$U(\boldsymbol{\xi}) := \left(\frac{1}{2} \sum_{i<j} k_{ij} \left(1 - \cos\left(\frac{\sigma_i}{m_i}\xi_i - \frac{\sigma_j}{m_j}\xi_j - \varphi_{ij} \right) \right) - \right.$$
$$\left. \sum_{i=1}^n \frac{\sigma_i}{m_i} P_i \xi_i \right) \left(\prod_{i=1}^n \left(\frac{m_i}{\sigma_i} \right)^2 \right),$$
$$F(\boldsymbol{\eta}) := \frac{1}{2} \langle \boldsymbol{\eta}, \boldsymbol{S}^{-1} \boldsymbol{\Gamma} \boldsymbol{\eta} \rangle.$$

In this new state coordinate, the problem (3.20) subject to (3.21b) and the endpoint PDF constraints, takes the form

$$\inf_{(\tilde{\rho}^{\boldsymbol{u}}, \boldsymbol{u})} \int_0^T \int_{\mathcal{X}} \|\boldsymbol{u}(\boldsymbol{\xi}, \boldsymbol{\eta}, t)\|_2^2 \, \tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, \boldsymbol{\eta}, t) \, \mathrm{d}\boldsymbol{\xi} \, \mathrm{d}\boldsymbol{\eta} \, \mathrm{d}t \tag{3.31a}$$

$$\frac{\partial \tilde{\rho}^{\boldsymbol{u}}}{\partial t} = \nabla_{\boldsymbol{\eta}} \cdot \left(\tilde{\rho}^{\boldsymbol{u}} \left(-\boldsymbol{u} + \widetilde{\boldsymbol{\Upsilon}} \nabla_{\boldsymbol{\xi}} U(\boldsymbol{\xi}) + \nabla_{\boldsymbol{\eta}} F(\boldsymbol{\eta}) \right) \right)$$
$$- \langle \boldsymbol{\eta}, \nabla_{\boldsymbol{\xi}} \tilde{\rho}^{\boldsymbol{u}} \rangle + \Delta_{\boldsymbol{\eta}} \tilde{\rho}^{\boldsymbol{u}}, \tag{3.31b}$$

$$\tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, \boldsymbol{\eta}, 0) = \rho_0 \left(\left(\boldsymbol{I}_2 \otimes \boldsymbol{S}\boldsymbol{M}^{-1} \right) \begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{pmatrix} \right) \left(\prod_{i=1}^n \frac{\sigma_i^2}{m_i^2} \right),$$

$$\tilde{\rho}^{\boldsymbol{u}}(\boldsymbol{\xi}, \boldsymbol{\eta}, T) = \rho_T \left(\left(\boldsymbol{I}_2 \otimes \boldsymbol{S}\boldsymbol{M}^{-1} \right) \begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{pmatrix} \right) \left(\prod_{i=1}^n \frac{\sigma_i^2}{m_i^2} \right). \tag{3.31c}$$

Applying Proposition 3.1 and Theorem 3.1 to (3.31), we next derive a boundary-

coupled system of linear PDEs akin to (3.25), for the Schrödinger factors $(\varphi, \hat{\varphi})$, given by

$$\frac{\partial \hat{\varphi}}{\partial t} = -\langle \boldsymbol{\eta}, \nabla_{\boldsymbol{\xi}} \hat{\varphi} \rangle + \nabla_{\boldsymbol{\eta}} \cdot \left(\hat{\varphi} (\widetilde{\boldsymbol{\Upsilon}} \nabla_{\boldsymbol{\xi}} \boldsymbol{U}(\boldsymbol{\xi}) + \nabla_{\boldsymbol{\eta}} \widetilde{\boldsymbol{f}}(\boldsymbol{\eta}))) \right) + \Delta_{\boldsymbol{\eta}} \hat{\varphi}, \tag{3.32a}$$

$$\frac{\partial \varphi}{\partial t} = -\langle \boldsymbol{\eta}, \nabla_{\boldsymbol{\xi}} \varphi \rangle + \langle \widetilde{\boldsymbol{\Upsilon}} \nabla_{\boldsymbol{\xi}} \boldsymbol{U}(\boldsymbol{\xi}) + \nabla_{\boldsymbol{\eta}} \widetilde{\boldsymbol{f}}(\boldsymbol{\eta}), \nabla_{\boldsymbol{\eta}} \varphi \rangle - \Delta_{\boldsymbol{\eta}} \varphi, \tag{3.32b}$$

$$\hat{\varphi}_0(\boldsymbol{\xi}) \varphi_0(\boldsymbol{\xi}) = \rho_0 \left(\left(\boldsymbol{I}_2 \otimes \boldsymbol{S} \boldsymbol{M}^{-1} \right) \begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{pmatrix} \right) \left(\prod_{i=1}^{n} \frac{\sigma_i^2}{m_i^2} \right), \tag{3.32c}$$

$$\hat{\varphi}_T(\boldsymbol{\xi}) \varphi_T(\boldsymbol{\xi}) = \rho_T \left(\left(\boldsymbol{I}_2 \otimes \boldsymbol{S} \boldsymbol{M}^{-1} \right) \begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{pmatrix} \right) \left(\prod_{i=1}^{n} \frac{\sigma_i^2}{m_i^2} \right). \tag{3.32d}$$

The optimal decision variables $(\tilde{\rho}^{\mathrm{opt}}, \boldsymbol{u}^{\mathrm{opt}})$ for problem (3.31) are obtained from the solution of (3.32) as

$$\tilde{\rho}^{\mathrm{opt}}(\boldsymbol{\xi}, \boldsymbol{\eta}, t) = \hat{\varphi}(\boldsymbol{\xi}, \boldsymbol{\eta}, t) \varphi(\boldsymbol{\xi}, \boldsymbol{\eta}, t), \tag{3.33}$$

$$\boldsymbol{u}^{\mathrm{opt}}(\boldsymbol{\xi}, \boldsymbol{\eta}, t) = \nabla_{\begin{pmatrix} \xi \\ \eta \end{pmatrix}} \log \varphi(\boldsymbol{\xi}, \boldsymbol{\eta}, t).$$

The optimally controlled joint state PDF ρ^{opt} for (3.20) in the second order case, is then obtained as

$$\rho^{\mathrm{opt}}(\boldsymbol{\theta}, \boldsymbol{\omega}, t) = \tilde{\rho}^{\mathrm{opt}} \left(\left(\boldsymbol{I}_2 \otimes \boldsymbol{M} \boldsymbol{S}^{-1} \right) \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{\omega} \end{pmatrix}, t \right) \left(\prod_{i=1}^{n} \frac{m_i^2}{\sigma_i^2} \right).$$

The optimal control in the original coordinates is

$$\left(\boldsymbol{I}_2 \otimes \boldsymbol{S} \boldsymbol{M}^{-1} \right) \nabla_{\boldsymbol{\theta}} \log \varphi \left(\left(\boldsymbol{I}_2 \otimes \boldsymbol{M} \boldsymbol{S}^{-1} \right) \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{\omega} \end{pmatrix}, t \right).$$

As in the first order case, our algorithmic approach is to solve (3.32) via fixed point recursion over the pair $(\hat{\varphi}_0, \varphi_T)$ that is provably contractive w.r.t. the Hilbert's projective metric. In particular, to solve the backward Kolmogorov PDE (3.32b), we

use the Feynman-Kac formula. The PDE (3.32a) is the so-called *kinetic Fokker-Planck equation* [123, p. 40], and to solve the same, we propose a modified version of the proximal recursion proposed in [5, Sec. V.B]. Our modification concerns with the distance functional in the proximal recursion, i.e., we consider the following analogue of (3.28):

$$\widetilde{W}^2_{h,\widetilde{\boldsymbol{\Upsilon}}}\left(\tilde{\varrho}, \tilde{\varrho}_{k-1}\right) :=$$
$$\inf_{\pi \in \Pi(\tilde{\varrho}, \tilde{\varrho}_{k-1})} \int_{\mathbb{T}^{2n} \times \mathbb{R}^{2n}} s_{h,\widetilde{\boldsymbol{\Upsilon}}}\left(\boldsymbol{\xi}, \boldsymbol{\eta}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\eta}}\right) \, \mathrm{d}\pi\left(\boldsymbol{\xi}, \boldsymbol{\eta}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\eta}}\right), \qquad (3.34)$$

where $h > 0$ is the step-size in proximal recursion, $\Pi\left(\tilde{\varrho}, \tilde{\varrho}_{k-1}\right)$ is the set of joint probability measures over the product space $\mathbb{T}^{2n} \times \mathbb{R}^{2n}$ that have finite second moments and marginal PDFs $\tilde{\varrho}, \tilde{\varrho}_{k-1}$. The "ground cost" in (3.34) is

$$s_{h,\widetilde{\boldsymbol{\Upsilon}}}\left(\boldsymbol{\xi}, \boldsymbol{\eta}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\eta}}\right) :=$$
$$\left\langle \left(\bar{\boldsymbol{\eta}} - \boldsymbol{\eta} + h\widetilde{\boldsymbol{\Upsilon}}\nabla U(\boldsymbol{\xi})\right), \widetilde{\boldsymbol{\Upsilon}}^{-1}\left(\bar{\boldsymbol{\eta}} - \boldsymbol{\eta} + h\widetilde{\boldsymbol{\Upsilon}}\nabla U(\boldsymbol{\xi})\right) \right\rangle$$
$$+ 12 \left\langle \left(\frac{\bar{\boldsymbol{\xi}} - \boldsymbol{\xi}}{h} - \frac{\bar{\boldsymbol{\eta}} - \boldsymbol{\eta}}{h}\right), \widetilde{\boldsymbol{\Upsilon}}^{-1}\left(\frac{\bar{\boldsymbol{\xi}} - \boldsymbol{\xi}}{h} - \frac{\bar{\boldsymbol{\eta}} - \boldsymbol{\eta}}{h}\right) \right\rangle. \qquad (3.35)$$

In the next Section, we bring these ideas together to detail the algorithms for computing the optimal solutions in both the first and second order cases.

3.4.5 Algorithms

In the following, we first outline the proximal algorithm for solving the forward Kolmogorov PDEs (3.25a) and (3.32a). Then we presents the Feynman–Kac algorithm for solving the backward Kolmogorov PDEs (3.25b) and (3.32b). Finally, we summarizes the overall algorithm for solving (3.25) and (3.32).

Proximal algorithm

For solving IVPs involving the forward Kolmogorov PDEs (3.25a) and (3.32a), we employ proximal recursions over the space of measurable positive functions over discrete time $t_{k-1} := (k-1)h$ where the index $k \in \mathbb{N}$, and $h > 0$ is (here constant) time step-size. These recursions are of the form

$$\hat{\phi}_k = \text{prox}^d_{h\hat{\phi}}\left(\hat{\phi}_{k-1}\right) := \arg\inf_{\hat{\phi}} \frac{1}{2}\left(d\left(\hat{\phi}, \hat{\phi}_{k-1}\right)\right)^2 + h\Psi\left(\hat{\phi}\right) \qquad (3.36)$$

where $\hat{\phi}_{k-1}(\cdot) := \hat{\phi}(\cdot, t_{k-1})$, $d(\cdot, \cdot)$ is a distance-like functional, Ψ is an energy-like functional, and $\hat{\phi}_0$ is suitable initial condition. The recursion (3.36) reads as "the proximal operator of the functional $h\Psi$ w.r.t. the distance d". The pair (d, Ψ) is constructed in a way that the sequence of functions $\{\hat{\phi}_{k-1}\}_{k\in\mathbb{N}}$ generated by (3.36) satisfies $\hat{\phi}_{k-1}(\cdot) \to \hat{\varphi}(\cdot, t)$ in $L^1(\mathcal{X})$ as $h \downarrow 0$.

For (3.25a), we set $d \equiv W_{\Upsilon}$ given by (3.28), and $\Psi(\hat{\phi}) \equiv \int_{\prod_{i=1}^n [0, 2\pi/\sigma_i)} \left(\tilde{V} + \log\hat{\phi}\right)\hat{\phi}d\boldsymbol{\xi}$. For (3.32a), we set $d \equiv W_{h, \tilde{\Upsilon}}$ given by (3.34), and

$$\Psi(\hat{\phi}) \equiv \int_{\left(\prod_{i=1}^n [0, 2\pi m_i/\sigma_i)\right)\times\mathbb{R}^n} \left(U + F + \log\hat{\phi}\right)\hat{\phi}\,\mathrm{d}\boldsymbol{\xi}\,\mathrm{d}\boldsymbol{\eta}$$

. For a discussion on the convergence guarantees and on implementation of these proximal updates via fixed point recursions, we refer the readers to [5]; see also [38, Sec. V-B,C].

Feynman-Kac algorithm

For solving IVPs involving the backward Kolmogorov PDEs (3.25b) and (3.32b), we employ the Feynman-Kac path integral formulation [93, Ch. 8.2], [124], [125, Ch. 3.3], [126]. We consider a general form of backward PDEs as

$$\frac{\partial\varphi}{\partial t} = \mathcal{L}_{\text{backward}}\varphi = \langle\nabla_{\tilde{\boldsymbol{x}}}\varphi, \underline{f}\rangle + \text{trace}\left(G(t, \tilde{\boldsymbol{x}})^{\top}\mathbf{Hess}(\varphi)G(t, \tilde{\boldsymbol{x}})\right) \qquad (3.37)$$

47

To find the solution of the above PDE at $t_0 = \tau$, we can use the following algorithm based on Feynman–Kac formula.

Algorithm 1 Feynman-Kac Algorithm for solving backward PDE IVP

1: **procedure** FEYNMANKAC$(\varphi_T(\tilde{\boldsymbol{x}}_T), \tilde{\boldsymbol{x}}_T, T, \tilde{\boldsymbol{x}}_\tau, \tau, \underline{f}, G,$
   $N_r, \text{nSample}, \dim, h, \lambda, \varsigma)$

2:     $\tilde{\boldsymbol{x}}_r \leftarrow [\mathbf{0}_{\text{nSample} \times \dim \times N_r}]$ ▷ initialize

3:     $\varphi_r \leftarrow [\mathbf{0}_{\text{nSample} \times N_r}]$

4:     numSteps= $\frac{T-\tau}{h}$

5:     **for** $i \leftarrow 1$ to N_r **do**

6:         $\tilde{\boldsymbol{x}}_{\text{temp}} = [\boldsymbol{x}_\tau, \mathbf{0}_{nSample \times \dim \times \text{numSteps}}]$

7:         **for** $k \leftarrow 1$ to numSteps **do**

8:             $\tilde{\boldsymbol{x}}_{\text{temp}}(:,:,k+1) = \tilde{\boldsymbol{x}}_{\text{temp}}(:,:,k) + h\underline{f}(k, \tilde{\boldsymbol{x}}(:,:,k)) + G(k, \tilde{\boldsymbol{x}}_k)(w_{k+1} - w_k)$
   ▷ Euler-Maruyama update

9:         **end for**

10:        $\tilde{\boldsymbol{x}}_r(:,:,i) \leftarrow \tilde{\boldsymbol{x}}_{\text{temp}}(:,:,\text{numSteps}+1)$

11:        $\varphi_r(:,i) \leftarrow \text{ElasticNet}(\varphi_T(\tilde{\boldsymbol{x}}_T), \tilde{\boldsymbol{x}}_T,$
   $\tilde{\boldsymbol{x}}_r(:,:,i), \lambda, \varsigma)$

12:    **end for**

13:    **return** $\varphi(\tau, \tilde{\boldsymbol{x}}_\tau) \leftarrow \frac{1}{N_r} \sum_{i=1}^{N_r} \varphi_r(:,i)$

14: **end procedure**

In line 11 of the above algorithm, we implement an elastic net regression with the ς_1 as the L_1 norm term and ς_2 as the L_2 norm term coefficient to estimate the value of φ_T at $\tilde{\boldsymbol{x}}_r(:,:,i)$ by knowing the values of $\varphi_T(\tilde{\boldsymbol{x}}_T)$ and $\tilde{\boldsymbol{x}}_T$. We use the ADMM algorithm [58, Ch. 5] to implement the ElasticNet regression.

Overall algorithm

We now bring together the ideas from scattered point cloud-based computation and Feynman Kac formula and outline the overall algorithm to solve the SBPs for the first and second-order Kuramoto oscillators. We perform a recursion over the pair$(\hat{\varphi}_0(\tilde{\boldsymbol{x}}), \varphi_1(\tilde{\boldsymbol{x}}))$, and the computational steps for the same are:

Step 1. Guess $\varphi_0(\tilde{\boldsymbol{x}})$ (everywhere positive).

Step 2. Compute $\hat{\varphi}_0(\tilde{\boldsymbol{x}}) = \tilde{\rho}_0(\tilde{\boldsymbol{x}})/\varphi_0(\tilde{\boldsymbol{x}})$

Step 3. Solve IVP (3.25a) or (3.32a) till $t = 1$ to obtain $\hat{\varphi}_1(\tilde{\boldsymbol{x}})$.

Step 4. Compute $\varphi_1(\tilde{\boldsymbol{x}}) = \tilde{\rho}_1(\tilde{\boldsymbol{x}})/\hat{\varphi}_1(\tilde{\boldsymbol{x}})$

Step 5. Use Algorithm 1 to calculate $\varphi_0(\tilde{\boldsymbol{x}}) = \varphi(0, \tilde{\boldsymbol{x}})$ for (3.25b) or (3.32b).

Step 6. Repeat until the pair $(\hat{\varphi}_0(\tilde{\boldsymbol{x}}), \varphi_1(\tilde{\boldsymbol{x}}))$ has converged

Step 7. Compute the Schrodinger factors at any time t, i.e., the pair $(\hat{\varphi}(t, \tilde{\boldsymbol{x}}), \varphi(t, \tilde{\boldsymbol{x}}))$ using the IVPs (3.25) and (3.32).

Step 8. Compute the discrete optimal pair $(\tilde{\rho}^{\mathrm{opt}}, \boldsymbol{u}^{\mathrm{opt}})$ from (3.26) or (3.33), with the obtained $(\hat{\varphi}(t, \tilde{\boldsymbol{x}}), \varphi(t, \tilde{\boldsymbol{x}}))$.

Step 9. Bring back the variables to the original coordinate $(\boldsymbol{\theta}, \boldsymbol{\omega})$.

3.4.6 Numerical example

First order case

We consider an instance of (3.16) with $n = 2$ oscillators, i.e., $\boldsymbol{\theta} \in \mathcal{X} = \mathbb{T}^2$. We generated the following parameters uniformly random from the respective intervals: $P_i \in [0, 10]$, $\sigma_i \in [1, 5]$ for $i = 1, 2$, and $k_{12} \in [0.7, 1.2]$, $\varphi_{12} \in [0, \frac{\pi}{2})$.

We set the final time $T = 1$, and ρ_0, ρ_T as in (3.22) (see Fig. 3.1) with $\boldsymbol{\kappa}_0 = (1, 1)^{\top}$, $\boldsymbol{\kappa}_T = (0.01, 0.01)^{\top}$, $\boldsymbol{m}_0 = (\pi, \pi)^{\top}$, $\boldsymbol{m}_T = (0, 0)^{\top}$, $\boldsymbol{\Lambda}_0 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, $\boldsymbol{\Lambda}_T = 0.1\boldsymbol{\Lambda}_0$. We solve (3.25) following the steps in Section 3.4.5. Specifically, we solve the backward PDE (3.25b) via Algorithm 1 with parameters $N_r = 100, h = 0.1, \mathrm{nSample} = 441, \lambda_1 = \lambda_2 = 0.01$. To solve the forward PDE (3.25a), we used the PROXRECUR algorithm from [5, Sec. III-B.1] with algorithmic parameters $\varepsilon = 1, \beta = 0.1, \delta = 0.1, L = 300$ together with the modifications mentioned in Section 3.4.5.

Fig. 3.2a shows the snapshots of the *optimally controlled joint* $\rho^{\text{opt}}(\boldsymbol{\theta}, t)$ steering ρ_0 to ρ_T over time horizon $[0, 1]$. Fig. 3.2b shows the snapshots of the uncontrolled joint $\rho^{\text{unc}}(\boldsymbol{\theta}, t)$ from the same ρ_0. The snapshots of the magnitude of optimal control are depicted in Fig. 3.2c.

(a) $t = 0$ (b) $t = 1$
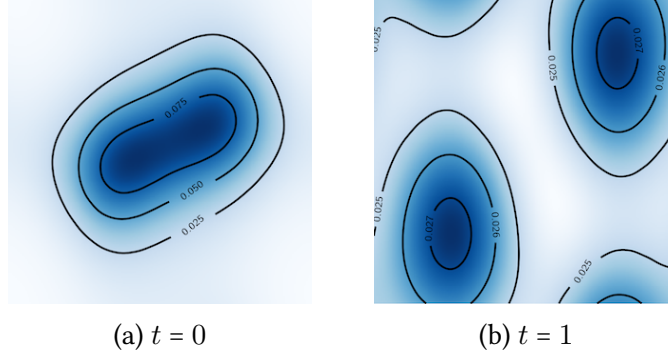
Figure 3.1: Endpoint von Mises $\boldsymbol{\theta}$ PDFs over $\mathbb{T}^2$.

Second order case

We next consider an instance of (3.18) with $n = 2$ oscillators, i.e., $(\boldsymbol{\theta}, \boldsymbol{\omega}) \in \mathcal{X} = \mathbb{T}^2 \times \mathbb{R}^2$. We set $T = 1$, and use $\{P_i, \sigma_i\}_{i=1,2}$, k_{12}, φ_{12} as in the first order case above. We consider the initial joint PDF $\rho_0(\boldsymbol{\theta}, \boldsymbol{\omega}) \equiv \bar{\rho}_0(\boldsymbol{\theta}) \times \text{Unif}\left([0, 0.2]^2\right)$, and the terminal joint PDF $\rho_T(\boldsymbol{\theta}, \boldsymbol{\omega}) \equiv \bar{\rho}_T(\boldsymbol{\theta}) \times \text{Unif}\left([0, 0.2]^2\right)$ where the $\boldsymbol{\theta}$ marginals $\bar{\rho}_0, \bar{\rho}_T$ are identical to ρ_0, ρ_T in the first order case, and $\text{Unif}(\cdot)$ denotes the uniform PDF. In other words, the endpoint joint PDFs ρ_0, ρ_T are supported on the compact set $\mathbb{T}^2 \times [0, 0.2]^2$.

We solve (3.32) using the same computational set up as in the subsection above except that the PROXRECUR algorithm [5, Sec. III-B.1] for solving the forward PDE (3.32a) is suitably modified as mentioned in Section 3.4.5.

Fig. 3.3a shows the snapshots of the $\boldsymbol{\theta}$ marginals of the optimally controlled joints $\rho^{\text{opt}}(\boldsymbol{\theta}, \boldsymbol{\omega}, t)$. Fig. 3.3b shows the $\boldsymbol{\theta}$ marginal snapshots of the uncontrolled joints. The snapshots of the magnitude of optimal control are depicted in Fig. 3.3c. A comparison
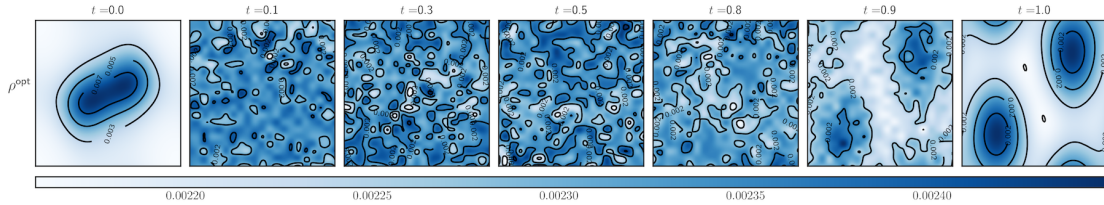
(a) Contour plots of the optimally controlled state PDFs $\rho^{\mathrm{opt}}(\boldsymbol{\theta}, t)$ over $\mathbb{T}^2$. Each subplot is a snapshot in time $t \in [0, 1]$.
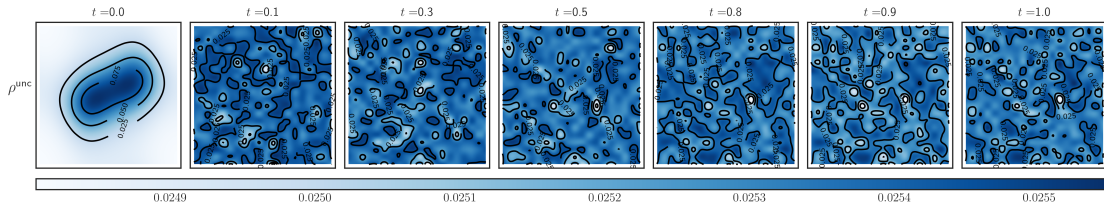
(b) Contour plots of the uncontrolled state PDFs $\rho^{\mathrm{unc}}(\boldsymbol{\theta}, t)$ over $\mathbb{T}^2$. Each subplot is a snapshot in time $t \in [0, 1]$.
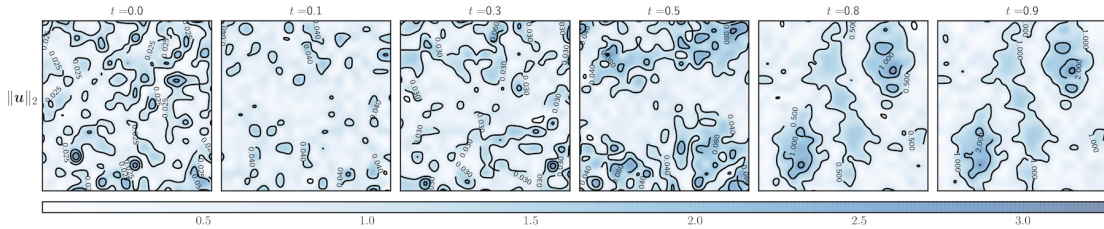
(c) Contour plots of the 2-norm magnitude of the optimal control over $\mathbb{T}^2$. Each subplot is a snapshot in time $t \in [0, 1]$.

Figure 3.2: Simulation results for the optimal PDF steering for the *first order* Kuramoto oscillators over $t \in [0, 1]$. The color denotes the value of the plotted variable; see colorbar (dark hue = high, light hue = low).
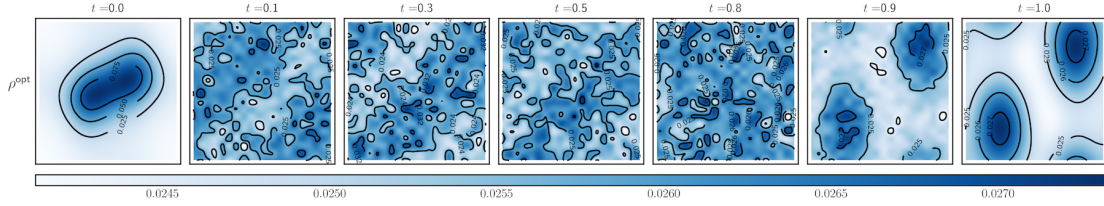
(a) Contour plots of $\boldsymbol{\theta}$ marginals of the optimally controlled joints $\rho^{\mathrm{opt}}(\boldsymbol{\theta}, \boldsymbol{\omega}, t)$ over $\mathbb{T}^2$. Each subplot is a snapshot in time $t \in [0, 1]$.
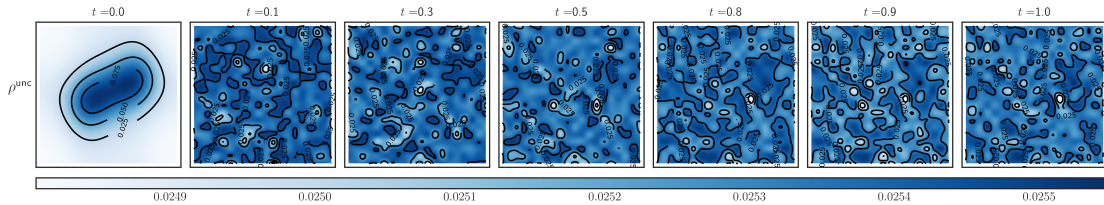
(b) Contour plots of $\boldsymbol{\theta}$ marginals of the uncontrolled joints $\rho^{\mathrm{unc}}(\boldsymbol{\theta}, \boldsymbol{\omega}, t)$ over $\mathbb{T}^2$. Each subplot is a snapshot in time $t \in [0, 1]$.
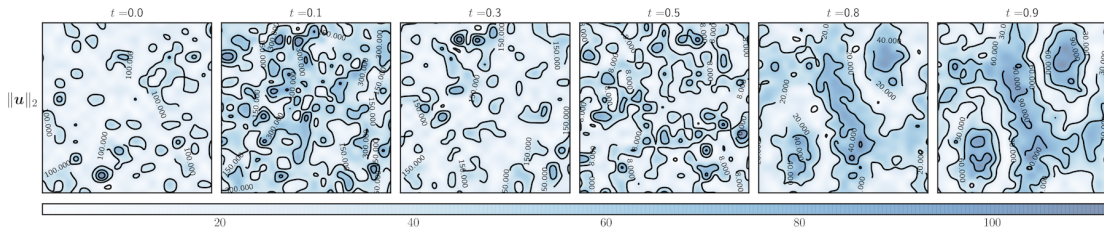
(c) Contour plots of the 2-norm magnitude of the optimal control over $\mathbb{T}^2$. Each subplot is a snapshot in time $t \in [0, 1]$.

Figure 3.3: Simulation results for the optimal PDF steering for the *second order* Kuramoto oscillators over $t \in [0, 1]$. The color denotes the value of the plotted variable; see colorbar (dark hue = high, light hue = low).

of Figs. 3.2c and 3.3c reveals that in the second order case, the prior dynamics being mixed conservative-dissipative, the optimal control entails forcing that is about two orders of magnitude above the same for the first order case. Fig. 3.4 shows four optimally controlled sample paths on $\mathbb{T}^2$ for the first order case (*in red*) and another four for the second order case (*in blue*).
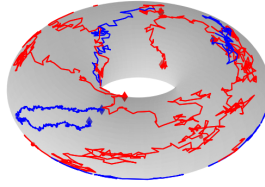
Figure 3.4: The optimally controlled first order (*in red*) and second order (*in blue*) Kuramoto sample paths on $\mathbb{T}^2$ for the numerical simulation in Section 3.4. The *circled* and *diamond markers* denote the initial and terminal angular coordinates, respectively.

Order parameter

In the coupled oscillator context, a measure of synchronization, or lack thereof, is the order parameter $r := \frac{1}{n}\left|\sum_{j=1}^{n}\exp(\iota\theta_j)\right| \in [0,1]$ where $\iota := \sqrt{-1}$; see e.g., [102, Sec. 3.2]. For instance, $r = 0$ implies lack of synchrony, and $r = 1$ implies synchronized motion in the state space. Fig. 3.5 shows the snapshots of the order parameter PDFs under optimal control for the aforesaid numerical simulation. As the optimal control steers the stochastic state $\boldsymbol{\theta}$ from unimodal to bimodal, the r PDFs in Fig. 3.5 slightly flatten over this transfer horizon and develop a secondary peak around $r = 0.5$.
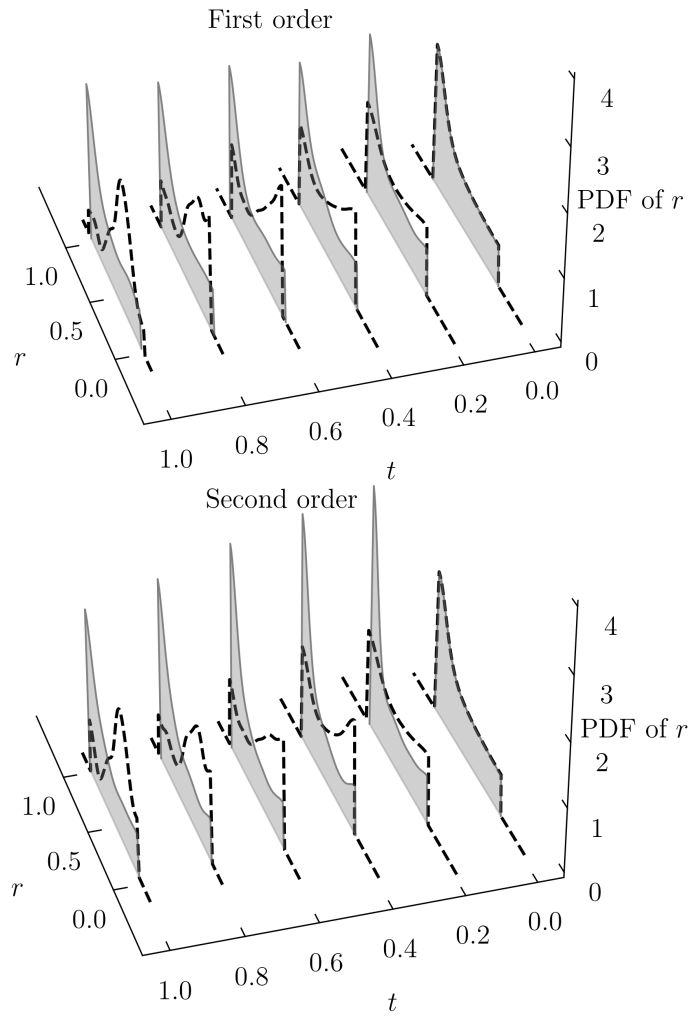
Figure 3.5: PDFs of r for the numerical simulation in Section 3.4.

4 | Control Non-affine Generalized Schrödinger Bridge

4.1 Problem Formulation

Motivated by feedback control of colloidal self-assembly (SA), in this chapter, we consider a control non-affine system for the *minimum energy* generalized SBP. Specifically, we aim to solve (2.13) given that:

$$q(\cdot) \equiv 0, \quad r(\cdot) \equiv \frac{1}{2} \| \cdot \|_2^2,$$

where the associated drift and diffusion coefficient pair $(\boldsymbol{f}, \boldsymbol{g})$ in (2.14), can either be given as an Itô SDE that models the dynamics of colloidal SA pathways, as detailed in [127, 128] or they not available from first principle physics, but are instead approximated in a data-driven manner (e.g., using neural networks). This is particularly challenging since both $\boldsymbol{f}$ and $\boldsymbol{g}$ can, in general, be non-autonomous (i.e., may have explicit t-dependence) as well as nonlinear in state, and non-affine in control.

Specifically, we focus on learning the solution of the nonlinear stochastic optimal control problems over a given fixed time horizon $[0, T]$ of the form

$$\inf_{\boldsymbol{u} \in \mathcal{U}} \quad \mathbb{E}_{\mu^{\boldsymbol{u}}} \left[\int_0^T \frac{1}{2} \| \boldsymbol{u}(t, \boldsymbol{x}) \|_2^2 \, \mathrm{d}t \right] \tag{4.1a}$$

$$\text{subject to} \quad \mathrm{d}\boldsymbol{x} = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u})\mathrm{d}t + \sqrt{2}\, \boldsymbol{g}(t, \boldsymbol{x}, \boldsymbol{u})\mathrm{d}\boldsymbol{w}, \tag{4.1b}$$

$$\boldsymbol{x}(t = 0) \sim \mu_0 \text{ (given)}, \ \boldsymbol{x}(t = T) \sim \mu_T \text{ (given)}, \tag{4.1c}$$

where μ_0, μ_T denote the prescribed probability measures over the state space $\mathcal{X} \subseteq \mathbb{R}^n$ at $t = 0$ and $t = T$, respectively. The constraint in (4.1b) is a controlled Itô SDE with the state vector $\boldsymbol{x} \in \mathcal{X}$, the control vector $\boldsymbol{u} \in \mathbb{R}^m$, and the standard Wiener process $\boldsymbol{w} \in \mathbb{R}^p$. For the solution to the SDE (4.1b) to be for colloidal SA systems, the state

55

vector $\boldsymbol{x}$ represents suitable order parameters. The *drift coefficient* $\boldsymbol{f}$ is a vector field given by mapping $\boldsymbol{f} : [0, T] \times \mathcal{X} \times \mathcal{U} \mapsto \mathbb{R}^n$, and the *diffusion coefficient* $\boldsymbol{g}$ is a matrix field given by mapping $\boldsymbol{g} : [0, T] \times \mathcal{X} \times \mathcal{U} \mapsto \mathbb{R}^{n \times p}$. For the SDE solutions to be well-posed, we will detail suitable smoothness assumptions on $\boldsymbol{f}$ and $\boldsymbol{g}$.

Associated with the diffusion coefficient $\boldsymbol{g}$, is a *diffusion tensor* $\boldsymbol{G} := \boldsymbol{g}\boldsymbol{g}^\top \in \mathbb{S}^n_+$, which being an outer product, is a symmetric positive semidefinite matrix field $\boldsymbol{G} : [0, T] \times \mathcal{X} \times \mathcal{U} \mapsto \mathbb{S}^n_+$. In (4.1a), we suppose that the set of admissible controls $\mathcal{U}$ comprises of finite energy Markovian inputs within a prescribed time horizon, i.e.,

$$\mathcal{U} := \left\{ \boldsymbol{u} : [0, T] \times \mathcal{X} \mapsto \mathbb{R}^m \mid \langle \boldsymbol{u}, \boldsymbol{u} \rangle < \infty \right\}, \tag{4.2}$$

The superscript $\boldsymbol{u}$ in $\mu^{\boldsymbol{u}}$ indicates that the joint measure depends on the choice of control $\boldsymbol{u}$. Thus, the objective in (4.1a) is to minimize the control effort in steering the state statistics from μ_0 to μ_T under a prespecified time horizon and controlled stochastic dynamics constraints, over all admissible control policies $\boldsymbol{u}(t, \boldsymbol{x})$ in $\mathcal{U}$.

In feedback control of colloidal SA systems, the objective generally is to design control policies that steer the system from an initial disordered stochastic state to a desired terminal ordered crystalline stochastic state [129, 130]. These stochastic states are naturally encoded in terms of suitable order parameters. As such, formulation (4.1) is particularly appealing in this context because it allows for directly shaping the multivariate distribution of order parameters via optimal control synthesis. The drift and diffusion coefficients $\boldsymbol{f}, \boldsymbol{g}$ in equation (4.1b) allow for the representation of the free energy landscape, which is crucial for circumventing kinetic traps or local minima when directing the system towards a desired end state, typically a global minimum within the solution space.

4.1.1   Smoothness of the learnt drift and diffusion coefficients

For an admissible Markovian policy $u(t, x) \in \mathcal{U}$, we assume that the coefficients f and g satisfy

(A1) non-explosion and Lipschitz conditions: there exist constants c_1, c_2 such that

$$\|f(t, x, u(t, x))\|_2 + \|g(t, x, u(t, x))\|_2 \leq c_1 \left(1 + \|x\|_2\right),$$

and that

$$\|f(t, x, u(t, x)) - f(t, \widetilde{x}, u(t, \widetilde{x}))\|_2 \leq c_2 \|x - \widetilde{x}\|_2$$

for all $x, \widetilde{x} \in \mathcal{X}$, $t \in [0, T]$;

(A2) uniformly lower bounded diffusion: there exists constant c_3 such that the diffusion tensor $G = gg^\top$ satisfies

$$\langle x, G\left(t, x, u(t, x)\right) x \rangle \geq c_3 \|x\|_2^2$$

for all $t \in [0, T]$.

The assumption **(A1)** guarantees [93, p. 66] existence-uniqueness for the sample path of the SDE (2.13b). The assumptions **(A1), (A2)** together guarantee [94, Ch. 1] that the generator associated with (2.13b) yields absolutely continuous probability measures μ^u for all $t > 0$ provided the prescribed initial probability measure $\mu_0 := \mu^u(t = 0, x)$ is absolutely continuous.

In this work, we assume that the given endpoint measures μ_0, μ_T are absolutely continuous, i.e., $\mu_0 = \rho_0(x)\mathrm{d}x$, $\mu_T = \rho_T(x)\mathrm{d}x$ where ρ_0, ρ_T are the corresponding endpoint joint state PDFs. If the solution for (4.1) exists, then under the stated regularity assumptions on f and g, the corresponding controlled measure $\mu^u(t, x)$ will remain absolutely continuous with $\mathrm{d}\mu^u(t, x) = \rho^u(t, x)\mathrm{d}x$ for admissible $u \in \mathcal{U}$. We next discuss reformulating (4.1) in terms of the controlled joint state PDF ρ^u.

4.1.2 PDF steering problem

With the assumptions in Section 4.1.1, the GSBP (4.1) can be rewritten as a state PDF steering problem:

$$\inf_{(\rho^{\boldsymbol{u}}, \boldsymbol{u})} \int_0^T \int_{\mathcal{X}} \frac{1}{2} \|\boldsymbol{u}(t, \boldsymbol{x})\|_2^2 \, \rho^{\boldsymbol{u}}(t, \boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t \tag{4.3a}$$

$$\text{subject to} \quad \frac{\partial \rho^{\boldsymbol{u}}}{\partial t} = -\nabla_{\boldsymbol{x}} \cdot (\rho^{\boldsymbol{u}} \boldsymbol{f}) + \langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle, \tag{4.3b}$$

$$\rho^{\boldsymbol{u}}(0, \boldsymbol{x}) = \rho_0, \quad \rho^{\boldsymbol{u}}(T, \boldsymbol{x}) = \rho_T. \tag{4.3c}$$

The constraint (4.3b) is the controlled Fokker-Planck-Kolmogorov (FPK) PDE which governs the flow of the joint state PDF $\rho^{\boldsymbol{u}}$ associated with the SDE (2.13b). For a derivation of (4.3b) from (2.13b), see e.g., [131, Prop. 3.3]. For the term $\langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle$ in (4.3b), note from (1.18) that

$$\langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle = \sum_{i,j} \frac{\partial^2}{\partial x_i \partial x_j} \left(G_{ij}(t, \boldsymbol{x}, \boldsymbol{u}(t, \boldsymbol{x})) \rho^{\boldsymbol{u}}(t, \boldsymbol{x}) \right).$$

The boundary conditions (4.3c) at $t = 0$ and $t = T$ involve the prescribed initial and terminal joint state PDFs ρ_0 and ρ_T, respectively.

We note that when $\boldsymbol{f} \equiv \boldsymbol{u}$, $\boldsymbol{g}$ (and hence $\boldsymbol{G}$) $\equiv \boldsymbol{I}_n$, then (4.3b) reduces to the controlled heat PDE, and problem (4.3) reduces to the classical SBP. Furthermore, when $\boldsymbol{f} \equiv \boldsymbol{u}$, $\boldsymbol{g} \equiv \boldsymbol{0}$, then (4.3b) reduces to the Liouville PDE [76] for integrator dynamics $\dot{\boldsymbol{x}} = \boldsymbol{u}$, and problem (4.3) reduces to the dynamic OMT.

Remark 4.1. *To better understand the correspondence between (4.1) and (4.3), notice that (4.3a) is simply a re-writing of (4.1a) by "opening up" the expectation operator w.r.t. the controlled state probability measure $\mathrm{d}\mu^{\boldsymbol{u}}(t, \boldsymbol{x}) = \rho^{\boldsymbol{u}}(t, \boldsymbol{x})\mathrm{d}\boldsymbol{x}$. The constraint (4.3b) is the PDF dynamics induced by the sample path dynamics (2.13b). Intuitively, the term $\frac{1}{2}\|\boldsymbol{u}\|_2^2 \rho^{\boldsymbol{u}}\mathrm{d}\boldsymbol{x}$ is a generalized kinetic energy, and the state-time integral (4.3a) encodes*

total control effort over the finite horizon $[0, T]$. The FPK PDE (4.3b) is a continuity equation expressing the conservation of probability mass under the drift coefficient $\boldsymbol{f}$ and the diffusion coefficient $\boldsymbol{g}$ (thus the diffusion tensor $\boldsymbol{G}$).

4.1.3   Existence and uniqueness of solution

Under the assumptions stated already in Section 4.1.1, the controlled PDF $\rho^{\boldsymbol{u}}$ exists for $\boldsymbol{u} \in \mathcal{U}$. For the existence-uniqueness of solution for the variational problem (4.3), we further assume that

(A3) the PDF $\rho^{\boldsymbol{u}}$ remains positive and continuous for all $t \in [0, T]$.

Then, following [119, Thm. 3.2], [132], problem (4.3) is guaranteed to admit a unique solution; see also [133, Sec. 10].

We next deduce the first order optimality conditions for the GSBP (4.3) in the form of a coupled system of $m+2$ PDEs with boundary conditions, where m is the number of control inputs. With respect to the existing literature on the conditions of optimality for GSBPs, this system of PDEs for non-affine control is the most general, and is a new result.

4.1.4   Conditions for optimality

We start with the Lagrangian associated with the GSBP (4.3):

$$
\begin{aligned}
\mathcal{L}(\rho^{\boldsymbol{u}}, \boldsymbol{u}, \psi) := \int_0^T \int_{\mathcal{X}} \Big\{ & \frac{1}{2} \|\boldsymbol{u}(t, \boldsymbol{x})\|_2^2 \rho^{\boldsymbol{u}}(t, \boldsymbol{x}) + \psi(t, \boldsymbol{x}) \times \\
& \left(\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x}} . (\rho^{\boldsymbol{u}} \boldsymbol{f}) - \langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle \right) \Big\} \mathrm{d}\boldsymbol{x} \, \mathrm{d}t
\end{aligned}
\tag{4.4}
$$

where $\psi(t, \boldsymbol{x})$ is a $C^2([0, T]; \mathcal{X})$ Lagrange multiplier. Let

$$
\mathcal{P}_{0T}(\mathcal{X}) := \Big\{ \rho(t, \boldsymbol{x}) \geq 0 \mid \int_{\mathcal{X}} \rho \mathrm{d}\boldsymbol{x} = 1,
$$

$$\rho(t=0, \boldsymbol{x}) = \rho_0, \quad \rho(t=T, \boldsymbol{x}) = \rho_T\}. \tag{4.5}$$

Performing the unconstrained minimization of the Lagrangian $\mathcal{L}$ over $\mathcal{P}_{0T}(\mathcal{X}) \times \mathcal{U}$, where $\mathcal{U}$ is given in (4.2), we get the following result.

Theorem 4.1. *(Optimal control and optimal state PDF)*

The pair $(\rho_{\mathrm{opt}}^{\boldsymbol{u}}(t, \boldsymbol{x}), \boldsymbol{u}_{\mathrm{opt}}(t, \boldsymbol{x}))$ that solves (4.3), must satisfy the following system of $m+2$ coupled PDEs:

$$\frac{\partial \psi}{\partial t} = \frac{1}{2}\|\boldsymbol{u}_{\mathrm{opt}}\|_2^2 - \langle \nabla_{\boldsymbol{x}}\psi, \boldsymbol{f}\rangle - \langle \boldsymbol{G}, \mathbf{Hess}(\psi)\rangle, \tag{4.6a}$$

$$\frac{\partial \rho_{\mathrm{opt}}^{\boldsymbol{u}}}{\partial t} = -\nabla_{\boldsymbol{x}} \cdot (\rho_{\mathrm{opt}}^{\boldsymbol{u}}\boldsymbol{f}) + \langle \mathbf{Hess}, \boldsymbol{G}\rho_{\mathrm{opt}}^{\boldsymbol{u}}\rangle, \tag{4.6b}$$

$$\boldsymbol{u}_{\mathrm{opt}} = \nabla_{\boldsymbol{u}_{\mathrm{opt}}}\left(\langle \nabla_{\boldsymbol{x}}\psi, \boldsymbol{f}\rangle + \langle \boldsymbol{G}, \mathbf{Hess}(\psi)\rangle\right), \tag{4.6c}$$

with boundary conditions

$$\rho_{\mathrm{opt}}^{\boldsymbol{u}}(0, \boldsymbol{x}) = \rho_0, \quad \rho_{\mathrm{opt}}^{\boldsymbol{u}}(T, \boldsymbol{x}) = \rho_T, \tag{4.7}$$

where $\psi(t, \boldsymbol{x})$ is a $C^2([0,T]; \mathcal{X})$ value function.

Remark 4.2. *The conditions of optimality (4.6) relate the primal variables $(\rho_{\mathrm{opt}}^{\boldsymbol{u}}(t, \boldsymbol{x}), \boldsymbol{u}_{\mathrm{opt}}(t, \boldsymbol{x}))$ with the dual variable (i.e., Lagrange multiplier) $\psi(t, \boldsymbol{x})$. Specifically, the HJB PDE (4.6a) and the controlled FPK PDE (4.6b) express the dual and the primal feasibility, respectively. The optimal control policy equation (4.6c) expresses the primal-dual relation.*

The system (4.6) for our non-affine GSBP comprises of $m+2$ coupled PDEs in three unknowns: $\rho_{\mathrm{opt}}, \boldsymbol{u}_{\mathrm{opt}}, \psi$, where m is the number of control inputs. This is because (4.6c) itself gives m PDEs coupled in ψ and $\boldsymbol{u}_{\mathrm{opt}}$, while the equation pair (4.6a)-(4.6b) are coupled in $\rho_{\mathrm{opt}}^{\boldsymbol{u}}, \boldsymbol{u}_{\mathrm{opt}}, \psi$. To the best of the authors' knowledge, this is the first work to derive and numerically solve the conditions of optimality for control-non-

affine GSBP in multi-dimensional state space. As detailed in the Theorem 4.1, the resulting system of equations is fundamentally different from the control-affine SBPs in that the optimal control is no longer an explicit functional of the (sub)gradient of the associated value function solving a HJB PDE. As a result, existing approaches from the literature such as the Hopf-Cole transform [134, 135] followed by a contractive fixed point recursion [38, 136], or Feynman-Kac path integral techniques That we used in **Chapter 3**, cannot be used to numerically solve our system of equations. Leveraging recent advances in NNs, we propose a computational framework to learn the solution for this system of $m + 2$ PDEs and boundary conditions.

Table 4.1 summarizes how known results in the literature can be recovered as special cases of (4.6).

4.2 Case Study: Colloidal Self-assembly Sample Path Model

In this section, we consider a one-dimensional special case of (4.1b), as represented by the Itô SDE given below:

$$\mathrm{d}\langle C_6 \rangle = D_1(\langle C_6 \rangle, \pi)\, \mathrm{d}t + \sqrt{2D_2(\langle C_6 \rangle, \pi)}\, \mathrm{d}w \tag{4.8}$$

where t denotes time, and the state variable $\langle C_6 \rangle \in [0, 6]$ is an order parameter denoting the average number of hexagonally close packed particles around each particle. This specific model, derived from [127, 128], provides a focused exploration of SA phenomena in colloidal systems through a mathematical lens. We consider the dynamics (4.8) over a fixed time horizon $[0, T]$. The control input $u := \pi(\langle C_6 \rangle, t) \in$

$\mathbb{R}$ denotes electric field voltage [137] that results from a Markovian control policy
$\pi : [0,6] \times [0,T] \mapsto \mathbb{R}$, and w denotes the standard Wiener process in $\mathbb{R}$. In (4.8),
the functionals $D_1(\cdot,\cdot)$ and $D_2(\cdot,\cdot)$ are referred to as the *drift* and the *diffusion land-
scapes*, respectively. In the colloidal SA context, both D_1, D_2 are nonlinear in state and
non-affine in control.

Typically, the drift landscape D_1 is expressed in terms of the so-called *free energy
landscape F* and the *diffusion landscape D_2*, as

$$D_1(\langle C_6 \rangle, \pi) = \frac{\partial}{\partial \langle C_6 \rangle} D_2(\langle C_6 \rangle, \pi)$$
$$- \frac{D_2(\langle C_6 \rangle, \pi)}{k_\mathrm{B}\,\theta} \frac{\partial}{\partial \langle C_6 \rangle} F(\langle C_6 \rangle, \pi), \tag{4.9}$$

where the Boltzmann constant $k_\mathrm{B} = 1.38066 \times 10^{-23}$ Joules per Kelvin, and θ denotes
a suitable temperature in Kelvin. In Section 4.2.5, we will give illustrative numerical
results for specific choices of the diffusion and the free energy landscapes. For an
admissible Markovian policy $\pi(\cdot,t)$, we assume that the landscapes D_1, D_2 satisfy the
assumptions **(A1)** and **(A2)** presented in Section 4.1.1.

4.2.1 Related works

Colloidal SA is the process by which discrete components (e.g., micro-/nano-particles
in solution) spontaneously organize into an ordered state [138]. The spontaneous self-
organization central to colloidal SA enables "bottom-up" materials synthesis, which
can allow for manufacturing advanced, highly-ordered crystalline structures in an
inherently parallelizable and cost-effective manner [130, 139]. The fact that colloidal
SA can begin with micro- and/or nano-scale building blocks of varying complexity
indicates that this bottom-up engineering approach can be used to synthesize novel
metamaterials with unique optical, electrical, or mechanical properties [130, 139, 140].

Colloidal SA is an inherently stochastic (i.e., random) process prone to kinetic arrest due to particle Brownian motion [130, 139, 141, 142]. This leads to variability in materials manufacturing and possibly high defect rates, which can severely compromise the viability of using colloidal SA to reproducibly manufacture advanced materials. This lack of reproducibility in turn prevents colloidal SA from achieving cost-effective and scalable manufacturing of such materials [130, 139, 143, 144]. Thus, the thermodynamic and kinetic driving forces that govern colloidal SA, will need to be precisely and systematically modulated to consistently and efficiently direct colloidal SA systems towards high-value mass-producible structures and materials.

To more reproducibly drive collodial SA systems towards desired structures, it has been proposed to design a model-based feedback control policy wherein a global actuator (e.g., electric field voltage) is manipulated based on currently available information on the system state and a dynamical system model [127, 128, 145]. Work in [127] presents a model predictive control (MPC) method for controlling colloidal SA. These authors consider the dynamical model for the stochastic colloidal assembly process based on actuator-parameterized Langevin equations. In [128, 145], the system is guided towards the desired highly-ordered structure based on a Markov decision process optimal control policy.

In comparison, the perspective and approach taken in this chapter towards optimal control of a colloidal SA process are significantly different, as we seek to control the time evolution of the joint probability distribution supported over the states of colloidal SA. Our technical approach and contributions are as follows.

(1) We show that the problem of controlling colloidal SA over a finite-time horizon can be naturally formulated as a control non-affine generalized Schrödinger Bridge. The notion of lifting the stochastic control of colloidal SA directly onto

the space of state probability measure as an SBP, is novel.

(2) While there exists a growing literature on numerically solving the SBPs with
nonlinear prior dynamics [38,39,71,91,146], these works leverage specific struc-
tures of the underlying drift and diffusion terms in an SDE. In contrast, the
typical colloidal SA application, as considered here, requires more generic con-
siderations since the drift and diffusion coefficients can be nonlinear w.r.t to the
states as well as non-affine in control. We show that, unlike the existing SBP
conditions of optimality, our setting leads to a system of three coupled PDEs
with endpoint PDF constraints. These three PDEs are the controlled FPK PDE,
the HJB PDE, and a policy PDE.

(3) The resulting system of three coupled PDEs is not amenable to existing compu-
tational approaches for the SBPs, such as the contractive fixed point recursions
on the so-called Schrödinger factors. To address this challenge, we employ the
notion of physics-informed neural networks (PINN) (e.g., [40,41]) to train a deep
neural network approximating the solution of our coupled PDE system and the
boundary conditions.

The distinct feature of the proposed control methodology is that it derives an opti-
mal control policy that steers a given probability distribution of the order parameters
of a colloidal SA system to a desired one over a finite-time horizon. The computation
does not require making parametric approximations of the statistics, such as the Gaus-
sian mixture or exponential family. It also avoids approximating the nonlinearities in
the drift and diffusion *a priori*, e.g., via Taylor series.

Our technical contribution is a new control methodology. To make the exposi-
tion concrete, we use a specific univariate SDE model from the literature [127, 128]

to demonstrate the proof-of-concept for our proposed method. The specificity of the
model is used from the outset as a didactic writing style, even though our proposed
method is general, i.e., *not* contingent on the nonlinearity structure or dimensionality
of the SDE model.

4.2.2   Controlled self-assembly as distribution steering

We propose reformulating the problem of designing a control policy $\pi(\langle C_6 \rangle, t)$ for
the controlled self-assembly subject to (4.8), to that of steering the *statistics* of the
stochastic state $\langle C_6 \rangle$ from a prescribed initial probability measure μ_0 at $t = 0$ to a
prescribed terminal probability measure μ_T at $t = T$. This is motivated by the fact
that $\langle C_6 \rangle \approx 0$ implies disordered crystalline structure while $\langle C_6 \rangle \approx 5.1$ implies an
ordered structure. So steering the stochastic state from disordered to ordered naturally
transcribes to steering a high concentration of probability mass around 0 to the same
around 5.1. Note the target value of $\langle C_6 \rangle$ is 5.1 and not 6 due to edge effects in the
lattice structure.

We emphasize here that we use the term "statistics" in nonparametric sense, i.e.,
we allow arbitrary probability measures μ_0, μ_T supported on the compact set $[0, 6]$,
and ask for provable steering of μ_0 to μ_T via control, not just steering of first few
statistical moments of μ_0 to μ_T such as mean and variance. Even if μ_0, μ_T have finite
dimensional sufficient statistics, the transient $\langle C_6 \rangle$ probability measures induced by
the nonlinear SDE (4.8) for a given control policy π, may not have so. This motivates
formulating the control synthesis as a two point boundary value problem over the
(infinite dimensional) manifold of state probability measures. So, for this case study,
we focus on the nonlinear stochastic optimal control model-based non-affine problems
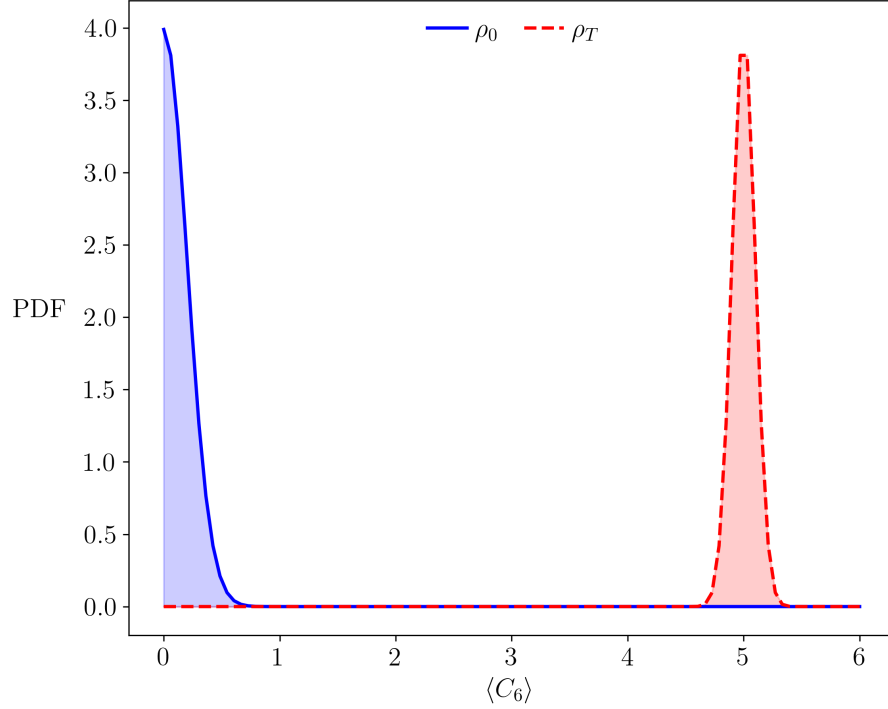
Figure 4.1: The prescribed initial PDF ρ_0 (solid line) at the initial time $t = 0$, and the prescribed terminal PDF ρ_T (dashed line) at the final time $t = T$. Both PDFs are supported over $[0, 6]$, which is the range of values for the state variable $\langle C_6 \rangle$ denoting a crystallinity order parameter. In particular, $\langle C_6 \rangle \approx 0$ implies a disordered state and $\langle C_6 \rangle \approx$ 5-6 implies a highly ordered state.

over a given fixed time horizon $[0, T]$ of the form

$$\inf_{\pi \in \mathcal{U}} \quad \int_0^T \mathbb{E}_{\mu^\pi} \left[\frac{1}{2} \pi^2 \right] \mathrm{d}t \tag{4.10}$$

$$\text{subject to} \quad \mathrm{d}\langle C_6 \rangle = D_1(\langle C_6 \rangle, \pi)\mathrm{d}t + \sqrt{2D_2(\langle C_6 \rangle, \pi)} \, \mathrm{d}w,$$

$$\langle C_6 \rangle(t = 0) \sim \mu_0 \text{ (given)},$$

$$\langle C_6 \rangle(t = T) \sim \mu_T \text{ (given)},$$

where $\mu^\pi \equiv \mu^\pi(\langle C_6 \rangle, t)$ denotes the controlled state probability measure at time t. In other words, we design state feedback for dynamically reshaping uncertainties subject to the dynamical constraint (4.8), endpoint statistical constraints, and the deadline constraint.

Recall that the input is the electrical field voltage, and the minimum effort objective
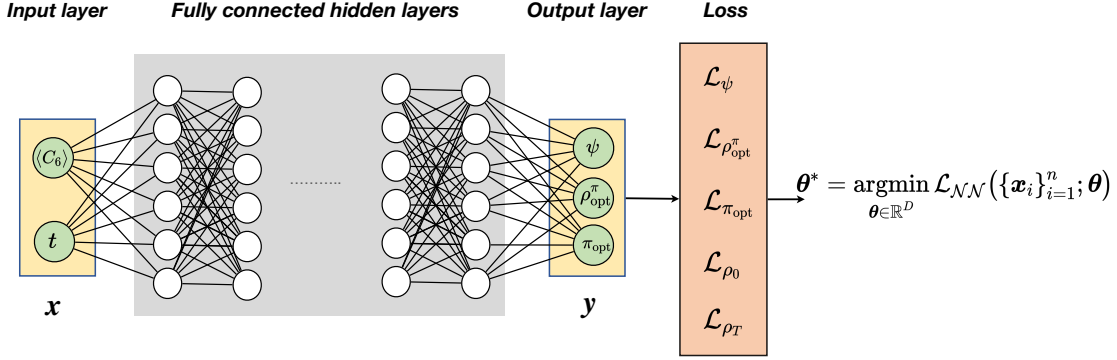
Figure 4.2: The architecture of the physics-informed neural network with the system order parameter and time as the input features $\boldsymbol{x} := (\langle C_6 \rangle, t)$. The output $\boldsymbol{y}$ comprises of the value function, optimally controlled PDF, and optimal control policy, i.e., $\boldsymbol{y} := (\psi, \rho^\pi_{\mathrm{opt}}, \pi_{\mathrm{opt}})$.

is a natural candidate to promote control parsimony.

We suppose that the endpoint probability measures μ_0, μ_T are absolutely continuous with respective PDFs ρ_0, ρ_T; see Fig. 4.1. Then, we can rewrite (4.10) as the variational problem:

$$\inf_{(\rho^\pi, \pi)} \int_0^T \int_{\mathbb{R}} \frac{1}{2} \pi^2(\langle C_6 \rangle, t) \rho^\pi(\langle C_6 \rangle, t) \, \mathrm{d}\langle C_6 \rangle \, \mathrm{d}t \tag{4.11a}$$

$$\text{subject to} \quad \frac{\partial \rho^\pi}{\partial t} = -\frac{\partial}{\partial \langle C_6 \rangle}(D_1 \rho^\pi) + \frac{\partial^2}{\partial \langle C_6 \rangle^2}(D_2 \rho^\pi), \tag{4.11b}$$

$$\rho^\pi(\langle C_6 \rangle, 0) = \rho_0, \quad \rho^\pi(\langle C_6 \rangle, T) = \rho_T, \tag{4.11c}$$

where (4.11b) is the *controlled* FPK PDE. The feasible pair $(\rho^\pi, \pi) \in \mathcal{P}_{0T} \times \mathcal{U}$.

4.2.3 Conditions for optimality

The next theorem is a specialization of Theorem 4.1, from which we derive the first-order conditions for optimality for problem (4.11). These conditions take the form of three coupled PDEs with endpoint boundary conditions while tacitly assuming existence and uniqueness. In Section 4.2.4, we will numerically solve this system via PINN.

Theorem 4.2. *(First order conditions for optimality)*

The pair $\left(\rho_{\mathrm{opt}}^{\pi}(\langle C_6\rangle, t), \pi_{\mathrm{opt}}(\langle C_6\rangle, t)\right)$ that solves (4.11), must satisfy the system of coupled PDEs

$$\frac{\partial \psi}{\partial t} = \frac{1}{2}(\pi_{\mathrm{opt}})^2 - D_1 \frac{\partial \psi}{\partial \langle C_6\rangle} - D_2 \frac{\partial^2 \psi}{\partial \langle C_6\rangle^2}, \tag{4.12a}$$

$$\frac{\partial \rho_{\mathrm{opt}}^{\pi}}{\partial t} = -\frac{\partial}{\partial \langle C_6\rangle}(D_1 \rho_{\mathrm{opt}}^{\pi}) + \frac{\partial^2}{\partial \langle C_6\rangle^2}(D_2 \rho_{\mathrm{opt}}^{\pi}), \tag{4.12b}$$

$$\pi_{\mathrm{opt}}(\langle C_6\rangle, t) = \frac{\partial \psi}{\partial \langle C_6\rangle}\frac{\partial D_1}{\partial \pi_{\mathrm{opt}}} + \frac{\partial^2 \psi}{\partial \langle C_6\rangle^2}\frac{\partial D_2}{\partial \pi_{\mathrm{opt}}}, \tag{4.12c}$$

in unknowns $\left(\psi(\langle C_6\rangle, t), \rho_{\mathrm{opt}}^{\pi}(\langle C_6\rangle, t), \pi_{\mathrm{opt}}(\langle C_6\rangle, t)\right)$ with boundary conditions

$$\rho_{\mathrm{opt}}^{\pi}(\langle C_6\rangle, 0) = \rho_0, \qquad \rho_{\mathrm{opt}}^{\pi}(\langle C_6\rangle, T) = \rho_T. \tag{4.13}$$

4.2.4 Solving the conditions for optimality using PINN

We propose leveraging recent advances in neural network based computational frameworks to jointly learn the solutions of (4.12)-(4.13). In the following, we discuss training of a PINN [40, 41] to numerically solve (4.12)-(4.13), which is a system of three coupled PDEs together with the endpoint PDF boundary conditions.

The proposed architecture of the PINN is shown in Fig. 4.2. In our problem, $\boldsymbol{x} := (\langle C_6\rangle, t)$ comprises of the features given to the DNN, and the DNN output $\boldsymbol{y} := (\psi, \rho_{\mathrm{opt}}^{\pi}, \pi_{\mathrm{opt}})$ comprises of the value function, optimally controlled PDF, and optimal policy. We parameterize the output of the fully connected feed-forward DNN via $\boldsymbol{\theta} \in \mathbb{R}^D$, i.e.,

$$\boldsymbol{y}(\boldsymbol{x}) \approx \mathcal{NN}(\boldsymbol{x}; \boldsymbol{\theta}), \tag{4.14}$$

where $\mathcal{NN}(\cdot; \boldsymbol{\theta})$ denotes the neural network approximant parameterized by $\boldsymbol{\theta}$, and D is the dimension of the parameter space (i.e., the total number of to-be-trained weight,

bias and scaling parameters for the DNN).

The overall loss function for the network, denoted as $\mathcal{L}_{\mathcal{NN}}$, consists of the sum of the losses associated with the three equations in (4.12) and the losses associated with the boundary conditions (4.13). Specifically, let $\mathcal{L}_{\psi}$ be the loss term for the HJB PDE (4.12a), let $\mathcal{L}_{\rho^{\pi}_{\mathrm{opt}}}$ be the loss term for the FPK PDE (4.12b), and let $\mathcal{L}_{\pi_{\mathrm{opt}}}$ be the loss term for the control policy equation (4.12c). Likewise, let $\mathcal{L}_{\rho_0}$ and $\mathcal{L}_{\rho_T}$ denote the loss terms for the corresponding endpoint constraints (4.13). Then

$$\mathcal{L}_{\mathcal{NN}} := \mathcal{L}_{\psi} + \mathcal{L}_{\rho^{\pi}_{\mathrm{opt}}} + \mathcal{L}_{\pi_{\mathrm{opt}}} + \mathcal{L}_{\rho_0} + \mathcal{L}_{\rho_T}, \tag{4.15}$$

where each summand loss term in (4.15) is evaluated on a set of n collocation points $\{\boldsymbol{x}_i\}_{i=1}^n$ in the domain of the feature space $\Omega := [0,6] \times [0,T]$, i.e., $\{\boldsymbol{x}_i\}_{i=1}^n \subset \Omega$, and

$$\mathcal{L}_{\psi} := \frac{1}{n} \sum_{i=1}^n \left(\frac{\partial \psi}{\partial t}\Big|_{\boldsymbol{x}_i} - \frac{1}{2}(\pi_{\mathrm{opt}})^2\Big|_{\boldsymbol{x}_i} + D_1 \frac{\partial}{\partial \langle C_6 \rangle} \psi \Big|_{\boldsymbol{x}_i} + D_2 \frac{\partial^2}{\partial \langle C_6 \rangle^2} \psi \Big|_{\boldsymbol{x}_i} \right)^2,$$

$$\mathcal{L}_{\rho^{\pi}_{\mathrm{opt}}} := \frac{1}{n} \sum_{i=1}^n \left(\frac{\partial \rho^{\pi}_{\mathrm{opt}}}{\partial t}\Big|_{\boldsymbol{x}_i} + \frac{\partial}{\partial \langle C_6 \rangle}(D_1 \rho^{\pi}_{\mathrm{opt}})\Big|_{\boldsymbol{x}_i} - \frac{\partial^2}{\partial \langle C_6 \rangle^2}(D_2 \rho^{\pi}_{\mathrm{opt}})\Big|_{\boldsymbol{x}_i} \right)^2,$$

$$\mathcal{L}_{\pi_{\mathrm{opt}}} := \frac{1}{n} \sum_{i=1}^n \left(\pi_{\mathrm{opt}}\big|_{\boldsymbol{x}_i} - \frac{\partial}{\partial \langle C_6 \rangle} \psi \frac{\partial}{\partial \pi_{\mathrm{opt}}} D_1 \Big|_{\boldsymbol{x}_i} - \frac{\partial^2}{\partial \langle C_6 \rangle^2} \psi \frac{\partial}{\partial \pi_{\mathrm{opt}}} D_2 \Big|_{\boldsymbol{x}_i} \right)^2,$$

$$\mathcal{L}_{\rho_0} := \frac{1}{n} \sum_{i=1}^n \left(\rho^{\pi}_{\mathrm{opt}}(\cdot, t=0)\big|_{\boldsymbol{x}_i} - \rho_0(\cdot)\big|_{\boldsymbol{x}_i} \right)^2,$$

$$\mathcal{L}_{\rho_T} := \frac{1}{n} \sum_{i=1}^n \left(\rho^{\pi}_{\mathrm{opt}}(\cdot, t=T)\big|_{\boldsymbol{x}_i} - \rho_T(\cdot)\big|_{\boldsymbol{x}_i} \right)^2,$$

for each collocation point $\boldsymbol{x}_i$, $i = 1, \cdots, n$. For training the PINN, we minimize the overall loss (4.15) by solving

$$\boldsymbol{\theta}^* = \operatorname*{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^D} \mathcal{L}_{\mathcal{NN}}(\{\boldsymbol{x}_i\}_{i=1}^n; \boldsymbol{\theta}). \tag{4.16}$$

In the next Section, we detail the simulation set up and report the numerical results.
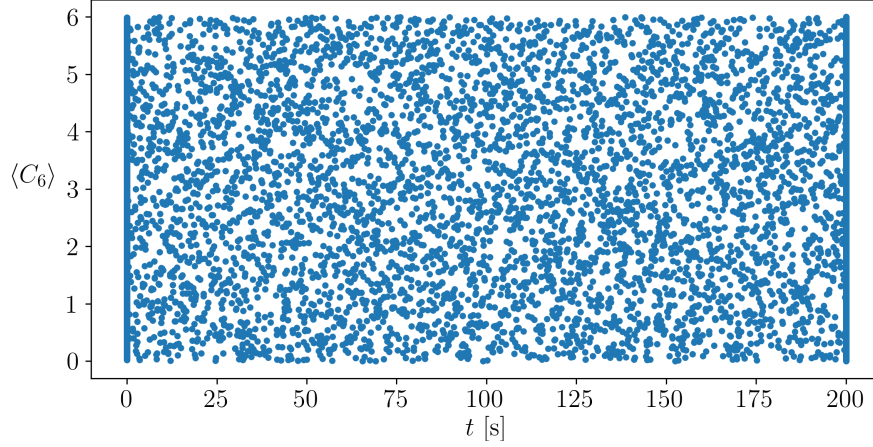
Figure 4.3: Training data in the domain $\Omega = [0, 6] \times [0, 200]$.
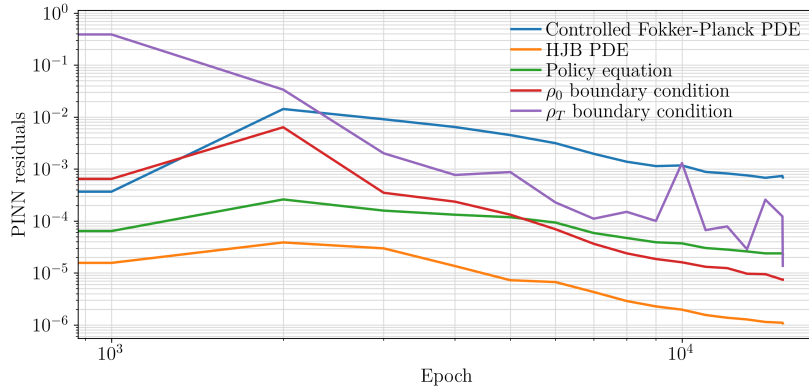
Figure 4.4: The PINN residuals for solving the conditions for optimality (4.12)-(4.13).

4.2.5   Numerical example

We consider the colloidal SA from [127, 128], where the free energy and the diffusion
landscapes are

$$F(\langle C_6 \rangle, \pi) = a\, k_{\mathrm{B}}\, \theta \left(\langle C_6 \rangle - b - c\pi \right)^2, \tag{4.17}$$

$$D_2(\langle C_6 \rangle, \pi) = d\, \exp\left(-(\langle C_6 \rangle - b - c\pi)^2 \right) + f, \tag{4.18}$$

with known parameters $a = 10, b = 2.1, c = 0.75, d = 4.5 \times 10^{-3}, f = 0.5 \times 10^{-3}$,
$k_{\mathrm{B}} = 1.38066 \times 10^{-23}$ Joules per Kelvin, and $\theta = 293$ Kelvin.

We use the DeepXDE library [41] for training the PINN. In particular, we choose

a neural network with 3 hidden layers with 70 neurons in each layer. The activation

functions are chosen to be $\tanh(\cdot)$. The input-output structure of the network is as

explained in Section 4.2.4. For solving (4.16), we use the Adam optimizer [147].

We fix the final time $T = 200$ s, and consider the endpoint PDFs ρ_0, ρ_T shown in

Fig. 4.1, represented as truncated normal PDFs (see e.g., [148, Sec. 2.2]):

$$\rho_i(x) := \begin{cases} \dfrac{1}{\sigma_i} \dfrac{\phi\left(\frac{x-\mu_i}{\sigma_i}\right)}{\Phi\left(\dfrac{b-\mu_i}{\sigma_i}\right) - \Phi\left(\dfrac{a-\mu_i}{\sigma_i}\right)} & \text{for } a \le x \le b, \\[4mm] 0 & \text{otherwise,} \end{cases}$$

where $i \in \{0, T\}$, $\mu_0 = 0, \mu_T = 5, \sigma_0 = 0.2, \sigma_T = 0.1, a = 0, b = 6$. The functions

$\phi(\cdot)$ and $\Phi(\cdot)$ denote the standard normal PDF, and the standard normal cumulative

distribution function, respectively. Recall from Section 4.2.2 that our proposed method

is applicable for arbitrary compactly supported endpoint PDFs.

As shown in Fig. 4.3, we choose 1000 training points at each of the initial ($t = 0$ s)

and terminal ($t = 200$ s) times, and another 5000 state-time points inside the domain

$\Omega := [0, 6] \times [0, 200]$. We used the residual-based adaptive refinement method in [41]

for generating these training points. As shown in Fig. 4.4, after 15,000 training epochs,

the residuals for all loss functions go below 10^{-3}. The training was performed in

Python 3 on a MacBook Air with 1.1 GHz Quad-Core Intel Core i5 processor and 8 GB

memory. The computational time for training was 2097.40 seconds.

Fig. 4.5 shows the optimal policy $\pi_{\text{opt}}\left(\langle C_6 \rangle, t\right)$ obtained as the output of the trained

PINN. The value function $\psi\left(\langle C_6 \rangle, t\right)$ obtained as another output of the trained PINN

is shown in Fig. 4.6. In Fig. 4.7, the snapshots of the optimally controlled PDFs ρ_{opt}^π ob-

tained from the trained PINN are shown as solid curves with grey filled areas. To verify

these results, we sampled 1000 initial states from the given ρ_0 using the Metropolis-
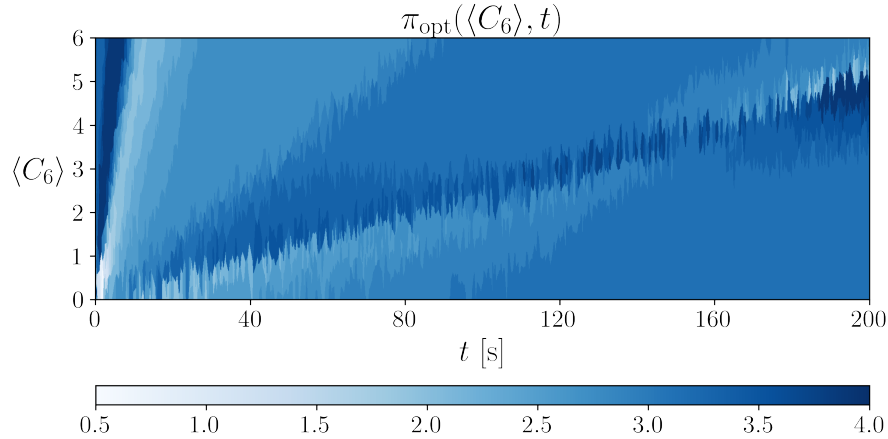
Figure 4.5: The optimal policy $\pi_{\mathrm{opt}}\left(\langle C_6\rangle, t\right)$ obtained as an output of the trained PINN solving the conditions for optimality (4.12)-(4.13).

Hastings [149] Markov Chain Monte Carlo algorithm, and then performed closed-loop simulations using the optimal policy $\pi_{\mathrm{opt}}\left(\langle C_6\rangle, t\right)$ (shown in Fig. 4.5) obtained from the PINN. The stem plots shown in Fig. 4.7 depict the kernel density estimates (KDE) of these closed-loop trajectories. The KDE stems provide empirical approximations for the closed-loop optimally controlled PDFs, which match very well with the learnt solutions from PINN.

Fig. 4.8 shows the 1000 random sample paths for the closed loop simulation using the learnt optimal policy $\pi_{\mathrm{opt}}\left(\langle C_6\rangle, t\right)$ that provably steers the given ρ_0 from $t = 0$ to the given ρ_T at $t = T$ over $[0, T] \equiv [0, 200]$ s.

4.3 Case Study: Colloidal Self-assembly Model Free

In this section, we shift our attention to scenarios where the drift and diffusion coefficient pair $(\boldsymbol{f}, \boldsymbol{g})$ in (4.1b), cannot be derived directly from fundamental physics as what we had in Section 4.2. Instead, these coefficients are estimated through data-driven methods. Specifically, a pair of neural networks (NNs) are trained to approxi-
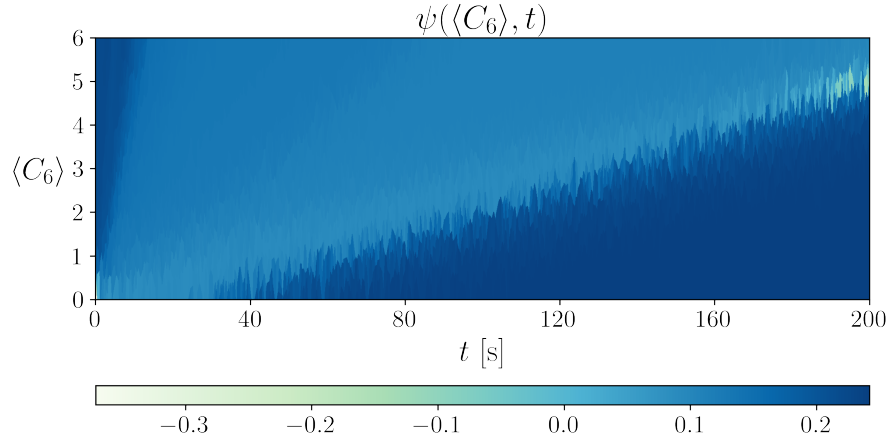
Figure 4.6: The value function $\psi\left(\langle C_6\rangle, t\right)$ obtained as an output of the trained PINN solving the conditions for optimality (4.12)-(4.13).

mate $\boldsymbol{f}$ and $\boldsymbol{g}$ in (4.1), utilizing molecular dynamics (MD) simulation data. In practice, the drift and diffusion coefficients are difficult to model from first principles. This is because accurately capturing the interplay between various forces and interactions, such as van der Waals forces, electrostatic interactions, and solvent-mediated interactions, is challenging. As a result, empirical or semi-empirical approaches [127], as well as coarse-grained or phenomenological models [145], are often employed to approximate these coefficients based on either experimental data or MD simulation data.

Another modeling difficulty specific to colloidal SA is that both $\boldsymbol{f}, \boldsymbol{g}$ are typically nonlinear in state $\boldsymbol{x}$, as well as non-affine in control $\boldsymbol{u}$. Furthermore, $\boldsymbol{f}$ and $\boldsymbol{g}$ both have explicit time dependence in practice. To circumvent these modeling issues, in this work, we propose a learning and control framework where $\boldsymbol{f}$ and $\boldsymbol{g}$ are given learnt from high-fidelity MD simulation data as the outputs of NN representations $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, respectively. With these learnt representations for $\boldsymbol{f}$ and $\boldsymbol{g}$, we propose a computational framework–based on another neural network–to numerically solve (4.1) for control synthesis.
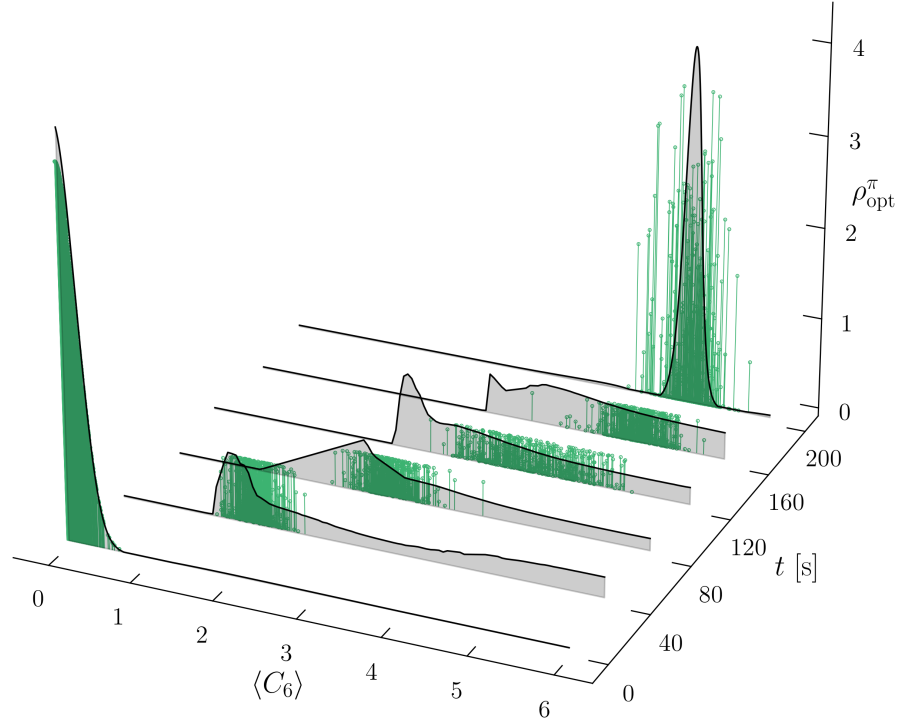
Figure 4.7: Snapshots of the optimally controlled joint PDFs ρ_{opt}^{π} steering the state $\langle C_6 \rangle$ distribution from the given ρ_0 to ρ_T, as in Fig. 4.1, over the given time horizon $[0, T] \equiv [0, 200]$ s subject to the controlled noisy nonlinear sample path dynamics (4.8). The solid black curves with grey filled areas are obtained from the PINN. The stem plots are the KDE approximants of the optimally controlled PDF snapshots obtained from the closed-loop sample paths, as explained in Section 4.2.5.

4.3.1   Neural Schrödinger bridge

Our overall approach is to learn $(\boldsymbol{f}, \boldsymbol{g})$ as fully connected feed-forward NN representations, denoted by $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, respectively. Both these NNs are designed to be functions of the current time $t \in [0, T]$, the system state $\boldsymbol{x}$, and the control input $\boldsymbol{u}$. These two networks are trained to predict the future states of the system based on the tuple $(t, \boldsymbol{x}, \boldsymbol{u})$. Training of these networks using MD simulations is detailed in Section 4.3.4. Fig. 4.9 gives an overview of the proposed learning and control framework. We consider both $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$ to have tangent hyperbolic, i.e., $\tanh(\cdot)$ activation
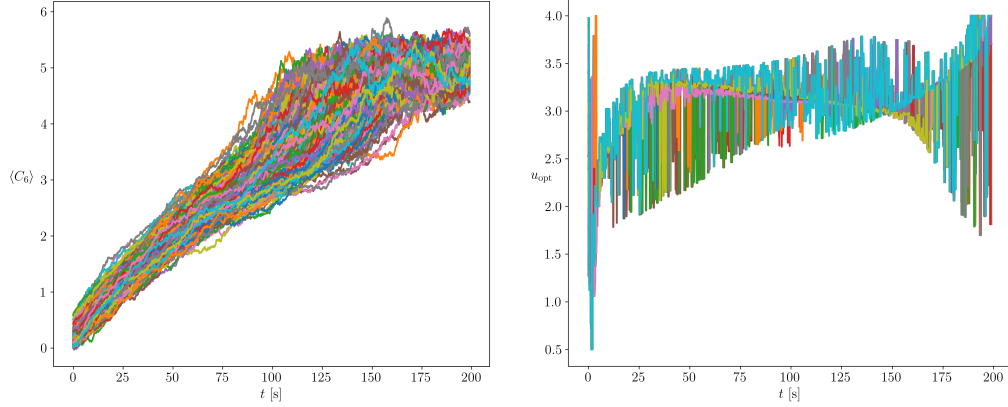
(a) Optimally controlled $\langle C_6 \rangle$ state trajec-
tories.

(b) Optimal control u_{opt} trajectories.

Figure 4.8: The 1000 random sample paths resulting from the closed loop simulation using
the learnt optimal policy $\pi_{\mathrm{opt}}(\langle C_6 \rangle, t)$.

functions. Tangent hyperbolic nonlinearities are known to be slope-restricted [150,
Proposition 2]. As a result, the output of a fully connected feed-forward NN with
$\tanh$ activation remains component-wise slope-restricted. Consequently, $\boldsymbol{f}, \boldsymbol{g}$ being
the respective outputs of the networks $\mathcal{N}_{\mathrm{Drift}}, \mathcal{N}_{\mathrm{Diffusion}}$, are guaranteed [151, Theorem
2] to be Lipschitz continuous. Motivated by the Lipschitz continuity of the outputs
from $\mathcal{N}$Drift and $\mathcal{N}$Diffusion under an admissible Markovian policy $\boldsymbol{u}(t, \boldsymbol{x}) \in \mathcal{U}$, the
assumptions **(A1)** and **(A2)** presented in Section 4.1.1 are deemed valid for this sce-
nario.

4.3.2   Related works

Feedback control has emerged as a promising approach to enhance the reproducibility
of colloidal SA systems towards desired structures. Previous studies [140, 152] demon-
strated the effectiveness of proportional-integral control on simple test systems. How-
ever, applying such basic control approaches to complex colloidal SA systems with
possible kinetically arrested dynamics may not yield satisfactory results. Alterna-

tive approaches like model predictive control (MPC) or dynamic programming have been suggested. For instance, [127] presents an MPC approach based on energy landscapes estimated from MD simulations. However, this method can be computationally demanding for large-scale systems or systems with complex interactions, especially considering that the solution time for MPC might exceed the sampling time of SA, particularly for fast dynamics. This challenge becomes even more compounded as the size and complexity of the SA model grows.

On the other hand, [145] utilizes a dynamic programming-based approach, which results in a lookup table of optimal actions for given states. Despite its theoretical elegance, dynamic programming suffers from the 'curse of dimensionality', rendering it impractical for systems of higher complexity due to the exponential growth in computational resources required. For both these methods, the accuracy of the control relies heavily on the quality of the underlying model. To this end, model-free reinforcement learning can alleviate modeling challenges in optimal control of colloidal SA systems [153, 154]. Furthermore, recent advances in NNs have provided a promising alternative for modeling the hidden physics of stochastic dynamic systems without making assumptions about the final equations representing the physics of the system (e.g., [155–158]). We employ the NNs, $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, to represent the energy and diffusion landscapes of a colloidal SA control problem, critical to guiding the system to a desired final state.

This section makes the following specific contributions.

- We introduce a novel computational approach named 'neural Schrödinger bridge', which utilizes neural networks (NNs) in two distinct ways:

    1. A pair of NNs are trained to approximate the functions $\boldsymbol{f}$ and $\boldsymbol{g}$, utilizing
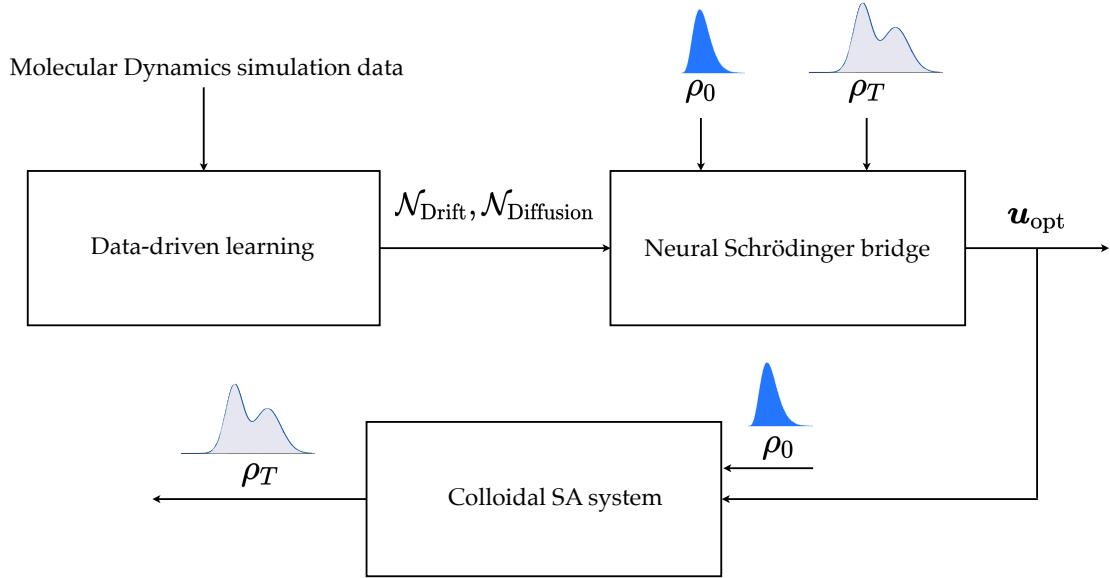
Figure 4.9: An overview of the proposed learning and control framework for solving the generalized Schrödinger Bridge Problem (4.1) for colloidal self-assembly. Here, ρ_0 and ρ_T denote the probability density functions associated with the endpoint measures μ_0 and μ_T, respectively.

    molecular dynamics (MD) simulation data.

2. The optimality conditions of the GSBP, as functions of these NN representations, are solved using PINNs.

- We elucidate that the optimality conditions for such GSBPs necessitate solving a novel system of $m + 2$ coupled PDEs with two boundary conditions, where m is the number of control inputs. This system of PDEs represents a new direction in the theory of Schrödinger bridges and related stochastic optimal control problems.

- Our work extends beyond connecting the colloidal self-assembly problem with GSBP, by tackling the challenges posed by nonlinear, non-affine, and explicitly time-dependent control dynamics that emerge from data-driven learning of the

controlled neural SDEs. The standard computational techniques for GSBPs are insufficient in this context.

- To overcome the limitations of standard PINNs in enforcing distributional endpoint constraints, we propose a PINN architecture with Sinkhorn losses. This innovation not only addresses the constraints more effectively but also differentiates through these losses for training, presenting a potentially independently interesting architecture. This contribution is of broad interest to researchers in control theory and machine learning, particularly those using diffusion models for learning and control.

We clarify here that, from a methodological viewpoint, the proposed framework is different from two recent works [159] and [160], which also bring together SBPs and NNs. In [159], the main idea was to learn the *uncontrolled* f, g as NNs, i.e., to learn an *unforced* neural SDE using the population samples via SBP. The unforced SDE was learnt via a stochastic version of the principle of least action, i.e., by appealing to how SBP can be seen as a stochastic version of dynamic OMT. The work in [160] proposed learning a classical SBP between unpaired images. Different from these works, our colloidal SA context requires learning the *controlled* f, g as controlled neural SDEs before proceeding for optimal control synthesis – the latter is an instance of GSBP, which is then solved via a new variant of PINN that we propose herein.

4.3.3 Solving the conditions for optimality using PINN with Sinkhorn losses

In this Section, we propose a new variant of the PINN designed for numerically solving (4.6)-(4.7). As we explain next, the entropy-regularized squared Wasserstein dis-
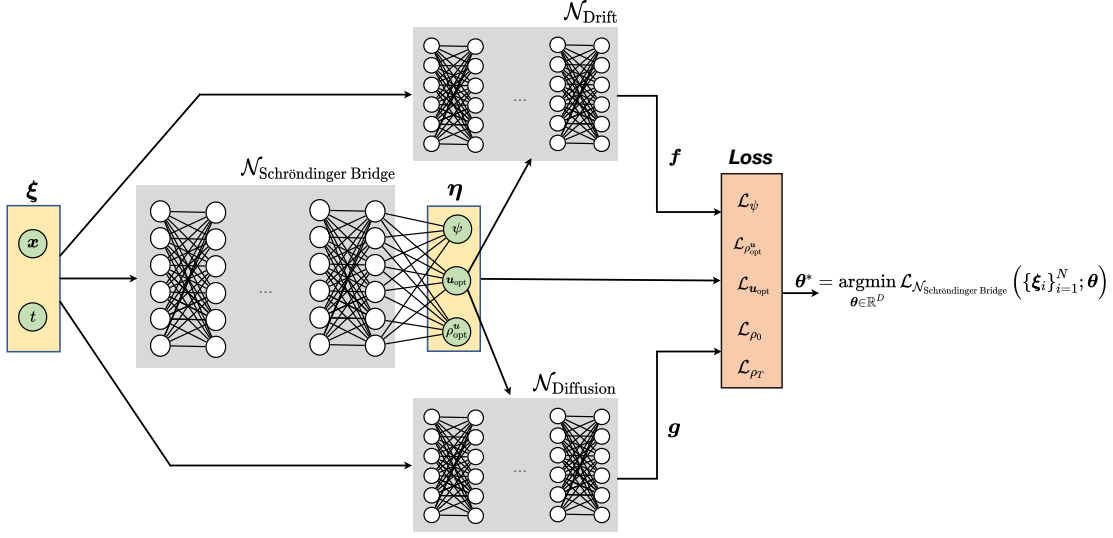
Figure 4.10: The architecture of the physics-informed neural network with the system state $\boldsymbol{x}$, and the time t as the input features $\boldsymbol{\xi} := (\boldsymbol{x}, t)$. The network output $\boldsymbol{\eta}$ comprises of the value function, optimally controlled PDF, and optimal control policy, i.e., $\boldsymbol{\eta} := (\psi, \rho_{\mathrm{opt}}^{\boldsymbol{u}}, \boldsymbol{u}_{\mathrm{opt}})$. The networks $\mathcal{N}_{\mathrm{Drift}}$ and $\mathcal{N}_{\mathrm{Diffusion}}$ are fully trained from MD simulation.

tance, which is given in (1.3), is better suited for automatic differentiation w.r.t. neural network parameters for PINN training, which is what we need for our boundary conditions (4.1c) (or equivalently (4.7)).

Learning with Sinkhorn losses

To better understand the advantage of learning with Sinkhorn losses, consider the squared Euclidean distance matrix $\boldsymbol{C} \in \mathbb{R}^{d \times d}$, and for a given pair of d-dimensional probability vectors $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$, let $\Pi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)$ denote the set of all coupling matrices, i.e.,

$$
\Pi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2) := \left\{ \boldsymbol{M} \in \mathbb{R}^{d \times d} \mid \boldsymbol{M} \geq 0 \text{ (element-wise)}, \right.
$$

$$
\left. \boldsymbol{M}\mathbf{1} = \boldsymbol{\mu}_1, \boldsymbol{M}^\top \mathbf{1} = \boldsymbol{\mu}_2 \right\}. \tag{4.19}
$$

The dimension d here represents the number of samples involved, i.e., the dimensionality of the standard simplex in which $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2$ belong to. Then the discrete version of

(1.3) becomes

$$W_\varepsilon^2\left(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2\right) = \min_{\boldsymbol{M} \in \Pi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)} \langle \boldsymbol{C} + \varepsilon \log \boldsymbol{M}, \boldsymbol{M} \rangle \tag{4.20}$$

where $\varepsilon > 0$ is a fixed regularization parameter. The convex problem (4.20) can be solved using the Sinkhorn recursions [161, 162] a.k.a. iterative proportional fitting procedure (IPFP) [163]. These recursions are motivated by the observation that the minimizer of (4.20) must be a diagonal scaling of the known matrix $\boldsymbol{\Gamma} := \exp\left(\frac{-\boldsymbol{C}}{2\varepsilon}\right) \in \mathbb{R}_{>0}^{d \times d}$ where the exponential is element-wise, i.e.,

$$\boldsymbol{M} = \mathrm{diag}(\boldsymbol{v}_1) \, \boldsymbol{\Gamma} \, \mathrm{diag}(\boldsymbol{v}_2) \tag{4.21}$$

for to-be-determined $\boldsymbol{v}_1, \boldsymbol{v}_2 \in \mathbb{R}^d$.

Starting with some initial guess, the Sinkhorn recursions alternate between updating $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ until convergence:

$$\boldsymbol{v}_1^{k+1} \leftarrow \boldsymbol{\mu}_1 \oslash \left(\boldsymbol{\Gamma} \boldsymbol{v}_2^k\right), \tag{4.22a}$$

$$\boldsymbol{v}_2^{k+1} \leftarrow \boldsymbol{\mu}_2 \oslash \left(\boldsymbol{\Gamma}^\top \boldsymbol{v}_1^{k+1}\right), \tag{4.22b}$$

for recursion index $k = 0, 1, \cdots$; the symbol $\oslash$ denotes the element-wise (Hadamard) division. The updates (4.22a)-(4.22b) can be seen as alternating Kullback-Leibler projections [51, 164] with guaranteed linear convergence.

When $\varepsilon = 0$ in (4.20), we get an LP corresponding to the discrete version of (1.3). This LP has d^2 unknowns with $d^2 + 2d$ constraints, and solving the same as standard network flow problem has $\widetilde{O}\left(d^2 \sqrt{2d}\right)$ complexity [165] which is impractical for large d. Furthermore, using (1.3) as the endpoint loss for training a PINN to learn the solution of (4.6)-(4.7), requires us to compute

$$\texttt{AutoDiff}_{\boldsymbol{\theta}} W^2 \left(\mu_i, \mu_i^{\text{epoch index}}(\boldsymbol{\theta})\right) \ \forall \, i \in \{0, T\} \tag{4.23}$$

for each epoch of the training, where $\texttt{AutoDiff}_{\boldsymbol{\theta}}$ refers to the standard reverse mode
automatic differentiation w.r.t. PINN training parameter $\boldsymbol{\theta}$. Evaluating (4.23) then
amounts to differentiating through a very large scale LP which is computationally
challenging even for moderately large d.

In contrast, using (1.3) as the endpoint loss for training a PINN to learn the solution
of (4.6)-(4.7), requires us to compute

$$\texttt{AutoDiff}_{\boldsymbol{\theta}} W_{\varepsilon}^2 \left(\mu_i, \mu_i^{\text{epoch index}}(\boldsymbol{\theta}) \right) \ \forall \ i \in \{0, T\} \tag{4.24}$$

for a fixed $\varepsilon > 0$. Because the Sinkhorn recursions (4.22) involve a series of differen-
tiable linear operations, it is amenable to Pytorch auto-differentiation to support back-
propagation. Thus using W_{ε}^2 instead of W^2 as the endpoint distributional losses incur
lesser computational overhead allowing us to train PINNs for nontrivial GSBPs. This
advantage of Sinkhorn losses over Wasserstein losses, has also been pointed out in a
different context in [166]. Rigorous consistency results have appeared in [167] show-
ing that the derivatives of the iterates from Sinkhorn recursion computed through
automatic differentiation, indeed converge to the derivatives of the corresponding
Sinkhorn loss.

Proposed PINN architecture

Our proposed architecture for the PINN is shown in Fig. 4.10. For the GSBP considered
here, the state-time $\boldsymbol{\xi} := (\boldsymbol{x}, t)$ comprises the features that are inputs to the network,
and the network output $\boldsymbol{\eta} := (\psi, \rho_{\text{opt}}^{\boldsymbol{u}}, \boldsymbol{u}_{\text{opt}})$ comprises of the value function, optimally
controlled PDF, and optimal policy.

The proposed PINN is a fully connected feed-forward NN with multiple hidden

layers, and we parameterize its output using the network parameter $\boldsymbol{\theta} \in \mathbb{R}^D$, i.e.,

$$\boldsymbol{\eta}(\boldsymbol{\xi}) \approx \mathcal{N}_{\text{Schrödinger Bridge}}(\boldsymbol{\xi}; \boldsymbol{\theta}), \tag{4.25}$$

where $\mathcal{N}_{\text{Schrödinger Bridge}}(\cdot; \boldsymbol{\theta})$ denotes the NN approximant parameterized by $\boldsymbol{\theta}$. Here D denotes the dimension of the parameter space, i.e., the total number of to-be-trained weight, bias and scaling parameters for the NN. For all neurons, we use the $\tanh$ activation functions.

As mentioned in Section 4.3.1, the explicit expressions for $\boldsymbol{f}$ and $\boldsymbol{g}$, the drift and diffusion coefficients, are not available from first-principle physics. We learn these coefficients from MD simulation data (see Section 4.3.4, 4.3.4). As shown in Fig. 4.10, the networks $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, represent the learnt drift $\boldsymbol{f}$ and the learnt diffusion $\boldsymbol{g}$, respectively, which are used to evaluate the loss function $\mathcal{L}_{\mathcal{N}_{\text{Schrödinger Bridge}}}$ for the PINN.

The PINN loss function $\mathcal{L}_{\mathcal{N}_{\text{Schrödinger Bridge}}}$ consists of the sum of the losses associated with the $m + 2$ equations in (4.6), and the losses associated with the boundary conditions (4.7). Specifically, let $\mathcal{L}_\psi$ be the MSE loss for the HJB PDE (4.6a). Likewise, let $\mathcal{L}_{\rho_{\text{opt}}^{\boldsymbol{u}}}$ be the MSE loss for the FPK PDE (4.6b), and because the control policy has m components $(u_1, \ldots, u_m)$, let $\mathcal{L}_{u_{j_{\text{opt}}}}\big|_{j=1,\ldots,m}$ be the corresponding MSE loss term for each control policy component in (4.6c).

However, the MSE losses are insufficient to capture the distributional mismatch for endpoint boundary conditions (4.7). Per Section 4.3.3, we use the Sinkhorn losses as the boundary condition losses $\mathcal{L}_{\rho_0}$ and $\mathcal{L}_{\rho_T}$, and differentiate through the corresponding Sinkhorn recursions.

Thus,

$$\mathcal{L}_{\mathcal{N}_{\text{Schrödinger Bridge}}} := \mathcal{L}_{\rho_0} + \mathcal{L}_{\rho_T} + \mathcal{L}_\psi + \mathcal{L}_{\rho_{\text{opt}}^{\boldsymbol{u}}} + \sum_{j=1}^{m} \mathcal{L}_{u_{j_{\text{opt}}}}, \tag{4.26}$$

where each summand loss term in (4.26) is evaluated on a set of N collocation points
$\{\boldsymbol{\xi}_i\}_{i=1}^N$ in the domain of the feature space $\Omega := \mathcal{X} \times [0, T]$, i.e., $\{\boldsymbol{\xi}_i\}_{i=1}^N \subset \Omega$. For instance,
the equation error losses are of the form

$$\mathcal{L}_\psi := \frac{1}{N} \sum_{i=1}^N \left(\frac{\partial \psi}{\partial t}\bigg|_{\boldsymbol{\xi}_i} - \frac{1}{2} \|\boldsymbol{u}_{\mathrm{opt}}\|_2^2 \bigg|_{\boldsymbol{\xi}_i} + \langle \nabla \psi, \boldsymbol{f} \rangle|_{\boldsymbol{\xi}_i} \right.$$
$$\left. + \langle \boldsymbol{G}, \mathbf{Hess}(\psi) \rangle|_{\boldsymbol{\xi}_i} \right)^2,$$

$$\mathcal{L}_{\rho_{\mathrm{opt}}^{\boldsymbol{u}}} := \frac{1}{N} \sum_{i=1}^N \left(\frac{\partial \rho_{\mathrm{opt}}^{\boldsymbol{u}}}{\partial t}\bigg|_{\boldsymbol{\xi}_i} + \nabla \cdot (\rho_{\mathrm{opt}}^{\boldsymbol{u}} \boldsymbol{f})|_{\boldsymbol{\xi}_i} \right.$$
$$\left. - \langle \mathbf{Hess}, \boldsymbol{G} \rho_{\mathrm{opt}}^{\boldsymbol{u}} \rangle|_{\boldsymbol{\xi}_i} \right)^2,$$

$$\mathcal{L}_{u_{j_{\mathrm{opt}}}}\big|_{j=1,\ldots,m} := \frac{1}{N} \sum_{i=1}^N \left(u_{j_{\mathrm{opt}}}\big|_{\boldsymbol{\xi}_i} - \frac{\partial}{\partial u_{j_{\mathrm{opt}}}} \left(\langle \nabla_{\boldsymbol{x}} \psi, \boldsymbol{f} \rangle \right. \right.$$
$$\left. \left. + \langle \boldsymbol{G}, \mathbf{Hess}(\psi) \rangle \right)|_{\boldsymbol{\xi}_i} \right)^2,$$

where u_{opt}^j denotes the jth component of the optimal control $\boldsymbol{u}_{\mathrm{opt}}$.

We implemented the Sinkhorn recursions with the log-sum-exp (LSE) technique
[168, Section 3] to maintain numerical stability at the expense of minor memory over-
head. We employed mini-batching for sampling our PINN output, and used the same
sample indices to sample from our prescribed ρ_0, ρ_T. The squared Euclidean distance
matrix $\boldsymbol{C}$ mentioned in Section 4.3.3 was constructed from the output batch points.

We used the PINN software library [41] with a Pytorch backend to perform nu-
merical experiments using the above loss functions. The PINN library [41] was not
written for Schrödinger bridge-type problems, so we needed to modify it to suit our
needs. One modification was to program PINN to compute loss between outputs and
distributions directly and integrate the Sinkhorn iteration algorithm into the library.
We also modified it to perform the mini-batching we needed. In summary, for training

the PINN, the overall loss (4.26) was minimized over $\boldsymbol{\theta} \in \mathbb{R}^D$ by solving

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta} \in \mathbb{R}^D}{\operatorname{argmin}} \, \mathcal{L}_{\mathcal{N}_{\text{Schrödinger Bridge}}}(\{\boldsymbol{\xi}_i\}_{i=1}^N; \boldsymbol{\theta}). \tag{4.27}$$

The next section details the simulation setup and reports the numerical results.

4.3.4   Numerical example

We now present a numerical case study of a colloidal SA system where the drift co-efficient $\boldsymbol{f}$ and the diffusion coefficient $\boldsymbol{g}$ are not analytically available, instead they are learnt as NN representations $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, from MD simulation data. Such learnt representations are nonlinear in the state $\boldsymbol{x}$ and non-affine in control $\boldsymbol{u}$. We then solve the GSBP (4.1) using the PINN architecture proposed in Section 4.3.3 to design a minimum effort controller steering the distribution in the order-parameter space to synthesize the body-centered cubic (BCC) crystal structure over the given time horizon. Fig. 4.11 shows an initial disordered structure and a final BCC structure.

System description

We consider the *in-silico* representation of isotropic colloidal particles with identical Lennard-Jones interaction potentials within an NPT (isothermal-isobaric) ensemble. The Lennard-Jones potential is used to model particle interactions in the system and is defined as

$$U(r) := 4\epsilon \left(\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^6 \right), \tag{4.28}$$

where r denotes the particle radius, and ϵ denotes the depth of the potential energy well and thus quantifies the strength of attractive forces between particles. The symbol

σ denotes the distance at which the potential energy is nullified, thereby demarcating the intermolecular potential's shift from attraction to repulsion depending on particle size [169, p. 234].

<div align="center">(a)   (b)</div>
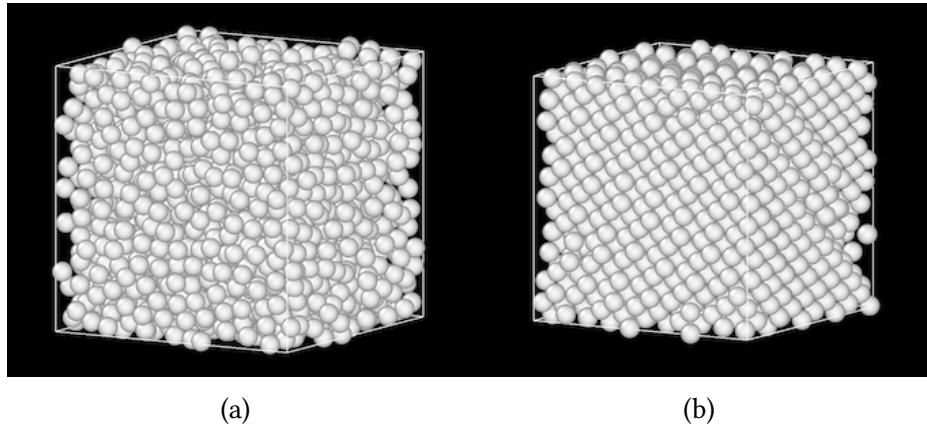
Figure 4.11: (a) An initial disordered crystalline structure. (b) A final BCC structure with minor defects. These images were generated using OVITO [1].

An ensemble of 2048 particles is initialized at a given temperature and pressure. While the positions of these particles may be considered as the most natural states of a colloidal SA system, they result in an unmanageably high-dimensional state space. To circumvent this difficulty, we seek a lower-dimensional representation. To this end, the Steinhardt bond order parameters $\langle C_{10} \rangle$ and $\langle C_{12} \rangle$ are used in this work, which are directly defined in terms of the particle positions. To calculate these parameters [170] from the MD simulation data, we proceed through a series of steps, as discussed below.

We first extract the positional information for each particle from the MD simulation data (see Section 4.3.4). Next, we identify the neighbors for each particle based on the Voronoi method [171]. Using this information, we calculate the spherical harmonics, Y_{lm}, indexed by two quantum numbers, viz. the azimuthal or orbital quantum number, denoted by l, and the magnetic quantum number, denoted by m.

The azimuthal quantum number defines the shape of the orbital, and for our con-

text $l \in [1, 12]$. The magnetic quantum number represents the orientation of the orbital in space, and for our context $m \in [-l, l]$, see e.g., [172, p. 545]. Accordingly, the spherical harmonics are defined as

$$Y_{lm}(\theta, \phi) := \sqrt{\frac{2l + 1}{4\pi} \frac{(l - m)!}{(l + m)!}} P_l^m(\cos\theta)e^{im\phi}, \tag{4.29}$$

where θ and ϕ represent the polar and azimuthal angles, respectively. In (4.29), the P_l^m denote the associated Legendre polynomials [173, p. 331–339], which is a class of functions that arise in the solution to Laplace's equation in spherical coordinates.

Let $\nu(i)$ denote the number of neighbors of particle i, and let r_{ij} signify the positional vector between particle i and its neighbor j. Subsequently, the lth bond order parameter $C_l(i)$, for each particle i, is computed as [170]

$$C_l(i) = \left(\frac{4\pi}{2l + 1} \sum_{m=-l}^{l} \left| \frac{1}{\nu(i)} \sum_{j=1}^{\nu(i)} Y_{lm}(r_{ij}) \right|^2 \right)^{\frac{1}{2}}, \tag{4.30}$$

where the index $i \in [0, \nu]$ and ν represents the total number of particles in the ensemble ($\nu = 2048$ in our case study). Furthermore, index $j \in [0, \nu(i)]$. In (4.30), normalizing by the number of neighbors ensures that the final system order parameter is size-independent and thus, scalable across different systems. That is, the normalization ensures that $C_l(i) \in [0, 1]$.

Next, the individual bond order parameters $C_l(i)$ are averaged over all particles in the ensemble to calculate the averaged lth Steinhardt bond order parameter

$$\langle C_l \rangle = \frac{1}{\nu} \sum_{i=1}^{\nu} C_l(i), \tag{4.31}$$

which can be used to describe the state of a colloidal SA system.[1] Therefore, the physics-based order parameters serve as a reduced-dimensionality conduit that en-

---

[1]In actual self-assembly systems, image analysis techniques can be used to locate and track particle centers, as well as compute local and global order parameters in real-time (e.g., see [140]).

hances the efficiency and effectiveness of our subsequent analyses by circumventing the need to work with high-dimensional particle position data.[2]

In this work, we specifically choose the order parameters $\langle C_{10} \rangle$ and $\langle C_{12} \rangle$ for their efficacy in distinguishing between the body-centered cubic (BCC) and the face-centered cubic (FCC) structures. The values of $\langle C_{10} \rangle$ and $\langle C_{12} \rangle$ for defect-free assembled BCC and FCC structures do not overlap, which enables differentiation between the two structure types.

In summary, the controlled dynamics of our colloidal SA system is described by the SDE (2.13b), where the state and inputs are defined as

$$\boldsymbol{x} := (\langle C_{10} \rangle, \langle C_{12} \rangle) \in \mathcal{X} \equiv [0, 1]^2,$$

$$\boldsymbol{u} := (\text{temperature}, \text{pressure}) \in \mathcal{U}.$$

We denote the components of the optimal control policy $\boldsymbol{u}_{\text{opt}}$ as $u_1^{\text{opt}}, u_2^{\text{opt}}$ respectively.

Learning drift and diffusion coefficients

To learn the NN representations $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, which model the drift and diffusion coefficients $\boldsymbol{f}$ and $\boldsymbol{g}$ in the SDE (2.13b), we performed MD simulations for the above-described system using the Python package HOOMD-blue [175] with final time $T = 200$ s. The simulation data consisted of 200 state trajectories, i.e., the trajectories of the order parameters $\langle C_{10} \rangle, \langle C_{12} \rangle$ for $t \in [0, T]$, that represent the time evolution of position of the $\nu = 2048$ particles of the colloidal SA system, mentioned earlier in Section 4.3.4. Each state trajectory was generated under different linear temperature and pressure ramp rates (i.e., $\boldsymbol{u}$), which were sampled using a Latin Hypercube design and scaled to $[-0.005, 0.005]$, the input range for the simulation. To generate the

---

[2]In this work, the Steinhart bond order parameters were calculated using the Python package Freud [174].
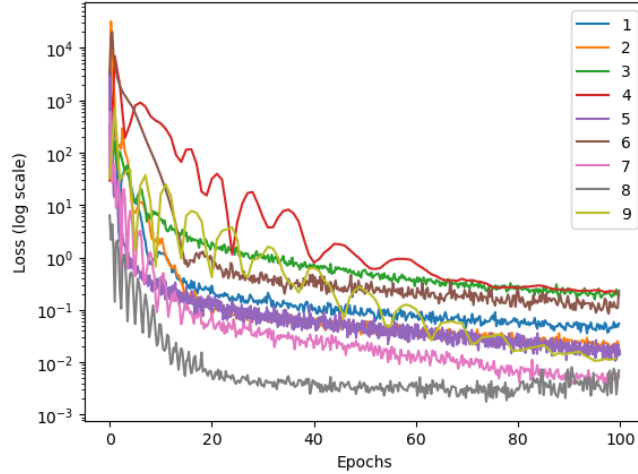
Figure 4.12: Validation losses for nine different neural network models $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, for the drift and diffusion terms in the SDE (2.13b), with legend numbers corresponding to model numbers in Table 4.2.

training and test data for learning the NN models, the state trajectories were sampled 500 times.

Building on our earlier work [156], $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$ were trained on the MD data with a controlled neural SDE. The NNs are designed to be functions of the current time $t \in [0, T]$, the system state x, and the control input u. The networks are passed to (2.13b) to predict the state evolution. The MSE loss is computed for each time step, and the learning process aims to minimize the total MSE loss between the networks' predicted states and the actual observed states from the MD simulation trajectories. For model optimization, we used the Adam optimizer [147] which adjusts the learning rate on a per-parameter basis. The learning rate was initially set to a predefined constant, as per Table 4.2, and was subsequently adjusted using an exponential learning rate scheduler with a decay rate of 0.999. This scheduler reduces the learning rate multiplicatively after each epoch. The data was partitioned into a 70/20/10 distribution for the training, testing, and validation subsets, respectively. The implementation

of these networks was done with the torchsde [176] Python package.

To determine the best architecture for the NNs $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$, we used hyper-parameter (depth and width, batch size, learning rate) turning as detailed in Table 4.2; a total of nine models were trained and evaluated. All of the NN architectures follow a sequential design of fully connected layers, with 5 input units and an output layer of 2 units. The architectures vary in the number of hidden layers and their nodes, all using $\texttt{tanh}$ activation functions. Architecture 1 employs one hidden layer with 200 nodes; architecture 2 utilizes a hidden layer of 1000 nodes; and architecture 3 deploys six hidden layers with 200 nodes each. The batch size, defining the number of samples to be processed before updating the model, is tuned for learning. Lastly, we adjust the learning rate, a factor determining how much the model's parameters should be adjusted with respect to the calculated error, for balanced and steady learning without risking instability or slow convergence. The MSE was used as the loss function for all models.

Fig. 4.12 shows the validation MSE loss for all models, which are evaluated by using the $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$ in (2.13b) to predict the state x, and then comparing the predicted states with those obtained from MD simulations. These validation results demonstrate that all models converge, indicating that the training time was sufficient. As seen in Table 4.2 and Fig. 4.12, model 7 exhibited the best performance evidenced by its minimal validation loss. Consequently, we used model 7 for representing the colloidal SA dynamics in the form of (2.13b). Its corresponding $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$ are used for the optimal control synthesis for the GSBP. On an NVIDIA GTX 1080, each model undergoes training that, on average, takes 10 seconds per training step. To complete 100 epochs, this process requires approximately 1.2 hours per model. The approximate inference time for the model is 0.0123 seconds.
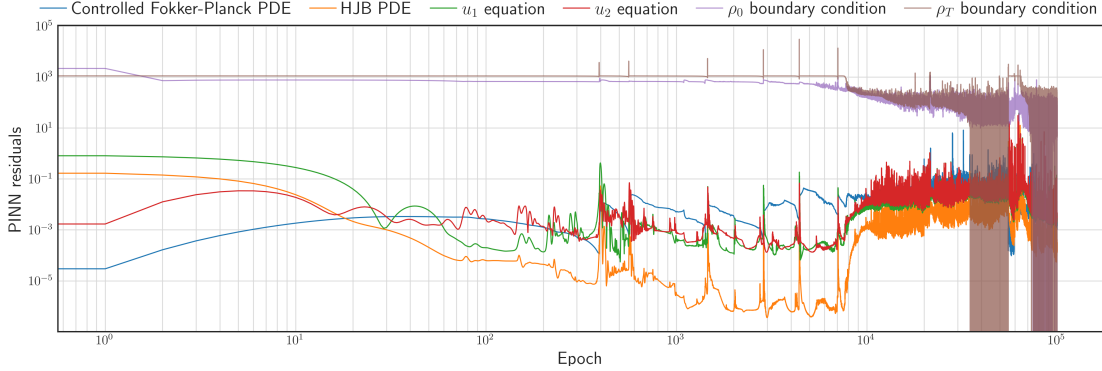
Figure 4.13: The PINN residuals in solving the conditions of optimality (4.6)-(4.7) for the simulation in Section 4.3.4.

Controller synthesis

With the $\boldsymbol{f}, \boldsymbol{g}$ learnt as per Section 4.3.4 for the colloidal SA system described in Section 4.3.4, we considered the GSBP (4.3) over fixed time horizon $[0, T]$, where the final time $T = 200$ s, the initial state $\boldsymbol{x}(t = 0) \sim \rho_0 = \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{\Sigma}_0)$, and the terminal state $\boldsymbol{x}(t = T) \sim \rho_T = \mathcal{N}(\boldsymbol{m}_T, \boldsymbol{\Sigma}_T)$. Here, the notation $\mathcal{N}(\boldsymbol{m}, \boldsymbol{\Sigma})$ stands for a joint Gaussian distribution with mean vector $\boldsymbol{m}$ and covariance matrix $\boldsymbol{\Sigma}$. We used

$$\boldsymbol{m}_0 = (0.2, 0.2)^\top, \ \boldsymbol{m}_T = (0.4, 0.375)^\top, \ \boldsymbol{\Sigma}_0 = \boldsymbol{\Sigma}_T = 0.1 \boldsymbol{I}_2. \tag{4.32}$$

In particular, the statistics of the initial state $\boldsymbol{x}(t = 0) \sim \mathcal{N}(\boldsymbol{m}_0, \boldsymbol{\Sigma}_0)$ is chosen to coincide with that used in the MD simulation in Section 4.3.4. The mean $\boldsymbol{m}_T$ for the target terminal state $\boldsymbol{x}(t = T) \sim \mathcal{N}(\boldsymbol{m}_T, \boldsymbol{\Sigma}_T)$ was chosen to represent the BCC crystal structure. Hence, the control objective for the GSBP represents the problem of designing a minimum effort Markovian controller that provably steers the stochastic order parameters in a way to synthesize BCC crystal structure over the prescribed time horizon.

We used the PINN $\mathcal{N}_{\text{Schrödinger Bridge}}$ proposed in Section 4.3.3 for numerically solving the GSBP conditions of optimality (4.6)-(4.7). For our PINN implementation, the
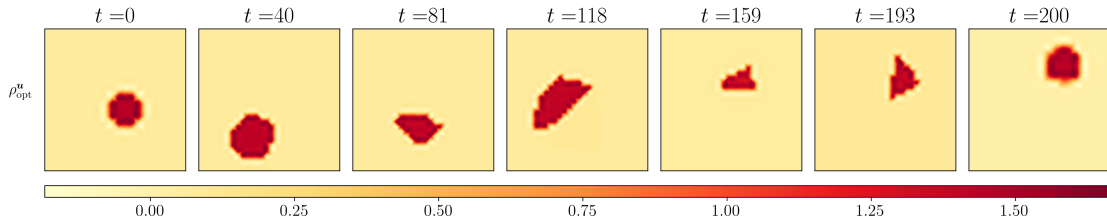
(a) Contour plots of the optimally controlled state PDFs $\rho^{\boldsymbol{u}}_{\mathrm{opt}}(t, \boldsymbol{x})$ over the state space $[0, 1]^2$.
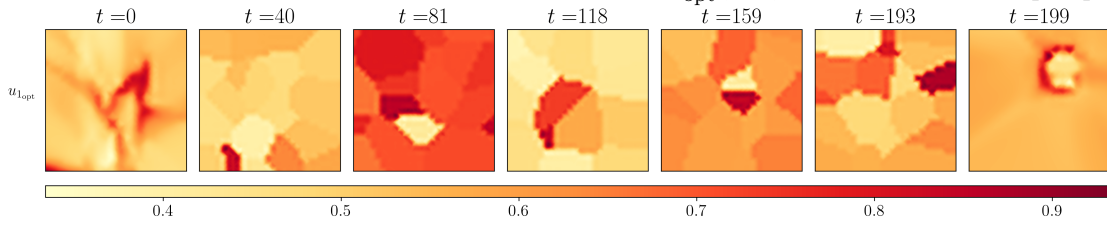
(b) Contour plots of the optimal control component $u_{1_{\mathrm{opt}}}(t, \boldsymbol{x})$ over the state space $[0, 1]^2$.
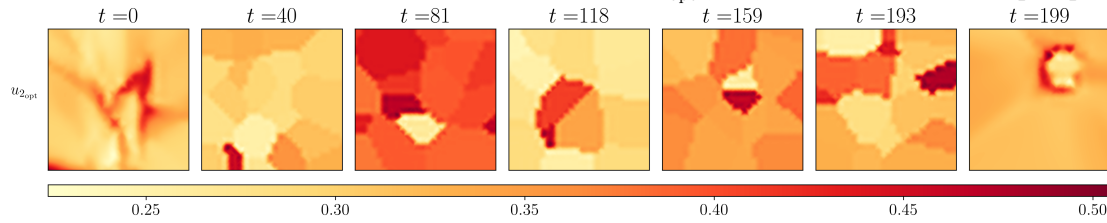
(c) Contour plots of the optimal control component $u_{2_{\mathrm{opt}}}(t, \boldsymbol{x})$ over the state space $[0, 1]^2$.
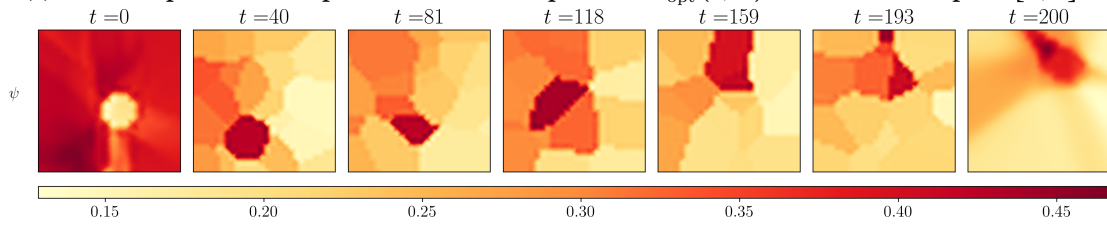
(d) Contour plots of the value function $\psi(t, \boldsymbol{x})$ over the state space $[0, 1]^2$.

Figure 4.14: Results for the GSBP simulation detailed in Section 4.3.4 over time $t \in [0, 200]$. The color denotes the value of the plotted variable; see colorbar (dark red = high, light yellow = low).

(a) Optimally controlled $\langle C_{10} \rangle$ state trajectories.

(b) Optimally controlled $\langle C_{12} \rangle$ state trajectories.
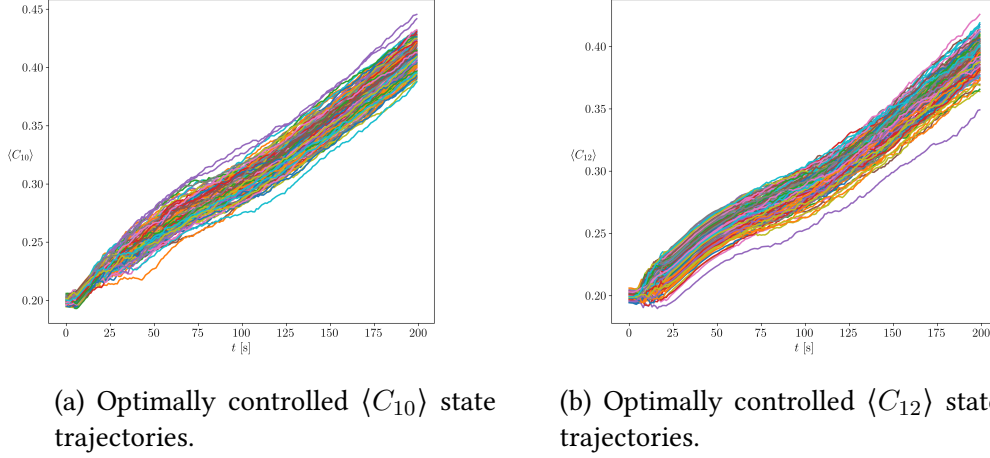
Figure 4.15: The 150 random sample paths resulting from closed-loop simulations using the learnt optimal policy $\boldsymbol{u}_{\mathrm{opt}}\left(t, \langle C_{10} \rangle, \langle C_{12} \rangle\right)$.

domain for state-time collocation is $\Omega = [0,1]^2 \times [0,200]$. Our network consisted of 4 hidden layers, each containing 70 neurons, all with $\tanh$ activation functions. We trained the PINN for $100{,}000$ epochs using the Adam optimizer [147] with a learning rate of 10^{-3}. All our training were performed on a computing platform with NVIDIA Quadro p1000, 640 Cuda cores, and 64 GB RAM. For the collocation, we used $N = 3000$ pseudorandom samples, drawn using Sobol distribution, between the endpoint boundary conditions at $t = 0$ and $t = 200$. We also uniformly randomly sampled $3{,}000$ samples every $20{,}000$ epochs to satisfy compute constraints. For computing the Sinkhorn losses at the endpoint boundary conditions, we used an entropic regularization parameter of $\varepsilon = 0.1$ as in (4.20). For the computing platform mentioned above, training the proposed PINN on average takes 2 seconds per epoch, so to complete $100{,}000$ epochs, it takes approximately 55.5 hours.

Fig. 4.13 shows the PINN residuals in (4.26), and Fig. 4.14 shows the corresponding GSBP solutions obtained from the trained PINN. In particular, Fig. 4.14a shows the evolution of the optimally controlled transient joint PDFs $\rho_{\mathrm{opt}}^{\boldsymbol{u}}(t, \boldsymbol{x})$ interpolating

the fixed ρ_0, ρ_T mentioned above. Notice that, even though the initial and terminal stochastic states are both chosen to have Gaussian statistics, the transient joints in Fig. 4.14a are non-Gaussian. This is expected since the learnt $\boldsymbol{f}, \boldsymbol{g}$, as well as the optimal controller $\boldsymbol{u}_{\mathrm{opt}}$ (see Fig. 4.14b-4.14c), are nonlinear in state. A comparison of Fig. 4.14b and Fig. 4.14c with Fig. 4.14d also shows that the optimal controls are high (resp. low) in regions where the value function ψ changes rapidly, i.e., when the (sub)gradient of ψ is large (resp. small).

To further illustrate the GSBP results, we performed a closed loop sample path simulation for 150 initial state samples $\boldsymbol{x}(t=0) \sim \rho_0$ (with the same ρ_0 mentioned before) using the learnt optimal control policy $\boldsymbol{u}_{\mathrm{opt}}(t, \langle C_{10} \rangle, \langle C_{12} \rangle)$ that provably steers the given ρ_0 from $t=0$ to the given ρ_T (BCC crystal) at $t=T=200$ s. The corresponding closed-loop state sample paths shown in Fig. 4.15 demonstrate that the optimal policy indeed steers the controlled stochastic state from around $(0.2, 0.2)$ to around $(0.4, 0.375)$ with high probability, as specified per problem data (4.32).

For the closed-loop simulations, we constructed a k-d tree [177] (with leaf size = 2) for fast querying of the PINN-trained optimal control policy $\boldsymbol{u}_{\mathrm{opt}}(t, \langle C_{10} \rangle, \langle C_{12} \rangle)$. This construction takes 1.785 seconds. During the numerical integration of the SDE, querying the optimal control policy takes 0.227 milliseconds. Without the k-d tree construction, this querying is 1000x slower (approximately 0.22 seconds). With k-d tree-based querying, to simulate a closed-loop sample path as in Fig. 4.15 using the Euler-Maruyama scheme with 500 equispaced time steps in $[0, T]$, taking approximately 177 seconds. These experiments suggest that the proposed control approach is practically viable for colloidal SA.

Special case	Drift $\boldsymbol{f}$	Diffusion $\boldsymbol{g}$	Form of (4.6a)	Form of (4.6b)	Form of (4.6c)
OMT [75]	$\boldsymbol{u}$	$\boldsymbol{0}_{n\times p}$	$\dfrac{\partial \psi}{\partial t} + \dfrac{1}{2}\|\nabla_{\boldsymbol{x}}\psi\|_2^2 = 0$	$\dfrac{\partial \rho^{\boldsymbol{u}}_{\mathrm{opt}}}{\partial t} = -\nabla_{\boldsymbol{x}} \cdot \left(\rho^{\boldsymbol{u}}_{\mathrm{opt}} \nabla_{\boldsymbol{x}}\psi\right)$	$\boldsymbol{u}_{\mathrm{opt}} = \nabla_{\boldsymbol{x}}\psi$
SBP [79, 80]	$\boldsymbol{u}$	$\boldsymbol{I}_{n\times n}$	$\dfrac{\partial \psi}{\partial t} + \dfrac{1}{2}\|\nabla_{\boldsymbol{x}}\psi\|_2^2 + \Delta_{\boldsymbol{x}}\psi = 0$	$\dfrac{\partial \rho^{\boldsymbol{u}}_{\mathrm{opt}}}{\partial t} = -\nabla_{\boldsymbol{x}} \cdot \left(\rho^{\boldsymbol{u}}_{\mathrm{opt}} \nabla_{\boldsymbol{x}}\psi\right) + \Delta_{\boldsymbol{x}}\psi$	$\boldsymbol{u}_{\mathrm{opt}} = \nabla_{\boldsymbol{x}}\psi$
Control affine GSBP [38]	$\widetilde{\boldsymbol{f}}(t,\boldsymbol{x})$ $+\boldsymbol{B}(t)\boldsymbol{u}$	$\boldsymbol{B}(t) \in \mathbb{R}^{n\times m}$	$\dfrac{\partial \psi}{\partial t} + \dfrac{1}{2}\|\boldsymbol{B}(t)^{\top}\nabla_{\boldsymbol{x}}\psi\|_2^2$ $+\langle \nabla_{\boldsymbol{x}}\psi, \widetilde{\boldsymbol{f}}\rangle$ $+\langle \boldsymbol{B}(t)\boldsymbol{B}(t)^{\top}, \mathbf{Hess}(\psi)\rangle = 0$	$\dfrac{\partial \rho^{\boldsymbol{u}}_{\mathrm{opt}}}{\partial t} =$ $-\nabla_{\boldsymbol{x}} \cdot \left(\rho^{\boldsymbol{u}}_{\mathrm{opt}} \left(\widetilde{\boldsymbol{f}} + \boldsymbol{B}(t)\boldsymbol{B}(t)^{\top}\nabla_{\boldsymbol{x}}\psi\right)\right)$ $+\langle \boldsymbol{B}(t)\boldsymbol{B}(t)^{\top}, \mathbf{Hess}(\rho^{\boldsymbol{u}}_{\mathrm{opt}})\rangle$	$\boldsymbol{u}_{\mathrm{opt}} = \boldsymbol{B}(t)^{\top}\nabla_{\boldsymbol{x}}\psi$

Table 4.1: Special cases of the GSBP (4.1) (equivalently (4.3)) and corresponding reductions of the optimality conditions (4.6).

Model number	Learning rate	Batch size (of epoch)	Model architecture	Validation loss	Training loss
1	10^{-3}	1/4	1	0.390	0.057
2	10^{-2}	1/4	1	0.460	0.024
3	10^{-4}	1/4	1	1.260	0.120
4	10^{-3}	1	1	3.890	0.200
5	10^{-3}	1/4	1	0.200	0.016
6	10^{-3}	1/4	2	9.290	0.072
7	10^{-3}	1/4	3	0.030	0.003
8	10^{-4}	1/4	3	0.031	0.007
9	10^{-3}	1	3	0.110	0.015

Table 4.2: Comparison of different model architectures and hyperparameters for learning the NN representations $\mathcal{N}_{\text{Drift}}$ and $\mathcal{N}_{\text{Diffusion}}$ for the drift f and diffusion g, respectively. The different architectures vary in the number of hidden layers and their nodes, all using $\texttt{tanh}$ activation function. Architecture 1 employs one hidden layer with 200 nodes, architecture 2 utilizes a hidden layer of 1000 nodes, and architecture 3 deploys six hidden layers with 200 nodes each.

5 | A Controlled Mean Field Model for Chiplet Population Dynamics

This Chapter is motivated by micro-assembly applications, such as printer systems [178, 179] and manufacturing of photovoltaic solar cells, where an array of electrodes can be used to generate spatio-temporally non-homogeneous electric potential landscapes for dynamically assembling the "chiplets"–micron sized particles immersed in dielectric fluid–into desired patterns. In such applications, the electric potentials generated by the array of electrodes induce non-uniform dielectrophoretic forces on the chiplets, thereby resulting in a population-level chiplet dynamics. The purpose of the present work is to propose a controlled mean field model for the same.

There have been several works [180–184] on the modeling and dielectrophoretic control of chiplet population. However, a continuum limit macroscopic dynamics that accounts for both chiplet-to-chiplet and chiplet-to-electrode nonlocal interactions, as considered herein, has not appeared before.

The mean field limit pursued here involves considering the number of chiplets and electrodes as infinity, i.e., to think both of them as continuum population. There are two reasons why this could be of interest. *First,* the continuum limit helps approximate and better understand the dynamics for large but finitely many chiplets and electrodes, which is indeed the situation in the engineering applications mentioned before. *Second,* the distributed control synthesis problem for large but finite population becomes computationally intractable, as noted in recent works [183, 185, 186]. A controlled mean field model opens up the possibility of designing a controller in the continuum limit with optimality guarantees. Such a controller can then be applied to a large but finite population with sub-optimality bounds. We clarify here that in this

work, we only present the mean field model and its properties. We leave the control synthesis problem for our follow up work.

As in prior works such as [183], we consider the chiplet dynamics in two dimensional position coordinate. Specifically, let $\boldsymbol{x}(t) \in \mathbb{R}^2$ denote the position vector of a chiplet at any fixed time $t \in [0, \infty)$, and let

$$u : \mathbb{R}^2 \times [0, \infty) \mapsto [u_{\min}, u_{\max}] \subset \mathbb{R}$$

denote a causal deterministic control policy, i.e., $u = u(\boldsymbol{x}, t)$. The control u represents the electrode voltage input, and in practice, the typical voltage range $[u_{\min}, u_{\max}] = [-400, 400]$ Volt. We denote the collection of admissible control policies as $\mathcal{U}$. For a typical experimental set up detailing the sensing-control architecture, see [183, Sec. II].

A viscous drag force balances the controlled force vector field $\boldsymbol{f}^u$ induced by the joint effect of the chiplet-to-chiplet and chiplet-to-electrode interactions. At the low Reynolds number context relevant here, the viscous drag force is proportional to $\dot{\boldsymbol{x}}$, where the proportionality constant μ denotes the viscous coefficient of the dielectric fluid. Ignoring the acceleration due to negligible mass of a chiplet, the dynamics then takes a form

$$\underbrace{\mu \dot{\boldsymbol{x}}}_{\text{viscous drag force}} = \underbrace{\boldsymbol{f}^u}_{\text{controlled interaction force}} + \quad \text{noise} \tag{5.1}$$

where the noise may result from stochastic forcing due to environmental fluctuations (e.g., dielectric fluid impurities) and/or unmodeled dynamics.

5.1   Controlled Mean Field Model

In this Section, we introduce the chiplet population dynamics. Such model has its origin in the physical processes enabling silicon microchips to be manipulated by both electrophoretic and dielectrophoretic forces when they are placed in dielectric carriers such as Isopar-M [187]. These carriers have low conductivity which allows long-range Coulomb interactions. In general, the dielectrophoretic forces dominate, and they are induced by the potential energy generated by electrostatic potentials created in electrodes. The electrodes are formed by depositing nm-scale Molybdenum-Chromium (MoCr) onto a glass substrate via vapor deposition and then directly patterning them with a laser ablation tool. The electrodes are then insulated from the chiplets and dielectric fluid by thermally laminating a micrometer-scale thick perfluoroalkoxy (PFA) film. The dielectric forces act on the chiplets, while viscous drag forces proportional to their velocities oppose their motion. Due to the negligible mass of the chiplets, their acceleration can be ignored.

Let us denote the *normalized chiplet population density function* (PDF) at time t as $\rho(\boldsymbol{x}, t)$. By definition, $\rho \geq 0$ and $\int_{\mathbb{R}^2} \rho \, \mathrm{d}\boldsymbol{x} = 1$ for all t.

We make the following assumptions.

A1. Under an admissible control policy $u \in \mathcal{U}$, the chiplet normalized population distribution over the two dimensional Euclidean configuration space remains absolutely continuous w.r.t. the Lebesgue measure $\mathrm{d}\boldsymbol{x}$ for all $t \in [0, \infty)$. In other words, the corresponding PDFs $\rho(\boldsymbol{x}, t)$ exist for all $t \in [0, \infty)$.

A2. Under an admissible control policy $u \in \mathcal{U}$, we have $\rho \in \mathcal{P}_2(\mathbb{R}^2)$ for all t.

The sample path dynamics of a chiplet position is governed by a controlled non-

local vector field

$$\boldsymbol{f}^u : \mathbb{R}^2 \times [0, \infty) \times \mathcal{U} \times \mathcal{P}_2(\mathbb{R}^2) \mapsto \mathbb{R}^2$$

induced by a controlled *interaction potential* $\phi^u : \mathbb{R}^2 \times \mathbb{R}^2 \times [0, \infty) \mapsto \mathbb{R}$, i.e.,

$$\boldsymbol{f}^u(\boldsymbol{x}, t, u, \rho) := -\nabla \left(\rho * \phi^u\right), \tag{5.2}$$

where $*$ denotes *generalized convolution* in the sense

$$\left(\rho * \phi^u\right)(\boldsymbol{x}, t) := \int_{\mathbb{R}^2} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t)\rho(\boldsymbol{y}, t)\mathrm{d}\boldsymbol{y}.$$

The superscript u in ϕ^u emphasizes that the potential depends on the choice of control policy. In particular,

$$\phi^u(\boldsymbol{x}, \boldsymbol{y}, t) := \phi_{\text{cc}}^u(\boldsymbol{x}, \boldsymbol{y}, t) + \phi_{\text{ce}}^u(\boldsymbol{x}, \boldsymbol{y}, t), \tag{5.3a}$$

$$\phi_{\text{cc}}^u(\boldsymbol{x}, \boldsymbol{y}, t) := \frac{1}{2} C_{\text{cc}} \left(\|\boldsymbol{x} - \boldsymbol{y}\|_2\right) \left(\bar{u}(\boldsymbol{y}, t) - \bar{u}(\boldsymbol{x}, t)\right)^2, \tag{5.3b}$$

$$\phi_{\text{ce}}^u(\boldsymbol{x}, \boldsymbol{y}, t) := \frac{1}{2} C_{\text{ce}} \left(\|\boldsymbol{x} - \boldsymbol{y}\|_2\right) \left(u(\boldsymbol{y}, t) - \bar{u}(\boldsymbol{x}, t)\right)^2, \tag{5.3c}$$

for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^2$ and

$$\bar{u}(\boldsymbol{x}, t) := \frac{\int_{\mathbb{R}^2} C_{\text{ce}} \left(\|\boldsymbol{x} - \boldsymbol{y}\|_2\right) u(\boldsymbol{y}, t)\rho(\boldsymbol{y}, t)\mathrm{d}\boldsymbol{y}}{\int_{\mathbb{R}^2} C_{\text{ce}} \left(\|\boldsymbol{x} - \boldsymbol{y}\|_2\right) \rho(\boldsymbol{y}, t)\mathrm{d}\boldsymbol{y}}. \tag{5.4}$$

The subscripts cc and ce denote the chiplet-to-chiplet and chiplet-to-electrode interactions, respectively. As before, the superscript u highlights the dependence on the choice of control policy. In (5.3b)-(5.3c), C_{cc} and C_{ce} respectively denote the chiplet-to-chiplet and chiplet-to-electrode capacitances. These capacitances can be determined using two dimensional electrostatic COMSOL® [188] simulations for a symmetric chiplet geometry. Such simulation model comprises two metal plates with dimensions defined by the chiplet and electrode geometry, surrounded by a dielectric with properties identical to those of the Isopar-M solution. The capacitances are computed

from the charges that result on each conductor when an electric potential is applied

to one and the other is grounded. Once the capacitance among chiplets and electrodes

at different distances are computed, differentiable parameterized capacitance function

approximations (e.g., linear combination of error functions) can be fitted to that data.

In words, (5.3a) says that the total controlled interaction potential ϕ^u is a sum of the

chiplet-to-chiplet interaction potential ϕ_{cc}^u given by (5.3b), and the chiplet-to-electrode

interaction potential ϕ_{ce}^u given by (5.3c).

The expressions for (5.3b), (5.3c), (5.4) arise from capacitive electrical circuit ab-

straction that lumps the interaction between the electrodes and the chiplets. In [183,

Sec. III], such an abstraction was detailed for a finite population of n chiplets and m

electrodes. The expressions (5.3b), (5.3c), (5.4) generalize those in the limit $n, m \to \infty$.

On the other hand, specializing (5.3b), (5.3c), (5.4) for a finite population $\{\boldsymbol{x}_i\}_{i \in [\![n]\!]}$ with

$\rho \equiv \frac{1}{n} \sum_{i=1}^n \delta_{\boldsymbol{x}_i}$ where $\delta_{\boldsymbol{x}_i}$ denotes the Dirac delta at $\boldsymbol{x}_i \in \mathbb{R}^2$, indeed recovers the devel-

opment in [183, Sec. III].

Remark 5.1. *An immediate observation from* (5.3) *is that even though the potential* ϕ_{cc}^u

is symmetric in $\boldsymbol{x}, \boldsymbol{y}$*, the potential* ϕ_{ce}^u *is not. Therefore, the overall controlled interaction*

potential ϕ^u *is not symmetric in* $\boldsymbol{x}, \boldsymbol{y}$*.*

Without loss of generality, we assume unity viscous coefficient in (5.1), i.e., $\mu = 1$

(since otherwise we can re-scale the $\boldsymbol{f}^u$). In addition, assuming the chiplet velocity is

affected by additive standard Gaussian White noise, the sample path dynamics of the

ith chiplet position $\boldsymbol{x}_i(t)$ then evolves as per a controlled interacting diffusion, i.e., as

a Itô stochastic differential equation (SDE) with *nonlocal* nonlinear drift:

$$\mathrm{d}\boldsymbol{x}_i = \boldsymbol{f}^u(\boldsymbol{x}_i, t, u, \rho)\,\mathrm{d}t + \sqrt{2\beta^{-1}}\,\mathrm{d}\boldsymbol{w}_i(t), \quad i \in [\![n]\!], \tag{5.5}$$

where $\boldsymbol{f}^u$ is given by (5.2), $\beta > 0$ denotes inverse temperature, and $\boldsymbol{w}_i(t) \in \mathbb{R}^2$ denote

i.i.d. realizations of a standard Wiener process that is $\mathcal{F}_t$-adapted on a complete filtered probability space with sigma-algebra $\mathcal{F}$ and associated filtration $(\mathcal{F}_t)_{t \geq 0}$. In particular, $\mathcal{F}_0$ contains all $\mathbb{P}$-null sets and $\mathcal{F}_t$ is right continuous.

The study of SDEs with nonlocal nonlinear drift originated in [189], and has grown into a substantial literature, see e.g., [190, 191]. In statistical physics, such models are often referred to as "propagation of chaos"–a terminology due to Kac [192]. A novel aspect of the model (5.5) w.r.t. the existing literature is that the interaction potential ϕ^u has a nonlinear dependence on the control policy $u(\boldsymbol{x}, t)$ as evident from (5.3).

5.1.1   Existence-uniqueness of solution for (5.5)

For a given causal control policy $u \in \mathcal{U}$, it is known [193, Thm. 2.4] that an interacting diffusion of the form (5.5) with initial condition $\boldsymbol{x}_{i0} \sim \rho_0$ admits unique weak solution provided the following four conditions hold:

(i) the drift $\boldsymbol{f}^u$ is jointly Borel measurable w.r.t. $\mathbb{R}^2 \times [0, \infty) \times \mathcal{P}(\mathbb{R}^2)$,

(ii) the diffusion coefficient $\sqrt{2\beta^{-1}} \boldsymbol{I}_2$ is invertible, and the driftless SDE $\mathrm{d}\boldsymbol{z}(t) = \sqrt{2\beta^{-1}} \, \mathrm{d}\boldsymbol{w}(t)$ admits unique strong solution,

(iii) the drift $\boldsymbol{f}^u$ is uniformly bounded,

(iv) there exists $\kappa > 0$ such that

$$\| \boldsymbol{f}^u (\boldsymbol{x}, t, u(\boldsymbol{x}, t), \rho) - \boldsymbol{f}^u (\boldsymbol{x}, t, u(\boldsymbol{x}, t), \widetilde{\rho}) \|_2$$

$$\leq \kappa \, \mathrm{dist}_{\mathrm{TV}} (\rho, \widetilde{\rho}) \quad \text{uniformly in } (\boldsymbol{x}, t) \in \mathbb{R}^2 \times [0, \infty).$$

We assume that the capacitances $C_{\mathrm{cc}}, C_{\mathrm{ce}}$ in (5.3)-(5.4) are sufficiently smooth, and the control u can be parameterized to ensure smoothness for guaranteeing that $\nabla_{\boldsymbol{x}} \phi_{\mathrm{cc}}^u, \nabla_{\boldsymbol{x}} \phi_{\mathrm{ce}}^u$ (and thus $\nabla_{\boldsymbol{x}} \phi^u$) are $\| \cdot \|_2$ Lipschitz and uniformly bounded.

As $\nabla_{\boldsymbol{x}} \phi^u$ is bounded, $\boldsymbol{f}^u = \int_{\mathbb{R}^2} \nabla_{\boldsymbol{x}} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t) \rho(\boldsymbol{y}) \mathrm{d}\boldsymbol{y}$, which being an average of

Lipschitz, is itself Lipschitz and thus continuous. Since $\boldsymbol{f}^u$ is continuous, the preimage of any Borel set in $\mathbb{R}^2$ under $\boldsymbol{f}^u$ is a measurable set in $\mathbb{R}^2 \times [0, \infty) \times \mathcal{U} \times \mathcal{P}_2(\mathbb{R}^2)$. Thus, condition (i) holds.

Condition (ii) holds for any $\beta > 0$ since $\boldsymbol{z}(t)$ is a Wiener process with variance $2\beta^{-1}$.

For (iii), we find $\operatorname*{ess\,sup}_{(\boldsymbol{x},t)\in\mathbb{R}^2\times[0,\infty)} \left\| \boldsymbol{f}^u\left(\boldsymbol{x}, t, u(\boldsymbol{x}, t), \rho\right) \right\|_\infty$

$$
= \operatorname*{ess\,sup}_{(\boldsymbol{x},t)\in\mathbb{R}^2\times[0,\infty)} \left\| \int_{\mathbb{R}^2} \nabla_{\boldsymbol{x}} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t) \rho(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} \right\|_\infty
$$

$$
\leq \operatorname*{ess\,sup}_{(\boldsymbol{x},t)\in\mathbb{R}^2\times[0,\infty)} \int_{\mathbb{R}^2} \left\| \nabla_{\boldsymbol{x}} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t) \rho(\boldsymbol{y}) \right\|_\infty \mathrm{d}\boldsymbol{y}
$$

$$
\leq \int_{\mathbb{R}^2} \operatorname*{ess\,sup}_{(\boldsymbol{x},t)\in\mathbb{R}^2\times[0,\infty)} \left\| \nabla_{\boldsymbol{x}} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t) \rho(\boldsymbol{y}) \right\|_\infty \mathrm{d}\boldsymbol{y}
$$

$$
= \int_{\mathbb{R}^2} \operatorname*{ess\,sup}_{(\boldsymbol{x},t)\in\mathbb{R}^2\times[0,\infty)} \left\| \nabla_{\boldsymbol{x}} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t) \right\|_\infty \rho(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} \tag{5.6}
$$

where we used the Leibniz rule, triangle inequality, and that $\rho \geq 0$. Per assumption, $\nabla_{\boldsymbol{x}} \phi^u$ is uniformly bounded, and we have: $(5.6) \leq M \int_{\mathbb{R}^2} \rho(\boldsymbol{y}) \mathrm{d}\boldsymbol{y} = M$ for some constant $M > 0$.

Condition (iv) holds because

$$
\left\| \boldsymbol{f}^u\left(\boldsymbol{x}, t, u(\boldsymbol{x}, t), \rho\right) - \boldsymbol{f}^u\left(\boldsymbol{x}, t, u(\boldsymbol{x}, t), \widetilde{\rho}\right) \right\|_2
$$

$$
= \left\| \nabla_{\boldsymbol{x}} \int_{\mathbb{R}^2} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t)(\rho(\boldsymbol{y}) - \widetilde{\rho}(\boldsymbol{y})) \mathrm{d}\boldsymbol{y} \right\|_2
$$

$$
= \left\| \int_{\mathbb{R}^2} \left(\nabla_{\boldsymbol{x}} \phi^u(\boldsymbol{x}, \boldsymbol{y}, t) \right) (\rho(\boldsymbol{y}) - \widetilde{\rho}(\boldsymbol{y})) \mathrm{d}\boldsymbol{y} \right\|_2
$$

$$
\leq c \operatorname{dist}_{\mathrm{BL}}(\rho, \widetilde{\rho}) \leq \kappa \operatorname{dist}_{\mathrm{TV}}(\rho, \widetilde{\rho}) \quad \forall (\boldsymbol{x}, t) \in \mathbb{R}^2 \times [0, \infty)
$$

for some constant $c > 0$, $\kappa := 2c$, and the second to last inequality follows from $\nabla_{\boldsymbol{x}} \phi^u$ being bounded and Lipschitz.

Thus, we can guarantee the existence-uniqueness of sample path $\boldsymbol{x}_i(t)$, $i \in [\![n]\!]$,

solving the interacting diffusion (5.5).

5.1.2 Derivation of the Controlled Mean Field Model

Our next result (Theorem 5.1) derives the macroscopic mean field dynamics as a *nonlinear* Fokker-Planck-Kolmogorov partial differential equation (PDE), and establishes the consistency of the mean field dynamics in the continuum limit vis-à-vis the finite population dynamics.

Theorem 5.1. *Supposing **A1**, consider a population of n interacting chiplets, where the ith chiplet position $\boldsymbol{x}_i \in \mathbb{R}^2$, $i \in [\![n]\!]$, evolves via (5.5). Denote the Dirac measure concentrated at $\boldsymbol{x}_i$ as $\delta_{\boldsymbol{x}_i}$ and let the random empirical measure $\rho^n := \frac{1}{n} \sum_{i=1}^n \delta_{\boldsymbol{x}_i}$. Consider the empirical version of the dynamics (5.5) given by*

$$\mathrm{d}\boldsymbol{x}_i = \boldsymbol{f}^u\left(\boldsymbol{x}_i, t, u, \rho^n\right) \mathrm{d}t + \sqrt{2\beta^{-1}}\,\mathrm{d}\boldsymbol{w}_i(t),$$

with respective initial conditions $\boldsymbol{x}_{0i} \in \mathbb{R}^2$, $i \in [\![n]\!]$, which are independently sampled from a given PDF ρ_0 supported on a subset of $\mathbb{R}^2$. Then, as $n \to \infty$, almost surely $\rho^n \rightharpoonup \rho$ where the deterministic function ρ is a PDF that evolves as per the macroscopic dynamics

$$\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\nabla \cdot \left(\rho \boldsymbol{f}^u\right) + \beta^{-1}\Delta\rho \\
&= \nabla \cdot \left(\rho \nabla \left(\rho * \phi^u + \beta^{-1}\left(1 + \log\rho\right)\right)\right),
\end{aligned} \tag{5.7}$$

with the initial condition

$$\rho(\cdot, t=0) = \rho_0 \in \mathcal{P}\left(\mathbb{R}^2\right) \ (given). \tag{5.8}$$

Notice that the Cauchy problem (5.7)-(5.8) involves a *nonlinear nonlocal PDE* which in turn depends on control policy u.

The solution $\rho(\boldsymbol{x}, t)$, $\boldsymbol{x} \in \mathbb{R}^2$, $t \in [0, \infty)$, for the Cauchy problem (5.7)-(5.8) is

understood in weak sense. In other words, for all compactly supported smooth test functions $\theta \in C_c^\infty \left(\mathbb{R}^2, [0, \infty) \right)$, the solution $\rho(\boldsymbol{x}, t)$ satisfies

$$\int_0^\infty \int_{\mathbb{R}^2} \left(\frac{\partial \theta}{\partial t} + L_\rho \theta \right) \rho \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t + \int_{\mathbb{R}^2} \rho_0(\boldsymbol{x}) \theta(\boldsymbol{x}, 0) \, \mathrm{d}\boldsymbol{x} = 0 \tag{5.9}$$

where L_ρ is defined as in (B.3). Note that, the reason why ρ satisfying (5.9) for all $\theta \in C_c^\infty \left(\mathbb{R}^2, [0, \infty) \right)$ is called a "weak solution" of (5.7)-(5.8) is because such ρ may not be sufficiently smooth to satisfy (5.7). In the next Section, we provide a variational interpretation of the solution for problem (5.7)-(5.8).

5.2 Chiplet Population Dynamics as Wasserstein Gradient Flow

The structure of the PDE in (5.7) motivates defining an *energy functional*

$$\Phi(\rho) := \Phi_{\mathrm{cc}}(\rho) + \Phi_{\mathrm{ce}}(\rho) + \mathbb{E}_\rho \left[\beta^{-1} \log \rho \right]$$

$$= \mathbb{E}_\rho \left[\rho * \phi^u + \beta^{-1} \log \rho \right] \tag{5.10}$$

where $\mathbb{E}_\rho$ denotes the expectation w.r.t. the PDF ρ, and

$$\Phi_{\mathrm{cc}}(\rho) := \int_{\mathbb{R}^2 \times \mathbb{R}^2} \phi_{\mathrm{cc}}^u(\boldsymbol{x}, \boldsymbol{y}) \rho(\boldsymbol{x}) \rho(\boldsymbol{y}) \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{y}, \tag{5.11a}$$

$$\Phi_{\mathrm{ce}}(\rho) := \int_{\mathbb{R}^2 \times \mathbb{R}^2} \phi_{\mathrm{ce}}^u(\boldsymbol{x}, \boldsymbol{y}) \rho(\boldsymbol{x}) \rho(\boldsymbol{y}) \mathrm{d}\boldsymbol{x} \, \mathrm{d}\boldsymbol{y}. \tag{5.11b}$$

In (5.10), the term $\mathbb{E}_\rho \left[\rho * \phi^u \right]$ quantifies the *interaction energy* while the scaled negative entropy term $\beta^{-1} \mathbb{E}_\rho \left[\log \rho \right]$ quantifies the *internal energy*. We have the following result.

Theorem 5.2. *Let* $\Phi : \mathcal{P}_2 \left(\mathbb{R}^2 \right) \mapsto \mathbb{R}$ *be the energy functional given in* (5.10). *Then,*

(i) the chiplet population dynamics given by (5.2), (5.3), (5.7) *is Wasserstein gradient flow*

of the functional Φ, i.e.,

$$\frac{\partial \rho}{\partial t} = -\nabla^W \Phi(\rho). \tag{5.12}$$

(ii) Φ is a Lyapunov functional that is decreasing along the flow generated by (5.7), i.e.,
$\frac{\mathrm{d}}{\mathrm{d}t}\Phi \leq 0$.

Remark 5.2. *Theorem 5.2 shows that for an admissible control policy $u \in \mathcal{U}$, the chiplet*

population dynamics (5.7)-(5.8) can be seen as gradient descent of the energy functional

Φ on the manifold $\mathcal{P}_2(\mathbb{R}^2)$ w.r.t. the Wasserstein metric. We point out that the state-

ment of Theorem 5.2 remains valid in the deterministic limit, i.e., when the noise strength

$\sqrt{2\beta^{-1}} \downarrow 0$. In that case, the functional Φ in (5.10) comprises of only the interaction en-

ergy term $\mathbb{E}_\rho[\rho * \phi^u]$, and $\frac{\delta\Phi}{\delta\rho} = \rho * \phi^u$. Other than this, the proof of Theorem 5.2 remains

unchanged.

Remark 5.3. *In the recent systems-control literature, the Wasserstein gradient flow in-*

terpretations and related proximal algorithms [5, 38] for several linear and nonlinear

Fokker-Planck-Kolmogorov PDEs in prediction and density control have appeared. New

gradient flow interpretations have also appeared [28–30] for well-known filtering equa-

tions. We next point out that the Wasserstein gradient flow interpretation deduced in

Theorem 5.2 allows approximating the weak solution of (5.12) by recursive evaluation of

a Wasserstein proximal operator on the manifold $\mathcal{P}_2(\mathbb{R}^2)$.

Theorem 5.3. *For a given control policy $u \in \mathcal{U}$ and potentials (5.3), let $\widehat{\Phi}(\varrho, \varrho_{k-1}) :=$*

$\mathbb{E}_\varrho[\varrho_{k-1} * \phi^u + \beta^{-1}\log\varrho]$, $\varrho, \varrho_{k-1} \in \mathcal{P}_2(\mathbb{R}^2)$, $k \in \mathbb{N}$. Consider the Wasserstein proximal

recursion:

$$\varrho_k = \mathrm{prox}_{\tau\widehat{\Phi}}^W(\varrho_{k-1})$$

$$:= \underset{\varrho \in \mathcal{P}_2(\mathbb{R}^2)}{\arg\inf}\left\{\frac{1}{2}W^2(\varrho, \varrho_{k-1}) + \tau\,\widehat{\Phi}(\varrho, \varrho_{k-1})\right\} \tag{5.13}$$

over discrete time $t_{k-1} := (k-1)\tau$ with fixed step-size $\tau > 0$, and with initial condition

$\varrho_0 \equiv \rho_0 \in \mathcal{P}_2(\mathbb{R}^2)$. *Let $\rho(\boldsymbol{x}, t)$ be the weak solution of (5.12) for the same $u \in \mathcal{U}$ and the*

functional Φ given by (5.10)-(5.11). Using the sequence of functions $\{\varrho_{k-1}\}_{k \in \mathbb{N}}$ generated

by the recursion (5.13), define an interpolation $\varrho_\tau : \mathbb{R}^2 \times [0, \infty) \mapsto [0, \infty)$ as

$$\varrho_\tau(\boldsymbol{x}, t) := \varrho_{k-1}(\boldsymbol{x}, \tau) \quad \forall\, t \in [(k-1)\tau, k\tau), \quad k \in \mathbb{N}.$$

Then $\varrho_\tau(\boldsymbol{x}, t) \xrightarrow{\tau \downarrow 0} \rho(\boldsymbol{x}, t)$ in $L^1(\mathbb{R}^2)$ for all $t \in [0, \infty)$.

Proof. Follows the development in [194, Sec. 12.3–12.5]. $\square$

Remark 5.4. *For a given control policy $u \in \mathcal{U}$, the Wasserstein proximal recursion (5.13)*

can in turn be leveraged for numerically updating the PDFs over discrete time with a

small step-size τ. To illustrate Theorem (5.2), we fixed a linear control policy $u = \langle \boldsymbol{k}, \boldsymbol{x} \rangle$

with gain $\boldsymbol{k} := (8.5 \times 10^{-3}, -1 \times 10^{-2})^\top$, and solved (5.13) with $\tau = 0.1$ via [5, Algo-

rithm 1] for $n = 400$ uniformly spaced grid samples in the domain $[-4\,\mathrm{mm}, 4\,\mathrm{mm}]^2$

starting from an initial bivariate Gaussian $\rho_0 = \mathcal{N}((0.5, 0.5)^\top, 0.1\boldsymbol{I}_2)$. Fig. 5.1 shows

the corresponding decay of the energy functional Φ in (5.10)-(5.11), computed using

these PDFs obtained from the Wasserstein proximal updates. As in [183, Sec. III], our

simulation used capacitances $C_{\mathrm{cc}}(\|\boldsymbol{x} - \boldsymbol{y}\|_2), C_{\mathrm{ce}}(\|\boldsymbol{x} - \boldsymbol{y}\|_2)$ in (5.3b)-(5.3c) of the form

$\sum_{i=1}^{n} a_i \left[\mathrm{erf}((\|\boldsymbol{x} - \boldsymbol{y}\|_2 + \delta)/c_i) - \mathrm{erf}((\|\boldsymbol{x} - \boldsymbol{y}\|_2 - \delta)/c_i)\right]$ *where $\mathrm{erf}(\cdot)$ denotes the error func-*

tion, the parameters a_i, c_i are sampled uniformly random in $[0, 1]$, and δ (half of the

electrode pitch) = 10 *micrometer.*
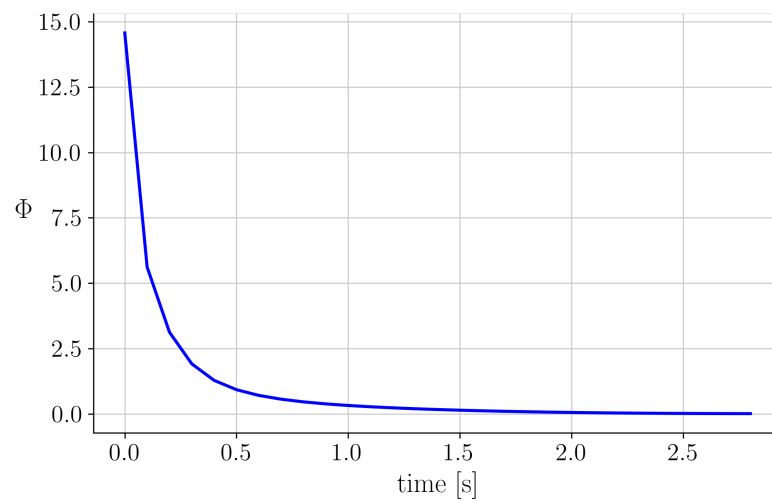
Figure 5.1: The energy functional Φ given by (5.10)-(5.11) versus time for the simulation set up summarized in Remark 5.4.

6 | Centralized Computing: Mean Field Learning

6.1 Empirical Risk Minimization for Supervised Learning

We consider a supervised learning problem where the dataset comprises of the features $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^{n_x}$, and the labels $y \in \mathcal{Y} \subseteq \mathbb{R}$, i.e., the samples of the dataset are tuples of the form

$$(\boldsymbol{x}, y) \in \mathcal{X} \times \mathcal{Y} \subseteq \mathbb{R}^{n_x} \times \mathbb{R}.$$

The objective of the supervised learning problem is to find the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^p$ such that $y \approx f(\boldsymbol{x}, \boldsymbol{\theta})$ where f is some function class parameterized by $\boldsymbol{\theta}$. In other words, f maps from the feature space $\mathcal{X}$ to the label space $\mathcal{Y}$. To this end, we consider a shallow neural network with a single hidden layer having n_{H} neurons. Then, the parameterized function f admits representation

$$f(\boldsymbol{x}, \boldsymbol{\theta}) := \frac{1}{n_{\mathrm{H}}} \sum_{i=1}^{n_{\mathrm{H}}} \Phi\left(\boldsymbol{x}, \boldsymbol{\theta}_i\right), \tag{6.1}$$

where

$$\Phi\left(\boldsymbol{x}, \boldsymbol{\theta}_i\right) := a_i \sigma(\langle \boldsymbol{w}_i, \boldsymbol{x} \rangle + b_i) \tag{6.2}$$

for all $i \in [n_{\mathrm{H}}] := \{1, 2, \cdots, n_{\mathrm{H}}\}$, and $\sigma(\cdot)$ is a smooth activation function. The parameters $a_i, \boldsymbol{w}_i$ and b_i are the scaling, weights, and bias of the ith hidden neuron, respectively, and together comprise the specific parameter realization $\boldsymbol{\theta}_i \in \mathbb{R}^p, i \in [n_{\mathrm{H}}]$.

We stack the parameter vectors of all hidden neurons as

$$\overline{\boldsymbol{\theta}} := \left(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{n_{\mathrm{H}}}\right)^{\top} \in \mathbb{R}^{p n_{\mathrm{H}}}$$

and consider minimizing the following quadratic loss:

$$l(y, \boldsymbol{x}, \overline{\boldsymbol{\theta}}) \equiv l(y, f(\boldsymbol{x}, \overline{\boldsymbol{\theta}})) := \underbrace{\left(y - f(\boldsymbol{x}, \overline{\boldsymbol{\theta}})\right)^2}_{\text{quadratic loss}}. \tag{6.3}$$

We suppose that the training data follows the joint probability distribution γ, i.e., $(\boldsymbol{x}, y) \sim \gamma$. Define the *population risk* R as the expected loss given by

$$R(f) := \mathbb{E}_{(\boldsymbol{x},y)\sim\gamma}[l(y, \boldsymbol{x}, \overline{\boldsymbol{\theta}})]. \tag{6.4}$$

In practice, γ is unknown, so we approximate the population risk with the *empirical risk*

$$R(f) \approx \frac{1}{n_{\text{data}}} \sum_{j=1}^{n_{\text{data}}} l\left(y_j, \boldsymbol{x}_j, \overline{\boldsymbol{\theta}}\right) \tag{6.5}$$

where n_{data} is the number of data samples. Then, the supervised learning problem reduces to the empirical risk minimization problem

$$\min_{\overline{\boldsymbol{\theta}} \in \mathbb{R}^{pn_{\text{H}}}} R(f). \tag{6.6}$$

Problem (6.6) is a large but finite dimensional optimization problem that is nonconvex in the decision variable $\overline{\boldsymbol{\theta}}$. The standard approach is to employ first or second order search algorithms such as the variants of SGD or ADAM [147].

6.1.1 Learning algorithm dynamics: the mean field limit

While universal function approximation theorems for neural networks have been known for long [195, 196], such guarantees do not account the dynamics of the learning algorithms. Starting in 2018, several works [15–18] pointed out that the first order learning dynamics for two layer neural networks in the infinite width (i.e., over-parametrization) limit leads to a PDE that enjoys a remarkable structure.

Specifically, the idea is to consider the continuum limit of hidden layer neuronal

populations by letting $n_H \to \infty$. Then (6.1) becomes the ensemble average

$$f = \int_{\mathbb{R}^p} \Phi(\boldsymbol{x}, \boldsymbol{\theta}) \underbrace{\mathrm{d}\mu(\boldsymbol{\theta})}_{\text{hidden neuronal population mass}} = \int_{\mathbb{R}^p} \Phi(\boldsymbol{x}, \boldsymbol{\theta}) \rho(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta} = \mathbb{E}_{\boldsymbol{\theta}}[\Phi(\boldsymbol{x}, \boldsymbol{\theta})] \quad (6.7)$$

where μ denotes the joint population measure supported on the hidden neuronal parameter space $\mathbb{R}^p$. Assuming the absolute continuity of the joint population measure μ for all times, we write $\mathrm{d}\mu(\boldsymbol{\theta}) = \rho(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}$ where ρ denotes the joint parametric PDF. Consequently, the risk functional R, now viewed as a functional of the joint PDF ρ, takes the form

$$\begin{aligned}
F(\rho) &:= R(f(\boldsymbol{x}, \rho)) = \mathbb{E}_{\Delta}\left[\frac{1}{2}\left\|\boldsymbol{y} - \int_{\mathbb{R}^p} \Phi(\boldsymbol{x}, \boldsymbol{\theta})\rho(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}\right\|_2^2\right] \\
&= F_0 + \int_{\mathbb{R}^p} V(\boldsymbol{\theta})\rho(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta} + \int_{\mathbb{R}^{2p}} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}})\rho(\boldsymbol{\theta})\rho(\tilde{\boldsymbol{\theta}})\mathrm{d}\boldsymbol{\theta}\mathrm{d}\tilde{\boldsymbol{\theta}},
\end{aligned} \quad (6.8)$$

where

$$\begin{aligned}
F_0 &:= \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y})}\left[\|\boldsymbol{y}\|_2^2\right], \\
V(\boldsymbol{\theta}) &:= \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y})}[-2\boldsymbol{y}\Phi(\boldsymbol{x}, \boldsymbol{\theta})], \\
U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) &:= \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y})}[\Phi(\boldsymbol{x}, \boldsymbol{\theta})\Phi(\boldsymbol{x}, \widetilde{\boldsymbol{\theta}})].
\end{aligned} \quad (6.9)$$

Hence the supervised learning problem, in this mean field limit, becomes an infinite dimensional static optimization problem

$$\min_{\rho} F(\rho). \quad (6.10)$$

Notice from (6.9) that F is a sum of three functionals: the first summand being F_0 which is independent of ρ, the second summand being a potential energy given by expected value of "drift potential" V which is linear in ρ, and the last summand being a bilinear interaction energy involving an "interaction potential" U.

The main result in [15] was that using first order stochastic gradient descent learning dynamics induces a gradient flow of the functional F w.r.t. the 2-Wasserstein met-

ric W_2, i.e., the mean field learning dynamics results in a joint PDF trajectory $\rho(t, \boldsymbol{\theta})$. Then the minimizer in (6.10) can be computed from the large t limit of the following nonlinear PDE:

$$\frac{\partial \rho}{\partial t} = -\nabla^{W_2} F(\rho) = -\nabla \cdot \left(\rho \nabla \frac{\delta F}{\delta \rho} \right). \tag{6.11}$$

In particular, [15] considered the regularized risk functional[1]

$$F_\beta(\rho) := F(\rho) + \beta^{-1} \int_{\mathbb{R}^p} \rho \log \rho \, \mathrm{d}\boldsymbol{\theta}, \quad \beta > 0, \tag{6.12}$$

by adding a strictly convex regularizer (scaled negative entropy) to the unregularized risk F. In that case, the sample path dynamics corresponding to the macroscopic dynamics (6.11) precisely becomes the noisy stochastic gradient descent

$$\mathrm{d}\boldsymbol{\theta} = -\nabla_{\boldsymbol{\theta}} \left(V(\boldsymbol{\theta}) + \int_{\mathbb{R}^p} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \rho(\tilde{\boldsymbol{\theta}}) \mathrm{d}\tilde{\boldsymbol{\theta}} \right) \mathrm{d}t + \sqrt{2\beta^{-1}} \, \mathrm{d}\boldsymbol{w}, \tag{6.13}$$

where $\boldsymbol{w}$ is the standard Wiener process in $\mathbb{R}^p$. In this regularized case, (6.11) becomes

$$\frac{\partial \rho}{\partial t} = \nabla_{\boldsymbol{\theta}} \cdot \left(\rho \left(V(\boldsymbol{\theta}) + \int_{\mathbb{R}^p} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \rho(\tilde{\boldsymbol{\theta}}) \mathrm{d}\tilde{\boldsymbol{\theta}} \right) \right) + \beta^{-1} \Delta_{\boldsymbol{\theta}} \rho. \tag{6.14}$$

In [15], asymptotic guarantees were obtained for the solution of (6.14) to converge to the minimizer of F_β. Our idea outlined next, is to solve the minimization of F_β using measure-valued proximal recursions.

6.1.2 Proximal mean field learning

For numerically computing the solution of the PDE IVP (6.14), we propose proximal recursions over $\mathcal{P}_2(\mathbb{R}^p)$, defined as the manifold of joint PDFs supported over $\mathbb{R}^p$ having finite second moments. Symbolically,

$$\mathcal{P}_2(\mathbb{R}^p) := \left\{ \text{Lebesgue integrable } \rho \text{ over } \mathbb{R}^p \mid \rho \geq 0, \int_{\mathbb{R}^p} \rho \, \mathrm{d}\boldsymbol{\theta} = 1, \int_{\mathbb{R}^p} \boldsymbol{\theta}^\top \boldsymbol{\theta} \rho \, \mathrm{d}\boldsymbol{\theta} < \infty \right\}.$$

---

[1]The parameter $\beta > 0$ is referred to as the inverse temperature.

Proximal updates generalize the concept of gradient steps, and are of significant interest in both finite and infinite dimensional optimization [61, 197–201]. For a given input, these updates take the form of a structured optimization problem:

proximal update

$$
= \underset{\text{decision variable}}{\arg \inf} \left\{ \frac{1}{2} \text{dist}^2 \left(\text{decision variable}, \text{input} \right) \right.
$$

$$
\left. + \text{time step} \times \text{functional} \left(\text{decision variable} \right) \right\},
$$

(6.15)

for some suitable notion of distance $\text{dist}(\cdot, \cdot)$ on the space of decision variables, and some associated functional. It is usual to view (6.15) as an *operator* mapping input $\mapsto$ proximal update, thus motivating the term *proximal operator*.

The connection between (6.15) and the gradient flow comes from recursively evaluating (6.15) with some initial choice for the input.

For suitably designed pair $(\text{dist}(\cdot, \cdot), \text{functional})$, in the small time step limit, the sequence of proximal updates generated by (6.15) converge to the infimum of the functional. In other words, the gradient descent of the functional w.r.t. dist may be computed as the fixed point of the proximal operator (6.15). For a parallel between gradient descent and proximal recursions in finite and infinite dimensional gradient descent, see e.g., [28, Sec. I]. Infinite dimensional proximal recursions over the manifold of PDFs have recently appeared in uncertainty propagation [24, 202], stochastic filtering [28–30], and stochastic optimal control [38, 39].

In our context, the decision variable $\rho \in \mathcal{P}_2 (\mathbb{R}^p)$ and the distance metric $\text{dist} \equiv W_2$. Specifically, we propose recursions over discrete time $t_{k-1} := (k-1)h$ where the index $k \in \mathbb{N}$, and $h > 0$ is a constant time step-size. Leveraging that (6.14) describes gradient flow of the functional F_β w.r.t. the W_2 distance metric, the associated proximal

recursion is of the form

$$\varrho_k = \text{prox}_{hF_\beta}^{W_2}(\varrho_{k-1}) := \underset{\varrho \in \mathcal{P}_2(\mathbb{R}^p)}{\arg\inf} \left\{ \frac{1}{2} \left(W_2(\varrho, \varrho_{k-1}) \right)^2 + h\, F_\beta(\varrho) \right\} \tag{6.16}$$

where $\varrho_{k-1}(\cdot) := \varrho(\cdot, t_{k-1})$, and $\varrho_0 \equiv \rho_0$. The notation $\text{prox}_{hF_\beta}^{W_2}(\varrho_{k-1})$ can be parsed as

"the proximal operator of the scaled functional hF_β w.r.t. the distance W_2, acting on

the input $\varrho_{k-1} \in \mathcal{P}_2(\mathbb{R}^p)$". Our idea is to evaluate the recursion in the small h limit,

i.e., for $h \downarrow 0$.

To account for the nonconvex bilinear term appearing in (6.12), following [203, Sec.

4], we employ the approximation:

$$\int_{\mathbb{R}^{2p}} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \varrho(\boldsymbol{\theta}) \varrho(\tilde{\boldsymbol{\theta}}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\tilde{\boldsymbol{\theta}} \approx \int_{\mathbb{R}^{2p}} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \varrho(\boldsymbol{\theta}) \varrho_{k-1}(\tilde{\boldsymbol{\theta}}) \mathrm{d}\boldsymbol{\theta} \mathrm{d}\tilde{\boldsymbol{\theta}} \quad \forall k \in \mathbb{N}.$$

We refer to the resulting approximation of F_β as $\hat{F}_\beta$, i.e.,

$$\hat{F}_\beta(\varrho, \varrho_{k-1}) := \int_{\mathbb{R}^p} \left(F_0 + V(\boldsymbol{\theta}) + \left(\int_{\mathbb{R}^p} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \varrho_{k-1}(\tilde{\boldsymbol{\theta}}) \mathrm{d}\tilde{\boldsymbol{\theta}} \right) + \beta^{-1} \log \varrho(\boldsymbol{\theta}) \right) \varrho(\boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}.$$

Notice in particular that $\hat{F}_\beta$ depends on both ϱ, ϱ_{k-1}, $k \in \mathbb{N}$. Consequently, this ap-

proximation results in a *semi-implicit* variant of (6.16), given by

$$\varrho_k := \underset{\varrho \in \mathcal{P}_2(\mathbb{R}^p)}{\arg\inf} \left\{ \frac{1}{2} \left(W_2(\varrho, \varrho_{k-1}) \right)^2 + h\hat{F}_\beta(\varrho, \varrho_{k-1}) \right\}. \tag{6.17}$$

We have the following consistency guarantee, among the solution of the PDE IVP

(6.14) and that of the variational recursions (6.17). The rigorous statement and proof

are provided in Appendix C.

The rigorous statement and proof are provided in Appendix C.

Theorem 6.1. *(**Informal**) Consider the regularized risk functional (6.12) wherein F is*

given by (6.8)-(6.9). In the small time step ($h \downarrow 0$) limit, the proximal updates (6.17) with

$\varrho_0 \equiv \rho_0$ converge to the solution for the PDE IVP (6.14).

We next detail the proposed algorithmic approach to numerically solve (6.17).

6.2 PROXLEARN: Proposed Proximal Algorithm

The overall workflow of our proposed proximal mean field learning framework is shown in Fig. 6.1. We generate N samples from the known initial joint PDF ϱ_0 and store them as a weighted point cloud $\{\boldsymbol{\theta}_0^i, \varrho_0^i\}_{i=1}^N$. Here, $\varrho_0^i := \varrho_0\left(\boldsymbol{\theta}_0^i\right)$ for all $i \in [N]$. In other words, the weights of the samples are the joint PDF values evaluated at those samples.

For each $k \in \mathbb{N}$, the weighted point clouds $\left\{\boldsymbol{\theta}_k^i, \varrho_k^i\right\}_{i=1}^N$ are updated through the two-step process outlined in our proposed Algorithm 2, referred to as PROXLEARN. At a high level, lines 9–18 in Algorithm 1 perform nonlinear block co-ordinate recursion on internally defined vectors $\boldsymbol{z}, \boldsymbol{q}$ whose converged values yield the proximal update (line 19). We next explain where these recursions come from detailing both the derivation of PROXLEARN and its convergence guarantee.

6.2.1 Derivation of PROXLEARN

The main idea behind our derivation that follows, is to regularize and dualize the discrete version of the optimization problem in (6.17). This allows us to leverage certain structure of the optimal solution that emerges from the first order conditions for optimality, which in turn helps design a custom numerical recursion.

Specifically, to derive the recursion given in PROXLEARN, we first write the discrete version of (6.17) as

$$\varrho_k = \arg\min_{\boldsymbol{\varrho}}\left\{\min_{\boldsymbol{M}\in\Pi(\boldsymbol{\varrho}_{k-1},\boldsymbol{\varrho})} \frac{1}{2}\langle\boldsymbol{C}_k, \boldsymbol{M}\rangle + h\left\langle\boldsymbol{v}_{k-1} + \boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1} + \beta^{-1}\log\boldsymbol{\varrho}, \boldsymbol{\varrho}\right\rangle\right\}, \quad k \in \mathbb{N},$$

(6.18)

where

$$\Pi\left(\boldsymbol{\varrho}_{k-1}, \boldsymbol{\varrho}\right) := \{\boldsymbol{M} \in \mathbb{R}^{N \times N} \mid \boldsymbol{M} \geq \boldsymbol{0} \text{ (elementwise)}, \boldsymbol{M}\boldsymbol{1} = \boldsymbol{\varrho}_{k-1}, \, \boldsymbol{M}^{\top}\boldsymbol{1} = \boldsymbol{\varrho}\}, \quad (6.19)$$

$$\boldsymbol{v}_{k-1} \equiv V\left(\boldsymbol{\theta}_{k-1}\right), \tag{6.20}$$

$$\boldsymbol{U}_{k-1} \equiv U\left(\boldsymbol{\theta}_{k-1}, \tilde{\boldsymbol{\theta}}_{k-1}\right), \tag{6.21}$$

and $\boldsymbol{C}_k \in \mathbb{R}^{N \times N}$ denotes the squared Euclidean distance matrix, i.e.,

$$\boldsymbol{C}_k(i, j) := \|\boldsymbol{\theta}_k^i - \boldsymbol{\theta}_{k-1}^j\|_2^2 \quad \forall (i, j) \in [N] \times [N].$$

We next follow a "regularize-then-dualize" approach. In particular, we regularize (6.18) by adding the entropic regularization $H(\boldsymbol{M}) := \langle \boldsymbol{M}, \log \boldsymbol{M} \rangle$, and write

$$\boldsymbol{\varrho}_k = \arg\min_{\boldsymbol{\varrho}} \left\{ \min_{\boldsymbol{M} \in \Pi(\boldsymbol{\varrho}_{k-1}, \boldsymbol{\varrho})} \frac{1}{2} \langle \boldsymbol{C}_k, \boldsymbol{M} \rangle + \epsilon H(\boldsymbol{M}) + h \left\langle \boldsymbol{v}_{k-1} + \boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1} + \beta^{-1}\log\boldsymbol{\varrho}, \boldsymbol{\varrho} \right\rangle \right\},$$

$$(6.22)$$

where $k \in \mathbb{N}$ and $\epsilon > 0$ is a regularization parameter.

Following [204, Lemma 3.5], [5, Sec. III], the Lagrange dual problem associated with (6.22) is

$$\left(\boldsymbol{\lambda}_0^{\text{opt}}, \boldsymbol{\lambda}_1^{\text{opt}}\right) =$$

$$\arg\max_{\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1 \in \mathbb{R}^N} \left\{ \langle \boldsymbol{\lambda}_0, \boldsymbol{\varrho}_{k-1} \rangle - \hat{F}_\beta^\star\left(-\boldsymbol{\lambda}_1\right) - \frac{\epsilon}{h} \left(\exp\left(\boldsymbol{\lambda}_0^{\top}h/\epsilon\right)\exp\left(-\boldsymbol{C}_k/2\epsilon\right)\exp\left(\boldsymbol{\lambda}_1 h/\epsilon\right)\right) \right\} \tag{6.23}$$

where

$$\hat{F}_\beta^\star(\cdot) := \sup_{\vartheta}\{\langle \cdot, \vartheta \rangle - \hat{F}_\beta(\vartheta)\} \tag{6.24}$$

is the Legendre-Fenchel conjugate of the free energy $\hat{F}_\beta$ in (6.17), and the optimal coupling matrix $\boldsymbol{M}^{\text{opt}} := \left[m^{\text{opt}}(i, j)\right]_{i,j=1}^{N}$ in (6.22) has the Sinkhorn form

$$m^{\text{opt}}(i, j) = \exp\left(\boldsymbol{\lambda}_0(i)h/\epsilon\right)\exp\left(-\boldsymbol{C}_k(i, j)/(2\epsilon)\right)\exp\left(\boldsymbol{\lambda}_1(j)h/\epsilon\right). \tag{6.25}$$

To solve (6.23), considering (6.12), we write the "discrete free energy" as

$$\hat{F}_\beta(\boldsymbol{\varrho}) = \left\langle \boldsymbol{v}_{k-1} + \boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1} + \beta^{-1}\log\boldsymbol{\varrho}, \boldsymbol{\varrho} \right\rangle. \tag{6.26}$$

Its Legendre-Fenchel conjugate, by (6.24), is

$$\hat{F}_\beta^\star(\boldsymbol{\lambda}) := \sup_\vartheta \{\boldsymbol{\lambda}^\top\boldsymbol{\varrho} - \boldsymbol{v}_{k-1}^\top\boldsymbol{\varrho} - \boldsymbol{\varrho}^\top\boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1} - \beta^{-1}\boldsymbol{\varrho}^\top\log\boldsymbol{\varrho}\}. \tag{6.27}$$

Setting the gradient of the objective in (6.27) w.r.t. $\boldsymbol{\varrho}$ to zero, and solving for $\boldsymbol{\varrho}$ gives

the maximizer

$$\boldsymbol{\varrho}_{\max} = \exp\left(\beta\left(\boldsymbol{\lambda} - \boldsymbol{v}_{k-1} - \beta^{-1}\mathbf{1} - \boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1}\right)\right). \tag{6.28}$$

Substituting (6.28) back into (6.27), we obtain

$$\hat{F}_\beta^\star(\boldsymbol{\lambda}) = \beta^{-1}\mathbf{1}\exp\left(\beta\left(\boldsymbol{\lambda} - \boldsymbol{v}_{k-1} - \boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1}\right) - \mathbf{1}\right). \tag{6.29}$$

Fixing $\boldsymbol{\lambda}_0$, and taking the gradient of the objective in (6.23) w.r.t. $\boldsymbol{\lambda}_1$ gives

$$\exp\left(\boldsymbol{\lambda}_1 h/\epsilon\right) \odot \left(\exp\left(-\boldsymbol{C}_k/2\epsilon\right)^\top \exp\left(\boldsymbol{\lambda}_0 h/\epsilon\right)\right) =$$
$$\exp(-\beta\boldsymbol{v}_{k-1} - \beta\boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1} - \mathbf{1}) \odot \left(\exp\left(\boldsymbol{\lambda}_1 h/\epsilon\right)\right)^{-\frac{\beta\epsilon}{h}}. \tag{6.30}$$

Likewise, fixing $\boldsymbol{\lambda}_1$, and taking the gradient of the objective in (6.23) w.r.t. $\boldsymbol{\lambda}_0$, gives

$$\exp\left(\boldsymbol{\lambda}_0 h/\epsilon\right) \odot \left(\exp\left(-\boldsymbol{C}_k/2\epsilon\right)\exp\left(\boldsymbol{\lambda}_1 h/\epsilon\right)\right) = \boldsymbol{\varrho}_{k-1}. \tag{6.31}$$

Next, letting $\boldsymbol{\Gamma}_k := \exp\left(-\boldsymbol{C}_k/2\epsilon\right)$, $\boldsymbol{q} := \exp\left(\boldsymbol{\lambda}_0 h/\epsilon\right)$, $\boldsymbol{z} := \exp\left(\boldsymbol{\lambda}_1 h/\epsilon\right)$, and $\boldsymbol{\xi}_{k-1} :=$
$\exp(-\beta\boldsymbol{v}_{k-1} - \beta\boldsymbol{U}_{k-1}\boldsymbol{\varrho}_{k-1} - \mathbf{1})$, we express (6.30) as

$$\boldsymbol{z} \odot \left(\boldsymbol{\Gamma}_k^\top\boldsymbol{q}\right) = \boldsymbol{\xi}_{k-1} \odot \boldsymbol{z}^{-\frac{\beta\epsilon}{h}}, \tag{6.32}$$

and (6.31) as

$$\boldsymbol{q} \odot \left(\boldsymbol{\Gamma}_k\boldsymbol{z}\right) = \boldsymbol{\varrho}_{k-1}. \tag{6.33}$$

Finally using (6.19), we obtain

$$\boldsymbol{\varrho}_k = \left(\boldsymbol{M}^{\mathrm{opt}}\right)^\top \mathbf{1} = \sum_{j=1}^N m^{\mathrm{opt}}(j,i) = \boldsymbol{z}(i) \sum_{j=1}^N \boldsymbol{\Gamma}_k(j,i) \boldsymbol{q}(j) = \boldsymbol{z} \odot \boldsymbol{\Gamma}_k^\top \boldsymbol{q}. \qquad (6.34)$$

In summary, (6.34) allows us to numerically perform the proximal update.

Remark 6.1. *Note that in Algorithm 2 (i.e., ProxLearn) presented in Section 6.2, the lines 11, 12, and 19 correspond to (6.32), (6.33) and (6.34), respectively.*

6.2.2   Convergence of ProxLearn

Our next result provides the convergence guarantee for our proposed ProxLearn algorithm derived in Section 6.2.1.

Proposition 6.1. *The recursions given in lines 7–18 in Algorithm 2 ProxLearn, converge to a unique fixed point $(\boldsymbol{q}^{\mathrm{opt}}, \boldsymbol{z}^{\mathrm{opt}}) \in \mathbb{R}_{>0}^N \times \mathbb{R}_{>0}^N$. Consequently, the proximal update (6.34) (i.e., the evaluation at line 19 in Algorithm 2) is unique.*

Proof. Notice that the mappings $(\boldsymbol{q}(:,\ell), \boldsymbol{z}(:,\ell)) \mapsto (\boldsymbol{q}(:,\ell+1), \boldsymbol{z}(:,\ell+1))$ given in lines 11 and 12 in Algorithm 2, are cone preserving since these mappings preserve the product orthant $\mathbb{R}_{>0}^N \times \mathbb{R}_{>0}^N$. This is a direct consequence of the definition of $\boldsymbol{q}, \boldsymbol{z}$ in terms of $\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1$.

Now the idea is to show that the recursions in lines 11 and 12 in Algorithm 2, as composite nonlinear maps, are in fact contractive w.r.t. a suitable metric on this cone. Following [5, Theorem 3], the $\boldsymbol{z}$ iteration given in line 11 in Algorithm 2, ProxLearn, for $\ell = 1, 2, \ldots$, is strictly contractive in the Thompson's part metric [62] and thanks to the Banach contraction mapping theorem, converges to a unique fixed point $\boldsymbol{z}^{\mathrm{opt}} \in \mathbb{R}_{>0}^N$.

We note that our definition of $\boldsymbol{\xi}_{k-1}$ is slightly different compared to the same in [5, Theorem 3], but this does not affect the proof. From definition of $\boldsymbol{C}_k$, we have

$C_k \in [0, \infty)$ which implies $\Gamma_k(i, j) \in (0, 1]$. Therefore, Γ_k is a positive linear map for each $k \in \mathbb{N}$. Thus, by (linear) Perron-Frobenius theorem, the linear maps Γ_k are contractive. Consequently the q iterates also converge to unique fixed point $q^{\mathrm{opt}} \in \mathbb{R}_{>0}^N$.

Since converged pair $(q^{\mathrm{opt}}, z^{\mathrm{opt}}) \in \mathbb{R}_{>0}^N \times \mathbb{R}_{>0}^N$ is unique, so is the proximal update (6.34), i.e., the evaluation at line 19 in Algorithm 2. $\qquad \square$

We next discuss the implementation details for the proposed ProxLearn algorithm.

6.2.3   Implementation of ProxLearn

We start by emphasizing that ProxLearn updates both the parameter sample locations $\boldsymbol{\theta}_k^i$ and the joint PDF values ϱ_k^i evaluated at those locations, without gridding the parameter space. In particular, the PDF values are updated online, not as an offline *a posteriori* function approximation as in traditional Monte Carlo algorithms.

We will apply ProxLearn, as outlined here, in Section 6.3. In Section 6.4, we will detail additional modifications of ProxLearn to showcase its flexibility.

Required inputs of ProxLearn are the inverse temperature β, the time step-size h, a regularization parameter ε, and the number of samples N. Additional required inputs are the training feature data $\boldsymbol{X} := \begin{bmatrix} \boldsymbol{x}_1 \dots \boldsymbol{x}_{n_{\mathrm{data}}} \end{bmatrix}^\top \in \mathbb{R}^{n_{\mathrm{data}} \times n_x}$ and the corresponding training labels $\boldsymbol{y} := \begin{bmatrix} y_1 \dots y_{n_{\mathrm{data}}} \end{bmatrix}^\top \in \mathbb{R}^{n_{\mathrm{data}}}$, as well the weighted point cloud $\{\boldsymbol{\theta}_{k-1}^i, \varrho_{k-1}^i\}_{i=1}^N$ for each $k \in \mathbb{N}$. Furthermore, ProxLearn requires two internal parameters as user input: the numerical tolerance δ, and the maximum number of iterations L.
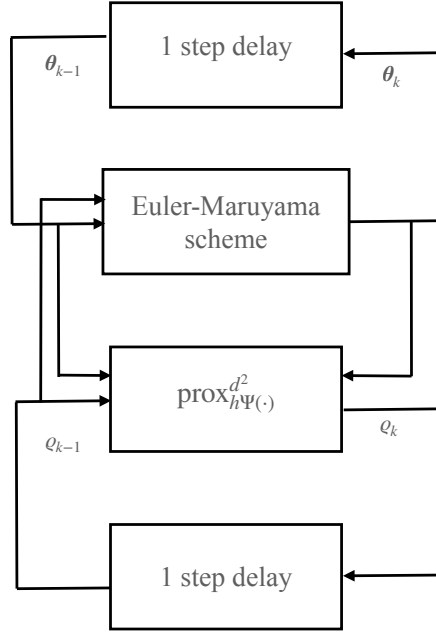
Figure 6.1: Schematic of the proposed proximal algorithm for mean field learning, updating scattered point cloud $\left\{\boldsymbol{\theta}_{k-1}^{i}, \varrho_{k-1}^{i}\right\}_{i=1}^{N}$ for $k \in \mathbb{N}$. The location of the points $\left\{\boldsymbol{\theta}_{k-1}^{i}\right\}_{i=1}^{N}$ are updated via the Euler-Maruyama scheme; the corresponding probability weights are computed via proximal updates highlighted within the dashed box. Explicitly performing the proximal updates via the proposed algorithm, and thereby enabling mean filed learning as an interacting weighted particle system, is our novel contribution

.

For $k \in \mathbb{N}$, let

$$
\boldsymbol{\Theta}_{k-1} := \begin{pmatrix} (\boldsymbol{\theta}_{k-1}^{1})^{\top} \\ (\boldsymbol{\theta}_{k-1}^{2})^{\top} \\ \vdots \\ (\boldsymbol{\theta}_{k-1}^{N})^{\top} \end{pmatrix} \in \mathbb{R}^{N \times p}, \quad \boldsymbol{\varrho}_{k-1} := \begin{pmatrix} \varrho_{k-1}^{1} \\ \varrho_{k-1}^{2} \\ \vdots \\ \varrho_{k-1}^{N} \end{pmatrix} \in \mathbb{R}_{>0}^{N}.
$$

In line 2 of Algorithm 2, PROXLEARN updates the locations of the parameter vector

samples $\boldsymbol{\theta}_{k}^{i}$ in $\mathbb{R}^{p}$ via Algorithm 3, EULERMARUYAMA. This location update takes the

form:

$$\boldsymbol{\theta}_k^i = \boldsymbol{\theta}_{k-1}^i - h\nabla\left(V\left(\boldsymbol{\theta}_{k-1}^i\right) + \omega\left(\boldsymbol{\theta}_{k-1}^i\right)\right) + \sqrt{2\beta^{-1}}\left(\boldsymbol{\eta}_k^i - \boldsymbol{\eta}_{k-1}^i\right), \qquad (6.35)$$

where $\omega(\cdot) := \int U(\cdot, \tilde{\boldsymbol{\theta}})\varrho(\tilde{\boldsymbol{\theta}})\mathrm{d}\tilde{\boldsymbol{\theta}}$, and $\boldsymbol{\eta}_{k-1}^i := \boldsymbol{\eta}^i(t = (k-1)h)$, $\forall k \in \mathbb{N}$.

To perform this update, EulerMaruyama constructs a matrix $\boldsymbol{P}_{k-1}$ whose (i,j)th element is $\boldsymbol{P}_{k-1}(i,j) = \Phi(\boldsymbol{x}_j, \boldsymbol{\theta}_{k-1}^i)$. From $\boldsymbol{P}_{k-1}$, we construct $\boldsymbol{v}_{k-1}$ and $\boldsymbol{U}_{k-1}$ as in lines 3 and 5. In line 6 of Algorithm 3, EulerMaruyama uses the automatic differentiation module of PyTorch Library, Backward [205], to calculate the gradients needed to update $\boldsymbol{\Theta}_{k-1}$ to $\boldsymbol{\Theta}_k$ $\forall k \in \mathbb{N}$.

Once $\boldsymbol{\Theta}_k$, $\boldsymbol{v}_{k-1}$, and $\boldsymbol{U}_{k-1}$ have been constructed via EulerMaruyama, ProxLearn maps the $N \times 1$ vector $\boldsymbol{\varrho}_{k-1}$ to the proximal update $\boldsymbol{\varrho}_k$.

We next illustrate the implementation of ProxLearn for binary and multi-class classification case studies. A GitHub repository containing our code for the implementation of these applications can be found at `https://github.com/zalexis12/Proximal-Mean-Field-Learning.git`. Please refer to the Readme file therein for an outline of the structure of our code.

6.3   Case study: Binary Classification

In this Section, we report numerical results for our first case study, where we apply the proposed ProxLearn algorithm for binary classification.

For this case study, we perform two implementations on different computing platforms. Our first implementation is on a PC with 3.4 GHz 6-Core Intel Core i5 processor, and 8 GB RAM. For runtime improvement, we then use a Jetson TX2 with a NVIDIA Pascal GPU with 256 CUDA cores, 64-bit NVIDIA Denver and ARM Cortex-A57 CPUs.

Algorithm 2 Proximal Algorithm

1: **procedure** PROXLEARN($\varrho_{k-1}, \Theta_{k-1}, \beta, h, \varepsilon, N, X, y, \delta, L$)
2:     $v_{k-1}, U_{k-1}, \Theta_k \leftarrow$ EULERMARUYAMA($h, \beta, \Theta_{k-1}, X, y, \varrho_{k-1}$)   ▷ Update the location of the samples
3:     $C_k(i,j) \leftarrow \left\| \theta_k{}^i - \theta_{k-1}^j \right\|_2^2$
4:     $\Gamma_k \leftarrow \exp(-C_k/2\varepsilon)$ ▷ Lines 4-8 define the terms needed in re-expressing (6.30) as (6.32)
5:     $\xi_{k-1} \leftarrow \exp(-\beta v_{k-1} - \beta U_{k-1}\varrho_{k-1} - 1)$
6:     $z_0 \leftarrow \mathrm{rand}_{N\times 1}$
7:     $z \leftarrow [z_0, 0_{N\times(L-1)}]$
8:     $q \leftarrow [\varrho_{k-1} \oslash (\Gamma_k z_0), 0_{N\times(L-1)}]$
9:     $\ell = 1$
10:    **while** $\ell \leq L$ **do**
11:        $z(:,\ell+1) \leftarrow (\xi_{k-1} \oslash (\Gamma_k^\top q(:,\ell)))^{\frac{1}{1+\beta\varepsilon/h}}$ ▷ Following (6.32)
12:        $q(:,\ell+1) \leftarrow \varrho_{k-1} \oslash (\Gamma_k z(:,\ell+1))$ ▷ Following (6.33)
13:        **if** $\|q(:,\ell+1) - q(:,\ell)\| < \delta$ and $\|z(:,\ell+1) - z(:,\ell)\| < \delta$ **then**
14:            Break
15:        **else**
16:            $\ell \leftarrow \ell + 1$
17:        **end if**
18:    **end while**
19:    **return** $\varrho_k \leftarrow z(:,\ell) \odot (\Gamma_k^\top q(:,\ell))$ ▷ Use (6.34) to map ϱ_{k-1} to ϱ_k
20: **end procedure**

6.3.1   WDBC data set

We apply the proposed algorithm to perform a binary classification on the Wisconsin Diagnostic Breast Cancer (henceforth, WDBC) data set available at the UC Irvine machine learning repository [206]. This data set consists of the data of scans from 569 patients. There are $n_x = 30$ features from each scan. Scans are classified as "benign" (which we label as −1) or "malignant" (labeled as +1).

In (6.1), we define $\Phi\left(x, \theta_{k-1}^i\right) := a_{k-1}^i \tanh(\langle w_{k-1}^i, x \rangle + b_{k-1}^i) \; \forall i \in [N]$ after $(k-1)$ updates. The parameters a_{k-1}^i, w_{k-1}^i and b_{k-1}^i are the scaling, weight and bias of the ith sample after $(k-1)$ updates, respectively. Letting $p := n_x + 2$, the parameter vector of

---

Algorithm 3 Euler-Maruyama Algorithm

1: **procedure** EULERMARUYAMA$(h, \beta, \boldsymbol{\Theta}_{k-1}, \boldsymbol{X}, \boldsymbol{y}, \boldsymbol{\varrho}_{k-1})$
2:      $\boldsymbol{P}_{k-1} \leftarrow \boldsymbol{\Phi}(\boldsymbol{\Theta}_{k-1}, \boldsymbol{X})$ $\triangleright$ Lines 2-4 construct the argument of the gradient in (6.35)
3:      $\boldsymbol{U}_{k-1} \leftarrow 1/n_{\text{data}} \boldsymbol{P}_{k-1} \boldsymbol{P}_{k-1}^{\top}$
4:      $\boldsymbol{u}_{k-1} \leftarrow \boldsymbol{U}_{k-1} \boldsymbol{\varrho}_{k-1}$
5:      $\boldsymbol{v}_{k-1} \leftarrow -2/n_{\text{data}} \boldsymbol{P}_{k-1} \boldsymbol{y}$
6:      $\boldsymbol{D} \leftarrow$ BACKWARD $(\boldsymbol{u}_{k-1} + \boldsymbol{v}_{k-1})$ $\triangleright$ Approximate the gradient of (6.35) using PyTorch library BACKWARD [205]
7:      $\boldsymbol{G} \leftarrow \sqrt{2h/\beta} \times \text{randn}_{N \times p}$
8:      $\boldsymbol{\Theta}_{k} \leftarrow \boldsymbol{\Theta}_{k-1} + h \times \boldsymbol{D} + \boldsymbol{G}$ $\triangleright$ Complete the location update via (6.35)
9: **end procedure**

---

the ith sample after $(k-1)$ updates is

$$\boldsymbol{\theta}_{k-1}^{i} := \begin{pmatrix} a_{k-1}^{i} \\ b_{k-1}^{i} \\ \boldsymbol{w}_{k-1}^{i} \end{pmatrix} \in \mathbb{R}^{p}, \quad \forall\, i \in [N].$$

We set $\varrho_0 \equiv \text{Unif}([0.9, 1.1] \times [-0.1, 0.1] \times [-1, 1]^{n_x})$, a uniform joint PDF supported over $n_p = n_x + 2 = 32$ dimensional mean field parameter space.

We use 70% of the entire data set as training data. As discussed in Section 6.2, we learn the mean field parameter distribution via weighted scattered point cloud evolution using PROXLEARN. We then use the confusion matrix method [207] to evaluate the accuracy of the obtained model over the test data, which is the remaining 30% of the full data set, containing n_{test} points.

For each test point $\boldsymbol{x}_{\text{test}} \in \mathbb{R}^{n_x}$, we construct

$$\boldsymbol{\varphi}(\boldsymbol{x}_{\text{test}}) := \begin{pmatrix} \Phi(\boldsymbol{x}_{\text{test}}, \boldsymbol{\theta}_{k-1}^{1}) \\ \Phi(\boldsymbol{x}_{\text{test}}, \boldsymbol{\theta}_{k-1}^{2}) \\ \vdots \\ \Phi(\boldsymbol{x}_{\text{test}}, \boldsymbol{\theta}_{k-1}^{N}) \end{pmatrix} \in \mathbb{R}^{N}$$

where $\boldsymbol{\theta}^i_{k-1}$ is obtained from the training process. We estimate $f_{\mathrm{MeanField}}$ in (6.7) in two ways. First, we estimate $f_{\mathrm{MeanField}}$ as a sample average of the elements of $\boldsymbol{\varphi}$. Second, we estimate $f_{\mathrm{MeanField}}$ by numerically approximating the integral in (6.7) using the propagated samples $\{\boldsymbol{\theta}^i_{k-1}, \varrho^i_{k-1}\}^N_{i=1}$ for $k \in \mathbb{N}$. We refer to these as the "unweighted estimate" and "weighted estimate," respectively. While the first estimate is an empirical average, the second uses the weights $\{\varrho^i_{k-1}\}^N_{i=1}$ obtained from the proposed proximal algorithm. The $f_{\mathrm{MeanField}}$ "unweighted estimate" and "weighted estimate" are then passed through the SOFTMAX and ARGMAX functions respectively, to produce the predicted labels.

6.3.2   Numerical experiments

We set the number of samples $N = 1000$, numerical tolerance $\delta = 10^{-3}$, the maximum number of iterations $L = 300$, and the regularizing parameter $\varepsilon = 1$. Additionally, we set the time step to $h = 10^{-3}$. We run the simulation for different values of the inverse temperature β, and list the corresponding classification accuracy in Table 6.1. The "weighted estimate," the ensemble average using proximal updates, produces more accurate results, whereas the "unweighted estimate," the empirical average, is found to be more sensitive to the inverse temperature β.

For each fixed β, we perform 10^6 proximal recursions incurring approx. 33 hours of computational time. Fig. 6.2a shows the risk functional, computed as the averaged loss over the test data using each of the two estimates described above.

To improve the runtime of our algorithm, we run our code on a Jetson TX2 module, converting data and variables to PyTorch variables.

We begin calculations in Float32, switching to Float64 only when needed to avoid not-a-number (NaN) errors. This switch typically occurs after 2×10^5 to 3×10^5 iter-

(a) Regularized risk calculated via CPU.   (b) Regularized risk calculated via GPU.
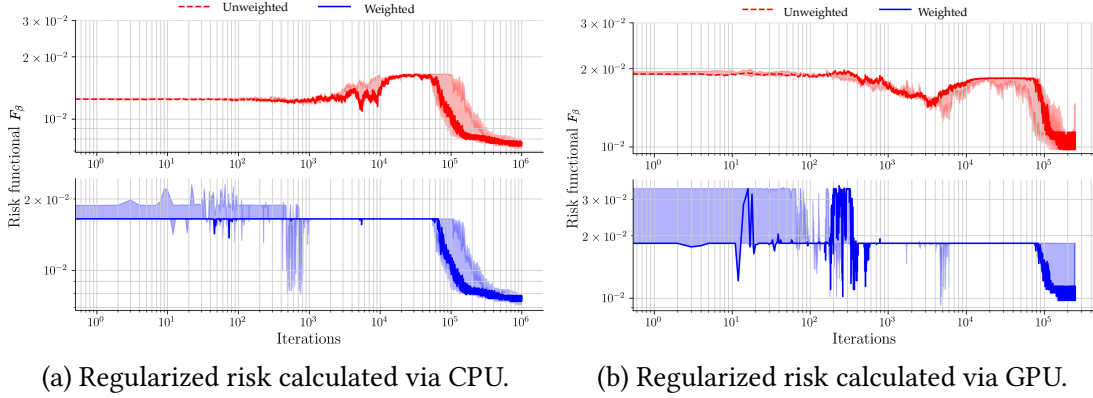
Figure 6.2: The solid line shows the regularized risk functional F_β versus the number of proximal recursions shown for the WDBC dataset with $\beta = 0.05$. The shadow shows the F_β variation range for different values of $\beta \in \{0.03, 0.05, 0.07\}$.

Table 6.1: Classification accuracy of the proposed computational framework for the WDBC Dataset

β	Unweighted	Weighted
0.03	91.17%	92.35%
0.05	92.94%	92.94%
0.07	78.23%	92.94%

ations. As shown in Table 6.2, we train the neural network to a comparable accuracy in only 2.5×10^5 iterations. The new runtime is around 6% of the original runtime for the CPU-based computation. Fig. 6.2b shows the risk functional calculated via this updated code. We parallelize these calculations, taking advantage of the GPU capacity of the Jetson TX2.

We utilize these improvements in runtime to additionally experiment with our choice of ε. Table 6.3 reports the final values of the regularized risk functional $\hat{F}_\beta$ and corresponding runtimes for varying ε. As expected, larger ε entails more smoothing and lowers runtime. The corresponding final regularized risk values show no significant variations, suggesting numerical stability.

Table 6.2: Classification accuracy on Jetson Tx2, after 2.5×10^5 iterations

β	Unweighted	Weighted	Runtime (hr)
0.03	91.18%	91.18%	1.415
0.05	91.18%	92.94%	1.533
0.07	90.59%	91.76%	1.704

Table 6.3: Comparing final $\hat{F}_\beta$ and runtimes for various ε

ε	Unweighted Final $\hat{F}_\beta$	Runtime (s)
0.1	1.4348×10^{-2}	32863
0.5	1.3740×10^{-2}	11026
1	1.0412×10^{-2}	5022
5	1.1293×10^{-2}	4731
10	9.8849×10^{-3}	4766

6.3.3 Computational complexity

In this case study, we determine the computational complexity of PROXLEARN as follows. Letting

$$\boldsymbol{a}_{k-1} := \left(a_{k-1}^1 \quad \cdots \quad a_{k-1}^N \right)^\top, \ \ \boldsymbol{b}_{k-1} := \left(b_{k-1}^1 \quad \cdots \quad b_{k-1}^N \right)^\top, \ \ \boldsymbol{W}_{k-1} := \left(\boldsymbol{w}_{k-1}^1 \quad \cdots \quad \boldsymbol{w}_{k-1}^N \right)^\top,$$

we create the $N \times n_{\text{data}}$ matrix

$$\boldsymbol{P}_{k-1} := (\boldsymbol{a}_{k-1} \mathbf{1}^\top) \odot \tanh(\boldsymbol{W}_{k-1} \boldsymbol{X}^\top + \boldsymbol{b}_{k-1} \mathbf{1}^\top), \tag{6.36}$$

which has complexity $O(n_{\text{data}} N n_x)$. The subsequent creation of matrix $\boldsymbol{U}_{k-1}$ in line 3 of Algorithm 3 has complexity $O(n_{\text{data}} N^2)$, and creating $\boldsymbol{v}_{k-1}$ and $\boldsymbol{u}_{k-1}$ takes complexity $O(N n_{\text{data}})$ and $O(N^2)$ respectively.

The complexity in calculating the relevant derivatives of $\boldsymbol{v}_{k-1}$ and $\boldsymbol{u}_{k-1}$ is $O(N^2 n_{\text{data}} n_x)$ (these derivatives are calculated in Appendices C and C). Updating $\boldsymbol{\Theta}_{k-1}$ using

Table 6.4: Comparison of average classification accuracy from [3, Table 1] to our algorithm, ProxLearn

Dataset	JKO-ICNN	SWGF+RealNVP	ProxLearn, Weighted	ProxLearn, Unweighted
Banana	0.550 ± 10^{-2}	0.559 ± 10^{-2}	0.551 ± 10^{-2}	$0.535 \pm 5 \cdot 10^{-2}$
Diabetes	$0.777 \pm 7 \cdot 10^{-3}$	$0.778 \pm 2 \cdot 10^{-3}$	$0.736 \pm 2 \cdot 10^{-2}$	0.731 ± 10^{-2}
Twonorm	$0.981 \pm 2 \cdot 10^{-4}$	$0.981 \pm 6 \cdot 10^{-4}$	$0.972 \pm 2 \cdot 10^{-3}$	$0.972 \pm 2 \cdot 10^{-3}$

these results has complexity of $O(Np) = O(Nn_x)$. Therefore, the process of updating Θ_{k-1} via EulerMaruyama is $O(N^2 n_{\text{data}} n_x)$.

The significant complexity in the remainder of ProxLearn arises in the construction of matrix C_k in line 3 and the matrix-vector multiplications within the while loop in lines 11, 12.

The creation of the $N \times N$ matrix C_k, in which each element is the vector norm of a $n_x \times 1$ vector, is $O(n_x N^2)$. In a worst-case scenario, the while loop runs L times. The operations of leading complexity within the while loop are the multiplications of the Γ_k matrix of size $N \times N$ with the $N \times 1$ vectors, which have a complexity of $O(N^2)$. Therefore, the while loop has a complexity of $O(LN^2)$.

Updating ϱ_{k-1} thus has a complexity of $O((n_x + L)N^2)$. In practice, the while loop typically ends far before reaching the maximum number of iterations L.

From this analysis, we find that the overall complexity of ProxLearn is $O(N^2(n_{\text{data}} n_x + L))$. In comparison, the per iteration complexity for JKO-ICNN is $O(N_{\text{inner}}(N + 1)N_{\text{batch}} + N^3)$ where N_{inner} denotes the number of inner optimization steps, and N_{batch} denotes the batch size. The per iteration complexity for SWGF is $O(N_{\text{inner}} N_{\text{proj}} N_{\text{batch}} \log N_{\text{batch}})$ where N_{proj} denotes the number of projections to approximate the sliced Wasserstein distance.

6.3.4   Comparisons to existing results

From Fig. 6.2, we observe that there is a significant burn in period for the risk functional curves. These trends in learning curves agree with those reported in [15]. In particular, [15, Fig. 3] and Fig. 7.3 in that reference's *Supporting Information*, show convergence trends very similar to our Fig. 6.2: slow decay until approx. 10^5 iterations and then a significant speed up. The unusual convergence trend was explicitly noted in [15]: "We observe that SGD converges to a network with very small risk, but this convergence has a nontrivial structure and presents long flat regions". It is interesting to note that [15] considered an experiment that allowed rotational symmetry and simulated the radial (i.e., with one spatial dimension) discretized PDE, while we used the proposed proximal recursion directly in the neuron population ensemble to solve the PDE IVP, i.e., similar convergence trends were observed using different numerical methods applied to the same mean field PDE IVP. This makes us speculate that the convergence trend is specific to the mean field PDE dynamics itself, and is less about the particular numerical algorithm. This observation is consistent with recent works such as [208] which investigate the mean field learning dynamics in two layer ReLU networks and in that setting, show that the learning can indeed be slow depending on the target function.

As a first study, our numerical results achieve reasonable classification accuracy compared to the state-of-art, even though our proposed meshless proximal algorithm is very different from the existing implementations. We next compare the numerical performance with existing methods as in [56] and [3]. We apply our proximal algorithm for binary classification to three datasets also considered in [56] and [3]: the banana, diabetes, and twonorm datasets.

The banana dataset consists of 5300 data points, each with $n_x = 2$ features, which

we rescale to lie between 0 and 8. We set $\beta = 0.05$, draw our initial weights $\boldsymbol{w}$ from Unif $\left([-2, 2]^{n_x}\right)$ and bias b from Unif $\left([-0.3, 0.3]\right)$, and set $\boldsymbol{\varrho}_0 \equiv$ Unif $(0, 1000)$. We run our code for 3500 iterations, splitting the data evenly between test and training data.

The diabetes dataset consists of $n_x = 8$ features from each of 768 patients. Based on our experimental results, we make the following adjustments to our algorithm: we re-define $\beta = 0.65$, $\boldsymbol{\varrho}_0 \equiv$ Unif $(0, 1000)$, and draw our initial weights $\boldsymbol{w}$ from Unif $\left([-2, 2]^{n_x}\right)$. We rescale the data to lie between 0 and 1, and use half of the dataset for training purposes, and the remainder as test data. In this case, we run our code for 4.99×10^5 iterations.

The twonorm dataset consists of 7,400 samples drawn from two different normal distributions, with $n_x = 20$ features. We again consider 50% of the same as training data and used the remaining 50% as test data, and rescale the given data by a factor of 8. Based on our empirical observations, we redefine $\beta = 1.95$, and once more draw our initial weights $\boldsymbol{w}$ from Unif $\left([-2, 2]^{n_x}\right)$ and set $\boldsymbol{\varrho}_0 \equiv$ Unif $(0, 1000)$. In this case, we perform 10^4 proximal recursions in each separate run.

We run our code five times for each of the three datasets under consideration and compute "unweighted estimates" and "weighted estimates" in each case, as described above. These estimates assign each data point a value: negative values predict the label as 0, while positive values predict the label as 1. From these results, we calculate the weighted and unweighted accuracy by finding the percentage of predicted test labels that match the actual test labels. The average accuracy over all five runs is reported in Table 6.4, alongside the results reported in [3, Table 1]. We achieve comparable accuracy to these recent results.

6.4   Case study: Multi-class Classification

We next apply the proposed proximal algorithm to a ten-class classification problem using the Semeion Handwritten Digit (hereafter SHD) Data Set [206]. This numerical experiment is performed on the aforementioned Jetson TX2.

6.4.1   SHD data set

The SHD Data Set consists of 1593 handwritten digits. By viewing each digit as 16×16 pixel image, each image is represented by $n_x = 16^2 = 256$ features. Each feature is a boolean value indicating whether a particular pixel is filled. We subsequently re-scale these features such that $\boldsymbol{x}_i \in \{-1, 1\}^{n_x}$.

6.4.2   Adaptations to PROXLEARN for multi-class case

To apply PROXLEARN for a multi-class case, we make several adaptations. For instance, rather than attempting to determine $f(\boldsymbol{x}) \approx y$, we redefine $f(\boldsymbol{x})$ to represent a mapping of input data to the predicted likelihood of the correct label. We therefore redefine the variables, parameters, and risk function as follows.

Each label is represented by a 1×10 vector of booleans, stored in a $n_{\text{data}} \times 10$ matrix $\boldsymbol{Y}$ where $Y_{i,j} = 1$ if the ith data point $\boldsymbol{x}_i$ has label j, and $Y_{i,j} = 0$ otherwise.

We construct the $N \times n_{\text{data}}$ matrix $\boldsymbol{P}_{k-1}$ by defining the (j, i) element of $\boldsymbol{P}_{k-1}$ as

$$
\begin{aligned}
\boldsymbol{P}_{k-1}(j, i) &:= \Phi(\boldsymbol{\theta}_{k-1}^j, \boldsymbol{X}(i, :), \boldsymbol{Y}(i, :)) \\
&:= \left\langle \text{SOFTMAX}(\boldsymbol{X}(i, :)(\boldsymbol{\theta}_{k-1}^j)^\top), (\boldsymbol{Y}(i, :))^\top \right\rangle
\end{aligned} \tag{6.37}
$$

where $\langle \cdot, \cdot \rangle$ denotes the standard Euclidean inner product. The SOFTMAX function in (6.37) produces a 10×1 vector of non-negative entries that sum to 1. By taking the

inner product of this vector with the Boolean vector $Y(i,:)$, we define $P_{k-1}(j,i) = \Phi(\theta_{k-1}^j, X(i,:), Y(i,:))$ as the perceived likelihood that the data point i is labeled correctly by sample j. As our model improves, this value approaches 1, which causes the probability of an incorrect label to drop.

As this newly defined P_{k-1} does not call for bias or scaling, the weights alone are stored in the $N \times p$ matrix Θ_{k-1}. In this case, $p := 10n_x$, as each of the n_x features requires a distinct weight for each of the ten labels. For convenience, we reshape $\Theta_{k-1} = (\theta_{k-1}^1, ..., \theta_{k-1}^N)$, where each θ_{k-1}^i is a $10 \times n_x$ matrix. Therefore, Θ_{k-1} is a $10 \times n_x \times N$ tensor.

We redefine the unregularized risk to reflect our new Φ as follows:

$$F(\rho) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}\left(1 - \int_{\mathbb{R}^p} \Phi(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})\rho(\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}\right)^2. \tag{6.38}$$

Expanding the above, we arrive at a form that resembles (6.8), now with the following adjusted definitions:

$$F_0 := 1, \tag{6.39}$$

$$V(\boldsymbol{\theta}) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}\big[-2\Phi(\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{y})\big], \tag{6.40}$$

$$U(\boldsymbol{\theta},\tilde{\boldsymbol{\theta}}) := \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y})}\big[\Phi(\boldsymbol{\theta},\boldsymbol{x},\boldsymbol{y})\Phi(\tilde{\boldsymbol{\theta}},\boldsymbol{x},\boldsymbol{y})\big]. \tag{6.41}$$

We use the regularized risk functional F_β as in (6.12) where F now is given by (6.38). Due to the described changes in the structure of Θ_{k-1}, the creation of C_k in line 3 of PROXLEARN results in a $10 \times N \times N$ tensor, which we then sum along the ten element axis, returning an $N \times N$ matrix.

Finally, we add a scaling in line 7 of EULERMARUYAMA, scaling the noise by a factor of 1/100.

6.4.3 Numerical experiments

With the adaptations mentioned above, we set the inverse temperature $\beta = 0.5$, $\epsilon = 10$, the step size $h = 10^{-3}$, and $N = 100$. We draw the initial weights from $\mathrm{Unif}\left([-1, 1]^{10n_x}\right)$. We take the first $n_{\mathrm{data}} = 1000$ images as training data, reserving the remaining $n_{\mathrm{test}} = 593$ images as test data, and execute 30 independent runs of our code, each for 10^6 proximal recursions.

To evaluate the training process, we create a matrix $\boldsymbol{P}_{k-1}^{\mathrm{test}} \in \mathbb{R}^{N \times n_{\mathrm{test}}}$, using test data rather than training data, but otherwise defined as in (6.37). We then calculate a weighted approximation of F_β:

$$F_\beta \approx \frac{1}{n_{\mathrm{test}}} \left\| \mathbf{1} - (\boldsymbol{P}_{k-1}^{\mathrm{test}})^\top \boldsymbol{\varrho} \right\|_2^2, \tag{6.42}$$

and an unweighted approximation:

$$F_\beta \approx \frac{1}{n_{\mathrm{test}}} \left\| \mathbf{1} - \frac{1}{N}(\boldsymbol{P}_{k-1}^{\mathrm{test}})^\top \mathbf{1} \right\|_2^2, \tag{6.43}$$

which we use to produce the risk and weighted risk log-log plots shown in Fig. 6.3.

Notably, despite the new activation function and the adaptations described above, our algorithm produces similar risk plots in the binary and multi-class cases. The run time for 10^6 iterations is approximately 5.3 hours.

To evaluate our multi-class model, we calculate the percentage of accurately labeled test data by first taking ARGMAX $(\boldsymbol{X}_{\mathrm{test}} \boldsymbol{\Theta}_{k-1})$ along the ten dimensional axis, to determine the predicted labels for each test data point using each sample of $\boldsymbol{\Theta}_{k-1}$. We then compare these predicted labels with the actual labels. We achieved 61.079% accuracy for the test data, and 75.330% accuracy for the training data.
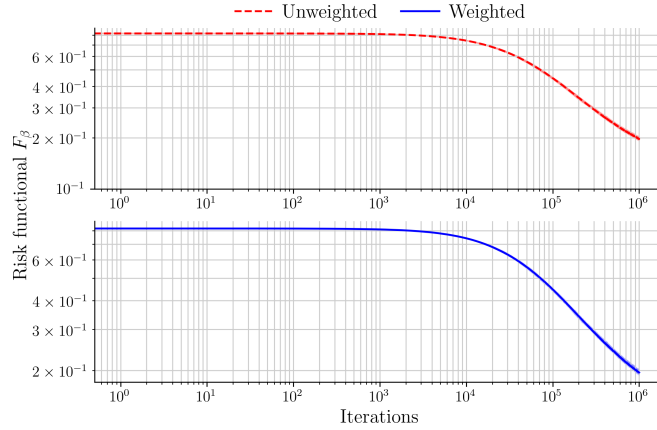
Figure 6.3: The solid line shows the average regularized risk functional F_β versus the number of proximal recursions shown for the Semeion dataset with $\beta = 0.5$. The narrow shadow shows the F_β variation range with the same β using the results of 30 independent runs, each starting from the same initial point cloud $\{\boldsymbol{\theta}_0^i, \varrho_0^i\}_{i=1}^N$.

6.4.4 Updated computational complexity of PROXLEARN

In the binary case, the creation of matrix $\boldsymbol{C}_k$ requires $O(n_x N^2)$ flops. In the case of multi-class classification concerning m classes, $\boldsymbol{C}_k$ is redefined as the sum of m such matrices. Therefore, creating the updated matrix $\boldsymbol{C}_k$ takes $O(m n_x N^2)$ flops. Thus, updating $\boldsymbol{\varrho}_{k-1}$ is of complexity $O((m n_x + L)N^2)$. The complexity of EULERMARUYAMA can be generalized from the discussion in Section 6.3.

6.5 Case study: Learning Sinusoid

To better visualize the functionality of the proposed algorithm, we perform a synthetic case study of learning a sinusoid following the set up as in [209, Sec. 2.1]. We performed 5000 iterations of PROXLEARN with $N = 1000$ samples (no mini-batch) from the initial PDF $\varrho_0(\boldsymbol{\theta} \equiv (a, b, w)) = \text{Unif}([-1, 1] \times [-1, 1] \times [-1.5, 1.5])$, and used al-
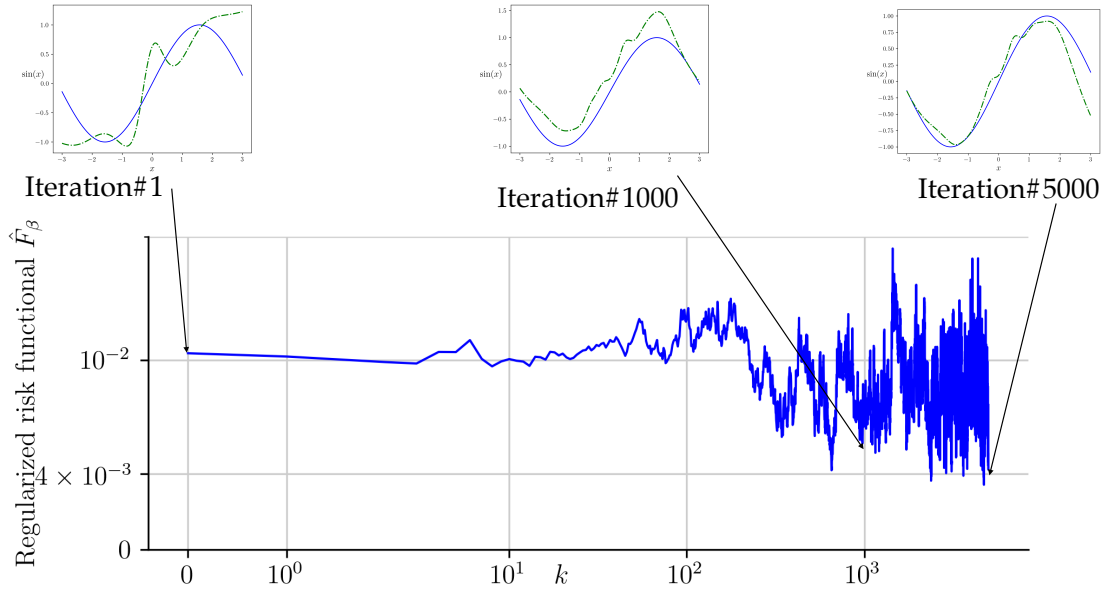
Figure 6.4: The evolution of the regularized risk $\hat{F}_\beta$ versus iteration index k for the proposed PROXLEARN. Inset plots compare the ground truth (sinusoid) with the output from the network at three specific iterations.

gorithm parameters $\beta = 0.3$, $h = 10^{-4}$, $\delta = 10^{-3}$, $\varepsilon = 10^{-3}$, $L = 10$. The evolution of the associated regularized risk functional and the learnt functions are shown in Fig. 6.4. Fig. 6.5 shows the function approximations learnt by our proposed algorithm at the end of 3200 iterations for 20 randomized runs with the same initial PDF and parameters as reported here.

To illustrate the effect of finite N on the algorithm's performance, we report the effect of varying N on the final regularized risk value $\hat{F}_\beta$ for a specific synthetic experiment. We observe that increasing N improves the final regularized risk, as expected intuitively.
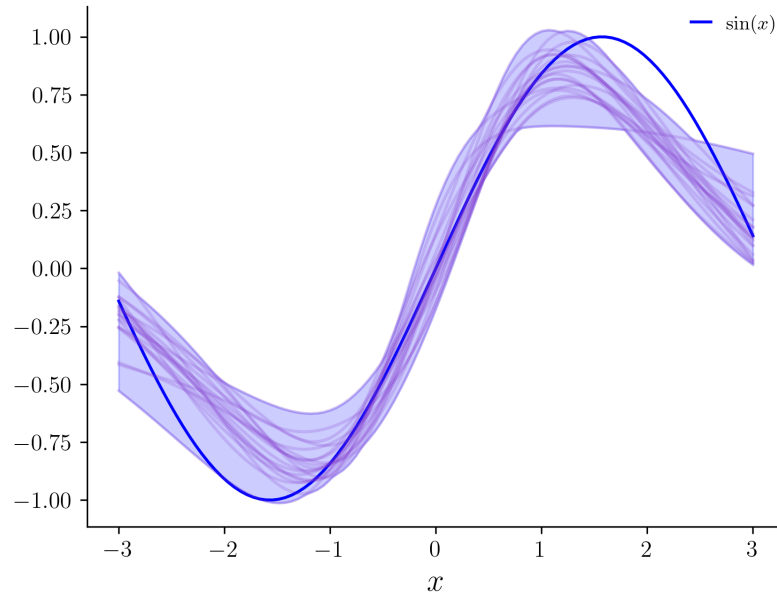
Figure 6.5: Comparison of the ground truth (here $\sin(x)$) with the learnt approximants obtained from the proposed PROXLEARN after 3200 iterations for 20 randomized runs. All randomized runs use the same initial PDF and parameters as reported here.

Table 6.5: Comparing final $\hat{F}_\beta$ for varying N

N	Final $\hat{F}_\beta$
500	0.01241931
700	0.01075817
1000	0.00806645
2000	0.00762518

7 | Distributed Computing: Wasserstein Consensus ADMM

7.1 Problem Formulation

Recall the notation that $\mathcal{P}_2(\mathbb{R}^d)$ denotes the space of Borel probability measures over $\mathbb{R}^d$ with finite second moments. Consider its subset

$$\mathcal{P}_{2,\mathrm{ac}}(\mathbb{R}^d) := \{\mu \in \mathcal{P}_2(\mathbb{R}^d) \mid \mu \text{ is absolutely continuous w.r.t. the Lebesgue measure}\}.$$

A probability measure $\mu \in \mathcal{P}_{2,\mathrm{ac}}(\mathbb{R}^d)$ admits a joint PDF $\rho(\boldsymbol{\theta}) := \frac{\mathrm{d}\mu}{\mathrm{d}\boldsymbol{\theta}}$ such that $\rho \geq 0$ for all $\boldsymbol{\theta} \in \mathbb{R}^d$ and $\int_{\mathbb{R}^d} \rho \, \mathrm{d}\boldsymbol{\theta} = 1$.

We consider measure-valued optimization problems of the form (1.1) where the objective F is additive, i.e., $F(\mu) = F_1(\mu) + F_2(\mu) + \cdots + F_n(\mu)$ for some finite integer $n > 1$, and the functionals $F_i : \mathcal{P}_2(\mathbb{R}^d) \mapsto \mathbb{R}$ are convex for all $i \in [n]$. If the optimization in (1.1) is instead over $\mathcal{P}_{2,\mathrm{ac}}(\mathbb{R}^d)$, then we can equivalently write it as

$$\arg\inf_{\rho} F_1(\rho) + F_2(\rho) + \cdots + F_n(\rho) \tag{7.1}$$

where the decision variable ρ is a joint PDF over $\mathbb{R}^d$ with finite second moment.

Instances of (1.1) are often encountered in machine learning [11, 15, 16, 18, 19, 57] and control [5, 38]. For example, given the measures $\mu_1, \ldots, \mu_n \in \mathcal{P}_2(\mathcal{X})$ and positive weights $w_1, \cdots, w_n$, computing the *Wasserstein barycenter* [54]

$$\arg\inf_{\mu \in \mathcal{P}_2(\mathcal{X})} \sum_{i=1}^{n} w_i W^2(\mu, \mu_i) \tag{7.2}$$

is an instance of (1.1) where $F_i(\mu) := w_i W^(\mu, \mu_i)$, and W denotes the 2-Wasserstein distance. As another example, consider maximum likelihood deconvolution, i.e., the problem of estimating an unknown probability measure μ from n noisy observations

$Y_i = X_i + Z_i$, $i \in [n]$

Most existing algorithms [4, 50, 52, 56, 203, 210–214] for this class of problems require centralized computation; relatively few works [215, 216] are available on solving specific instances of (1.1) via distributed computation. The main contribution of this work is to deduce a distributed algorithm for solving (1.1) by generalizing the Euclidean consensus ADMM to Wasserstein spaces. Our proposed algorithm realizes measure-valued operator splitting [217–219] but allows explicit distributed updates.

Motivation and contributions

While problem (1.1) appears across many disciplines, one particular motivation behind our work is to numerically solve the *transient* solutions for measure-valued PDE initial value problems (IVPs). These PDEs are often nonlinear and nonlocal (see e.g., the second case study in (7.6)), and difficult to solve scalably via traditional scientific computing methods such as finite difference. However, it is known that the flow induced by such PDE IVPs can often be seen [45, Ch. 11], [46] as gradient descent of a suitable free energy Lyapunov functional $F(\mu)$ w.r.t. the 2-Wasserstein metric over the space of measures. Then, high-level idea is to leverage this variational reformulation to compute the transient solutions for such IVPs by numerically performing Wasserstein gradient descent on (1.1).

The specific idea in this work is to further recognize that the functional F in practice has an additive structure $F(\cdot) = F_1(\cdot) + \cdots + F_n(\cdot)$, which comes from different spatial operators (e.g., advection, interaction, diffusion) appearing in the PDE. One of our contribution here is to show that it is possible to leverage this additive structure in F to generalize the Euclidean ADMM to the Wasserstein space. The proposed algorithm can then be seen as a nonlinear superposition principle where different comput-

136

ers solve different (simpler) PDE IVPs by performing proximal update on a modified version of F_i, and then combine the resulting updates in a nonlinear manner. Historically, this point of view is very close to the origin of operator splitting [220, 221] in the PDE community that motivated the development of ADMM [222], albeit in the finite-dimensional setting.

We clarify here that while augmented Lagrangian methods for infinite dimensional problems have been investigated before, they appeared in the Hilbert spaces [223] or reflexive Banach spaces [224, 225]. In contrast, our definition (7.4) for the Wasserstein augmented Lagrangian is novel. Our development is also different from the (standard) augmented Lagrangian for Wasserstein gradient flow as in [203, equation 2.12], and directly works on the Wasserstein space.

7.2 Main Idea

To leverage the additive structure of the objective in (1.1) for distributed computation, we start by rewriting it in the consensus form. Specifically, we relabel the argument of the functional F_i as μ_i for all $i \in [n]$, and then impose the consensus constraint $\mu_1 = \mu_2 = \cdots = \mu_n$. Letting $\mathcal{P}_2^{n+1}(\mathcal{X}) := \underbrace{\mathcal{P}_2(\mathcal{X}) \times \cdots \times \mathcal{P}_2(\mathcal{X})}_{n+1 \text{ times}}$, we thus transcribe (1.1) into

$$\underset{(\mu_1, \cdots, \mu_n, \zeta) \in \mathcal{P}_2^{n+1}(\mathcal{X})}{\arg\inf} F_1(\mu_1) + F_2(\mu_2) + \cdots + F_n(\mu_n) \tag{7.3a}$$

$$\text{subject to} \quad \mu_i = \zeta \quad \text{for all } i \in [n]. \tag{7.3b}$$

Denote an element of the base space as $\boldsymbol{\theta} \in \mathcal{X} \subseteq \mathbb{R}^d$. Akin to the standard (Euclidean) augmented Lagrangian, we define the *Wasserstein augmented Lagrangian*

$$L_\alpha(\mu_1, \cdots, \mu_n, \zeta, \nu_1, \cdots, \nu_n) := \sum_{i=1}^n \left\{ F_i(\mu_i) + \frac{\alpha}{2} W^2(\mu_i, \zeta) + \int_\mathcal{X} \nu_i(\boldsymbol{\theta}) \, (\mathrm{d}\mu_i - \mathrm{d}\zeta) \right\} \quad (7.4)$$

where $\nu_i(\boldsymbol{\theta})$, $i \in [n]$, are the Lagrange multipliers for the constraints in (7.3b), and $\alpha > 0$ is a regularization constant.

Motivated by the Euclidean ADMM, we then set up the recursions

$$\mu_i^{k+1} = \underset{\mu_i \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \ L_\alpha\left(\mu_1, \cdots, \mu_n, \zeta^k, \nu_1^k, \cdots, \nu_n^k\right) \quad (7.5a)$$

$$\zeta^{k+1} = \underset{\zeta \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \ L_\alpha\left(\mu_1^{k+1}, \cdots, \mu_n^{k+1}, \zeta, \nu_1^k, \cdots, \nu_n^k\right) \quad (7.5b)$$

$$\nu_i^{k+1} = \nu_i^k + \alpha\left(\mu_i^{k+1} - \zeta^{k+1}\right) \quad (7.5c)$$

where $i \in [n]$, and the recursion index $k \in \mathbb{N}_0$. For convenience, let

$$\nu_{\text{sum}}^k(\boldsymbol{\theta}) := \sum_{i=1}^n \nu_i^k(\boldsymbol{\theta}), \quad k \in \mathbb{N}_0. \quad (7.6)$$

We view (7.5a)-(7.5b) as primal updates, and (7.5c) as dual ascent.

Substituting (7.4) in (7.5), dropping the terms independent of the decision variable in the respective $\arg\inf$, re-scaling, and using (7.6), the recursions (7.5) simplify to

$$\mu_i^{k+1} = \underset{\mu_i \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \ \frac{1}{2} W^2\left(\mu_i, \zeta^k\right) + \frac{1}{\alpha}\left\{ F_i(\mu_i) + \int_\mathcal{X} \nu_i^k(\boldsymbol{\theta}) \mathrm{d}\mu_i \right\}$$

$$= \text{prox}^W_{\frac{1}{\alpha}\left(F_i(\cdot) + \int \nu_i^k \mathrm{d}(\cdot)\right)}\left(\zeta^k\right), \quad (7.7a)$$

$$\zeta^{k+1} = \underset{\zeta \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \ \sum_{i=1}^n \left\{ \frac{1}{2} W^2\left(\mu_i^{k+1}, \zeta\right) - \frac{1}{\alpha} \int_\mathcal{X} \nu_i^k(\boldsymbol{\theta}) \mathrm{d}\zeta \right\}$$

$$= \underset{\zeta \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \left\{ \left(\sum_{i=1}^n W^2\left(\mu_i^{k+1}, \zeta\right)\right) - \frac{2}{\alpha} \int_\mathcal{X} \nu_{\text{sum}}^k(\boldsymbol{\theta}) \mathrm{d}\zeta \right\}, \quad (7.7b)$$

$$\nu_i^{k+1} = \nu_i^k + \alpha\left(\mu_i^{k+1} - \zeta^{k+1}\right). \quad (7.7c)$$

We refer to (7.7) as the *Wasserstein consensus ADMM* generalizing its finite dimen-
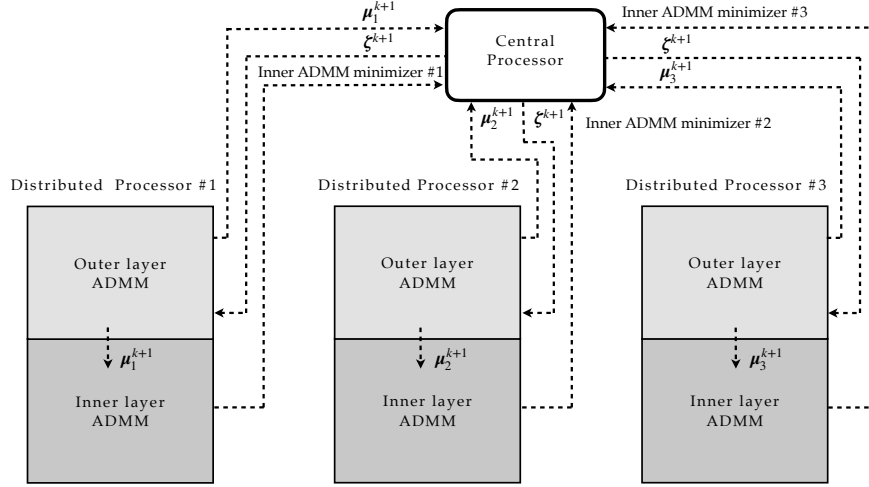
Figure 7.1: High level schematic of the proposed two-layer ADMM algorithm illustrated with one central and $n = 3$ distributed processors. The central processor updates ζ^{k+1}. The "upstairs" (lighter shade) of the distributed processors update $\boldsymbol{\mu}_i^{k+1}$ via *outer layer ADMM* (7.3.1). These distributed $\boldsymbol{\mu}_i^{k+1}$ updates and the centralized ζ^{k+1} values are passed to the "downstairs" (darker shade) of the distributed processors for updating ζ^{k+1} via an *inner layer ADMM* (7.3.2).

sional Euclidean counterpart in the sense (7.7a)-(7.7b) are analogues of the so-called x and z updates, respectively [61, Ch. 5.2.1]. However, important difference arises in (7.7b) compared to its Euclidean counterpart due to the sum of squares of Wasserstein distances. In the Euclidean case, the corresponding z update can be analytically performed in terms of the *arithmetic mean* of the x updates. While (7.7b) involves a *generalized mean* of the updates from (7.7a), we now have *Wasserstein barycentric proximal* of a linear functional in ν_{sum}^k w.r.t. n measures $\{\mu_1^{k+1}, \cdots, \mu_n^{k+1}\}$.

The proximal updates (7.7a) are closely related to the WGFs of the form (1.7) generated by the respective (scaled) free energy functionals

$$\Phi_i(\mu_i) := F_i(\mu_i) + \int_{\mathcal{X}} \nu_i^k \mathrm{d}\mu_i, \quad \mu_i \in \mathcal{P}_2(\mathcal{X}), \quad i \in [n]. \tag{7.8}$$

As per the assumptions on F_i, the functionals Φ_i are also proper lsc and convex along generalized geodesics defined by the 2-Wasserstein distance. As $1/\alpha \downarrow 0$, the sequence

$\Phi_i(\cdot) = F_i(\cdot) + \int \nu_i^k \mathrm{d}(\cdot)$	PDE in (5.12)	Name
$\int_{\mathbb{R}^d} \left(V(\boldsymbol{\theta}) + \nu_i^k(\boldsymbol{\theta}) \right) \mathrm{d}\mu_i(\boldsymbol{\theta})$	$\frac{\partial \widetilde{\mu}_i}{\partial t} = \nabla \cdot \left(\widetilde{\mu}_i \left(\nabla V + \nabla \nu_i^k \right) \right)$	Liouville equation
$\int_{\mathbb{R}^d} \left(\nu_i^k(\boldsymbol{\theta}) + \beta^{-1} \log \mu_i(\boldsymbol{\theta}) \right) \mathrm{d}\mu_i(\boldsymbol{\theta})$	$\frac{\partial \widetilde{\mu}_i}{\partial t} = \nabla \cdot \left(\widetilde{\mu}_i \nabla \nu_i^k \right) + \beta^{-1} \Delta \widetilde{\mu}_i$	Fokker-Planck equation
$\int_{\mathbb{R}^d} \nu_i^k(\boldsymbol{\theta}) \mathrm{d}\mu_i(\boldsymbol{\theta}) + \int_{\mathbb{R}^{2d}} U(\boldsymbol{\theta},\boldsymbol{\sigma}) \mathrm{d}\mu_i(\boldsymbol{\theta}) \mathrm{d}\mu_i(\boldsymbol{\sigma})$	$\frac{\partial \widetilde{\mu}_i}{\partial t} = \nabla \cdot \left(\widetilde{\mu}_i \left(\nabla \nu_i^k + \nabla (U \otimes \widetilde{\mu}_i) \right) \right)$	Propagation of chaos equation
$\int_{\mathbb{R}^d} \left(\nu_i^k(\boldsymbol{\theta}) + \frac{\beta^{-1}}{m-1} \mathbf{1}^\top \mu_i^m \right) \mathrm{d}\mu_i(\boldsymbol{\theta}), m > 1$	$\frac{\partial \widetilde{\mu}_i}{\partial t} = \nabla \cdot \left(\widetilde{\mu}_i \nabla \nu_i^k \right) + \beta^{-1} \Delta \widetilde{\mu}_i^m$	Porous medium equation

Table 7.1: Specific instances of the PDE in (5.12) for different choices of F_i, and hence Φ_i. The Euclidean gradient operator ∇ is w.r.t. $\boldsymbol{\theta} \in \mathbb{R}^d$. The operator $\otimes$ can be seen as a generalized convolution, given by $(U \otimes \widetilde{\mu}_i)(\boldsymbol{\theta}) := \int_{\mathbb{R}^d} U(\boldsymbol{\theta},\boldsymbol{\sigma}) \mathrm{d}\widetilde{\mu}_i(\boldsymbol{\sigma})$ where $U(\boldsymbol{\theta},\boldsymbol{\sigma})$ is symmetric and positive definite for all $(\boldsymbol{\theta},\boldsymbol{\sigma}) \in \mathbb{R}^d \times \mathbb{R}^d$.

$\{\mu_i^k(\alpha)\}_{k \in \mathbb{N}_0}$ generated by the updates (7.7a) converge to the measure-valued solution trajectory $\widetilde{\mu}_i(t, \cdot)$, $t \in [0, \infty)$ solving the IVP

$$\frac{\partial \widetilde{\mu}_i}{\partial t} = -\nabla^W \Phi_i \left(\widetilde{\mu}_i \right), \quad \widetilde{\mu}_i(t = 0, \cdot) = \widetilde{\mu}_i^0(\cdot), \quad i \in [n]. \tag{7.9}$$

Thus, in a rather generic setting, performing the proximal updates (7.7a) in parallel across the index $i \in [n]$, amounts to performing distributed time updates for the approximate transient solutions of the IVPs (7.9).

For specific choices of $i \in [n]$, important examples of F_i include $\int V(\boldsymbol{\theta}) \mathrm{d}\mu_i(\boldsymbol{\theta})$ (potential energy for some suitable advection potential V), $\beta^{-1} \int \log \mu_i(\boldsymbol{\theta}) \mathrm{d}\mu_i(\boldsymbol{\theta})$ (internal energy with the "inverse temperature" parameter $\beta > 0$), $\int_{\mathbb{R}^{2d}} U(\boldsymbol{\theta},\boldsymbol{\sigma}) \mathrm{d}\mu_i(\boldsymbol{\theta}) \mathrm{d}\mu_i(\boldsymbol{\sigma})$ (interaction energy for some symmetric positive definite interaction potential U). In Table 7.1, we summarize how the PDE in (5.12) specializes in such cases. An interesting observation for (7.7a) is that for each $i \in [n]$, the dual variables ν_i^k contribute as time-varying advection potentials irrespective of whether F_i already has an advection potential or not.

Remark 7.1. *The Lagrange multiplier ν_i for the ith measure consensus constraint (7.3b) is an element of the dual space of $\mathcal{P}_2$ comprising of bounded linear functionals of the elements of $\mathcal{P}_2$. Thus, when the primal updates for μ_i are identified with the corresponding WGFs, then the Lagrange multipliers ν_i become "algorithmic" advection potentials. For*

the same reason, the integral involving the Lagrange multiplier ends up being simply an Euclidean inner product post-discretization; see (7.10).

In the next Section, we propose a two-layer ADMM algorithm (see (7.1)) to solve (7.7).

7.3   Results

To numerically realize the recursions (7.7), we consider a sequence of discrete probability distributions $\{\boldsymbol{\mu}_1^k, \cdots, \boldsymbol{\mu}_n^k, \boldsymbol{\zeta}^k\}_{k \in \mathbb{N}_0}$ where each distribution is a probability vector of length $N \times 1$, representative of the respective probability values at N samples. Thus, for each fixed $k \in \mathbb{N}_0$, the tuple

$$\left(\boldsymbol{\mu}_1^k, \cdots, \boldsymbol{\mu}_n^k, \boldsymbol{\zeta}^k\right) \in \underbrace{\Delta^{N-1} \times \cdots \times \Delta^{N-1}}_{n+1 \text{ times}} =: \left(\Delta^{N-1}\right)^{n+1} \text{ (the product simplex)}.$$

Likewise, for each fixed $k \in \mathbb{N}_0$, the Lagrange multipliers $(\boldsymbol{\nu}_1^k, \cdots, \boldsymbol{\nu}_n^k) \in \mathbb{R}^{nN}$, and $\boldsymbol{\nu}_{\text{sum}}^k = \sum_{i=1}^n \boldsymbol{\nu}_i^k \in \mathbb{R}^N$.

Given probability vectors $\boldsymbol{\xi}, \boldsymbol{\eta} \in \Delta^{N-1}$, let

$$\Pi_N\left(\boldsymbol{\xi}, \boldsymbol{\eta}\right) := \{\boldsymbol{M} \in \mathbb{R}^{N \times N} \mid \boldsymbol{M} \geq \boldsymbol{0} \text{ (elementwise)}, \ \boldsymbol{M}\boldsymbol{1} = \boldsymbol{\xi}, \ \boldsymbol{M}^\top \boldsymbol{1} = \boldsymbol{\eta}\}.$$

Also, let $\boldsymbol{C} \in \mathbb{R}^{N \times N}$ denote the squared Euclidean distance matrix for the sampled data $\{\boldsymbol{\theta}_r\}_{r \in [N]}$ in $\mathbb{R}^d$, i.e., the entries of the matrix C are $C(i,j) := \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\|_2^2$ for all $i, j \in [N]$.

For each $i \in [n]$ and $k \in \mathbb{N}_0$, we write the discrete version of (7.7) as

$$\boldsymbol{\mu}_i^{k+1} = \text{prox}_{\frac{1}{\alpha}(F_i(\boldsymbol{\mu}_i) + \langle \boldsymbol{\nu}_i^k, \boldsymbol{\mu}_i \rangle)}^W \left(\boldsymbol{\zeta}^k\right)$$

$$= \underset{\boldsymbol{\mu}_i \in \Delta^{N-1}}{\arg\inf} \left\{ \min_{\boldsymbol{M} \in \Pi_N(\boldsymbol{\mu}_i, \boldsymbol{\zeta}^k)} \frac{1}{2}\langle \boldsymbol{C}, \boldsymbol{M} \rangle + \frac{1}{\alpha}\left(F_i(\boldsymbol{\mu}_i) + \langle \boldsymbol{\nu}_i^k, \boldsymbol{\mu}_i \rangle\right) \right\}, \qquad (7.10a)$$

$$\boldsymbol{\zeta}^{k+1} = \underset{\boldsymbol{\zeta} \in \Delta^{N-1}}{\arg\inf} \left\{ \left(\sum_{i=1}^{n} \underset{\boldsymbol{M}_i \in \Pi_N(\boldsymbol{\mu}_i^{k+1}, \boldsymbol{\zeta})}{\min} \frac{1}{2} \langle \boldsymbol{C}, \boldsymbol{M}_i \rangle \right) - \frac{2}{\alpha} \langle \boldsymbol{\nu}_{\text{sum}}^k, \boldsymbol{\zeta} \rangle \right\}, \tag{7.10b}$$

$$\boldsymbol{\nu}_i^{k+1} = \boldsymbol{\nu}_i^k + \alpha \left(\boldsymbol{\mu}_i^{k+1} - \boldsymbol{\zeta}^{k+1} \right), \tag{7.10c}$$

wherein (7.10a)-(7.10b) used the discrete version of (1.3).

Replacing the squared Wasserstein distance (1.3) in (7.7) by its Sinkhorn regularized version (1.8), modify the recursions (7.10) as

$$\boldsymbol{\mu}_i^{k+1} = \text{prox}_{\frac{1}{\alpha}\left(F_i(\boldsymbol{\mu}_i) + \langle \boldsymbol{\nu}_i^k, \boldsymbol{\mu}_i \rangle\right)}^{W_\varepsilon} \left(\boldsymbol{\zeta}^k \right)$$

$$= \underset{\boldsymbol{\mu}_i \in \Delta^{N-1}}{\arg\inf} \left\{ \underset{\boldsymbol{M} \in \Pi_N(\boldsymbol{\mu}_i, \boldsymbol{\zeta}^k)}{\min} \left\langle \frac{1}{2}\boldsymbol{C} + \varepsilon \log \boldsymbol{M}, \boldsymbol{M} \right\rangle + \frac{1}{\alpha} \left(F_i(\boldsymbol{\mu}_i) + \langle \boldsymbol{\nu}_i^k, \boldsymbol{\mu}_i \rangle \right) \right\}, \tag{7.11a}$$

$$\boldsymbol{\zeta}^{k+1} = \underset{\boldsymbol{\zeta} \in \Delta^{N-1}}{\arg\inf} \left\{ \left(\sum_{i=1}^{n} \underset{\boldsymbol{M}_i \in \Pi_N(\boldsymbol{\mu}_i^{k+1}, \boldsymbol{\zeta})}{\min} \left\langle \frac{1}{2}\boldsymbol{C} + \varepsilon \log \boldsymbol{M}_i, \boldsymbol{M}_i \right\rangle \right) - \frac{2}{\alpha} \langle \boldsymbol{\nu}_{\text{sum}}^k, \boldsymbol{\zeta} \rangle \right\}, \tag{7.11b}$$

$$\boldsymbol{\nu}_i^{k+1} = \boldsymbol{\nu}_i^k + \alpha \left(\boldsymbol{\mu}_i^{k+1} - \boldsymbol{\zeta}^{k+1} \right), \tag{7.11c}$$

where $\varepsilon > 0$ is a regularization parameter.

Remark 7.2. *For $\varepsilon \downarrow 0$, the solution of the inner minimization (7.11a) is known [52, Sec. 3] to be a consistent approximation of that in (7.10a). The Wasserstein proximal update (7.10a) can, in principle, be performed by the proximal gradient Jordan-Kinderlehrer-Otto (JKO) algorithm as in [47] with more general regularization. Our motivation for choosing Sinkhorn regularization is computational convenience. As we explain in (7.3.1), when we dualize the inner minimization problem in (7.11a), not only do we have strong duality, but we can also explicitly write the proximal update in terms of the multipliers, which can be obtained, in general, numerically via provably contractive block-coordinate ascent. Note that [47, Remark 1] mentions the computational convenience of performing the JKO update for the negative entropy regularization.*

Remark 7.3. *We note a prior work [226] that computes unregularized Wasserstein barycenter using multiblock ADMM. However, the nested minimization in (7.10b) is different*

from computing the barycenter in that it involves computing the Wasserstein barycentric proximal.

We next provide novel results and algorithmic details to perform the recursions (7.11).

7.3.1   The μ update

The Sinkhorn regularized recursion (7.11a) allows us to get semi-analytical handle on the nested minimization via strong duality. Specifically, consider the convex functions $F_i, G_i : \Delta^{N-1} \mapsto \mathbb{R}$ for all $i \in [n]$ where

$$G_i(\boldsymbol{\mu_i}) := F_i(\boldsymbol{\mu_i}) + \langle \boldsymbol{\nu}_i^k, \boldsymbol{\mu_i} \rangle,$$

and denote the Legendre-Fenchel conjugate of G_i as G_i^*. Following [204, Lemma 3.5], [5, Sec. III], the Lagrange dual problem associated with (7.11a) is

$$\left(\boldsymbol{\lambda}_{0i}^{\mathrm{opt}}, \boldsymbol{\lambda}_{1i}^{\mathrm{opt}}\right) = \underset{\boldsymbol{\lambda}_{0i}, \boldsymbol{\lambda}_{1i} \in \mathbb{R}^N}{\arg\max} \left\{ \langle \boldsymbol{\lambda}_{0i}, \boldsymbol{\zeta}_k \rangle - G_i^* \left(-\boldsymbol{\lambda}_{1i} \right) - \right.$$
$$\left. \alpha\varepsilon \left(\exp\left(\frac{\boldsymbol{\lambda}_{0i}^\top}{\alpha\varepsilon}\right) \exp\left(-\frac{\boldsymbol{C}}{2\varepsilon}\right) \exp\left(\frac{\boldsymbol{\lambda}_{1i}}{\alpha\varepsilon}\right) \right) \right\}, \ i \in [n]. \qquad (7.12)$$

Using (7.12), the proximal updates in (7.11a) can then be recovered from the following proposition.

Proposition 7.1. *([204, Lemma 3.5], [5, Theorem 1]) Given $\alpha, \varepsilon > 0$, the squared Euclidean distance matrix $\boldsymbol{C} \in \mathbb{R}^{N \times N}$, and the probability vector $\boldsymbol{\zeta}^k \in \Delta^{N-1}$, $k \in \mathbb{N}_0$. Let $\boldsymbol{0}$ denote the $N \times 1$ vector of zeros. For $i \in [n]$, the vectors $\boldsymbol{\lambda}_{0i}^{\mathrm{opt}}, \boldsymbol{\lambda}_{1i}^{\mathrm{opt}} \in \mathbb{R}^N$ in (7.12) solve the system*

$$\exp\left(\frac{\boldsymbol{\lambda}_{0i}^{\mathrm{opt}}}{\alpha\varepsilon}\right) \odot \left(\exp\left(-\frac{\boldsymbol{C}}{2\varepsilon}\right) \exp\left(\frac{\boldsymbol{\lambda}_{1i}^{\mathrm{opt}}}{\alpha\varepsilon}\right) \right) = \boldsymbol{\zeta}_k, \qquad (7.13a)$$

$$\mathbf{0} \in \partial_{\boldsymbol{\lambda}_{1i}^{\mathrm{opt}}} G_i^* \left(-\boldsymbol{\lambda}_{1i}^{\mathrm{opt}}\right) - \exp\left(\frac{\boldsymbol{\lambda}_{1i}^{\mathrm{opt}}}{\alpha\varepsilon}\right) \odot \left(\exp\left(-\frac{\boldsymbol{C}^{\top}}{2\varepsilon}\right) \exp\left(\frac{\boldsymbol{\lambda}_{0i}^{\mathrm{opt}}}{\alpha\varepsilon}\right)\right). \tag{7.13b}$$

The proximal update $\boldsymbol{\mu}_i^{k+1}$ in (7.11a) is given by

$$\boldsymbol{\mu}_i^{k+1} = \exp\left(\frac{\boldsymbol{\lambda}_{1i}^{\mathrm{opt}}}{\alpha\varepsilon}\right) \odot \left(\exp\left(-\frac{\boldsymbol{C}^{\top}}{2\varepsilon}\right) \exp\left(\frac{\boldsymbol{\lambda}_{0i}^{\mathrm{opt}}}{\alpha\varepsilon}\right)\right). \tag{7.14}$$

We point out that if $F_i(\boldsymbol{\mu}_i) = \beta^{-1}\langle \log\boldsymbol{\mu}_i, \boldsymbol{\mu}_i \rangle$ where $\beta > 0$, then Proposition 7.1 reduces exactly to [5, Theorem 1] allowing further simplification of (7.13b). As we explained toward the end of Section 7.2 (see also Table 7.1, second row), this specific choice of F_i is of practical interest. In this particular case, the system (7.13) can be solved via certain cone-preserving block coordinate iteration proposed in [5, Sec. III.B,C] that is provably contractive w.r.t. the Thompson metric [62] [227, Ch. 2.1]. This makes the proximal update (7.14) semi-analytical in the sense the pair $\left(\boldsymbol{\lambda}_{0i}^{\mathrm{opt}}, \boldsymbol{\lambda}_{1i}^{\mathrm{opt}}\right)$ needs to be numerically computed by performing the block coordinate iteration while "freezing" the index $k \in \mathbb{N}_0$. With the converged pair $\left(\boldsymbol{\lambda}_{0i}^{\mathrm{opt}}, \boldsymbol{\lambda}_{1i}^{\mathrm{opt}}\right)$, the evaluation (7.14) is analytical for each $k \in \mathbb{N}_0$.

In our context, another case of interest is when F_i and hence G_i, is linear in $\boldsymbol{\mu}_i$. We next show that the proximal update $\boldsymbol{\mu}_i^{k+1}$ in this case can be computed analytically, obviating the zero order hold sub-iterations mentioned above. We summarize this result in the following Theorem (proof in Appendix A).

Theorem 7.1. *Given* $\boldsymbol{a} \in \mathbb{R}^N \setminus \{\boldsymbol{0}\}$, *let* $\boldsymbol{\Phi}(\boldsymbol{\mu}) := \langle \boldsymbol{a}, \boldsymbol{\mu} \rangle$ *for* $\boldsymbol{\mu} \in \Delta^{N-1}$. *Let* $\boldsymbol{C} \in \mathbb{R}^{N \times N}$ *be the squared Euclidean distance matrix, and for* $\varepsilon > 0$, *let* $\boldsymbol{\Gamma} := \exp\left(-\boldsymbol{C}/2\varepsilon\right)$. *For any* $\boldsymbol{\zeta} \in \Delta^{N-1}$, $\alpha > 0$, *the proximal operator*

$$prox_{\frac{1}{\alpha}\boldsymbol{\Phi}}^{W_\varepsilon}(\boldsymbol{\zeta}) = \exp\left(-\frac{1}{\alpha\varepsilon}\boldsymbol{a}\right) \odot \left(\boldsymbol{\Gamma}^{\top}\left(\boldsymbol{\zeta} \oslash \left(\boldsymbol{\Gamma}\exp\left(-\frac{1}{\alpha\varepsilon}\boldsymbol{a}\right)\right)\right)\right). \tag{7.15}$$

Remark 7.4. *The case of advection PDE shown in the first row of Table 7.1 can be treated via Theorem 7.1 with* $\boldsymbol{\Phi}_i(\boldsymbol{\mu}) = \langle \boldsymbol{V} + \boldsymbol{\nu}_i^k, \boldsymbol{\mu} \rangle$, *for given* $i \in [n]$. *In this discrete version,*

$V \in \mathbb{R}^N$ is the advection potential.

The case of interaction PDE shown in the third row of Table 7.1 can also be treated via Theorem 7.1 with $\mathbf{\Phi}_i(\boldsymbol{\mu}) = \langle \boldsymbol{U}\boldsymbol{\mu}^k + \boldsymbol{\nu}_i^k, \boldsymbol{\mu} \rangle$, for given $i \in [n]$. In this discrete version, $\boldsymbol{U} \in \mathbb{R}^{N \times N}$ is the interaction potential.

In the following, we show how to solve (7.14) for two case studies. Theorem 7.2 is for the porous medium equation (the last row of Table 7.1) [228], and Theorem 7.3 is for the Wasserstein barycenter case study (1.10).

Theorem 7.2. *Let* $\mathbf{\Gamma} := \exp(-\boldsymbol{C}/2\varepsilon)$. *For the porous medium equation, the vectors* $\left(\boldsymbol{\lambda}_{0i}^{opt}, \boldsymbol{\lambda}_{1i}^{opt}\right)$ *can be found by solving for* $\boldsymbol{y}_i, \boldsymbol{z}_i$ *from the following system of equations:*

$$\boldsymbol{y}_i \odot \left(\mathbf{\Gamma}^\top \boldsymbol{y}_i\right) = \boldsymbol{\zeta}^k \tag{7.16a}$$

$$\boldsymbol{z}_i \odot \left(\mathbf{\Gamma}^\top \boldsymbol{y}_i\right) = (\beta)^{\frac{1}{m-1}} \left(\frac{m-1}{m}\right)^{\frac{m}{m-1}} \left(-\frac{m}{m-1}\right)$$

$$\left(\mathbf{1}^\top \left(-\alpha\varepsilon \ln(\boldsymbol{z}_i) - \boldsymbol{\nu}_i^k\right)^m\right)^{\frac{2-m}{m-1}} \left(-\alpha\varepsilon \ln(\boldsymbol{z}_i) - \boldsymbol{\nu}_i^k\right)^{m-1} \tag{7.16b}$$

and then inverting the maps

$$\boldsymbol{y}_i := \exp\left(\frac{\boldsymbol{\lambda}_{0i}}{\alpha\varepsilon}\right), \quad \boldsymbol{z}_i := \exp\left(\frac{\boldsymbol{\lambda}_{1i}}{\alpha\varepsilon}\right). \tag{7.17}$$

Let $\left(\boldsymbol{y}_i^{\mathrm{opt}}, \boldsymbol{z}_i^{\mathrm{opt}}\right)$ *be the solution of* (7.16). *The vector* $\boldsymbol{\mu}_i^{k+1}$ *in* (7.14), *can then be obtained as*

$$\boldsymbol{\mu}_i^{k+1} = \boldsymbol{z}_i^{\mathrm{opt}} \odot \left(\mathbf{\Gamma}^\top \boldsymbol{y}_i^{\mathrm{opt}}\right).$$

For the case that $m = 2$, the $\boldsymbol{z}_i$ update equation (7.16b) will be:

$$\boldsymbol{z}_i \odot \left(\mathbf{\Gamma}_k^\top \boldsymbol{y}_i\right) = \frac{\beta}{2}\left(\alpha\varepsilon \ln(\boldsymbol{z}_i) + \boldsymbol{\nu}_i^k\right). \tag{7.18}$$

Equation (7.18) is a transcendental equation, with the solution written in terms of the Lambert W function as

$$z_i = \frac{-b}{a} W\left(\frac{-a}{b} \exp\left(\frac{-c}{b}\right)\right) \tag{7.19}$$

where W is the Lambert W function and $\boldsymbol{a} = (\boldsymbol{\Gamma}_k^\top \boldsymbol{y}_i)$, $b = \frac{\beta}{2}\alpha\varepsilon$, and $\boldsymbol{c} = \frac{\beta}{2}\boldsymbol{\nu}_i^k$.

Theorem 7.3. *Given $\boldsymbol{\xi} \in \mathcal{P}_2(\mathcal{X})$ and $\boldsymbol{\nu}^k \in \mathbb{R}^N$ let $\Phi(\boldsymbol{\mu}) := w W^2(\boldsymbol{\mu}, \boldsymbol{\xi}) + \langle \boldsymbol{\nu}^k, \boldsymbol{\mu} \rangle$. Let $\boldsymbol{C} \in \mathbb{R}^{N \times N}$ be the squared Euclidean distance matrix, and for $\varepsilon > 0$ let $\boldsymbol{\Gamma} := \exp(-\boldsymbol{C}/2\varepsilon)$, $\boldsymbol{K} := \exp(-\boldsymbol{C}/\varepsilon)$. For any $\boldsymbol{\zeta} \in \Delta^{N-1}$, $\alpha > 0$, and $0 < w < 1$ let $(\boldsymbol{y}^{\mathrm{opt}}, \boldsymbol{z}^{\mathrm{opt}}) \in \mathbb{R}_{>0}^N \times \mathbb{R}_{>0}^N$ be the solution of*

$$\boldsymbol{y} \odot (\boldsymbol{\Gamma}^\top \boldsymbol{z}) = \boldsymbol{\zeta}, \tag{7.20a}$$

$$\boldsymbol{z} \odot (\boldsymbol{\Gamma}^\top \boldsymbol{y}) = \boldsymbol{z}^{-\frac{1}{\alpha w}} \odot \exp\left(\frac{-\boldsymbol{\nu}^k}{\varepsilon}\right) \odot \boldsymbol{K}\boldsymbol{\xi} \oslash \left(\boldsymbol{K}\left(\boldsymbol{z}^{-\frac{1}{\alpha w}} \odot \exp\left(\frac{-\boldsymbol{\nu}^k}{\varepsilon}\right)\right)\right) \tag{7.20b}$$

Then

$$prox_{\frac{1}{\alpha}\Phi}^{W_\varepsilon}(\boldsymbol{\zeta}) = \boldsymbol{z}^{\mathrm{opt}} \odot (\boldsymbol{\Gamma}^\top \boldsymbol{y}^{\mathrm{opt}}). \tag{7.21}$$

To address the solution of (7.20), we propose the following recursion for $\ell = 1, 2, \ldots$, which is contracting in Thompson metric (1.17) on $\mathbb{R}_{>0}^N$

$$\boldsymbol{z}(:, \ell+1) = \left(\boldsymbol{z}(:, \ell)^{-\frac{1}{\alpha w}} \odot \exp\left(\frac{-\boldsymbol{\nu}(:, \ell)^k}{\varepsilon}\right) \odot \boldsymbol{K}\boldsymbol{\xi} \oslash \left(\boldsymbol{K}\left(\right.\right.\right.$$
$$\left.\left.\left. \boldsymbol{z}(:, \ell)^{-\frac{1}{\alpha w}} \odot \exp\left(\frac{-\boldsymbol{\nu}(:, \ell)^k}{\varepsilon}\right)\right)\right)\right) \oslash (\boldsymbol{\Gamma}^\top \boldsymbol{z}(:, \ell)). \tag{7.22}$$

The preceding results show that the update (7.11a) can be numerically realized via (7.13)-(7.14). We next consider numerically realizing the update (7.11b).

7.3.2   The ζ update

The update (7.11b) can be seen as a problem of computing the Sinkhorn regularized Wasserstein barycenter with an extra linear regularization. We next show that as in Section 7.3.1, dualization also helps to solve problems of this type. In particular, the following Proposition from [53, Sec. 4.1], rephrased in our notation, will be useful in the sequel.

Proposition 7.2. *([53, Proposition 1]) Let*

$$W^2_{\varepsilon,\boldsymbol{\mu}_i}(\boldsymbol{\zeta}) := \min_{\boldsymbol{M}_i \in \Pi_N(\boldsymbol{\mu}_i,\boldsymbol{\zeta})} \left\langle \frac{1}{2}\boldsymbol{C} + \varepsilon \log \boldsymbol{M}_i, \boldsymbol{M}_i \right\rangle, \quad \varepsilon > 0,$$

for given $\boldsymbol{\mu}_i \in \Delta^{N-1}$ for all $i \in [n]$, and for a given squared Euclidean distance matrix $\boldsymbol{C} \in \mathbb{R}^{N \times N}$. Let the superscript * denote the Legendre-Fenchel conjugate. Given weights $w_1, \cdots, w_n > 0$, linear operator $\mathcal{A}$, and a convex real-valued function J, consider the variational problem

$$\boldsymbol{\zeta}^{\mathrm{opt}} = \arg\min_{\boldsymbol{\zeta} \in \Delta^{N-1}} J\left(\mathcal{A}\boldsymbol{\zeta}\right) + \sum_{i=1}^{n} w_i W^2_{\varepsilon,\boldsymbol{\mu}_i}(\boldsymbol{\zeta}). \tag{7.23}$$

The dual problem of (7.23) *is given by*

$$\left(\boldsymbol{u}_1^{\mathrm{opt}}, \cdots, \boldsymbol{u}_n^{\mathrm{opt}}, \boldsymbol{v}^{\mathrm{opt}}\right) = \arg\min_{(\boldsymbol{u}_1,\cdots,\boldsymbol{u}_n,\boldsymbol{v}) \in \mathbb{R}^{(n+1)N}} J^*\left(\boldsymbol{v}\right) + \sum_{i=1}^{n} w_i \left(W^2_{\varepsilon,\boldsymbol{\mu}_i}\right)^*\left(\boldsymbol{u}_i\right)$$

$$\text{subject to} \quad \mathcal{A}^*\boldsymbol{v} + \sum_{i=1}^{n} w_i \boldsymbol{u}_i = \boldsymbol{0}, \tag{7.24}$$

and the primal-dual relation giving the minimizer in (7.23) *is*

$$\boldsymbol{\zeta}^{\mathrm{opt}} = \nabla_{\boldsymbol{u}_i}\left(W^2_{\varepsilon,\boldsymbol{\mu}_i}\right)^*\left(\boldsymbol{u}_i^{\mathrm{opt}}\right) \in \Delta^{N-1}, \quad \text{for all } i \in [n]. \tag{7.25}$$

We recast (7.11b) in the form (7.23) by setting the probability vectors $\boldsymbol{\mu}_i \equiv \boldsymbol{\mu}_i^{k+1}$, the weights $w_1 = w_2 = \cdots = w_n = 1$, the operator $\mathcal{A}$ as identity, and the function

$J(\cdot) \equiv \langle -\frac{2}{\alpha} \boldsymbol{\nu}_{\text{sum}}^k, \cdot \rangle$. Since J is linear, we have

$$J^*(\boldsymbol{v}) = \begin{cases} 0 & \text{if} \quad \boldsymbol{v} = -\frac{2}{\alpha} \boldsymbol{\nu}_{\text{sum}}^k, \\ +\infty & \text{otherwise.} \end{cases} \tag{7.26}$$

Also, $\mathcal{A}$ being the identity operator, we get $\mathcal{A}^* \boldsymbol{v} = \boldsymbol{v}$. Therefore, the dual problem (7.24) corresponding to (7.11b) becomes

$$\left(\boldsymbol{u}_1^{\text{opt}}, \cdots, \boldsymbol{u}_n^{\text{opt}} \right) = \underset{(\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n) \in \mathbb{R}^{nN}}{\arg\min} \sum_{i=1}^n \left(W_{\varepsilon, \boldsymbol{\mu}_i^{k+1}}^2 \right)^* (\boldsymbol{u}_i)$$

$$\text{subject to} \quad \sum_{i=1}^n \boldsymbol{u}_i = \frac{2}{\alpha} \boldsymbol{\nu}_{\text{sum}}^k. \tag{7.27}$$

Consequently, the update (7.11b) can be performed by first solving the problem (7.27), and then evaluating the gradient of the Legendre-Fenchel conjugate (7.25) at the minimizer of (7.27). Furthermore, taking advantage of the structure of these Legendre-Fenchel conjugates allows us to deduce the following (proof in Appendix C).

Theorem 7.4. *Given $\alpha, \varepsilon > 0$, the squared Euclidean distance matrix $\boldsymbol{C} \in \mathbb{R}^{N \times N}$, and the probability vectors $\boldsymbol{\mu}_i^{k+1} \in \Delta^{N-1}$ for all $i \in [n]$, $k \in \mathbb{N}_0$, let $\boldsymbol{\Gamma} := \exp\left(-\boldsymbol{C}/2\varepsilon \right)$. The dual problem (7.27) corresponding to (7.11b) can be rewritten as*

$$\left(\boldsymbol{u}_1^{\text{opt}}, \cdots, \boldsymbol{u}_n^{\text{opt}} \right) = \underset{(\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n) \in \mathbb{R}^{nN}}{\arg\min} \sum_{i=1}^n \left\langle \boldsymbol{\mu}_i^{k+1}, \log \left(\boldsymbol{\Gamma} \exp \left(\boldsymbol{u}_i / \varepsilon \right) \right) \right\rangle$$

$$\text{subject to} \quad \sum_{i=1}^n \boldsymbol{u}_i = \frac{2}{\alpha} \boldsymbol{\nu}_{\text{sum}}^k. \tag{7.28}$$

The update $\boldsymbol{\zeta}^{k+1}$ in (7.11b) is given by

$$\boldsymbol{\zeta}^{k+1} = \exp \left(\boldsymbol{u}_i^{\text{opt}} / \varepsilon \right) \odot \left(\boldsymbol{\Gamma} \left(\boldsymbol{\mu}_i^{k+1} \oslash \left(\boldsymbol{\Gamma} \exp \left(\boldsymbol{u}_i^{\text{opt}} / \varepsilon \right) \right) \right) \right) \in \Delta^{N-1}, \quad \text{for all } i \in [n]. \tag{7.29}$$

We observe that (7.28) has a separable sum objective where each summand is a

weighted log-sum-exp (thus convex). Denoting these summands as

$$f_i(\boldsymbol{u}_i) := \left\langle \boldsymbol{\mu}_i^{k+1}, \log\left(\boldsymbol{\Gamma}\exp\left(\boldsymbol{u}_i/\varepsilon\right)\right)\right\rangle, \quad \boldsymbol{u}_i \in \mathbb{R}^N \quad \text{for all } i \in [n], \tag{7.30}$$

we write (7.28) in the scaled ADMM form (1.16):

$$\boldsymbol{u}_i^{\ell+1} = \text{prox}_{\frac{1}{\tau}f_i}^{\|\cdot\|_2}\left(\boldsymbol{z}_i^\ell - \widetilde{\boldsymbol{\nu}}_i^\ell\right), \quad i \in [n], \tag{7.31a}$$

$$\boldsymbol{z}^{\ell+1} = \text{proj}_{\mathcal{C}}\left(\boldsymbol{u}^{\ell+1} + \widetilde{\boldsymbol{\nu}}^\ell\right), \tag{7.31b}$$

$$\widetilde{\boldsymbol{\nu}}_i^{\ell+1} = \widetilde{\boldsymbol{\nu}}_i^\ell + \left(\boldsymbol{u}_i^{\ell+1} - \boldsymbol{z}_i^{\ell+1}\right), \quad i \in [n], \tag{7.31c}$$

where $\ell \in \mathbb{N}_0$ is the ADMM iteration index while holding the index k fixed, $\tau > 0$, and $\boldsymbol{u}^\ell := (\boldsymbol{u}_1^\ell, \cdots, \boldsymbol{u}_n^\ell) \in \mathbb{R}^{nN}$, $\boldsymbol{z}^\ell := (\boldsymbol{z}_1^\ell, \cdots, \boldsymbol{z}_n^\ell) \in \mathbb{R}^{nN}$, $\widetilde{\boldsymbol{\nu}}^\ell := (\widetilde{\boldsymbol{\nu}}_1^\ell, \cdots, \widetilde{\boldsymbol{\nu}}_n^\ell) \in \mathbb{R}^{nN}$ for all $\ell \in \mathbb{N}_0$. The constraint set $\mathcal{C}$ in (7.31b) corresponds to the equality constraint in (7.28), i.e.,

$$\mathcal{C} := \left\{(\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n) \in \mathbb{R}^{nN} \mid \boldsymbol{z}_1 + \cdots + \boldsymbol{z}_n = \frac{2}{\alpha}\boldsymbol{\nu}_{\text{sum}}^k\right\}. \tag{7.32}$$

To proceed further, we need the following Lemma (proof in Appendix D).

Lemma 7.1. *For any* $\boldsymbol{v} := (\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n) \in \mathbb{R}^{nN}$, *where the subvectors* $\boldsymbol{v}_i \in \mathbb{R}^N$ *for all* $i \in [n]$, *let* $\overline{\boldsymbol{v}} := \frac{1}{n}\sum_{i=1}^n \boldsymbol{v}_i \in \mathbb{R}^N$. *Then the Euclidean projection of* $\boldsymbol{v}$ *onto* $\mathcal{C}$ *in (7.32) is*

$$\text{proj}_{\mathcal{C}}(\boldsymbol{v}) = \left(\boldsymbol{v}_1 - \overline{\boldsymbol{v}} + \frac{2}{n\alpha}\boldsymbol{\nu}_{\text{sum}}^k, \cdots, \boldsymbol{v}_n - \overline{\boldsymbol{v}} + \frac{2}{n\alpha}\boldsymbol{\nu}_{\text{sum}}^k\right) \in \mathbb{R}^{nN}.$$

Thanks to Lemma 7.1, we can parallelize (7.31b) as

$$\boldsymbol{z}_i^{\ell+1} = \left(\boldsymbol{u}_i^{\ell+1} - \frac{1}{n}\sum_{i=1}^n \boldsymbol{u}_i^{\ell+1}\right) + \left(\widetilde{\boldsymbol{\nu}}_i^\ell - \frac{1}{n}\sum_{i=1}^n \widetilde{\boldsymbol{\nu}}_i^\ell\right) + \frac{2}{n\alpha}\boldsymbol{\nu}_{\text{sum}}^k, \quad i \in [n]. \tag{7.33}$$

Therefore, (7.28) can be solved in a distributed manner:

$$\boldsymbol{u}_i^{\ell+1} = \text{prox}_{\frac{1}{\tau}f_i}^{\|\cdot\|_2}\left(\boldsymbol{z}_i^\ell - \widetilde{\boldsymbol{\nu}}_i^\ell\right), \quad i \in [n], \tag{7.34a}$$

$$\boldsymbol{z}_i^{\ell+1} = \left(\boldsymbol{u}_i^{\ell+1} - \frac{1}{n}\sum_{i=1}^n \boldsymbol{u}_i^{\ell+1}\right) + \left(\widetilde{\boldsymbol{\nu}}_i^\ell - \frac{1}{n}\sum_{i=1}^n \widetilde{\boldsymbol{\nu}}_i^\ell\right) + \frac{2}{n\alpha}\boldsymbol{\nu}_{\text{sum}}^k, \quad i \in [n], \tag{7.34b}$$

149

$$\widetilde{\boldsymbol{\nu}}_i^{\ell+1} = \widetilde{\boldsymbol{\nu}}_i^{\ell} + \left(\boldsymbol{u}_i^{\ell+1} - \boldsymbol{z}_i^{\ell+1} \right), \quad i \in [n]. \tag{7.34c}$$

The proximal update (7.34a) does not admit an analytical solution. To compute (7.34a), we take advantage of the structured Hessian (see Appendix E) of the proximal objective and implement the Newton's method with variable step size computed by backtracking line search. The diagonal plus sum of rank one structured Hessian in Appendix D makes the per iteration complexity for the Newton's method to be $\mathcal{O}(N^2)$ flops instead of $\mathcal{O}(N^3)$ flops–the latter would be the case for Cholesky factorization-based solution of the associated linear system. Figure 7.2 shows that the typical convergence for the Newton's method occurs in approx. 5 iterations, much faster than gradient descent (see Figure 7.2 caption for details).

In the following (Section 7.4), we outline the overall implementation of our distributed computational framework to solve (7.11) and (7.34). first for each agent, we update $\boldsymbol{\mu}_i^1$ with $\mathrm{prox}_{\frac{1}{\alpha}\Phi}^{W_\varepsilon}(.)$ and generate samples from the known initial distribution. Then, with a random value for $\boldsymbol{\zeta}^0$ and $\boldsymbol{\nu}_i^0$, and

7.4   The Overall Algorithm

In Fig. 7.3, we detail the computational framework proposed in Section 7.3.1 and Section 7.3.2. We view Fig. 7.3 as an expanded version of the high level schematic given in Fig. 7.1, i.e., Fig. 7.3 depicts the low level details omitted in Fig. 7.1.

Note that in the inner layer ADMM, to update $\boldsymbol{z}_i$ in Eq. (7.34b), we need $\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{u}_i^{\ell+1}$ from the other distributed processors and the pipeline below the diagram in Fig. 7.3 gathers these data from all distributed processors and feeds to Eq. (7.34b).

In summary, the computational steps are as follows.

Step 0. Split the objective F as Eq. (7.3a) and relabel the argument of the func-
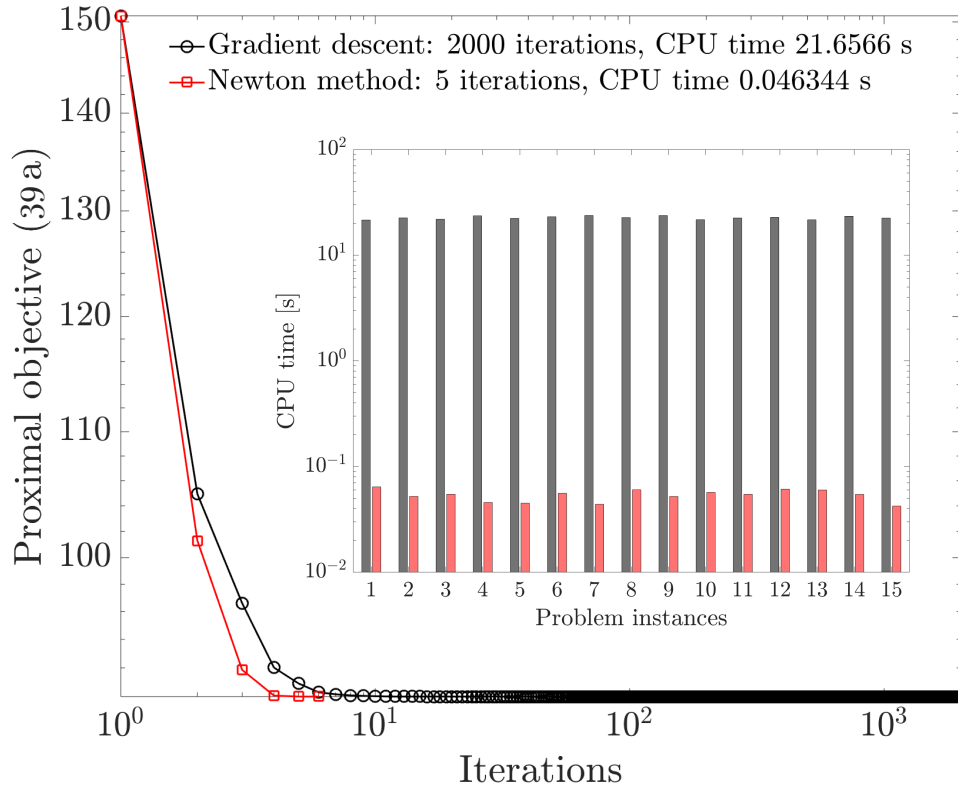
Figure 7.2: *Main plot:* A typical instance of the proximal optimization problem (7.34a) with $N = 441$, $\varepsilon = \tau = 0.1$, random initial guess, randomly generated input data (i.e., proximal argument in $\mathbb{R}^N$), random parameter $\boldsymbol{\mu} \in \Delta^{N-1}$, and the Euclidean distance matrix $\boldsymbol{C} \in \mathbb{R}^{N \times N}$ for uniform grid over $[-1, 1]^2$ with spatial discretization length 0.1 in both directions. The problem instance was solved via the gradient descent and the Newton's method with the same numerical tolerance 10^{-4}. The stopping criterion for the gradient descent was the norm of the gradient being less than or equal to the numerical tolerance. For the Newton's method, we used the standard stopping criterion [2, p. 487]: one half of the squared Newton decrement being less than or equal to the numerical tolerance. Both algorithms used variable step size via backtracking line search (see Appendix E) with parameters $\alpha_0 = 0.3$, $\beta_0 = 0.7$. For gradient descent, the proximal objective after the last iteration was equal to 90.018062265357955; the same for Newton's method was equal to 90.018072956312977. *Inset plot:* The CPU time comparisons for 15 instances of (7.34a) with randomly chosen initial guess, proximal argument and parameter $\boldsymbol{\mu} \in \Delta^{N-1}$ while keeping all other parameters fixed and same as before across all problem instances. The longer (resp. shorter) bars are for the gradient descent (resp. Newton's method). For all 15 problem instances, the gradient descent took 2000–2002 iterations while the Newton's method required 5–6 iterations. So the convergence trend shown in the main plot is typical.
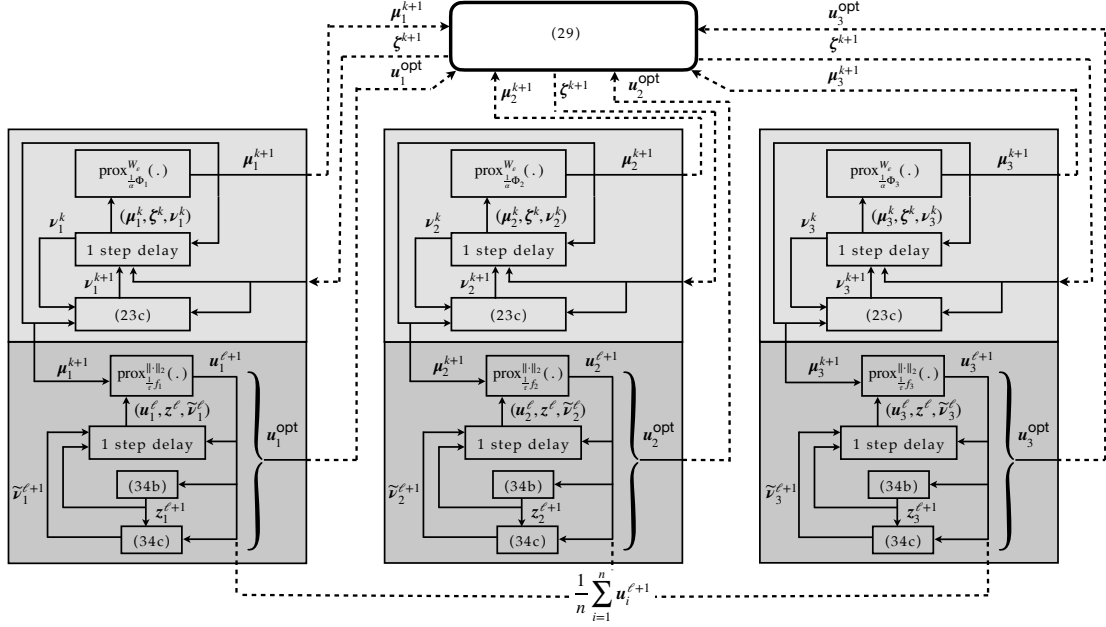
Figure 7.3: Detailed schematic of the proposed computational framework. As in Fig. 7.1, the lighter and darker shades correspond to the "upstairs" and "downstairs" computation in the distributed processors, respectively, which in turn, correspond to the outer and inner layer ADMM, respectively.

tionals F_i as μ_i $\forall i \in [n]$.

Step 1. Initialize μ_i^0, ζ^0 everywhere positive, and ν_i^0 arbitrary $\forall i \in [n]$.

Step 2. Perform distributed "upstairs" updates Eq. (7.11a) for μ_i^{k+1} via Proposition 7.1 (outer layer ADMM).

Step 3. Perform distributed "downstairs" updates u_i^{opt} from the inner layer ADMM Eq. (7.34).

Step 4. Perform centralized update for ζ^{k+1} using Eq. (7.29) (outer layer ADMM).

Step 5. Perform distributed "upstairs" updates for ν_i^{k+1} using Eq. (7.11c) (outer layer ADMM).

The above steps are repeated until a user-specified maximum number of outer layer iterations are done, or the maximum of the pairwise Wasserstein distances fall

below a prescribed tolerance.

7.5 Convergence Guarantee for the Inner Layer ADMM

In the following, we present sufficient conditions that guarantee the convergence of inner layer ADMM Eq. (7.34). To this end, we need two preparatory lemmas.

Lemma 7.2. *[229, p. 58, Thm. 2.1.6] A C^2 convex function f with $\operatorname{domain}(f) = \mathbb{R}^N$, has Lipschitz continuous gradient w.r.t. $\|\cdot\|_2$ with Lipschitz constant $L > 0$ if $v^\top \nabla^2 f(u) v \leq L \|v\|_2^2$ for all $u, v \in \mathbb{R}^N$.*

Lemma 7.3. *The C^2 convex function f given by Eq. (D.17) with $\operatorname{domain}(f) = \mathbb{R}^N$, has Lipschitz continuous gradient w.r.t. $\|\cdot\|_2$ with Lipschitz constant $L = \dfrac{1}{\varepsilon^2} \left\| \Gamma \mu^{k+1} \right\|_\infty$.*

Proof. Let $e := \exp(u/\varepsilon)$. From Eq. (D.20), for all $u, v \in \mathbb{R}^N$, we have

$$
\begin{aligned}
v^\top \nabla^2 f(u) v = \frac{1}{\varepsilon^2} v^\top \Big[&\operatorname{diag}\left(\left(\Gamma^\top \mu^{k+1} \right) \odot e \oslash (\Gamma e) \right) - \\
&\operatorname{diag}\left(\left(\Gamma^\top \mu^{k+1} \right) \oslash (\Gamma e)^2 \right) \Gamma \odot (e e^\top) \Big] v \leq \frac{1}{\varepsilon^2} v^\top \operatorname{diag}\left(\left(\Gamma^\top \mu^{k+1} \right) \odot e \oslash (\Gamma e) \right) v,
\end{aligned}
$$

(7.35)

since the quadratic term followed by the minus sign is nonnegative. Hence Eq. (7.35) yields

$$
v^\top \nabla^2 f(u) v \leq \frac{1}{\varepsilon^2} \left\| \left(\Gamma^\top \mu^{k+1} \right) \odot e \oslash (\Gamma e) \right\|_\infty \|v\|_2^2 \leq \frac{1}{\varepsilon^2} \left\| \Gamma^\top \mu^{k+1} \right\|_\infty \left\| e \oslash (\Gamma e) \right\|_\infty \|v\|_2^2.
$$

(7.36)

Recall that $\Gamma := \exp\left(-C/2\varepsilon \right)$ where $C \in \mathbb{R}^{N \times N}$ is a squared Euclidean distance matrix. So the entries of the symmetric matrix C are in $[0, \infty)$ and thus, the entries of the symmetric matrix Γ are in $(0, 1]$ with all diagonal entries being equal to 1.

Therefore, $\left\| \boldsymbol{e} \oslash (\boldsymbol{\Gamma} \boldsymbol{e}) \right\|_\infty \leq 1$, and Eq. (7.36) gives

$$\boldsymbol{v}^\top \nabla^2 f(\boldsymbol{u}) \boldsymbol{v} \leq \frac{1}{\varepsilon^2} \left\| \boldsymbol{\Gamma} \boldsymbol{\mu}^{k+1} \right\|_\infty \|\boldsymbol{v}\|_2^2 \qquad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^N,$$

where we dropped the transpose due to the symmetry of $\boldsymbol{\Gamma}$. Invoking Lemma 7.2,

we conclude the proof. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ □

Theorem 7.5. *Let $\boldsymbol{C}, \boldsymbol{\Gamma}$, and $\boldsymbol{\mu}_i^{k+1} \in \Delta^{N-1}$ for all $i \in [n]$, $k \in \mathbb{N}_0$, as in Theorem 7.4. If*

$$\tau > \frac{\sqrt{2}}{\varepsilon^2} \left\| \boldsymbol{\Gamma} \boldsymbol{\mu}_n^{k+1} \right\|_\infty, \tag{7.37}$$

then the sequence $(\boldsymbol{u}_1^\ell, \cdots, \boldsymbol{u}_n^\ell)$ generated by the inner layer ADMM given in Eq. (7.34)

converge to the optimal solutions of problem Eq. (7.28), i.e., $(\boldsymbol{u}_1^\ell, \cdots, \boldsymbol{u}_n^\ell) \xrightarrow{\ell \nearrow \infty} (\boldsymbol{u}_1^{\mathrm{opt}}, \cdots, \boldsymbol{u}_n^{\mathrm{opt}})$.

Proof. We start our proof by presenting a sufficient condition for convergence of

certain generic multi-block ADMM, and show that the inner layer ADMM given in

Eq. (7.34) satisfies these conditions.

To this end, we start with the following convex minimization problem:

$$\min_{(\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n) \in \mathbb{R}^{nN}} g_1(\boldsymbol{u}_1) + g_2(\boldsymbol{u}_2) + \cdots + g_n(\boldsymbol{u}_n)$$

$$\text{subject to } \boldsymbol{A}_1 \boldsymbol{u}_1 + \boldsymbol{A}_2 \boldsymbol{u}_2 + \cdots + \boldsymbol{A}_n \boldsymbol{u}_n = \boldsymbol{b},$$

$$\boldsymbol{u}_i \in \mathcal{U}_i \quad \text{for all } i \in [n], \tag{7.38}$$

where $\boldsymbol{A}_i \in \mathbb{R}^{N \times N}, \boldsymbol{b} \in \mathbb{R}^N$, the sets $\mathcal{U}_i \subseteq \mathbb{R}^N$ are closed convex, and $g_i : \mathcal{U}_i \to \mathbb{R}$ are

closed convex functions for all $i \in [n]$. The augmented Lagrangian for Eq. (7.38) is

$$\mathcal{L}_\tau(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n; \boldsymbol{\lambda}) := g_1(\boldsymbol{u}_1) + \cdots + g_n(\boldsymbol{u}_n) + \left\langle \boldsymbol{\lambda}, \sum_{i=1}^n \boldsymbol{A}_i \boldsymbol{u}_i - \boldsymbol{b} \right\rangle + \frac{\tau}{2} \left\| \sum_{i=1}^n \boldsymbol{A}_i \boldsymbol{u}_i - \boldsymbol{b} \right\|^2,$$

$$\tag{7.39}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^N$ is the Lagrange multiplier, and $\tau > 0$ is a penalty parameter. For

Eq. (7.38), consider the multi-block ADMM recursions:

$$
\begin{cases}
\boldsymbol{u}_1^{\ell+1} = \operatorname{argmin}_{\boldsymbol{u}_1 \in \mathcal{U}_1} \mathcal{L}_\tau \left(\boldsymbol{u}_1, \boldsymbol{u}_2^\ell, \ldots, \boldsymbol{u}_n^\ell; \boldsymbol{\lambda}^\ell \right), \\[4pt]
\boldsymbol{u}_2^{\ell+1} = \operatorname{argmin}_{\boldsymbol{u}_2 \in \mathcal{U}_2} \mathcal{L}_\tau \left(\boldsymbol{u}_1^{\ell+1}, \boldsymbol{u}_2, \boldsymbol{u}_3^\ell, \ldots, \boldsymbol{u}_n^\ell; \boldsymbol{\lambda}^\ell \right), \\[4pt]
\quad\vdots \\[4pt]
\boldsymbol{u}_n^{\ell+1} = \operatorname{argmin}_{\boldsymbol{u}_n \in \mathcal{U}_n} \mathcal{L}_\tau \left(\boldsymbol{u}_1^{\ell+1}, \boldsymbol{u}_2^{\ell+1}, \ldots, \boldsymbol{u}_{n-1}^{\ell+1}, \boldsymbol{u}_n; \boldsymbol{\lambda}^\ell \right), \\[4pt]
\boldsymbol{\lambda}^{\ell+1} = \boldsymbol{\lambda}^\ell + \tau \left(\sum_{i=1}^n \boldsymbol{A}_i \boldsymbol{u}_i^{\ell+1} - \boldsymbol{b} \right).
\end{cases}
\tag{7.40}
$$

For Eq. (7.38), when the following conditions [230, Corollary 3.5]:

c1. the matrices $\boldsymbol{A}_i$ have full column rank for all $i \in [n-1]$, and $\boldsymbol{A}_n = \boldsymbol{I}_N$,

c2. the sets $\mathcal{U}_i$ are closed convex for all $i \in [n]$,

c3. the mappings g_i are lower bounded for all $i \in [n]$,

c4. $\tau > \sqrt{2}L$ where L is Lipschitz constant (w.r.t. $\|\cdot\|_2$) for $\nabla_{\boldsymbol{u}_n} g_n$,

are satisfied, then as the recursion index $\ell \nearrow \infty$, the solution of the multi-block

ADMM Eq. (7.40) converges to the optimal solutions of Eq. (7.38). Notice that the

recursions Eq. (7.34) associated with the problem Eq. (7.28), are indeed an instance

of the generic recursions Eq. (7.40) associated with Eq. (7.38). In particular,

$$
g_i(\boldsymbol{u}_i) \equiv \left\langle \boldsymbol{\mu}_i^{k+1}, \log \left(\boldsymbol{\Gamma} \exp \left(\boldsymbol{u}_i / \varepsilon \right) \right) \right\rangle,
$$

where the probability vectors $\boldsymbol{\mu}_i^{k+1} \in \Delta^{N-1}$ for all $i \in [n]$, $k \in \mathbb{N}_0$. Thus motivated,

we check the conditions c1-c4.

Specifically, condition c1 for Eq. (7.28) is satisfied because $\boldsymbol{A}_i = \boldsymbol{I}_N$ for all $i \in [n]$.

Condition c2 for Eq. (7.28) holds since $\mathcal{U}_i = \mathbb{R}^N$ for all $i \in [n]$, which are closed as

well as affine (hence convex) sets.

For condition c3, we need to verify that the mappings $\boldsymbol{u}_i \mapsto g_i(\boldsymbol{u}_i) = \left\langle \boldsymbol{\mu}_i^{k+1}, \log\left(\boldsymbol{\Gamma}\exp\left(\boldsymbol{u}_i/\varepsilon\right)\right) \right\rangle$ are uniformly lower bounded. The lower bound for $\boldsymbol{u}_i \mapsto g_i(\boldsymbol{u}_i)$ can be found as the following unconstrained minimum

$$g_i^{\mathrm{opt}} := \min_{\boldsymbol{u}_i \in \mathbb{R}^N} \left\langle \boldsymbol{\mu}_i^{k+1}, \log\left(\boldsymbol{\Gamma}\exp\left(\boldsymbol{u}_i/\varepsilon\right)\right) \right\rangle, \tag{7.41}$$

which is the minimum of a convex combination of log-sum-exp composed with an affine map.

By choosing matrix $\boldsymbol{A}$ as an invertible matrix and introducing two new variables, $\tilde{\boldsymbol{u}} \in \mathbb{R}^N$ and $\boldsymbol{y} \in \mathbb{R}^N$, we reformulate problem Eq. (7.41) as:

$$\min_{\boldsymbol{y} \in \mathbb{R}^N} f_0(\boldsymbol{y}) := \mu_j \log \sum_{i=1}^N \exp(\boldsymbol{y}_i)$$

$$\text{subject to} \quad \boldsymbol{u}/\varepsilon = \boldsymbol{A}\tilde{\boldsymbol{u}},$$

$$\boldsymbol{A}\tilde{\boldsymbol{u}} + \log\boldsymbol{\gamma}_j = \boldsymbol{y}, \tag{7.42}$$

where $\boldsymbol{y}_i$ is ith element of vector $\boldsymbol{y}$. The Lagrangian of the reformulated problem is

$$L(\boldsymbol{u},\tilde{\boldsymbol{u}},\boldsymbol{y},\boldsymbol{\kappa},\boldsymbol{\eta}) = \mu_j \log \sum_{i=1}^N \exp(\boldsymbol{y}_i) + \boldsymbol{\eta}^\top\left(\boldsymbol{A}\tilde{\boldsymbol{u}} + \log\boldsymbol{\gamma}_j - \boldsymbol{y}\right) + \boldsymbol{\kappa}^\top\left(\boldsymbol{u}/\varepsilon - \boldsymbol{A}\tilde{\boldsymbol{u}}\right),$$

$$\tag{7.43}$$

where $\boldsymbol{\kappa}$ and $\boldsymbol{\eta}$ are the Lagrangian multipliers, and the corresponding Lagrange dual function is defined as

$$h(\boldsymbol{\eta},\boldsymbol{\kappa}) = \inf_{\boldsymbol{u},\tilde{\boldsymbol{u}},\boldsymbol{y}} \left\{ \mu_j \log \sum_{i=1}^N \exp(\boldsymbol{y}_i) + \boldsymbol{\eta}^\top\left(\boldsymbol{A}\tilde{\boldsymbol{u}} + \log\boldsymbol{\gamma}_j - \boldsymbol{y}\right) + \boldsymbol{\kappa}^\top\left(\boldsymbol{u}/\varepsilon - \boldsymbol{A}\tilde{\boldsymbol{u}}\right) \right\}.$$

$$\tag{7.44}$$

Minimizing over $\boldsymbol{u}$ results in $h(\boldsymbol{\eta},\boldsymbol{\kappa}) = -\infty$ unless $\boldsymbol{\kappa} = \boldsymbol{0}$. Substituting $\boldsymbol{\kappa} = \boldsymbol{0}$ in Eq. (7.44), we get

$$h(\boldsymbol{\eta}) = \inf_{\tilde{\boldsymbol{u}},\boldsymbol{y}} \left(\mu_j \log \sum_{i=1}^N \exp(\boldsymbol{y}_i) + \boldsymbol{\eta}^\top\left(\boldsymbol{A}\tilde{\boldsymbol{u}} + \log\boldsymbol{\gamma}_j - \boldsymbol{y}\right) \right). \tag{7.45}$$

Minimizing over $\tilde{\boldsymbol{u}}$ results in $h(\boldsymbol{\eta}) = -\infty$ unless $\boldsymbol{A}^\top \boldsymbol{\eta} = \boldsymbol{0}$. So,

$$h(\boldsymbol{\eta}) = \boldsymbol{\eta}^\top \log \boldsymbol{\gamma}_j + \inf_{\boldsymbol{y}} \left(\mu_j \log \sum_{i=1}^N \exp(\boldsymbol{y}_i) - \boldsymbol{\eta}^\top \boldsymbol{y} \right) = \boldsymbol{\eta}^\top \log \boldsymbol{\gamma}_j + f_0^*(\boldsymbol{\eta}), \qquad (7.46)$$

where the conjugate of f_0 is

$$f_0^* = \begin{cases} \left\langle \boldsymbol{\eta}, \log \dfrac{\boldsymbol{\eta}}{\mu_j} \right\rangle & \boldsymbol{\eta} \geq 0, \ \mathbf{1}^\top \boldsymbol{\eta} = 1, \\ \\ \infty & \text{otherwise.} \end{cases} \qquad (7.47)$$

Therefore, the dual problem of Eq. (7.42) is

$$\max_{\boldsymbol{\eta} \in \mathbb{R}^N} \ \boldsymbol{\eta}^\top \log \boldsymbol{\gamma}_j - \left\langle \boldsymbol{\eta}, \log \dfrac{\boldsymbol{\eta}}{\mu_j} \right\rangle$$

$$\text{subject to} \quad \boldsymbol{\eta} \geq 0,$$

$$\mathbf{1}^\top \boldsymbol{\eta} = 1,$$

$$\boldsymbol{A}^\top \boldsymbol{\eta} = 0. \qquad (7.48)$$

The solution to the above entropy maximization problem provides a lower bound for the mappings $\boldsymbol{u} \mapsto \mu_j \log \langle \boldsymbol{\gamma}_j, \exp(\boldsymbol{u}/\varepsilon) \rangle$ in $\mathbb{R}^N$, thus helping satisfy condition c3.

From Lemma 7.3, $\nabla_{\boldsymbol{u}_n} g_n$ has the Lipschitz constant $L = \dfrac{1}{\varepsilon^2} \left\| \boldsymbol{\Gamma}^\top \boldsymbol{\mu}^{k+1} \right\|_\infty$. So, by choosing

$$\tau > \dfrac{\sqrt{2}}{\varepsilon^2} \left\| \boldsymbol{\Gamma}^\top \boldsymbol{\mu}^{k+1} \right\|_\infty$$

we satisfy condition c4. This completes the proof. $\qquad \qquad \square$

7.6   Experiments

In this section, we provide numerical examples to demonstrate the proposed distributed computation framework.
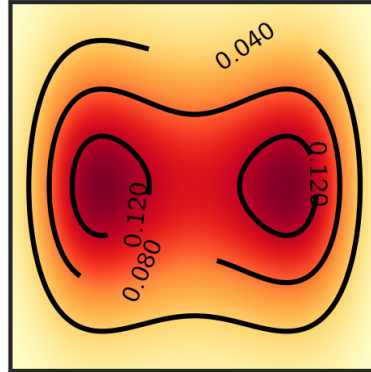
Figure 7.4: The analytical stationary solution for the FPK equation (7.49), given by $\boldsymbol{\mu}_\infty = \frac{1}{Z} \exp\left(-\beta\left((1 + x_1^4)/4 + (x_2^2 - x_1^2)/2\right)\right) \mathrm{d}\boldsymbol{x}$, where Z is the normalization constant.

7.6.1   Linear Fokker–Planck equation

In this example, we consider a linear variant of the Fokker–Planck equation,

$$\frac{\partial\mu}{\partial t} = \nabla \cdot (\mu\nabla V) + \beta^{-1}\Delta\mu, \quad \mu(\boldsymbol{x}, 0) = \mu_0(\boldsymbol{x}) \tag{7.49}$$

where $V(x_1, x_2) = \frac{1}{4}\left(1 + x_1^4\right) + \frac{1}{2}\left(x_2^2 - x_1^2\right)$ and $\boldsymbol{x} = (x_1, x_2) \in [-2, 2]^2$. As shown in [5], the stationary measure is $\mu_\infty(\boldsymbol{x}) \propto \exp(-\beta V(\boldsymbol{x}))\mathrm{d}\boldsymbol{x}$, which for our choice of V, is bimodal (see Figure 7.4).

Here, $n = 2$ and by looking at Table 7.1, we choose

$$F_1(\boldsymbol{\mu}) = \langle \boldsymbol{V}_k, \boldsymbol{\mu} \rangle \tag{7.50a}$$

$$F_2(\boldsymbol{\mu}) = \left\langle \beta^{-1}\log\boldsymbol{\mu}, \boldsymbol{\mu} \right\rangle \tag{7.50b}$$

where the drift potential vector $\boldsymbol{V}_k \in \mathbb{R}^N$ is given by $\boldsymbol{V}_k(i) := V\left(\boldsymbol{x}_k^i\right), i = 1, \ldots, N$. We artificially relabeled the argument of the functionals F_1 and F_2 as $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$,

respectively. Because F_1 is linear in $\boldsymbol{\mu}_1$, we use (7.15) with $\boldsymbol{\Phi}_1(\boldsymbol{\mu}) = \langle \boldsymbol{V}_{k-1} + \boldsymbol{\nu}_1^k, \boldsymbol{\mu} \rangle$ to analytically compute the proximal update $\boldsymbol{\mu}_1^{k+1}$. The simulation parameters are considered to be $\alpha = 12$, $\tau = 150$, $\beta = 1$, and $\varepsilon = 5 \times 10^{-2}$. We define $G_2(\boldsymbol{\mu_2}) :=$ $F_2(\boldsymbol{\mu_2}) + \langle \boldsymbol{\nu}_2^k, \boldsymbol{\mu_2} \rangle = \langle \beta^{-1} \log \boldsymbol{\mu}_2 + \boldsymbol{\nu}_2^k, \boldsymbol{\mu_2} \rangle$, and compute the proximal update $\boldsymbol{\mu}_2^{k+1}$ by (7.14). In this case, we use the PROXRECUR algorithm from [5, Sec. III-B.1] with algorithmic parameters, $\delta = 10^{-4}$, $\beta = 1$, and $L = 20$ to solve (7.14). For doing so, we generate $N = 1681$ samples from the initial distribution

$$\boldsymbol{\mu}_0 = \frac{1}{5}\mathcal{N}(\boldsymbol{m}_1, \boldsymbol{\Sigma}) + \frac{1}{5}\mathcal{N}(\boldsymbol{m}_2, \boldsymbol{\Sigma}) + \frac{1}{5}\mathcal{N}(\boldsymbol{m}_3, \boldsymbol{\Sigma}) + \frac{1}{5}\mathcal{N}(\boldsymbol{m}_4, \boldsymbol{\Sigma}) + \frac{1}{5}\mathcal{N}(\boldsymbol{m}_5, \boldsymbol{\Sigma})$$

with $\boldsymbol{m}_1 = (1, 1)^\top$, $\boldsymbol{m}_2 = (-1, -1)^\top$, $\boldsymbol{m}_3 = (1, -1)^\top$, $\boldsymbol{m}_4 = (-1, 1)^\top$, $\boldsymbol{m}_5 = (0, 0)^\top$, and $\boldsymbol{\Sigma} = 0.1\boldsymbol{I}_2$. We use $\mathcal{N}(\boldsymbol{m}, \boldsymbol{\Sigma})$ to denote a multivariate Gaussian distribution with mean vector $\boldsymbol{m}$ and covariance matrix $\boldsymbol{\Sigma}$.

The resulting evolution of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ are shown in Figure 7.5; it can be seen that after 5000 iterations of the outer layer ADMM (7.11), both $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, tend to the known stationary solution $\boldsymbol{\mu}_\infty$ given in Figure 7.4. We solve (7.34a) via the gradient descent method with a fixed step size as 0.001. The number of iterations of the inner layer ADMM given in (7.34) is 3. The simulation time was 99.89 sec. It is remarkable that all simulations are performed on a same MacBook Air with Intel Core i5 CPU, 1.1 GHz, and 8 GB RAM.

7.6.2   Nonlinear aggregation-drift-diffusion equation

Next, we consider an aggregation-drift equation of the form

$$\frac{\partial \mu}{\partial t} = \nabla \cdot (\mu \nabla U \circledast \mu) + \nabla \cdot (\mu \nabla V) + \beta^{-1} \Delta \mu^2 \tag{7.51}$$

(a) Contour plots of the transient solution of the joint measure μ_1

(b) Contour plots of the transient solution of the joint measure μ_2
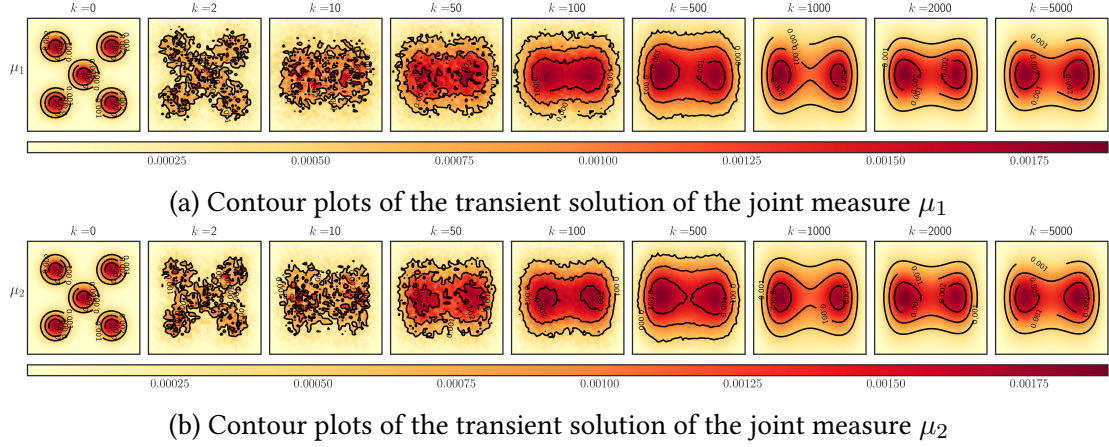
Figure 7.5: Evolution of the solution to the linear Fokker–Planck equation (7.49), with $V(x_1, x_2) = \frac{1}{4}\left(1 + x_1^4\right) + \frac{1}{2}\left(x_2^2 - x_1^2\right)$. The computational domain is $[-2, 2] \times [-2, 2]$. The color denotes the value of the plotted variable; see colorbar (dark red = high, light yellow = low).

where $U(x) = |x|^2/2 - \ln(|x|)$ and $V(x) = -\frac{1}{4}\ln(|x|)$. As given in [4, Sec. 4.3.2], in the limit $\beta^{-1} \downarrow 0$, the stationary measure, $\mu_\infty(\boldsymbol{x})$, is a torus with the inner and outer radius of $R_i = \sqrt{\frac{1}{4}}$, and $R_o = \sqrt{\frac{1}{4} + 1}$, respectively. To avoid evaluation of $U(x)$ and $V(x)$ at $x = 0$, we respectively set $U(0)$ and $V(0)$ to equal the average value of U and V on the cell of width $2h = 2 * 0.005$ centered at 0, i.e., $U(0) = \frac{1}{2h}\int_{-h}^{h} U(x)\mathrm{d}x$ and $V(0) = \frac{1}{2h}\int_{-h}^{h} V(x)\mathrm{d}x$.

In this example, we have three terms (interaction: $\nabla \cdot (\mu \nabla U \circledast \mu)$, drift: $\nabla \cdot (\mu \nabla V)$, and diffusion: $\beta^{-1}\Delta \mu^2$). We choose four different ways of splitting (7.51) and present the simulation results for each case of splitting.

In the first case, we choose $\nabla \cdot (\mu \nabla U \circledast \mu)$ and $\nabla \cdot (\mu \nabla V)$ with each other as the first term and $\beta^{-1}\Delta \mu^2$ as the second term

$$\partial_t \mu = \underbrace{\nabla \cdot (\mu \nabla V) + \beta^{-1}\Delta \mu^2}_{i=1} + \underbrace{\nabla \cdot (\mu \nabla U \circledast \mu)}_{i=2}.$$

In the second case, we choose $\nabla \cdot (\mu \nabla U \circledast \mu)$ and $\beta^{-1}\Delta \mu^2$ with each other as the first

term and $\nabla \cdot (\mu \nabla V)$ as the second term

$$\partial_t \mu = \underbrace{\nabla \cdot (\mu \nabla U \circledast \mu) + \beta^{-1} \Delta \mu^2}_{i=1} + \underbrace{\nabla \cdot (\mu \nabla V)}_{i=2}.$$

In the third case, we choose $\nabla \cdot (\mu \nabla U \circledast \mu)$ and $\nabla \cdot (\mu \nabla V)$ with each other as the first term and $\beta^{-1} \Delta \mu^2$ as the second term

$$\partial_t \mu = \underbrace{\nabla \cdot (\mu \nabla V) + \nabla \cdot (\mu \nabla U \circledast \mu)}_{i=1} + \underbrace{\beta^{-1} \Delta \mu^2}_{i=2}.$$

Finally in the fourth case, we choose $\nabla \cdot (\mu \nabla U \circledast \mu)$ as the first term, $\nabla \cdot (\mu \nabla V)$ as the second term and $\beta^{-1} \Delta \mu^2$ as the third term

$$\partial_t \mu = \underbrace{\nabla \cdot (\mu \nabla V)}_{i=1} + \underbrace{\nabla \cdot (\mu \nabla U \circledast \mu)}_{i=2} + \underbrace{\beta^{-1} \Delta \mu^2}_{i=3}.$$

The corresponding F_i and the Wasserstein distance between the solution of F_i and F_j i.e. $\boldsymbol{\mu}_j^k$ and $\boldsymbol{\mu}_j^k$ for each cases of splitting are given in Table 7.2. For each case, we artificially relabeled the argument of the functionals F_i as $\boldsymbol{\mu}_i$.

For the first case of splitting (the first row in Table.7.2), we define $G_1(\boldsymbol{\mu_1}) := F_1(\boldsymbol{\mu}_1) + \langle \boldsymbol{\nu}_1^k, \boldsymbol{\mu}_1 \rangle = \langle \boldsymbol{V}_k + \beta^{-1} \boldsymbol{\mu}_1 + \boldsymbol{\nu}_1^k, \boldsymbol{\mu}_1 \rangle$, and obtain the proximal update $\boldsymbol{\mu}_1^{k+1}$ by (7.14). Because F_2 is linear in $\boldsymbol{\mu}_2$, we use (7.15) with $\boldsymbol{\Phi}_2(\boldsymbol{\mu}_2) = \langle \boldsymbol{U}_k \boldsymbol{\mu}_2^k + \boldsymbol{\nu}_2^k, \boldsymbol{\mu}_2 \rangle$ to analytically compute the proximal update $\boldsymbol{\mu}_2^{k+1}$.

For the second case of splitting (the second row in Table.7.2), we define $G_1(\boldsymbol{\mu_1}) := F_1(\boldsymbol{\mu}_1) + \langle \boldsymbol{\nu}_1^k, \boldsymbol{\mu}_1 \rangle = \langle \boldsymbol{U}_k \boldsymbol{\mu}^k + \beta^{-1} \boldsymbol{\mu}_1 + \boldsymbol{\nu}_1^k, \boldsymbol{\mu}_1 \rangle$, and obtain the proximal update $\boldsymbol{\mu}_1^{k+1}$ by (7.14). Because F_2 is linear in $\boldsymbol{\mu}_2$, we use (7.15) with $\boldsymbol{\Phi}_2(\boldsymbol{\mu}_2) = \langle \boldsymbol{V}_k + \boldsymbol{\nu}_2^k, \boldsymbol{\mu}_2 \rangle$ to analytically compute the proximal update $\boldsymbol{\mu}_2^{k+1}$.

For the third case of splitting (the third row in Table.7.2), we define $G_2(\boldsymbol{\mu_2}) := F_2(\boldsymbol{\mu}_2) + \langle \boldsymbol{\nu}_2^k, \boldsymbol{\mu}_2 \rangle = \langle \beta^{-1} \boldsymbol{\mu}_2 + \boldsymbol{\nu}_2^k, \boldsymbol{\mu}_2 \rangle$, and obtain the proximal update $\boldsymbol{\mu}_2^{k+1}$ by (7.14). Because F_1 is linear in $\boldsymbol{\mu}_1$, we use (7.15) with $\boldsymbol{\Phi}_1(\boldsymbol{\mu}_1) = \langle \boldsymbol{V}_k + \boldsymbol{U}_k \boldsymbol{\mu}_1^k + \boldsymbol{\nu}_1^k, \boldsymbol{\mu}_1 \rangle$ to analytically

Splitting case	Functionals	Wasserstein distance
#1	$F_1(\boldsymbol{\mu}) = \langle \boldsymbol{V}_k + \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \rangle,$ $F_2(\boldsymbol{\mu}) = \langle \boldsymbol{U}_k\boldsymbol{\mu}^k, \boldsymbol{\mu} \rangle$	
#2	$F_1(\boldsymbol{\mu}) = \langle \boldsymbol{U}_k\boldsymbol{\mu}^k + \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \rangle,$ $F_2(\boldsymbol{\mu}) = \langle \boldsymbol{V}_k, \boldsymbol{\mu} \rangle$	
#3	$F_1(\boldsymbol{\mu}) = \langle \boldsymbol{U}_k\boldsymbol{\mu}^k + \boldsymbol{V}_k, \boldsymbol{\mu} \rangle,$ $F_2(\boldsymbol{\mu}) = \langle \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \rangle$	
#4	$F_1(\boldsymbol{\mu}) = \langle \boldsymbol{V}_k, \boldsymbol{\mu} \rangle,$ $F_2(\boldsymbol{\mu}) = \langle \boldsymbol{U}_k\boldsymbol{\mu}^k \rangle,$ $F_3(\boldsymbol{\mu}) = \langle \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \rangle$	

Table 7.2: Corresponding choice of functionals F_i, $i \in \{1,2,3\}$ for each cases of splitting (7.51) and the Wasserstein distance between the solution of F_i and F_j i.e. $\boldsymbol{\mu}_i^k$ and $\boldsymbol{\mu}_i^k$, $i,j \in \{1,2,3\}$, $i \neq j$ for 100 times executions of the code with the same initial samples. In the functional column, the drift potential vector $\boldsymbol{V}_k \in \mathbb{R}^N$ and the symmetric matrix $\boldsymbol{U}_k$ are given by $\boldsymbol{V}_k(i) := V\left(\boldsymbol{x}_k^i\right)$, $i = 1, \ldots, N$ and $\boldsymbol{U}_k(i,j) := U\left(\boldsymbol{x}_k^i - \boldsymbol{x}_k^j\right)$, $i,j = 1, \ldots, N$. We executed the code for each case of splitting 100 times and plot the averaged Wasserstein distance for each splitting case. The figures in the first three rows shows the averaged Wasserstein distance of the solution of each term after 10000 iterations for the cases that we split (7.51) to two terms and the shadow shows the variation range for each case of splitting. In the last row, each curve shows the averaged Wasserstein distance of the solution of each term after 10000 iterations for the cases that we split (7.51) to three terms. The shadow shows the variation range of each Wasserstein distances of $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, and $\boldsymbol{\mu}_3$. Because we start from the same initial distribution for $\boldsymbol{\mu}_i$, $i = \{1,2,3\}$, $W(\boldsymbol{\mu}_i^k, \boldsymbol{\mu}_j^k)$, $i,j \in \{1,2,3\}$, $i \neq j$ at $k = 0$ is zero.

compute the proximal update $\boldsymbol{\mu}_1^{k+1}$.

Finally, for the fourth case of splitting (the last row in Table.7.2), we define $G_3(\boldsymbol{\mu}_3) :=$

$F_3(\boldsymbol{\mu}_3) + \langle \boldsymbol{\nu}_3^k, \boldsymbol{\mu}_3 \rangle = \langle \beta^{-1}\boldsymbol{\mu}_3 + \boldsymbol{\nu}_3^k, \boldsymbol{\mu}_3 \rangle$, and obtain the proximal update $\boldsymbol{\mu}_3^{k+1}$ by (7.14).

Because F_2 is linear in $\boldsymbol{\mu}_2$, we use (7.15) with $\Phi_2(\boldsymbol{\mu}_2) = \langle \boldsymbol{U}_k\boldsymbol{\mu}_2^k + \boldsymbol{\nu}_2^k, \boldsymbol{\mu}_2 \rangle$ to analytically

compute the proximal update $\boldsymbol{\mu}_2^{k+1}$. Also, because F_1 is linear in $\boldsymbol{\mu}_1$, we use (7.15) with

$\Phi_1(\boldsymbol{\mu}_1) = \langle \boldsymbol{V}_k + \boldsymbol{U}_k\boldsymbol{\mu}_1^k + \boldsymbol{\nu}_1^k, \boldsymbol{\mu}_1 \rangle$ to analytically compute the proximal update $\boldsymbol{\mu}_1^{k+1}$.

In all four cases above, we use (7.19) to modify the PROXRECUR algorithm given in [5, Sec. III-B.1], and then solve (7.14) with the modified version of the PROXRECUR algorithm for the porous medium equation.

The resulting evolution of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ for the first case are shown in Figure 7.6; it can be seen that after 10000 iterations of the outer layer ADMM (7.11), both $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$, tend to the known stationary solution (torus with inner and outer radius of 0.5 and $\sqrt{\frac{5}{4}}$). The resulting evolution of $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ for the other cases behave almost the same. The averaged Wasserstein distance for each case of splitting is shown in Table. 7.2. All the simulation parameters and the initial distributions are the same as the previous example, only in the last case, $\alpha = 15$ and $\tau = 10$. The averaged simulation time after 100 execution for each case of splitting is 294.06 sec, 285.32 sec, 289.87 sec, and 108.99 sec, respectively.

(a) Contour plots of the transient solution of the joint measure μ_1

(b) Contour plots of the transient solution of the joint measure μ_2
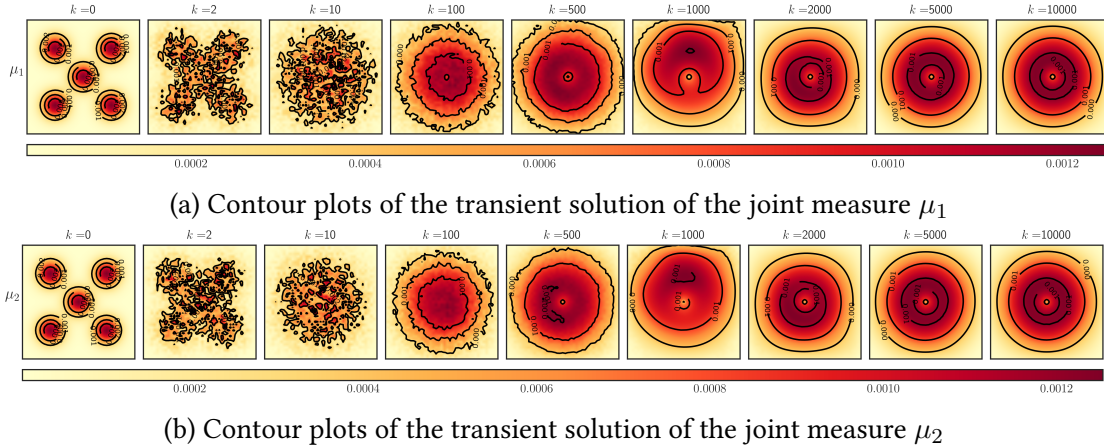
Figure 7.6: Evolution of the solution to the aggregation-drift equations (7.51), with $U(x) = |x|^2/2 - \ln(|x|)$ and $V(x) = -\frac{1}{4}\ln(|x|)$. The computational domain is $[-2, 2] \times [-2, 2]$. The color denotes the value of the plotted variable; see colorbar (dark red = high, light yellow = low).

Table 7.3 shows a comparison between how long it took for the centralized and proposed Wasserstein ADMM methods to run using the same simulation setup. It also displays the accuracy results by plotting the Wasserstein distances between the

centralized and Wasserstein ADMM iterations, based on the known stationary measure. These results provide two clear findings: Firstly, the proposed ADMM updates are faster (much faster when using three-way splitting) than the corresponding updates in the centralized approach. Secondly, as the iterations continue, the proposed algorithm outperforms the centralized method in terms of accuracy, as seen in the improvement of Wasserstein distance to the known stationary solution. In Table 7.4, we show how the final objective value changes for this case study based on different ADMM barrier parameter values (α). We maintained a constant inner ADMM iteration number of 3 throughout this analysis. We also performed simulations varying the inner ADMM iteration number while keeping $\alpha = 12$ fixed. The resulting fluctuations in the final objective value are detailed in Table 7.5.

7.6.3   Grouping of summand functionals

In (1.1), $F = F_1 + \cdots + F_n$, $n > 1$, where the summand functionals F_i, $i \in [n]$, are necessarily *distinct*. Suppose that we have n *indistinguishable* computing elements available for distributed computation. We can use any subset of them to implement our proposed algorithm depending on how we group the n distinct summand functionals. Clearly, the grouping $\{\{F_1, \cdots, F_n\}, \{0\}, \cdots, \{0\}\}$ corresponds to centralized computation. Then the number of ways to implement our distributed algorithm over n computing elements is

$$B_n - 1, \quad n = 2, 3, \cdots, \quad \text{where } B_n \text{ denotes the } n\text{th Bell number [231].} \qquad (7.52)$$

| Case	Functionals	Wasserstein distances
#1	$F_1(\boldsymbol{\mu}) = \left\langle \boldsymbol{V}_k + \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \right\rangle,$ $F_2(\boldsymbol{\mu}) = \left\langle \boldsymbol{U}_k \boldsymbol{\mu}^k, \boldsymbol{\mu} \right\rangle$	
#2	$F_1(\boldsymbol{\mu}) = \langle \boldsymbol{U}_k \boldsymbol{\mu}^k + \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \rangle,$ $F_2(\boldsymbol{\mu}) = \langle \boldsymbol{V}_k, \boldsymbol{\mu} \rangle$	
#3	$F_1(\boldsymbol{\mu}) = \langle \boldsymbol{U}_k \boldsymbol{\mu}^k + \boldsymbol{V}_k, \boldsymbol{\mu} \rangle,$ $F_2(\boldsymbol{\mu}) = \left\langle \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \right\rangle$	
#4	$F_1(\boldsymbol{\mu}) = \left\langle \boldsymbol{V}_k, \boldsymbol{\mu} \right\rangle,$ $F_2(\boldsymbol{\mu}) = \left\langle \boldsymbol{U}_k \boldsymbol{\mu}^k, \boldsymbol{\mu} \right\rangle,$ $F_3(\boldsymbol{\mu}) = \left\langle \beta^{-1}\boldsymbol{\mu}, \boldsymbol{\mu} \right\rangle$	

Table 7.3: For the aggregation-drift-diffusion nonlinear PDE case study in Section 7.6.2, comparison of the Wasserstein distances to the known stationary solution $\boldsymbol{\mu}_\infty$, from the iterates of the centralized ($\boldsymbol{\mu}^k_{\text{centralized}}$), and from the iterates of the proposed Wasserstein ADMM algorithm $\boldsymbol{\mu}^k_i$, $i \in 1, 2, 3$. The known $\boldsymbol{\mu}_\infty$ here is a uniform measure over an annulus with the inner radius $R_i = 1/2$ and the outer radius $R_o = \sqrt{5}/2$ [4, Sec. 4.3.2]. All Wasserstein distances are computed by solving the corresponding Kantorovich LPs as in Table 7.2. All simulations are done with the same setup as in Table 7.2. Material H, Table 2, i.e., with the same uniform grid over $[-2, 2]^2$ with 1681 samples, $\beta^{-1} = 0.0520$ as in [4, Sec. 4.3.2], and the same μ_0, U, V and other parameters reported in Section 7.6.2. For centralized computation, we used the proximal recursion algorithm in [5]. The figures in the last column show that after 10000 iterations, the Wasserstein distances between μ_i and μ_∞ in all cases are smaller than the corresponding Wasserstein distance between the centralized solution and μ_∞. The average runtime (averaged over 100 executions of the same code as in Table 2, Supp. Material) from the proposed Wasserstein ADMM algorithm in all cases remain below 300 sec, and especially it is recorded at 108.99 sec in case #4, significantly below the total runtime of the centralized variant (310.21 sec).

α	11	11.5	12	12.5	13	13.5	14	14.5	15
F^{10000}, case #1	10.9058	10.9224	10.8978	10.9064	10.8922	10.9203	10.9124	10.9203	10.9139
F^{10000}, case #2	11.0624	11.0598	11.0618	11.0578	11.0694	11.0692	11.0591	11.0570	11.0561
F^{10000}, case #3	11.0296	11.0325	11.0275	11.0312	11.0338	11.0301	11.0395	11.0351	11.0305
F^{10000}, case #4	16.5087	16.5012	16.5106	16.5080	16.5049	16.5029	16.5030	16.5018	16.5057

Table 7.4: Value of the objective $F^{10000} := \langle \boldsymbol{V}_k + \boldsymbol{U}_k \boldsymbol{\mu}^k + \beta^{-1} \boldsymbol{\mu}, \boldsymbol{\mu} \rangle|_{k=10000}$ at the final consensus iterate $\boldsymbol{\mu} \equiv \boldsymbol{\mu}^{10000}$ for cases in Figure 7.6 w.r.t. different values of ADMM barrier parameter $\alpha \in [11, 15]$.

Inner layer ADMM iter. #	3	4	5	6	7	8	9	10
F^{10000}, case #1	10.9263	10.8981	10.9165	10.8997	10.9124	10.9157	10.8813	10.9009
F^{10000}, case #2	11.0638	11.0546	11.0643	11.0625	11.0632	11.0583	11.0701	11.0678
F^{10000}, case #3	11.0368	11.0457	11.0374	11.0381	11.0363	11.0359	11.0318	11.0322
F^{10000}, case #4	16.5072	16.5023	16.5046	16.5001	16.5123	16.5039	16.5045	16.5034

Table 7.5: Value of the objective $F^{10000} := \langle \boldsymbol{V}_k + \boldsymbol{U}_k \boldsymbol{\mu}^k + \beta^{-1} \boldsymbol{\mu}, \boldsymbol{\mu} \rangle|_{k=10000}$ at the final consensus iterate $\boldsymbol{\mu} \equiv \boldsymbol{\mu}^{10000}$ for cases in Figure 7.6 w.r.t. different number for the Inner layer ADMM iteration.

The minus one in (7.52) discounts the centralized computation. The first few Bell numbers are $B_2 = 2, B_3 = 5, B_4 = 15, B_5 = 52, B_6 = 203, \cdots$.

For our first experiment in Section 7.6, $n = 2$ and there is $B_2 - 1 = 1$ way to implement the proposed algorithm. For our second experiment in Section 7.6, $n = 3$ and there are $B_3 - 1 = 4$ ways to implement the proposed algorithm.

More generally, if we have n distinct summand functionals with $r \leq n$ indistinguishable computing elements available, then the number of ways to implement our distributed algorithm is

$$\sum_{k=1}^{r} \left\{ \binom{n}{k} \right\}, \text{where } \left\{ \binom{n}{k} \right\} \text{ denote the Stirling numbers of the second kind [232, p. 244].}$$

(7.53)

For $r = n$, (7.53) reduces to (7.52).

7.6.4   Wasserstein barycenter

The Wasserstein Barycenter as described in [54] and [233], for a set of discrete distributions $\boldsymbol{\xi}_i \in \mathcal{P}_2(\mathcal{X})$, for $i = 1, \ldots, n$; is characterized by the solution to this optimization problem:

$$\underset{\boldsymbol{\mu} \in \mathcal{P}_2(\mathcal{X})}{\arg\inf} \sum_{i=1}^{n} w_i W^2 (\boldsymbol{\mu}, \boldsymbol{\xi}_i) \tag{7.54}$$

where $0 < w_i < 1$ and $\sum\limits_{i=1}^{n} w_i = 1$.

Optimization problem (7.54), can be written in the consensus form as (7.3) as below:

$$\underset{(\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_n, \boldsymbol{\zeta}) \in \mathcal{P}_2^{n+1}(\mathcal{X})}{\arg\inf} F_1(\boldsymbol{\mu}_1) + F_2(\boldsymbol{\mu}_2) + \cdots + F_n(\boldsymbol{\mu}_n) \tag{7.55a}$$

$$\text{subject to} \qquad \boldsymbol{\mu}_i = \boldsymbol{\zeta} \quad \text{for all } i \in [n], \tag{7.55b}$$

where $F_i(\boldsymbol{\mu}_i) = w_i W^2 (\boldsymbol{\mu}_i, \boldsymbol{\xi}_i)$.

In this example, we consider three distinct distributions represented by $\boldsymbol{\xi}_i$, $i = 1, 2, 3$, each characterized by unique geometric shapes: a donut, a heart, and an X, as depicted in Figures 7.7b, 7.7a, and 7.7c, respectively. Our goal is to compute the Wasserstein barycenter for these distributions, leveraging our novel distributed algorithm.

In this series of experiments, we explore the Wasserstein barycenter computations through four distinct scenarios.

- Between heart and donut shapes, with $F_1(\boldsymbol{\mu}_1) = \frac{1}{2} W^2 (\boldsymbol{\mu}_1, \boldsymbol{\xi}_1)$ and $F_2(\boldsymbol{\mu}_2) = \frac{1}{2} W^2 (\boldsymbol{\mu}_2, \boldsymbol{\xi}_2)$.

- Between heart and X shapes, with $F_1(\boldsymbol{\mu}_1) = \frac{1}{2} W^2 (\boldsymbol{\mu}_1, \boldsymbol{\xi}_1)$ and $F_3(\boldsymbol{\mu}_3) = \frac{1}{2} W^2 ($

(a) Heart-shaped distribution (ξ_1)

(b) Donut-shaped distribution (ξ_2)

(c) X-shaped distribution (ξ_3)

Figure 7.7: Visual representation of distinct distributions for ξ_i, $i = 1, 2, 3$, illustrating heart, donut, and X shapes, respectively, in the context of the Wasserstein Barycenter problem ((7.54)).

μ_3, ξ_3).

- Between donut and X shapes, with $F_2(\mu_2) = \frac{1}{2}W^2(\mu_2, \xi_2)$ and $F_3(\mu_3) = \frac{1}{2}W^2($

  μ_3, ξ_3).

- Among all three shapes: heart, donut, and X, with $F_1(\mu_1) = \frac{1}{3}W^2(\mu_1, \xi_1)$,

  $F_2(\mu_2) = \frac{1}{3}W^2(\mu_2, \xi_2)$, and $F_3(\mu_3) = \frac{1}{3}W^2(\mu_3, \xi_3)$.

The parameter n in (7.55) that shows the number of distributed processors is set $n = 2$ for pairwise comparisons and $n = 3$ for the comprehensive analysis involving all three shapes. For all F_i, we use (7.21) with $\Phi_i(\mu_i) := w_i W^2(\mu_i^k, \xi_i) + \langle \nu_i^k, \mu_i^k \rangle$ to compute the proximal update μ_i^{k+1}. We use (7.22) to modify the PROXRECUR algorithm given in [5, Sec. III-B.1], to solve (7.20) with the modified version of the PROXRECUR algorithm for this case study.

Figure 7.8 showcases the iterative evolution of Wasserstein barycenters as computed by three distributed processors, each tasked with a unique geometric distribution: Processor #1 with a donut shape, Processor #2 with a heart shape, and Proces-

sor #3 with an X shape. This figure offers a visual timeline of the algorithm's performance across several iterations, highlighting the dynamic process of shape adaptation and convergence.

Distinctively, each processor's primary assignment is denoted in bold blue, while contributions received from other processors are depicted in light blue. This color differentiation aids in visualizing the flow and integration of information across the distributed network. The triangle's center serves as a focal point for the amalgamation of all three shapes. Moreover, the Wasserstein barycenters resulting from pairwise combinations of the shapes are represented along the triangle's edges. For example, the edge connecting the heart and donut shapes illustrates the intermediate barycenter derived from these two distributions. As the iterations progress, we observe a gradual blending and convergence of the distinct shapes towards a cohesive barycenter. The final iteration reveals a notable harmonization of the shapes, encapsulating the essence of each original distribution while presenting a new, integrated form. This visual documentation eloquently captures the essence of distributed computation in Wasserstein barycenters.

(a) first iteration

(b) Iteration #10

(c) Iteration #2000

(d) Iteration #3500
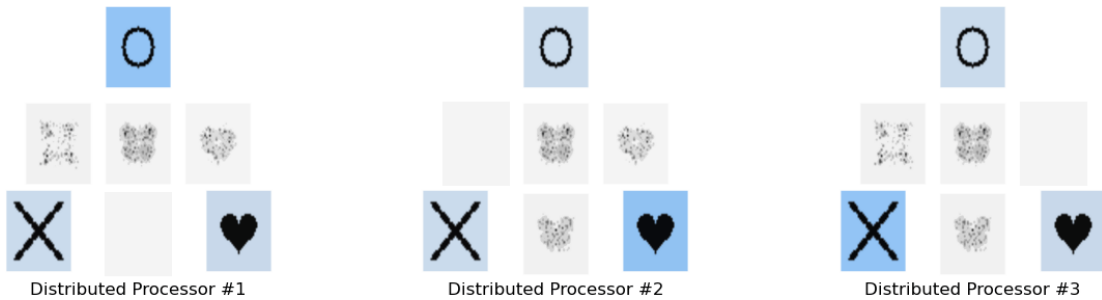
(e) Iteration #9600
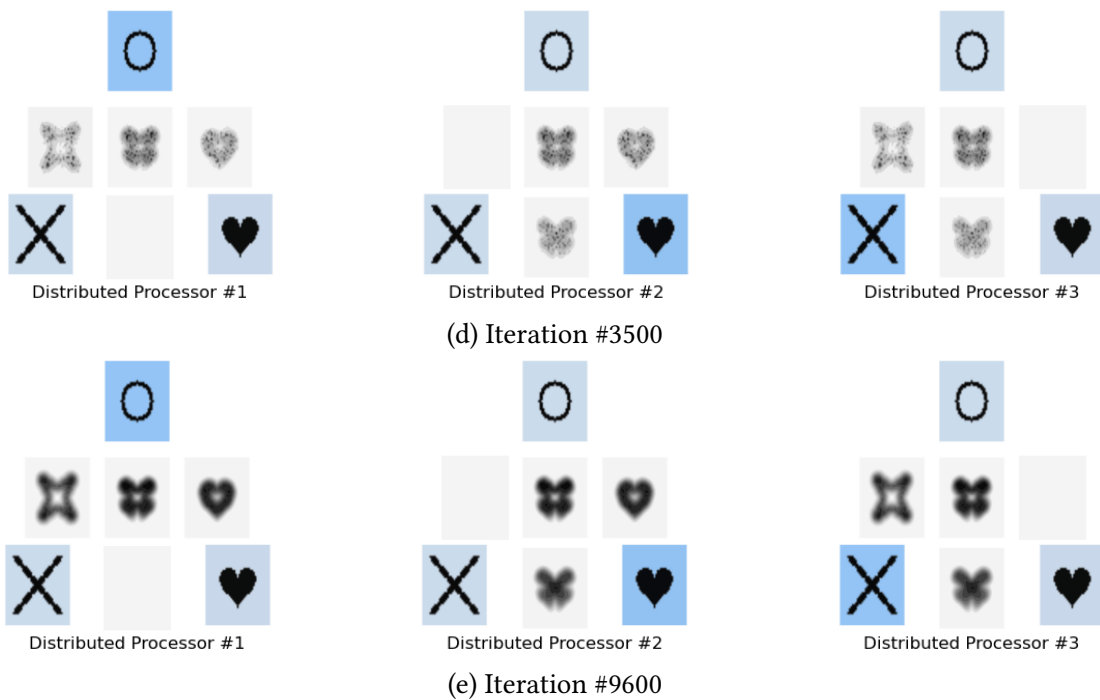
Figure 7.8: Dynamic evolution of distributed processors in computing Wasserstein barycenters: A series of snapshots illustrating the iterative refinement process across distributed processors, each designated with a specific shape (donut, heart, X). These images trace the convergence journey from the initial assignment through multiple iterations (1, 10, 2000, 3500, 9600), showcasing the collaborative exchange and progressive adaptation of shapes towards a unified solution.

8 | Summary and Future Work

Throughout this dissertation, we explored convex optimization in the space of probability measures, showcasing its ubiquity in stochastic control, stochastic modeling, and stochastic learning. The methods developed here hold promise for a wide range of applications, from enhancing machine learning models, simulating mean-field dynamics, optimizing Wasserstein barycenters, and controlling probability distributions.

Our investigation began with solving the generalized Schrödinger bridge problems that are nonlinear in state, and possibly non-affine in control. The effectiveness of our proposed algorithms were demonstrated in both model-based and data-driven settings.

We then introduced a controlled mean field model for the dynamics of chiplets in micro-assembly applications, offering a detailed analysis of their interactions and collective behavior. Our results establish consistency of the model in a limiting sense and demonstrate that the resulting PDF evolution can be seen as an infinite dimensional gradient descent of a Lyapunov-like energy functional w.r.t. the Wasserstein metric. Our future work for this part will investigate the synthesis of optimal control of the chiplet joint PDF w.r.t. suitable performance objective that allows steering an initial joint PDF to a desired terminal joint PDF. We note that the feedback synthesis for density steering subject to a controlled mean field nonlocal PDE is relatively less explored but has started appearing in recent works; see e.g., [234–236].

We then turned to stochastic learning. We designed a proximal mean field learning algorithm to train an over-parameterized subsumes shallow neural network in the over-parameterized regime. The proposed algorithm is meshless and non-parametric. It implements the Wasserstein proximal recursions realizing the gradient descent of

entropic-regularized risk. Numerical case studies in binary classification, multi-class classification, and learning 1D sinusoid demonstrate that the ideas of mean field learning can be attractive as a computational framework beyond purely theoretical interests.

Existing efforts to generalize the theoretical results for the mean field learning as in this work, from single to multi-hidden layer networks, have been pursued in two different limiting sense. One line of investigations [18, 237, 238] take the infinite width limit one hidden layer at a time while holding the (variable) widths of other hidden layers fixed. More precisely, if the ith hidden layer has N_i neurons, then the limit is taken by first normalizing that layer's output by $N_i^{\gamma_i}$ for some fixed $\gamma_i \in [1/2, 1]$ and then letting $N_i \to \infty$ for an index i while holding other N_j's fixed and finite, $j \neq i$. A different line of investigations [239, 240] consider the limit where the widths of all hidden layers simultaneously go to infinity. In this setting, the population distribution over the joint (across hidden layers) parameter space is shown to evolve under SGD as per a McKean-Vlasov type nonlinear IVP; see [239, Def. 4.4 and Sec. 5]. We anticipate that the proximal recursions proposed herein can be extended to this setting by effectively lifting the Wasserstein gradient flow to the space of measure-valued paths. Though not quite the same, but this is similar in spirit to how classical bi-mariginal Schrödinger bridge problems [32, 241] have been generalized to multi-marginal settings over the path space and have led to significant algorithmic advances in recent years [242–244].

Finally, we proposed a novel computational framework for distributed computation in solving measure-valued optimization problems. Our findings provide new insights in generalizing the well-known finite dimensional Euclidean ADMM to its Wasserstein and Sinkhorn counterparts. The proposed framework leverages existing

proximal and Jordan-Kinderlehrer-Otto (JKO) schemes, and open up the possibility of designing measure-valued operator splitting algorithms. While we provided convergence guarantee for the proposed inner layer ADMM, an important undertaking not pursued here is the convergence guarantee for the overall scheme. This is also the topic of our future work.

A | Proofs for Chapter 4

Proof of Theorem 4.1

Proof. For $\mathcal{X} \subseteq \mathbb{R}^n$, let $r_0 \in \overline{\mathbb{R}} := \mathbb{R} \cup \{-\infty, +\infty\}$ (two point compactification of the real line $\mathbb{R}$) be defined as $r_0 := \sup_{\boldsymbol{x} \in \mathcal{X}} \|\boldsymbol{x}\|_2$.

We write the Lagrangian (4.4) as the sum of three state-time integrals:

$$\int_0^T \int_{\mathcal{X}} \frac{1}{2} \|\boldsymbol{u}\|_2^2 \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x} \, \mathrm{d}t + \int_0^T \int_{\mathcal{X}} \psi \frac{\partial \rho^{\boldsymbol{u}}}{\partial t} \mathrm{d}\boldsymbol{x} \, \mathrm{d}t$$
$$+ \int_0^T \int_{\mathcal{X}} \left(\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x}} \cdot (\rho^{\boldsymbol{u}} \boldsymbol{f}) - \langle \boldsymbol{G}, \mathbf{Hess}\,(\rho^{\boldsymbol{u}}) \rangle \right) \psi \mathrm{d}\boldsymbol{x} \, \mathrm{d}t. \tag{A.1}$$

In above, for the second summand, we invoke the Fubini–Tonelli theorem to switch the order of integration and perform integration by parts w.r.t. t. This gives

$$\int_0^T \int_{\mathcal{X}} \psi \frac{\partial \rho^{\boldsymbol{u}}}{\partial t} \mathrm{d}\boldsymbol{x} \, \mathrm{d}t$$
$$= \int_{\mathcal{X}} \left(\int_0^T \psi \frac{\partial \rho^{\boldsymbol{u}}}{\partial t} \mathrm{d}t \right) \mathrm{d}\boldsymbol{x}$$
$$= \int_{\mathcal{X}} \left([\psi \rho^{\boldsymbol{u}}]_{t=0}^{t=T} - \int_0^T \frac{\partial \psi}{\partial t} \rho^{\boldsymbol{u}} \mathrm{d}t \right) \mathrm{d}\boldsymbol{x}$$
$$= \underbrace{\int_{\mathcal{X}} (\psi(T, \boldsymbol{x}) \rho_T(\boldsymbol{x}) - \psi(0, \boldsymbol{x}) \rho_0(\boldsymbol{x})) \, \mathrm{d}\boldsymbol{x}}_{\text{constant over } \mathcal{P}_{0T}(\mathcal{X}) \times \mathcal{U}} - \int_0^T \int_{\mathcal{X}} \frac{\partial \psi}{\partial t} \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x} \mathrm{d}t. \tag{A.2}$$

For the third summand in (A.1), we perform integration by parts w.r.t. $\boldsymbol{x}$, to obtain

$$\int_0^T \int_{\mathcal{X}} \left(\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x}} \cdot (\rho^{\boldsymbol{u}} \boldsymbol{f}) - \langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle \right) \psi \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t$$
$$= \int_0^T \left(\int_{\mathcal{X}} \left(\frac{\partial \rho^{\boldsymbol{u}}}{\partial t} + \nabla_{\boldsymbol{x}} \cdot (\rho^{\boldsymbol{u}} \boldsymbol{f}) \right) \psi \, \mathrm{d}\boldsymbol{x} - \langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle \psi \mathrm{d}\boldsymbol{x} \right) \mathrm{d}t$$
$$= \int_0^T \left(\lim_{\|\boldsymbol{x}\|_2 \to r_0} \left[\psi(t, \boldsymbol{x}) \int \frac{\partial \rho^{\boldsymbol{u}}}{\partial t} \mathrm{d}\boldsymbol{x} \right] \right) - \int_0^T \int_{\mathcal{X}} \langle \nabla_{\boldsymbol{x}} \psi, \boldsymbol{f} \rangle \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x} \mathrm{d}t$$
$$- \int_0^T \int_{\mathcal{X}} \langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle \psi \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t$$
$$= - \int_0^T \int_{\mathcal{X}} \langle \nabla_{\boldsymbol{x}} \psi, \boldsymbol{f} \rangle \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x} \mathrm{d}t - \int_0^T \int_{\mathcal{X}} \langle \mathbf{Hess}, \boldsymbol{G} \rho^{\boldsymbol{u}} \rangle \psi \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t \tag{A.3}$$

where we assumed that the limits at $\|\boldsymbol{x}\|_2 \to r_0$ are zero.

Now consider the second summand in (A.3), and perform two-fold integration by parts w.r.t. $\boldsymbol{x}$ as

$$
\int_{\mathcal{X}} \langle \mathbf{Hess}, \boldsymbol{G}\rho^{\boldsymbol{u}} \rangle \psi \, \mathrm{d}\boldsymbol{x}
$$

$$
= \int_{\mathcal{X}} \sum_{i,j} \frac{\partial^2}{\partial x_i \partial x_j} \left(G_{ij}\rho^{\boldsymbol{u}} \right) \psi \, \mathrm{d}\boldsymbol{x}
$$

$$
= \sum_{i,j} \int_{\mathcal{X}} \frac{\partial^2}{\partial x_i \partial x_j} \left(G_{ij}\rho^{\boldsymbol{u}} \right) \psi \, \mathrm{d}\boldsymbol{x}
$$

$$
= -\sum_{i,j} \int_{\mathcal{X}} \frac{\partial (G_{ij}\rho^{\boldsymbol{u}})}{\partial x_j} \frac{\partial \psi}{\partial x_i} \, \mathrm{d}\boldsymbol{x}
$$

$$
= \sum_{i,j} \int_{\mathcal{X}} (G_{ij}\rho^{\boldsymbol{u}}) \frac{\partial^2 \psi}{\partial x_j \partial x_i} \, \mathrm{d}\boldsymbol{x}
$$

$$
= \int_{\mathcal{X}} \sum_{i,j} (G_{ij}\rho^{\boldsymbol{u}}) \frac{\partial^2 \psi}{\partial x_j \partial x_i} \, \mathrm{d}\boldsymbol{x}
$$

$$
= \int_{\mathcal{X}} \langle \boldsymbol{G}, \mathbf{Hess}(\psi) \rangle \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x}, \tag{A.4}
$$

which helps rewrite (A.3) as

$$
-\int_0^T \int_{\mathcal{X}} \langle \nabla_{\boldsymbol{x}}\psi, \boldsymbol{f} \rangle \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x} \mathrm{d}t - \int_0^T \int_{\mathcal{X}} \langle \boldsymbol{G}, \mathbf{Hess}(\psi) \rangle \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x} \, \mathrm{d}t. \tag{A.5}
$$

Combining (A.2), (A.3), (A.5), and dropping the constant term, the Lagrangian (A.1) simplifies to

$$
\int_0^T \int_{\mathcal{X}} \left(\frac{1}{2}\|\boldsymbol{u}\|_2^2 - \frac{\partial \psi}{\partial t} - \langle \nabla_{\boldsymbol{x}}\psi, \boldsymbol{f} \rangle - \langle \boldsymbol{G}, \mathbf{Hess}(\psi) \rangle \right) \rho^{\boldsymbol{u}} \mathrm{d}\boldsymbol{x} \, \mathrm{d}t. \tag{A.6}
$$

Minimizing (A.6) w.r.t. $\boldsymbol{u}$ for a fixed PDF $\rho^{\boldsymbol{u}}$ yields (4.6c).

We then substitute (4.6c) back in (A.6), and equate the resulting expression to zero, to arrive at the dynamic programming equation

$$
\int_0^T \int_{\mathcal{X}} \left(\frac{1}{2}\|\nabla_{\boldsymbol{u}_{\mathrm{opt}}} \left(\langle \nabla_{\boldsymbol{x}}\psi, \boldsymbol{f} \rangle + \langle \boldsymbol{G}, \mathbf{Hess}(\psi) \rangle \right) \|_2^2 \right.
$$

$$
\left. - \frac{\partial \psi}{\partial t} - \langle \nabla_{\boldsymbol{x}}\psi, \boldsymbol{f} \rangle - \langle \boldsymbol{G}, \mathbf{Hess}(\psi) \rangle \right) \rho^{\boldsymbol{u}}(t, \boldsymbol{x}) \mathrm{d}\boldsymbol{x} \, \mathrm{d}t = 0. \tag{A.7}
$$

Since (A.7) should be satisfied for arbitrary $\rho^{\boldsymbol{u}}$, we get

$$\frac{\partial \psi}{\partial t} = \frac{1}{2} \| \nabla_{\boldsymbol{u}} \left(\langle \nabla_{\boldsymbol{x}} \psi, \boldsymbol{f} \rangle + \langle \boldsymbol{G}, \mathrm{Hess}(\psi) \rangle \right) \|_2^2 - \langle \nabla_{\boldsymbol{x}} \psi, \boldsymbol{f} \rangle$$
$$- \langle \boldsymbol{G}, \mathrm{Hess}(\psi) \rangle$$

which is the HJB PDE (4.6a). The FPK PDE (4.6b) and the boundary conditions (4.7) follow from the primal feasibility conditions (4.3b) and (4.3c), respectively. □

Proof of Theorem 4.2

Proof. Consider the Lagrangian associated with (4.11):

$$\mathcal{L}(\rho^\pi, \pi, \psi) := \int_0^T \int_{\mathbb{R}} \left\{ \frac{1}{2} \pi^2 \rho^\pi + \psi \times \left(\frac{\partial \rho^\pi}{\partial t} + \frac{\partial}{\partial \langle C_6 \rangle} (D_1 \rho^\pi) - \frac{\partial^2}{\partial \langle C_6 \rangle^2} (D_2 \rho^\pi) \right) \right\} \mathrm{d}x \, \mathrm{d}t$$
(A.8)

where $\psi(\langle C_6 \rangle, t)$ is a $C^1(\mathbb{R}; \mathbb{R}_{\geq 0})$ Lagrange multiplier.

Performing integration by parts, the Lagrangian (A.8) can be written as

$$\mathcal{L}(\rho^\pi, \pi, \psi) = \int_0^T \int_{\mathbb{R}} \left(\frac{1}{2} \pi^2 - \frac{\partial \psi}{\partial t} - D_1 \frac{\partial \psi}{\partial \langle C_6 \rangle} - D_2 \frac{\partial^2}{\partial \langle C_6 \rangle^2} \psi \right) \rho^\pi \, \mathrm{d}x \, \mathrm{d}t.$$
(A.9)

For ρ^π fixed, pointwise minimization of (A.9) with respect to π yields (4.12c).

By substituting (4.12c) in (A.9) and equating the resulting expression to zero, we get the dynamic programming equation

$$\int_0^T \int_{\mathbb{R}} \left(\frac{1}{2} \left(\frac{\partial \psi}{\partial \langle C_6 \rangle} \frac{\partial D_1}{\partial \pi} + \frac{\partial^2 \psi}{\partial \langle C_6 \rangle^2} \frac{\partial D_2}{\partial \pi} \right)^2 - \frac{\partial \psi}{\partial t} \right.$$
$$\left. - D_1 \frac{\partial \psi}{\partial \langle C_6 \rangle} - D_2 \frac{\partial^2 \psi}{\partial \langle C_6 \rangle^2} \right) \rho^\pi (\langle C_6 \rangle, t) \mathrm{d}\langle C_6 \rangle \, \mathrm{d}t = 0.$$
(A.10)

For (A.10) to be satisfied for arbitrary ρ^π, we must have

$$\frac{\partial \psi}{\partial t} = \frac{1}{2} \left(\frac{\partial \psi}{\partial \langle C_6 \rangle} \frac{\partial D_1}{\partial \pi} + \frac{\partial^2 \psi}{\partial \langle C_6 \rangle^2} \frac{\partial D_2}{\partial \pi} \right)^2 - D_1 \frac{\partial \psi}{\partial \langle C_6 \rangle} - D_2 \frac{\partial^2 \psi}{\partial \langle C_6 \rangle^2},$$
(A.11)

which upon using (4.12c), gives the HJB PDE (4.12a). The associated FPK PDE (4.12b)

and the boundary conditions (4.13) follow from (4.11b) and (4.11c), respectively. $\square$

B | Proofs for Chapter 5

Proof of Theorem 5.1

Proof. To describe the dynamics of ρ^n as $n \to \infty$, we start with investigating the time evolution of the quantity

$$\langle \varphi, \rho^n \rangle := \frac{1}{n} \sum_{i=1}^{n} \varphi(\boldsymbol{x}_i) \tag{B.1}$$

for any compactly supported test function $\varphi \in C_b^2(\mathbb{R}^2)$.

Using Ito's rule, we have

$$d\varphi(\boldsymbol{x}_i) = L_{\rho^n}\varphi(\boldsymbol{x}_i) dt + \nabla\varphi^\top(\boldsymbol{x}_i) \sqrt{2\beta^{-1}} d\boldsymbol{w}_i \tag{B.2}$$

wherein the infinitesimal generator

$$L_\rho \varphi(\boldsymbol{x}) := \langle \boldsymbol{f}^u(\boldsymbol{x}, t, u, \rho), \nabla_{\boldsymbol{x}}\varphi(x) \rangle + \beta^{-1}\Delta\varphi. \tag{B.3}$$

Thus,

$$\begin{aligned} d\langle \varphi, \rho^n \rangle &= \frac{1}{n} \sum_{i=1}^{n} d\varphi(\boldsymbol{x}_i) \\ &= \langle L_{\rho^n}\varphi, \rho^n \rangle dt + \frac{1}{n} \sum_{i=1}^{n} \sqrt{2\beta^{-1}} \nabla\varphi^\top(\boldsymbol{x}_i) d\boldsymbol{w}_i \\ &:= \langle L_{\rho^n}\varphi, \rho^n \rangle dt + dM_t^n \end{aligned} \tag{B.4}$$

where M_t^n is a local martingale.

Because $\varphi \in C_b^2(\mathbb{R}^2)$, we have $\left| \sqrt{2\beta^{-1}} \nabla\varphi^\top(\boldsymbol{x}_i) \right| \le C$ uniformly for some $C > 0$.

Notice that the quadratic variation of the noise term in (B.4) is

$$[M_t^n] := \frac{1}{n^2} \sum_{i=1}^{n} \int_0^t \left| \sqrt{2\beta^{-1}} \nabla\varphi^\top(\boldsymbol{x}_i(s)) \right|^2 ds \le \frac{tC^2}{n},$$

and using Doob's martingale inequality [245, Ch. 14.11],

$$\mathbb{E}\left(\sup_{t \le T} M_t^n\right)^2 \le \mathbb{E}\left(\sup_{t \le T}\left(M_t^n\right)^2\right) \le 4\mathbb{E}\left(\left(M_t^n\right)^2\right)$$

$$\le 4\mathbb{E}\left(\left[M_t^n\right]\right) \le \frac{4tC^2}{n}.$$

Hence in the limit $n \to \infty$, the noise term in (B.4) vanishes, resulting in a deterministic evolution equation.

For any $t > 0$, we take $\{\rho^n\}_{n=1}^{\infty}$ to be the (random) elements of $\Omega = C([0, \infty), \mathcal{P}(\mathbb{R}^2))$, the set of continuous functions from $[0, \infty)$ into $\mathcal{P}(\mathbb{R}^2)$ endowed with the topology of weak convergence. Following the argument of Oelschläger [246, Proposition 3.1], the sequence $\mathbb{P}_n$ of joint PDFs on Ω induced by the processes $\{\rho^n\}_{n=1}^{\infty}$, is relatively compact in $\mathcal{P}(\Omega)$, which is the space of probability measures on Ω. Oelschläger's proof makes use of the Prohorov's theorem [247, Ch. 5]. The relative compactness implies that the sequence $\mathbb{P}_n$ weakly converges (along a subsequence) to some $\mathbb{P}$, where $\mathbb{P}$ is the joint PDFs induced by the limiting process ρ. By Skorohod representation theorem [247, Theorem 6.7], the sequence $\{\rho^n\}_{n=1}^{\infty}$ converges $\mathbb{P}$-almost surely to ρ. Since the martingale term in (B.4) vanishes as $n \to \infty$, we obtain

$$\mathrm{d}\langle \varphi, \rho \rangle = \langle L_\rho \varphi, \rho \rangle \, \mathrm{d}t = \langle \varphi, L_\rho^* \rho \rangle \, \mathrm{d}t \tag{B.5}$$

where L^* is the adjoint (see e.g., [248, Ch. 2.3, 2.5], [249, p. 278]) of the generator L given by (B.3), and is defined as

$$L_m^* \rho(x, t) := -\nabla \cdot \left(\rho \boldsymbol{f}^u(\boldsymbol{x}, t, u, m)\right) + \beta^{-1} \Delta \rho$$

$$= \nabla \cdot \left(\rho \nabla \left(m * \phi^u + \beta^{-1}(1 + \log \rho)\right)\right)$$

where $m \in \mathcal{P}(\mathbb{R}^2)$. For any test function $\varphi \in C_b^2(\mathbb{R}^2)$, (B.5) is valid almost everywhere, and therefore, ρ is almost surely a weak solution to the nonlinear Fokker-

Planck-Kolmogorov PDE initial value problem (5.7)-(5.8). □

Proof of Theorem 5.2

Proof. (i) We start by noticing that the functional derivative

$$\frac{\delta \Phi}{\delta \rho} = \rho * \phi^u + \beta^{-1}(1 + \log \rho). \tag{B.6}$$

Next, we rewrite (5.7) as

$$\frac{\partial \rho}{\partial t} = \nabla \cdot \left(\rho \nabla \frac{\delta \Phi}{\delta \rho} \right), \tag{B.7}$$

which by definition (1.12), yields (5.12).

(ii) To show that Φ is decreasing along the flow generated by (5.7), we find

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t} \Phi &= \int \frac{\delta \Phi}{\delta \rho} \frac{\partial \rho}{\partial t} \mathrm{d}\boldsymbol{x} \\
&\overset{(B.7)}{=} \int \frac{\delta \Phi}{\delta \rho} \nabla \cdot \left(\rho \nabla \frac{\delta \Phi}{\delta \rho} \right) \mathrm{d}\boldsymbol{x} \\
&= -\int \left\langle \nabla \frac{\delta \Phi}{\delta \rho}, \rho \nabla \frac{\delta \Phi}{\delta \rho} \right\rangle \mathrm{d}\boldsymbol{x} \\
&= -\int \left\langle \nabla \frac{\delta \Phi}{\delta \rho}, \nabla \frac{\delta \Phi}{\delta \rho} \right\rangle \rho \mathrm{d}\boldsymbol{x} \\
&= -\mathbb{E}_\rho \left[\left\| \nabla \frac{\delta \Phi}{\delta \rho} \right\|_2^2 \right] \leq 0.
\end{aligned}
\tag{B.8}
$$

In order to get from the second line to the third line of (B.8), we used the duality[a] between the gradient and divergence operators, namely the fact that for differentiable scalar field $s(\boldsymbol{x})$ and vector field $\boldsymbol{v}(\boldsymbol{x})$, we have

$$\langle \nabla s, \boldsymbol{v} \rangle_{L_2} + \langle s, \nabla \cdot \boldsymbol{v} \rangle_{L_2} = 0, \tag{B.9}$$

where $\langle \boldsymbol{p}, \boldsymbol{q} \rangle_{L_2} := \int \langle \boldsymbol{p}, \boldsymbol{q} \rangle \mathrm{d}\boldsymbol{x}$. Specifically, in (B.8), $s \equiv \frac{\delta \Phi}{\delta \rho}$ and $\boldsymbol{v} \equiv \rho \nabla \frac{\delta \Phi}{\delta \rho}$. □

---

[a]In words, the gradient and the negative divergence are adjoint maps.

C | Proofs for Chapter 6

Proof of Theorem 6.1

We provide the formal statement followed by the proof.

Theorem 5.1. *Consider the regularized risk functional (6.12) wherein F is given by (6.8)-(6.9). Let $\rho(t, \boldsymbol{\theta})$ solve the IVP (6.14), and let $\{\varrho_{k-1}\}_{k \in \mathbb{N}}$ be the sequence generated by (6.17) with $\varrho_0 \equiv \rho_0$. Define the interpolation $\varrho_h : [0, \infty) \times \mathbb{R}^p \mapsto [0, \infty)$ as*

$$\varrho_h(t, \boldsymbol{\theta}) := \varrho_{k-1}(h, \boldsymbol{\theta}) \ \forall t \in [(k-1)h, kh), \ k \in \mathbb{N}.$$

Then $\varrho_h(t, \boldsymbol{\theta}) \xrightarrow{h \downarrow 0} \rho(t, \boldsymbol{\theta})$ in $L^1(\mathbb{R}^p)$.

Proof. Our proof follows the general development in [194, Sec. 12.3]. In the following, we sketch the main ideas.

We have the semi-implicit free energy

$$\hat{F}_\beta(\varrho, \varrho_{k-1}) = \underbrace{\mathbb{E}_\varrho \left[F_0 + V(\boldsymbol{\theta}) + \int_{\mathbb{R}^p} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \varrho_{k-1}(\tilde{\boldsymbol{\theta}}) \mathrm{d}\tilde{\boldsymbol{\theta}} \right]}_{=: \mathcal{V}_{\mathrm{advec}}(\varrho)} + \beta^{-1} \mathbb{E}_\varrho \left[\log \varrho \right], \quad k \in \mathbb{N},$$

wherein the summand

$$\mathcal{V}_{\mathrm{advec}}(\varrho) := \mathbb{E}_\varrho \left[F_0 + V(\boldsymbol{\theta}) + \int_{\mathbb{R}^p} U(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \varrho_{k-1}(\tilde{\boldsymbol{\theta}}) \mathrm{d}\tilde{\boldsymbol{\theta}} \right]$$

is linear in ϱ, and contributes as an effective advection potential energy. The remaining summand $\beta^{-1} \mathbb{E}_\varrho \left[\log \varrho \right]$ results in from diffusion regularization and contributes as an internal energy term.

We note from (6.9) that the functional $\mathcal{V}_{\mathrm{advec}}(\varrho)$ is lower bounded for all $\varrho \in \mathcal{P}_2(\mathbb{R}^p)$. Furthermore, $\mathcal{V}_{\mathrm{advec}}(\varrho)$ and $\nabla \mathcal{V}_{\mathrm{advec}}(\varrho)$ are uniformly Lipschitz continuous, i.e., there

exists $C_1 > 0$ such that

$$\|\nabla \mathcal{V}_{\text{advec}}(\varrho)\|_{L^\infty(\mathbb{R}^p)} + \|\nabla^2 \mathcal{V}_{\text{advec}}(\varrho)\|_{L^\infty(\mathbb{R}^p)} \leq C_1$$

for all $\varrho \in \mathcal{P}_2(\mathbb{R}^p)$ where the constant $C_1 > 0$ is independent of ϱ, and ∇^2 denotes the Euclidean Hessian operator.

Moreover, there exists $C_2 > 0$ such that for all $\varrho, \tilde{\varrho} \in \mathcal{P}_2(\mathbb{R}^p)$, we have

$$\|\nabla \mathcal{V}_{\text{advec}}(\varrho) - \nabla \mathcal{V}_{\text{advec}}(\tilde{\varrho})\|_{L^\infty(\mathbb{R}^p)} \leq C_2 W_2(\varrho, \tilde{\varrho}).$$

Thus, $\mathcal{V}_{\text{advec}}(\varrho)$ satisfy the hypotheses in [194, Sec. 12.2].

For $t \in [0, T]$, we say that $\rho(t, \boldsymbol{\theta}) \in C([0, T], \mathcal{P}_2(\mathbb{R}^p))$ is a weak solution of the IVP (6.14) if for any smooth compactly supported test function $\varphi \in C_c^\infty([0, \infty) \times \mathbb{R}^p)$, we have

$$\int_0^\infty \int_{\mathbb{R}^p} \left(\left(\frac{\partial \varphi}{\partial t} - \langle \nabla \varphi, \nabla \mathcal{V}_{\text{advec}}(\varrho)\rangle\right)\rho + \beta^{-1}\rho\Delta\varphi\right) \mathrm{d}\boldsymbol{\theta}\mathrm{d}t = -\int_{\mathbb{R}^p} \varphi(t = 0, \boldsymbol{\theta})\rho_0(\boldsymbol{\theta}). \quad \text{(C.1)}$$

Following [194, Sec. 12.2], under the stated hypotheses on $\mathcal{V}_{\text{advec}}(\varrho)$, there exists weak solution of the IVP (6.14) that is continuous w.r.t. the W_2 metric.

The remaining of the proof follows the outline below.

- Using the Dunford-Pettis' theorem, establish that the sequence of functions $\{\varrho_k(h, \boldsymbol{\theta})\}_{k\in\mathbb{N}}$ solving (6.17) is unique.

- Define the interpolation $\varrho_h(t) := \varrho_k(h, \boldsymbol{\theta})$ if $t \in ((k-1)h, kh]$ for all $k \in \mathbb{N}$. Then establish that $\varrho_h(t)$ solves a discrete approximation of (C.1).

- Finally combine the gradient estimates and pass to the limit $h \downarrow 0$, to conclude that $\rho_h(t)$ in this limit solves converges to the weak solution of (C.1) in strong $L^1(\mathbb{R}^p)$ sense.

For the detailed calculations on the passage to the limit, we refer the readers to [194,

Sec. 12.5]. □

Expressions involving the derivatives of v

We define matrices

$$T := \tanh\left(WX^\top + b1^\top\right),$$

$$S := \operatorname{sech}^2\left(WX^\top + b1^\top\right),$$

where 1 is a vector of all ones of size $n_{\text{data}} \times 1$, and the functions $\tanh(\cdot)$ and $\operatorname{sech}^2(\cdot)$ are elementwise. Notice that $T, S \in \mathbb{R}^{N \times n_{\text{data}}}$. Then

$$v = -\frac{2}{n_{\text{data}}} a \odot \left(\tanh(WX^\top + b1^\top)y\right) = -\frac{2}{n_{\text{data}}} a \odot (Ty).$$

Proposition C.1. *With the above notations in place, we have*

$$\frac{\partial v}{\partial a} 1 = \sum_{k=1}^{N} \frac{\partial v_k}{\partial a} = -\frac{2}{n_{\text{data}}} Ty, \tag{C.2}$$

and

$$\frac{\partial v}{\partial b} 1 = \sum_{k=1}^{N} \frac{\partial v_k}{\partial b} = -\frac{2}{n_{\text{data}}} a \odot Sy. \tag{C.3}$$

Furthermore,

$$\sum_{k=1}^{N} \frac{\partial v_k}{\partial W} = -\frac{2}{n_{\text{data}}} \left[(a1^\top) \odot \left(S\left(X \odot y1^\top\right)\right)\right]. \tag{C.4}$$

Proof. The kth element of v is $v_k = -\frac{2}{n_{\text{data}}} a_k \sum_{i=1}^{n_{\text{data}}} [T_{k,i} y_i]$. Thus,

$$\frac{\partial v_k}{\partial a_j} = \begin{cases} 0 & \text{for} \quad k \neq j, \\[2mm] -\frac{2}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} [T_{k,i} y_i] & \text{for} \quad k = j. \end{cases}$$

So the matrix $\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{a}}$ is diagonal, and

$$\left[\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{a}}\mathbf{1}\right]_k = -\frac{2}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} [\boldsymbol{T}_{k,i}\boldsymbol{y}_i] = -\frac{2}{n_{\text{data}}} [\boldsymbol{T}\boldsymbol{y}]_k .$$

Hence, we obtain

$$\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{a}}\mathbf{1} = -\frac{2}{n_{\text{data}}}\boldsymbol{T}\boldsymbol{y},$$

which is (C.2).

On the other hand,

$$\frac{\partial \boldsymbol{v}_k}{\partial \boldsymbol{b}_k} = \frac{\partial}{\partial \boldsymbol{b}_k}\left[-\frac{2}{n_{\text{data}}}\boldsymbol{a}_k \sum_{i=1}^{n_{\text{data}}} [\boldsymbol{T}_{k,i}\boldsymbol{y}_i]\right]$$

$$= -\frac{2}{n_{\text{data}}}\boldsymbol{a}_k \sum_{i=1}^{n_{\text{data}}} \frac{\partial}{\partial \boldsymbol{b}_k} [\boldsymbol{T}_{k,i}\boldsymbol{y}_i] .$$

Note that

$$\frac{\partial}{\partial \boldsymbol{b}_k} [\boldsymbol{T}_{k,i}\boldsymbol{y}_i] = \frac{\partial}{\partial \boldsymbol{b}_k} \tanh\left(\sum_{j=1}^{n_x} (\boldsymbol{W}_{k,j}\boldsymbol{X}_{i,j}) + \boldsymbol{b}_k\right)\boldsymbol{y}_i$$

$$= \operatorname{sech}^2\left(\sum_{j=1}^{n_x} (\boldsymbol{W}_{k,j}\boldsymbol{X}_{i,j}) + \boldsymbol{b}_k\right)\boldsymbol{y}_i = \boldsymbol{S}_{k,i}\boldsymbol{y}_i.$$

Therefore,

$$\frac{\partial \boldsymbol{v}_k}{\partial \boldsymbol{b}_j} = \begin{cases} 0 & \text{for} \quad k \neq j, \\ -\frac{2}{n_{\text{data}}}\boldsymbol{a}_k \sum_{i=1}^{n_{\text{data}}} [\boldsymbol{S}_{k,i}\boldsymbol{y}_i] & \text{for} \quad k = j. \end{cases}$$

As the matrix $\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{b}}$ is diagonal, we get

$$\left[\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{b}}\mathbf{1}\right]_k = -\frac{2}{n_{\text{data}}}\boldsymbol{a}_k \sum_{i=1}^{n_{\text{data}}} [\boldsymbol{S}_{k,i}\boldsymbol{y}_i] = -\frac{2}{n_{\text{data}}}\boldsymbol{a}_k [\boldsymbol{S}\boldsymbol{y}]_k ,$$

and so

$$\frac{\partial \boldsymbol{v}}{\partial \boldsymbol{b}}\mathbf{1} = -\frac{2}{n_{\text{data}}}\boldsymbol{a} \odot \boldsymbol{S}\boldsymbol{y},$$

which is indeed (C.3).

Likewise, we take an element-wise approach to the derivatives with respect to weights $\boldsymbol{W}_{k,m}$. Note that such a weight $\boldsymbol{W}_{k,m}$ will only appear in the kth element of $\boldsymbol{v}$, and so we only need to compute

$$\frac{\partial \boldsymbol{v}_k}{\partial \boldsymbol{W}_{k,m}} = -\frac{2}{n_{\text{data}}} \boldsymbol{a}_k \sum_{i=1}^{n_{\text{data}}} \frac{\partial}{\partial \boldsymbol{W}_{k,m}} \left[\boldsymbol{T}_{k,i} \boldsymbol{y}_i \right].$$

Since

$$\frac{\partial}{\partial \boldsymbol{W}_{k,m}} \left[\boldsymbol{T}_{k,i} \boldsymbol{y}_i \right] = \frac{\partial}{\partial \boldsymbol{W}_{k,m}} \tanh \left(\sum_{j=1}^{n_x} (\boldsymbol{W}_{k,j} \boldsymbol{X}_{j,i}) + \boldsymbol{b}_k \right) \boldsymbol{y}_i$$

$$= \text{sech}^2 \left(\sum_{j=1}^{n_x} (\boldsymbol{W}_{k,j} \boldsymbol{X}_{i,j}) + \boldsymbol{b}_k \right) \boldsymbol{X}_{i,m} \boldsymbol{y}_i = \boldsymbol{S}_{k,i} \boldsymbol{X}_{i,m} \boldsymbol{y}_i,$$

we have

$$\frac{\partial \boldsymbol{v}_k}{\partial \boldsymbol{W}_{m,j}} = \begin{cases} 0 & \text{for} \quad k \neq m, \\ -\frac{2}{n_{\text{data}}} \boldsymbol{a}_k \sum_{i=1}^{n_{\text{data}}} \left[\boldsymbol{S}_{k,i} \boldsymbol{X}_{i,m} \boldsymbol{y}_i \right] & \text{for} \quad k = m. \end{cases}$$

Thus,

$$\sum_{k=1}^{N} \frac{\partial \boldsymbol{v}_k}{\partial \boldsymbol{W}_{m,j}} = -\frac{2}{n_{\text{data}}} \boldsymbol{a}_m \sum_{i=1}^{n_{\text{data}}} \left[\boldsymbol{S}_{m,i} \boldsymbol{X}_{i,m} \boldsymbol{y}_i \right]$$

$$= -\frac{2}{n_{\text{data}}} \boldsymbol{a}_m \left[\boldsymbol{S} \left(\boldsymbol{X} \odot (\boldsymbol{y} \boldsymbol{1}^{\top}) \right) \right]_{m,j}.$$

Therefore, considering $\boldsymbol{1} \in \mathbb{R}^{n_x}$, we write

$$\sum_{k=1}^{N} \frac{\partial \boldsymbol{v}_k}{\partial \boldsymbol{W}} = -\frac{2}{n_{\text{data}}} \left[(\boldsymbol{a} \boldsymbol{1}^{\top}) \odot (\boldsymbol{S} (\boldsymbol{X} \odot \boldsymbol{y} \boldsymbol{1}^{\top})) \right],$$

thus arriving at (C.4). This completes the proof. $\qquad \square$

Expressions involving the derivatives of u

Expressions involving the derivatives of u, are summarized in the Proposition next.

These results find use in Sec. 6.3. We start by noting that

$$u = \frac{1}{n_{\text{data}}}(\mathbf{1}^\top a \odot T)(\mathbf{1}^\top a \odot T)^\top \rho$$

$$= \frac{1}{n_{\text{data}}}(\mathbf{1}^\top a \odot T)(a^\top \mathbf{1} \odot T^\top)\rho.$$

Proposition C.2. *With the above notations in place, we have*

$$\underbrace{\frac{\partial u}{\partial a}}_{N \times N} \underbrace{\mathbf{1}}_{N \times 1} = \frac{1}{n_{\text{data}}}\left[\left((\varrho a^\top) \odot (TT^\top)\right)\mathbf{1} + \mathbf{1}\left(a \odot \varrho\right)^\top TT^\top \mathbf{1}\right], \tag{C.5}$$

and

$$\underbrace{\frac{\partial u}{\partial b}}_{N \times N} \underbrace{\mathbf{1}}_{N \times 1} = \frac{1}{n_{\text{data}}}\left[\left((a\mathbf{1}^\top) \odot (ST^\top) \odot \left(\mathbf{1}\left(a \odot \varrho\right)^\top\right)\right)\mathbf{1}\right.$$

$$\left. + \left((\mathbf{1}a^\top) \odot (ST^\top) \odot \left((a \odot \varrho)\mathbf{1}^\top\right)\right)\mathbf{1}\right]. \tag{C.6}$$

Furthermore,

$$\sum_{k=1}^N \frac{\partial u}{\partial W_{i,j}} = \frac{1}{n_{\text{data}}}\sum_{k=1}^N \sum_{m=1}^{n_{\text{data}}} a_i a_k (\varrho_i + \varrho_k) T_{k,m} S_{i,m} X_{m,j}. \tag{C.7}$$

Proof. Letting $t_i^\top$ denote the ith row of T, we rewrite u as follows:

$$u = \frac{1}{n_{\text{data}}}\begin{bmatrix} a_1 t_1^\top \\ \vdots \\ a_N t_N^\top \end{bmatrix}\begin{bmatrix} \rho_1 a_1 t_1 + \cdots + \rho_N a_N t_N \end{bmatrix}$$

$$= \frac{1}{n_{\text{data}}} \begin{bmatrix} a_1 t_1^\top (\rho_1 a_1 t_1 + \cdots + \rho_N a_N t_N) \\ \vdots \\ a_N t_N^\top (\rho_1 a_1 t_1 + \cdots + \rho_N a_N t_N) \end{bmatrix}.$$

For $i \neq k$, we thus have

$$\frac{\partial u_i}{\partial a_k} = \frac{1}{n_{\text{data}}} a_i t_i^\top (\rho_k t_k) = \frac{1}{n_{\text{data}}} a_i \rho_k t_i^\top t_k.$$

Likewise, for $i = k$, we have

$$\frac{\partial u_k}{\partial a_k} = \frac{1}{n_{\text{data}}} a_k \rho_k t_k^\top t_k + \frac{1}{n_{\text{data}}} t_k^\top (\rho_1 a_1 t_1 + \cdots + \rho_N a_N t_N).$$

Combining the above, we obtain $\frac{\partial u}{\partial a}$, and hence (C.5) follows. On the other hand, for $i \neq k$, we have

$$\frac{\partial u_i}{\partial b_k} = \frac{1}{n_{\text{data}}} a_i t_i^\top \rho_k a_k \frac{\partial t_k}{\partial b_k} = \frac{1}{n_{\text{data}}} a_i t_i^\top \rho_k a_k s_k,$$

and for $i = k$, we obtain

$$\frac{\partial u_i}{\partial b_k} = \frac{1}{n_{\text{data}}} a_i \left(\frac{\partial t_k}{\partial b_k} \right)^\top (\rho_1 a_1 t_1 + \cdots + \rho_N a_N t_N)$$

$$+ \frac{1}{n_{\text{data}}} a_k t_k^\top \rho_k a_k \frac{\partial t_k}{\partial b_k}$$

$$= \frac{1}{n_{\text{data}}} a_i (s_k)^\top (\rho_1 a_1 t_1 + \cdots + \rho_N a_N t_N)$$

$$+ \frac{1}{n_{\text{data}}} a_k t_k^\top \rho_k a_k s_k.$$

Combining the above, we obtain $\frac{\partial u}{\partial b}$, and hence (C.6) follows.

Finally, noting that $W_{i,j}$ is in the ith row of T, for $i \neq k$, we obtain

$$\frac{\partial u_k}{\partial W_{i,j}} = \frac{1}{n_{\text{data}}} a_k t_k^\top \rho_i a_i \frac{\partial t_i}{\partial W_{i,j}} = \frac{1}{n_{\text{data}}} a_k t_k^\top \rho_i a_i (s_i \odot x_j),$$

where $\boldsymbol{x}_j$ is the jth column of $\boldsymbol{X}$. Likewise, for $i = k$, we get

$$
\frac{\partial \boldsymbol{u}_k}{\partial \boldsymbol{W}_{i,j}} = \frac{1}{n_{\text{data}}} a_k \left(\frac{\partial \boldsymbol{t}_k}{\partial \boldsymbol{W}_{i,j}} \right)^\top (\rho_1 a_1 \boldsymbol{t}_1 + \cdots + \rho_N a_N \boldsymbol{t}_N)
$$

$$
+ \frac{1}{n_{\text{data}}} a_k \boldsymbol{t}_k^\top \rho_k a_k \frac{\partial \boldsymbol{t}_k}{\partial \boldsymbol{W}_{i,j}}
$$

$$
= \frac{1}{n_{\text{data}}} a_k (\boldsymbol{s}_i \odot \boldsymbol{x}_j)^\top (\rho_1 a_1 \boldsymbol{t}_1 + \cdots + \rho_N a_N \boldsymbol{t}_N)
$$

$$
+ \frac{1}{n_{\text{data}}} a_k \boldsymbol{t}_k^\top \rho_k a_k (\boldsymbol{s}_i \odot \boldsymbol{x}_j).
$$

Combining the above, we obtain $\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{W}_{i,j}}$, thereby arriving at (C.7). □

190

D | Proofs for Chapter 7

Proof of Theorem 7.1

We start from (7.12) by dropping the indices i and k, and set $G(\boldsymbol{\mu}) = \langle \boldsymbol{a}, \boldsymbol{\mu} \rangle$, where $\boldsymbol{a} \in \mathbb{R}^N \smallsetminus \{\mathbf{0}\}$.

For notational ease, let $\boldsymbol{y} := \exp\left(\frac{\boldsymbol{\lambda}_0}{\alpha \varepsilon}\right), \boldsymbol{z} := \exp\left(\frac{\boldsymbol{\lambda}_1}{\alpha \varepsilon}\right) \in \mathbb{R}^N_{>0}$. Since G is linear, its Legendre-Fenchel conjugate is an indicator function:

$$
G^*(-\boldsymbol{\lambda}_1) = \begin{cases} 0 & \text{if} \quad \boldsymbol{\lambda}_1 = -\boldsymbol{a}, \\ +\infty & \text{otherwise.} \end{cases}
$$

Therefore, (7.12) yields

$$
\boldsymbol{\lambda}_0^{\text{opt}} = \arg\max_{\boldsymbol{\lambda}_0 \in \mathbb{R}^N} \left\{ \langle \boldsymbol{\lambda}_0, \boldsymbol{\zeta} \rangle - \alpha \varepsilon \langle \boldsymbol{y}, \boldsymbol{\Gamma} \boldsymbol{z} \rangle \right\}, \tag{D.1a}
$$

$$
\boldsymbol{\lambda}_1^{\text{opt}} = -\boldsymbol{a}. \tag{D.1b}
$$

From (D.1b),

$$
\boldsymbol{z}^{\text{opt}} = \exp\left(-\frac{1}{\alpha \varepsilon} \boldsymbol{a}\right). \tag{D.2}
$$

Setting the gradient of the objective in (D.1a) to zero, determines $\boldsymbol{\lambda}_0^{\text{opt}}$, or equivalently $\boldsymbol{y}^{\text{opt}}$ as

$$
\boldsymbol{y}^{\text{opt}} = \boldsymbol{\zeta} \oslash \left(\boldsymbol{\Gamma} \boldsymbol{z}^{\text{opt}}\right). \tag{D.3}
$$

From (7.14), the proximal update is

$$
\text{prox}_{\frac{1}{\alpha} \boldsymbol{\Phi}}^{W_\varepsilon} (\boldsymbol{\zeta}) = \boldsymbol{z}^{\text{opt}} \odot \left(\boldsymbol{\Gamma}^\top \boldsymbol{y}^{\text{opt}}\right)
$$

$$
\stackrel{\text{(D.3)}}{=} \boldsymbol{z}^{\text{opt}} \odot \left(\boldsymbol{\Gamma}^\top \left(\boldsymbol{\zeta} \oslash \left(\boldsymbol{\Gamma} \boldsymbol{z}^{\text{opt}}\right)\right)\right). \tag{D.4}
$$

Substituting (D.2) in (D.4), we arrive at (7.15).□

Proof of Theorem 7.2

A Lyapunov functional associated with the porous medium equation given in the last row of Table 7.1 is the free energy

$$
F_i(\mu_i) = \mathbb{E}_{\mu_i}\left[\nu_i^k + \beta^{-1}\frac{\mu_i^{m-1}}{m-1}\right].
\tag{D.5}
$$

From (D.5), the "discrete free energy" is

$$
F_i(\boldsymbol{\mu}_i) = \left\langle \boldsymbol{\nu}_i^k + \beta^{-1}\frac{\boldsymbol{\mu}_i^{m-1}}{m-1}, \boldsymbol{\mu}_i \right\rangle.
\tag{D.6}
$$

Its Legendre-Fenchel conjugate is given by

$$
F_i^\star(\boldsymbol{\lambda}_i) = \sup_{\boldsymbol{\mu}_i}\left\{\boldsymbol{\lambda}_i^\top\boldsymbol{\mu}_i - (\boldsymbol{\nu}_i^k)^\top\boldsymbol{\mu}_i - \frac{\beta^{-1}}{m-1}\mathbf{1}^\top\boldsymbol{\mu}_i^m\right\}.
\tag{D.7}
$$

Setting the gradient of the objective function in (D.7) w.r.t. $\boldsymbol{\mu}_i$ to zero, and solving for $\boldsymbol{\mu}_i$ yields

$$
(\boldsymbol{\mu}_i)_{max} = \left(\beta\frac{m-1}{m}\left(\boldsymbol{\lambda}_i - \boldsymbol{\nu}_i^k\right)\right)^{\frac{1}{m-1}}.
\tag{D.8}
$$

Substituting (D.8) back into (D.7), results

$$
F_i^\star(\boldsymbol{\lambda}_i) = \left(\beta\left(\frac{m-1}{m}\mathbf{1}^\top(\boldsymbol{\lambda}_i - \boldsymbol{\nu}_i^k)\right)^m\right)^{\frac{1}{m-1}}.
\tag{D.9}
$$

Fixing $\boldsymbol{\lambda}_{0i}$, and taking the gradient of the objective in (7.12) w.r.t. $\boldsymbol{\lambda}_{1i}$ gives

$$\nabla_{\boldsymbol{\lambda}_{1i}} F_i^\star \left(-\boldsymbol{\lambda}_{1i} \right) = \boldsymbol{z}_i \odot \left(\boldsymbol{\Gamma}^\top \boldsymbol{y}_i \right). \tag{D.10}$$

Substituting (D.9) into the left-hand-side of (D.10) results in (7.16b). The rest of the proof follows exactly the second part of the proof of [5, Theorem 1]. $\square$

Proof of Theorem 7.3

Proof. Following [233, Theorem 2.4], the Legendre-Fenchel conjugate of $\Phi(\boldsymbol{\mu})$ is

$$\Phi^\star(\boldsymbol{\lambda}) = -\langle \boldsymbol{C} - \boldsymbol{\lambda}\mathbb{1}^T, \boldsymbol{X}^\star \rangle + \varepsilon E \left(\boldsymbol{X}^\star \right) \tag{D.11}$$

where $\boldsymbol{X}^\star = \mathrm{diag}\left(\exp\left(\frac{\boldsymbol{\lambda}}{w\varepsilon} - \frac{\boldsymbol{\nu}}{\varepsilon} \right) \right) \boldsymbol{K} \, \mathrm{diag}(\boldsymbol{\xi} \oslash \boldsymbol{K} \exp\left(\frac{\boldsymbol{\lambda}}{w\varepsilon} - \frac{\boldsymbol{\nu}}{\varepsilon} \right))$ and $E(.)$ represents Shannon entropy function.

Let $\boldsymbol{y} := \exp\left(\frac{\boldsymbol{\lambda}_0}{\alpha\varepsilon} \right) \in \mathbb{R}^N_{>0}$, $\boldsymbol{z} := \exp\left(\frac{\boldsymbol{\lambda}_1}{\alpha\varepsilon} \right) \in \mathbb{R}^N_{>0}$, and drop the subscripts i in Eq. (7.12) for notational ease. Fixing $\boldsymbol{\lambda}_1$, and taking the gradient of the objective in Eq. (7.12) w.r.t. $\boldsymbol{\lambda}_0$ gives Eq. (7.20a).

On the other hand, fixing $\boldsymbol{\lambda}_0$, and taking the gradient of the objective in Eq. (7.12) w.r.t. $\boldsymbol{\lambda}_1$ gives

$$\nabla_{\boldsymbol{\lambda}_1} \Phi^\star \left(-\boldsymbol{\lambda}_1 \right) = \boldsymbol{z} \odot \left(\boldsymbol{\Gamma}^\top \boldsymbol{y} \right) \tag{D.12}$$

where from Eq. (D.11), we have

$$\begin{aligned}
\nabla_{\boldsymbol{\lambda}_1} \Phi^\star(-\boldsymbol{\lambda}_1) &= \exp\left(\frac{-\boldsymbol{\lambda}_1}{w\varepsilon} - \frac{\boldsymbol{\nu}^k}{\varepsilon} \right) \odot \left(\boldsymbol{K}\boldsymbol{\xi} \oslash \left(\boldsymbol{K} \exp\left(\frac{-\boldsymbol{\lambda}_1}{w\varepsilon} - \frac{\boldsymbol{\nu}^k}{\varepsilon} \right) \right) \right) \\
&= \boldsymbol{z}^{-\frac{1}{\alpha w}} \odot \exp\left(\frac{-\boldsymbol{\nu}^k}{\varepsilon} \right) \odot \boldsymbol{K}\boldsymbol{\xi} \oslash \left(\boldsymbol{K}\left(\boldsymbol{z}^{-\frac{1}{\alpha w}} \odot \exp\left(\frac{-\boldsymbol{\nu}^k}{\varepsilon} \right) \right) \right). \tag{D.13}
\end{aligned}$$

Using Eq. (D.13) into the left hand side of Eq. (D.12) results in Eq. (7.20b). Finally,

Eq. (7.14) yields the proximal update Eq. (7.21).

$\square$

Proof of Theorem 7.4

It is known [53, Theorem 2.4] that for given $\varepsilon > 0$ and $\boldsymbol{\mu} \in \Delta^{N-1}$, the Legendre-Fenchel conjugate $\left(W_{\varepsilon,\boldsymbol{\mu}}^2\right)^* (\boldsymbol{u})$ is $C^\infty(\mathbb{R}^N)$ w.r.t. $\boldsymbol{u} \in \mathbb{R}^N$, and the gradient $\nabla_{\boldsymbol{u}} \left(W_{\varepsilon,\boldsymbol{\mu}}^2\right)^* (\boldsymbol{u})$ is $1/\varepsilon$ Lipschitz. Furthermore, [53, Theorem 2.4] gives the explicit formula

$$\left(W_{\varepsilon,\boldsymbol{\mu}}^2\right)^* (\boldsymbol{u}) = -\varepsilon \big\langle \boldsymbol{\mu}, \log \left(\boldsymbol{\mu} \oslash \left(\boldsymbol{\Gamma} \exp \left(\boldsymbol{u}/\varepsilon\right)\right)\right) \big\rangle, \tag{D.14a}$$

$$\nabla_{\boldsymbol{u}} \left(W_{\varepsilon,\boldsymbol{\mu}}^2\right)^* (\boldsymbol{u}) = \exp \left(\boldsymbol{u}/\varepsilon\right) \odot \left(\boldsymbol{\Gamma} \left(\boldsymbol{\mu} \oslash \left(\boldsymbol{\Gamma} \exp \left(\boldsymbol{u}/\varepsilon\right)\right)\right)\right) \in \Delta^{N-1}. \tag{D.14b}$$

Using (D.14a) in the objective of (7.27) followed by algebraic simplification yields (7.28). Using (D.14b) in (7.25), we obtain (7.29). $\square$

Proof of Lemma 7.1

We re-write the constraint set $\mathcal{C}$ as

$$\mathcal{C} = \left\{ \boldsymbol{z} \in \mathbb{R}^{nN} \mid \boldsymbol{A}\boldsymbol{z} = \frac{2}{\alpha}\boldsymbol{\nu}_{\text{sum}}^k \right\}, \tag{D.15}$$

where $\boldsymbol{z} = (\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n)$, $\boldsymbol{z}_i \in \mathbb{R}^N$ for all $i \in [n]$, $\boldsymbol{A} := [\boldsymbol{I}_N, \cdots, \boldsymbol{I}_N] \in \mathbb{R}^{N \times nN}$, and $\boldsymbol{I}_N$ is the $N \times N$ identity matrix.

Following [250, Sec. 4], we have

$$\text{proj}_{\mathcal{C}} (\boldsymbol{v}) = \boldsymbol{v} - \boldsymbol{A}^\dagger \left(\boldsymbol{A}\boldsymbol{v} - \frac{2}{\alpha}\boldsymbol{\nu}_{\text{sum}}^k\right) \tag{D.16}$$

where the superscript † denotes the Moore-Penrose pseudoinverse. For our $\boldsymbol{A} \in$

$\mathbb{R}^{N \times nN}$, (D.16) simplifies to

$$
\begin{aligned}
\operatorname{proj}_C(\boldsymbol{v}) &= \boldsymbol{v} - \boldsymbol{A}^\top (\boldsymbol{A}\boldsymbol{A}^\top)^{-1} \left(\boldsymbol{A}\boldsymbol{v} - \frac{2}{\alpha} \boldsymbol{\nu}_{\text{sum}}^k \right) \\
&= \boldsymbol{v} - \begin{bmatrix} \frac{1}{n}\boldsymbol{I}_N \\ \vdots \\ \frac{1}{n}\boldsymbol{I}_N \end{bmatrix} \left(\sum_{i=1}^n \boldsymbol{v}_i - \frac{2}{\alpha} \boldsymbol{\nu}_{\text{sum}}^k \right) \\
&= \boldsymbol{v} - \begin{bmatrix} \overline{\boldsymbol{v}} - \frac{2}{n\alpha} \boldsymbol{\nu}_{\text{sum}}^k \\ \vdots \\ \overline{\boldsymbol{v}} - \frac{2}{n\alpha} \boldsymbol{\nu}_{\text{sum}}^k \end{bmatrix},
\end{aligned}
$$

thus completing the proof. $\square$

Gradient and Hessian of (7.30)

For notational ease, let us drop the indices $i \in [n]$ and $k \in \mathbb{N}_0$ for the time being, and focus on computing the gradient and Hessian of

$$
f(\boldsymbol{u}) := \left\langle \boldsymbol{\mu}, \log\left(\boldsymbol{\Gamma} \exp\left(\boldsymbol{u}/\varepsilon\right)\right) \right\rangle
$$

w.r.t. $\boldsymbol{u} \in \mathbb{R}^N$ for given $\boldsymbol{\mu} = (\mu_1, \cdots, \mu_N) \in \Delta^{N-1}$. Notice that f is twice continuously differentiable but is not everywhere strictly convex; e.g., f is affine along any line $\boldsymbol{u} = u_0 \mathbf{1}$ where u_0 is some nonzero real and $\mathbf{1}$ denotes the $N \times 1$ vector of ones.

Denote the jth row of the matrix $\boldsymbol{\Gamma} \in \mathbb{R}^{N \times N}$ as $\boldsymbol{\gamma}_j$, and write

$$
f(\boldsymbol{u}) = \sum_{j=1}^N \mu_j \log\langle \boldsymbol{\gamma}_j, \exp\left(\boldsymbol{u}/\varepsilon\right)\rangle. \tag{D.17}
$$

Using the chain rule in (D.17), we have

$$
\nabla_{\boldsymbol{u}} f = \frac{1}{\varepsilon} \sum_{j=1}^N \mu_j \frac{\boldsymbol{\gamma}_j \odot \exp\left(\boldsymbol{u}/\varepsilon\right)}{\langle \boldsymbol{\gamma}_j, \exp\left(\boldsymbol{u}/\varepsilon\right)\rangle} = \frac{1}{\varepsilon} \left(\boldsymbol{\Gamma}^\top \boldsymbol{\mu}\right) \odot \exp\left(\boldsymbol{u}/\varepsilon\right) \oslash \left(\boldsymbol{\Gamma} \exp\left(\boldsymbol{u}/\varepsilon\right)\right). \tag{D.18}
$$

Bringing back the indices $i \in [n]$ and $k \in \mathbb{N}_0$ as in (7.30), and letting $\boldsymbol{e}_i := \exp\left(\boldsymbol{u}_i/\varepsilon\right)$, the expression (D.18) gives

$$\nabla_{\boldsymbol{u}_i} f_i = \frac{1}{\varepsilon} \left(\boldsymbol{\Gamma}^\top \boldsymbol{\mu}_i^{k+1}\right) \odot \boldsymbol{e}_i \oslash \left(\boldsymbol{\Gamma} \boldsymbol{e}_i\right). \tag{D.19}$$

Likewise, we get the Hessian

$$\nabla_{\boldsymbol{u}_i}^2 f_i = \frac{1}{\varepsilon^2} \left[\operatorname{diag}\left(\left(\boldsymbol{\Gamma}^\top \boldsymbol{\mu}_i^{k+1}\right) \odot \boldsymbol{e}_i \oslash \left(\boldsymbol{\Gamma} \boldsymbol{e}_i\right)\right) - \operatorname{diag}\left(\left(\boldsymbol{\Gamma}^\top \boldsymbol{\mu}_i^{k+1}\right) \oslash \left(\boldsymbol{\Gamma} \boldsymbol{e}_i\right)^2\right) \boldsymbol{\Gamma} \odot \left(\boldsymbol{e}_i \boldsymbol{e}_i^\top\right) \right] \tag{D.20}$$

where $\left(\boldsymbol{\Gamma} \boldsymbol{e}_i\right)^2$ denotes the elementwise square of the vector $\boldsymbol{\Gamma} \boldsymbol{e}_i$.

Because the matrix $\boldsymbol{C}$ is symmetric, $\boldsymbol{\Gamma}$ is symmetric too, and we can drop the transpose from (D.20). Furthermore, since $\boldsymbol{\Gamma} \odot \left(\boldsymbol{e}_i \boldsymbol{e}_i^\top\right) = \operatorname{diag}\left(\boldsymbol{e}_i\right) \boldsymbol{\Gamma} \operatorname{diag}\left(\boldsymbol{e}_i\right)$, we can rewrite (D.20) as

$$\nabla_{\boldsymbol{u}_i}^2 f_i = \frac{1}{\varepsilon^2} \operatorname{diag}\left(\left(\boldsymbol{\Gamma} \boldsymbol{\mu}_i^{k+1}\right) \odot \boldsymbol{e}_i \oslash \left(\boldsymbol{\Gamma} \boldsymbol{e}_i\right)\right) \left[\boldsymbol{I}_N - \operatorname{diag}\left(\mathbf{1} \oslash \left(\boldsymbol{\Gamma} \boldsymbol{e}_i\right)\right) \boldsymbol{\Gamma} \operatorname{diag}\left(\boldsymbol{e}_i\right)\right]. \tag{D.21}$$

Notice that the matrix $\operatorname{diag}\left(\mathbf{1} \oslash \left(\boldsymbol{\Gamma} \boldsymbol{e}_i\right)\right) \boldsymbol{\Gamma} \operatorname{diag}\left(\boldsymbol{e}_i\right)$ is elementwise positive and row stochastic, and therefore, by Perron-Frobenius theorem, the matrix in square braces in (D.21) has zero as a simple eigenvalue. Thus, the Hessian (D.21) is positive semidefinite. The Hessian of the proximal objective in (7.34a) is $\boldsymbol{I}_N + \frac{1}{\tau} \nabla_{\boldsymbol{u}_i}^2 f_i$ where $\tau > 0$, and is, therefore, strictly positive definite.

Backtracking Line Search

For unconstrained minimization of an objective f_0 via recursive algorithms such as gradient descent or Newton's method, at each iteration, we compute the corresponding descent direction Δx at $x \in \operatorname{domain}(f_0)$. Then we apply the recursive update rule $x \leftarrow x + t\Delta x$ where t is a variable step size at that iteration. A standard method of

computing the step size is the backtracking line search [2, p. 464]. Given parameters $\alpha_0 \in (0, 0.5)$, $\beta_0 \in (0, 1)$, the backtracking line search starts with an initial step size $t = 1$, and while $f_0(x + t\Delta x) > f_0(x) + \alpha_0 t \langle \nabla f, \Delta x \rangle$, sets $t \leftarrow \beta_0 t$. The resulting value of t is used as the step size at that iteration.

Bibliography

[1] A. Stukowski, "Visualization and analysis of atomistic simulation data with OVITO-the Open Visualization Tool," *Modelling Simul. Mater. Sci. Eng.*, vol. 18, p. 015012, Jan. 2010.

[2] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization.* Cambridge university press, 2004.

[3] C. Bonet, N. Courty, F. Septier, and L. Drumetz, "Efficient gradient flows in sliced-wasserstein space," *arXiv preprint arxiv:2110.10972*, 2022.

[4] J. A. Carrillo, K. Craig, L. Wang, and C. Wei, "Primal dual methods for Wasserstein gradient flows," *Foundations of Computational Mathematics*, vol. 22, no. 2, pp. 389–443, 2022.

[5] K. F. Caluya and A. Halder, "Gradient flow algorithms for density propagation in stochastic systems," *IEEE Transactions on Automatic Control*, vol. 65, no. 10, pp. 3991–4004, 2019.

[6] O. Stramer and R. Tweedie, "Langevin-type models i: Diffusions with given stationary distributions and their discretizations," *Methodology and Computing in Applied Probability*, vol. 1, no. 3, pp. 283–306, 1999.

[7] O. Stramer and R. Tweedie, "Langevin-type models ii: Self-targeting candidates for mcmc algorithms," *Methodology and Computing in Applied Probability*, vol. 1, no. 3, pp. 307–328, 1999.

[8] S. F. Jarner and E. Hansen, "Geometric ergodicity of metropolis algorithms," *Stochastic processes and their applications*, vol. 85, no. 2, pp. 341–361, 2000.

[9] G. O. Roberts and O. Stramer, "Langevin diffusions and metropolis-hastings algorithms," *Methodology and computing in applied probability*, vol. 4, no. 4, pp. 337–357, 2002.

[10] S. Vempala and A. Wibisono, "Rapid convergence of the unadjusted langevin algorithm: Isoperimetry suffices," *Advances in neural information processing systems*, vol. 32, 2019.

[11] R. Zhang, C. Chen, C. Li, and L. Carin, "Policy optimization as Wasserstein gradient flows," in *International Conference on Machine Learning*, pp. 5737–5746, PMLR, 2018.

[12] C. Chu, J. Blanchet, and P. Glynn, "Probability functional descent: A unifying perspective on GANs, variational inference, and reinforcement learning," in *International Conference on Machine Learning*, pp. 1213–1222, PMLR, 2019.

[13] J. Zhang, A. Koppel, A. S. Bedi, C. Szepesvari, and M. Wang, "Variational policy gradient method for reinforcement learning with general utilities," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4572–4583, 2020.

[14] C. Kent, J. Blanchet, and P. Glynn, "Frank-wolfe methods in probability space," *arXiv preprint arXiv:2105.05352*, 2021.

[15] S. Mei, A. Montanari, and P.-M. Nguyen, "A mean field view of the landscape of two-layer neural networks," *Proceedings of the National Academy of Sciences*, vol. 115, no. 33, pp. E7665–E7671, 2018.

[16] L. Chizat and F. Bach, "On the global convergence of gradient descent for over-parameterized models using optimal transport," *Advances in neural information processing systems*, vol. 31, 2018.

[17] G. M. Rotskoff and E. Vanden-Eijnden, "Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error," *stat*, vol. 1050, p. 22, 2018.

[18] J. Sirignano and K. Spiliopoulos, "Mean field analysis of neural networks: A central limit theorem," *Stochastic Processes and their Applications*, vol. 130, no. 3, pp. 1820–1852, 2020.

[19] C. Domingo-Enrich, S. Jelassi, A. Mensch, G. Rotskoff, and J. Bruna, "A mean-field analysis of two-player zero-sum games," *Advances in neural information processing systems*, 2020.

[20] W. Krichene, K. F. Caluya, and A. Halder, "Global convergence of second-order dynamics in two-layer neural networks," *arXiv preprint arXiv:2007.06852*, 2020.

[21] A. Halder, K. F. Caluya, B. Travacca, and S. J. Moura, "Hopfield neural network flow: A geometric viewpoint," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4869–4880, 2020.

[22] R. Jordan, D. Kinderlehrer, and F. Otto, "The variational formulation of the Fokker–Planck equation," *SIAM journal on mathematical analysis*, vol. 29, no. 1, pp. 1–17, 1998.

[23] L. Ambrosio, N. Gigli, and G. Savaré, *Gradient flows: in metric spaces and in the space of probability measures.* Springer Science & Business Media, 2005.

[24] K. F. Caluya and A. Halder, "Proximal recursion for solving the Fokker-Planck equation," in *2019 American Control Conference (ACC)*, pp. 4098–4103, IEEE, 2019.

[25] H. J. Kushner, "On the differential equations satisfied by conditional probablitity densities of markov processes, with applications," *Journal of the Society for Industrial and Applied Mathematics, Series A: Control*, vol. 2, no. 1, pp. 106–119, 1964.

[26] R. L. Stratonovich, "Conditional markov processes," in *Non-linear transformations of stochastic processes*, pp. 427–453, Elsevier, 1965.

[27] R. Bucy, "Nonlinear filtering theory," *IEEE Transactions on Automatic Control*, vol. 10, no. 2, pp. 198–198, 1965.

[28] A. Halder and T. T. Georgiou, "Gradient flows in uncertainty propagation and filtering of linear Gaussian systems," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 3081–3088, IEEE, 2017.

[29] A. Halder and T. T. Georgiou, "Gradient flows in filtering and fisher-rao geometry," in *2018 Annual American Control Conference (ACC)*, pp. 4281–4286, IEEE, 2018.

[30] A. Halder and T. T. Georgiou, "Proximal recursion for the Wonham filter," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 660–665, IEEE, 2019.

[31] Y. Chen, T. T. Georgiou, and M. Pavon, "Controlling uncertainty," *IEEE Control Systems Magazine*, vol. 41, no. 4, pp. 82–94, 2021.

[32] Y. Chen, T. T. Georgiou, and M. Pavon, "Stochastic control liaisons: Richard Sinkhorn meets Gaspard Monge on a Schrodinger bridge," *SIAM Review*, vol. 63, no. 2, pp. 249–313, 2021.

[33] D. Milutinović and P. Lima, "Modeling and optimal centralized control of a large-size robotic population," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1280–1285, 2006.

[34] D. L. Milutinović and P. U. Lima, *Cells and robots: modeling and control of large-size agent populations.* Springer, 2007.

[35] M. Schwager, D. Rus, and J.-J. Slotine, "Unifying geometric, probabilistic, and potential field approaches to multi-robot deployment," *The International Journal of Robotics Research*, vol. 30, no. 3, pp. 371–383, 2011.

[36] S. G. Lee, Y. Diaz-Mercado, and M. Egerstedt, "Multirobot control using time-varying density functions," *IEEE Transactions on robotics*, vol. 31, no. 2, pp. 489–493, 2015.

[37] K. Elamvazhuthi and S. Berman, "Mean-field models in swarm robotics: A survey," *Bioinspiration & Biomimetics*, vol. 15, no. 1, p. 015001, 2019.

[38] K. F. Caluya and A. Halder, "Wasserstein proximal algorithms for the Schrödinger bridge problem: Density control with nonlinear drift," *IEEE Transactions on Automatic Control*, vol. 67, no. 3, pp. 1163–1178, 2021.

[39] K. F. Caluya and A. Halder, "Reflected Schrödinger bridge: Density control with path constraints," in *2021 American Control Conference (ACC)*, pp. 1137–1142, IEEE, 2021.

[40] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational physics*, vol. 378, pp. 686–707, 2019.

[41] L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis, "Deepxde: A deep learning library for solving differential equations," *SIAM Review*, vol. 63, no. 1, pp. 208–228, 2021.

[42] C. Villani, *Topics in optimal transportation*, vol. 58. American Mathematical Soc., 1st ed., 2003.

[43] Y. Brenier, "Polar factorization and monotone rearrangement of vector-valued functions," *Communications on pure and applied mathematics*, vol. 44, no. 4, pp. 375–417, 1991.

[44] C. Villani, *Optimal transport: old and new*, vol. 338. Springer, 2009.

[45] L. Ambrosio, N. Gigli, and G. Savaré, *Gradient flows: in metric spaces and in the space of probability measures.* Springer Science & Business Media, 2008.

[46] F. Santambrogio, "{Euclidean, metric, and Wasserstein} gradient flows: an overview," *Bulletin of Mathematical Sciences*, vol. 7, no. 1, pp. 87–154, 2017.

[47] A. Salim, A. Korba, and G. Luise, "The Wasserstein proximal gradient algorithm," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12356–12366, 2020.

[48] A. Genevay, M. Cuturi, G. Peyré, and F. Bach, "Stochastic optimization for large-scale optimal transport," in *NIPS 2016-Thirtieth Annual Conference on Neural Information Processing System*, pp. 3440–3448, 2016.

[49] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *Advances in neural information processing systems*, vol. 26, pp. 2292–2300, 2013.

[50] G. Carlier, V. Duval, G. Peyré, and B. Schmitzer, "Convergence of entropic schemes for optimal transport and gradient flows," *SIAM Journal on Mathematical Analysis*, vol. 49, no. 2, pp. 1385–1418, 2017.

[51] J.-D. Benamou, G. Carlier, M. Cuturi, L. Nenna, and G. Peyré, "Iterative Bregman projections for regularized transportation problems," *SIAM Journal on Scientific Computing*, vol. 37, no. 2, pp. A1111–A1138, 2015.

[52] G. Peyré, "Entropic approximation of Wasserstein gradient flows," *SIAM Journal on Imaging Sciences*, vol. 8, no. 4, pp. 2323–2351, 2015.

[53] M. Cuturi and G. Peyré, "A smoothed dual approach for variational wasserstein problems," *SIAM Journal on Imaging Sciences*, vol. 9, no. 1, pp. 320–343, 2016.

[54] M. Agueh and G. Carlier, "Barycenters in the wasserstein space," *SIAM Journal on Mathematical Analysis*, vol. 43, no. 2, pp. 904–924, 2011.

[55] C. Frogner and T. Poggio, "Approximate inference with Wasserstein gradient flows," in *International Conference on Artificial Intelligence and Statistics*, pp. 2581–2590, PMLR, 2020.

[56] P. Mokrov, A. Korotin, L. Li, A. Genevay, J. Solomon, and E. Burnaev, "Large-scale Wasserstein gradient flows," *arXiv preprint arXiv:2106.00736*, 2021.

[57] C. Bunne, L. Papaxanthos, A. Krause, and M. Cuturi, "Proximal optimal transport modeling of population dynamics," in *International Conference on Artificial Intelligence and Statistics*, pp. 6511–6528, PMLR, 2022.

[58] S. Boyd, N. Parikh, and E. Chu, *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

[59] R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. Jordan, "A general analysis of the convergence of ADMM," in *International Conference on Machine Learning*, pp. 343–352, PMLR, 2015.

[60] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *Journal of Scientific Computing*, vol. 78, no. 1, pp. 29–63, 2019.

[61] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[62] A. C. Thompson, "On certain contraction mappings in a partially ordered vector space," *Proceedings of the American Mathematical Society*, vol. 14, no. 3, pp. 438–443, 1963.

[63] R. M. Dudley, *Real analysis and probability.* volume 74 of Cambridge Studies in Advanced Mathematics, Cambridge University Press, 2002.

[64] A. M. Teter, I. Nodozi, and A. Halder, "Solution of the probabilistic lambert problem: Connections with optimal mass transport, schr\" odinger bridge and reaction-diffusion pdes," *arXiv preprint arXiv:2401.07961*, 2024.

[65] I. Nodozi, C. Yan, M. Khare, A. Halder, and A. Mesbah, "Neural schrödinger bridge with sinkhorn losses: Application to data-driven minimum effort control of colloidal self-assembly," *IEEE Transactions on Control Systems Technology*, 2023.

[66] I. Nodozi and A. Halder, "Wasserstein consensus admm," *arXiv preprint arXiv:2309.07351*, 2023.

[67] I. Nodozi, A. Halder, and I. Matei, "A controlled mean field model for chiplet population dynamics," *IEEE Control Systems Letters*, 2023.

[68] I. Nodozi, J. O'Leary, A. Mesbah, and A. Halder, "A physics-informed deep learning approach for minimum effort stochastic control of colloidal self-assembly," in *2023 American Control Conference (ACC)*, pp. 609–615, IEEE, 2023.

[69] C. Yan, I. Nodozi, and A. Halder, "Optimal mass transport over the euler equation," in *the 62nd IEEE Conference on Decision and Control (CDC)*, pp. 6819–6826, IEEE, 2023.

[70] A. Teter, I. Nodozi, and A. Halder, "Proximal mean field learning in shallow neural networks," *Transactions on Machine Learning Research*, 2023.

[71] I. Nodozi and A. Halder, "Schrödinger meets Kuramoto via Feynman-Kac: Minimum effort distribution steering for noisy nonuniform Kuramoto oscillators," *2022 IEEE Conference on Decision and Control (CDC), arXiv:2202.09734*, 2022.

[72] I. Nodozi and A. Halder, "A distributed algorithm for measure-valued optimization with additive objective," in *25th International Symposium on Mathematical Theory of Networks and Systems (MTNS)*, 2022.

[73] A. Halder and R. Bhattacharya, "Geodesic density tracking with applications to data driven modeling," in *2014 American Control Conference*, pp. 616–621, IEEE, 2014.

[74] L. V. Kantorovich, "On the translocation of masses," in *Dokl. Akad. Nauk. USSR (NS)*, vol. 37, pp. 199–201, 1942.

[75] J.-D. Benamou and Y. Brenier, "A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem," *Numerische Mathematik*, vol. 84, no. 3, pp. 375–393, 2000.

[76] A. Halder and R. Bhattacharya, "Dispersion analysis in hypersonic flight during planetary entry using stochastic Liouville equation," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 2, pp. 459–474, 2011.

[77] R. J. McCann, "A convexity principle for interacting gases," *Advances in mathematics*, vol. 128, no. 1, pp. 153–179, 1997.

[78] A. Figalli, *Optimal transportation and action-minimizing measures.* PhD thesis, Lyon, École normale supérieure (sciences), 2007.

[79] E. Schrödinger, "Über die umkehrung der naturgesetze," *Sitzungsberichte der Preuss. Phys. Math. Klasse*, vol. 10, pp. 144–153, 1931.

[80] E. Schrödinger, "Sur la théorie relativiste de l'électron et l'interprétation de la mécanique quantique," in *Annales de l'institut Henri Poincaré*, vol. 2, pp. 269–310, 1932.

[81] A. Wakolbinger, "Schrödinger bridges from 1931 to 1991," in *Proc. of the 4th Latin American Congress in Probability and Mathematical Statistics, Mexico City*, pp. 61–79, 1990.

[82] Y. Chen, T. T. Georgiou, and M. Pavon, "On the relation between optimal transport and Schrödinger bridges: A stochastic control viewpoint," *Journal of Optimization Theory and Applications*, vol. 169, no. 2, pp. 671–691, 2016.

[83] I. Nodozi and A. Halder, "Schrödinger meets kuramoto via feynman-kac: Minimum effort distribution steering for noisy nonuniform kuramoto oscillators," in *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 2953–2960, IEEE, 2022.

[84] V. De Bortoli, J. Thornton, J. Heng, and A. Doucet, "Diffusion Schrödinger bridge with applications to score-based generative modeling," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17695–17709, 2021.

[85] G. Wang, Y. Jiao, Q. Xu, Y. Wang, and C. Yang, "Deep generative learning via Schrödinger bridge," in *International Conference on Machine Learning*, pp. 10794–10804, PMLR, 2021.

[86] T. Chen, G.-H. Liu, and E. A. Theodorou, "Likelihood training of Schrödinger bridge using forward-backward SDEs theory," *arXiv preprint arXiv:2110.11291*, 2021.

[87] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, "Diffusion models in vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[88] M. Pavon, G. Trigila, and E. G. Tabak, "The data-driven Schrödinger bridge," *Communications on Pure and Applied Mathematics*, vol. 74, no. 7, pp. 1545–1573, 2021.

[89] F. Vargas, P. Thodoroff, A. Lamacraft, and N. Lawrence, "Solving Schrödinger bridges via maximum likelihood," *Entropy*, vol. 23, no. 9, p. 1134, 2021.

[90] A. Stromme, "Sampling from a Schrödinger bridge," in *International Conference on Artificial Intelligence and Statistics*, pp. 4058–4067, PMLR, 2023.

[91] K. F. Caluya and A. Halder, "Finite horizon density steering for multi-input state feedback linearizable systems," in *2020 American Control Conference (ACC)*, pp. 3577–3582, IEEE, 2020.

[92] S. Haddad, K. F. Caluya, A. Halder, and B. Singh, "Prediction and optimal feedback steering of probability density functions for safe automated driving," *IEEE Control Systems Letters*, vol. 5, no. 6, pp. 2168–2173, 2020.

[93] B. Oksendal, *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

[94] A. Friedman, *Partial differential equations of parabolic type*. Courier Dover Publications, 2008.

[95] B. Jamison, "Reciprocal processes," *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, vol. 30, pp. 65–86, 1974.

[96] E. Hopf, "The partial differential equation: $u_t + uu_x = \mu_{xx}$," *Communications on Pure and Applied Mathematics*, vol. 3, pp. 201–230, 1950.

[97] J. D. Cole, "On a quasi-linear parabolic equation occurring in aerodynamics," *Quarterly of Applied Mathematics*, vol. 9, pp. 225–236, 1951.

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

[99] J. A. Acebrón, L. L. Bonilla, C. J. P. Vicente, F. Ritort, and R. Spigler, "The Kuramoto model: A simple paradigm for synchronization phenomena," *Reviews of modern physics*, vol. 77, no. 1, p. 137, 2005.

[100] F. Dörfler and F. Bullo, "Synchronization in complex networks of phase oscillators: A survey," *Automatica*, vol. 50, no. 6, pp. 1539–1564, 2014.

[101] L. Hörmander, "Hypoelliptic second order differential equations," *Acta Mathematica*, vol. 119, no. 1, pp. 147–171, 1967.

[102] S. H. Strogatz, "From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators," *Physica D: Nonlinear Phenomena*, vol. 143, no. 1-4, pp. 1–20, 2000.

[103] A. Jadbabaie, N. Motee, and M. Barahona, "On the stability of the Kuramoto model of coupled nonlinear oscillators," in *Proceedings of the 2004 American Control Conference*, vol. 5, pp. 4296–4301, IEEE, 2004.

[104] N. Chopra and M. W. Spong, "On exponential synchronization of Kuramoto oscillators," *IEEE transactions on Automatic Control*, vol. 54, no. 2, pp. 353–357, 2009.

[105] H. Yin, P. G. Mehta, S. P. Meyn, and U. V. Shanbhag, "Synchronization of coupled oscillators is a game," *IEEE Transactions on Automatic Control*, vol. 57, no. 4, pp. 920–935, 2011.

[106] F. Dorfler and F. Bullo, "Synchronization and transient stability in power networks and nonuniform Kuramoto oscillators," *SIAM Journal on Control and Optimization*, vol. 50, no. 3, pp. 1616–1642, 2012.

[107] S. Sahyoun, S. Djouadi, and M. Shankar, "Optimal control of droop controlled inverters in islanded microgrids," *IFAC-PapersOnLine*, vol. 48, no. 30, pp. 363–368, 2015.

[108] M. Li and H. Dankowicz, "A unified analytical framework for optimal control problems on networks with input homogeneity," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 4, pp. 1822–1832, 2021.

[109] J. Wu and X. Li, "Global stochastic synchronization of Kuramoto-oscillator networks with distributed control," *IEEE Transactions on Cybernetics*, vol. 51, no. 12, pp. 5825–5835, 2021.

[110] E. A. Martens, E. Barreto, S. H. Strogatz, E. Ott, P. So, and T. M. Antonsen, "Exact results for the Kuramoto model with a bimodal frequency distribution," *Physical Review E*, vol. 79, no. 2, p. 026204, 2009.

[111] D. Benedetto, E. Caglioti, and U. Montemagno, "On the complete phase synchronization for the Kuramoto model in the mean-field limit," *Communications in Mathematical Sciences*, vol. 13, no. 7, pp. 1775–1786, 2015.

[112] L. Bertini, G. Giacomin, and K. Pakdaman, "Dynamical aspects of mean field plane rotators and the Kuramoto model," *Journal of Statistical Physics*, vol. 138, no. 1, pp. 270–290, 2010.

[113] F. Dorfler and F. Bullo, "Kron reduction of graphs with applications to electrical networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 1, pp. 150–163, 2012.

[114] Y. Chen, T. T. Georgiou, and M. Pavon, "Fast cooling for a system of stochastic oscillators," *Journal of Mathematical Physics*, vol. 56, no. 11, p. 113302, 2015.

[115] K. V. Mardia, G. Hughes, C. C. Taylor, and H. Singh, "A multivariate von Mises distribution with applications to bioinformatics," *Canadian Journal of Statistics*, vol. 36, no. 1, pp. 99–109, 2008.

[116] K. V. Mardia and J. Voss, "Some fundamental properties of a multivariate von Mises distribution," *Communications in Statistics-Theory and Methods*, vol. 43, no. 6, pp. 1132–1144, 2014.

[117] K. V. Mardia and P. E. Jupp, *Directional statistics*, vol. 494. John Wiley & Sons, 2009.

[118] D. W. Stroock, *Partial differential equations for probabilists*. Cambridge University Press Cambridge, 2008.

[119] B. Jamison, "Reciprocal processes," *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, vol. 30, no. 1, pp. 65–86, 1974.

[120] C. Villani, "Hypocoercivity," *arXiv preprint math/0609050*, 2006.

[121] I. Markou, *A Fokker-Planck study motivated by a problem in fluid-particle interactions.* PhD thesis, University of Maryland, College Park, 2014.

[122] P. Del Moral, "Feynman-Kac formulae," in *Feynman-Kac Formulae*, pp. 47–93, Springer, 2004.

[123] C. Villani, *Hypocoercivity*. American Mathematical Society, 2009.

[124] J. Yong, "Relations among ODEs, PDEs, FSDEs, BSDEs, and FBSDEs," in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 3, pp. 2779–2784, IEEE, 1997.

[125] J. Yong and X. Y. Zhou, *Stochastic controls: Hamiltonian systems and HJB equations*, vol. 43. Springer Science & Business Media, 1999.

[126] A. Palmer and D. Milutinović, "A hamiltonian approach using partial differential equations for open-loop stochastic optimal control," in *Proceedings of the 2011 American Control Conference*, pp. 2056–2061, IEEE, 2011.

[127] X. Tang, Y. Xue, and M. A. Grover, "Colloidal self-assembly with model predictive control," in *Proceedings of the American Control Conference*, pp. 4228–4233, 2013.

[128] Y. Xue, D. J. Beltran-Villegas, X. Tang, M. A. Bevan, and M. A. Grover, "Optimal design of a colloidal self-assembly process," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 5, pp. 1956–1963, 2014.

[129] J. A. Paulson, A. Mesbah, X. Zhu, M. C. Molaro, and R. D. Braatz, "Control of self-assembly in micro- and nano-scale systems," *Journal of Process Control*, vol. 27, pp. 38–49, 2015.

[130] X. Tang and M. A. Grover, "Control of microparticle assembly," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, pp. 491–514, 2022.

[131] G. A. Pavliotis, *Stochastic processes and applications*. Springer, 2016.

[132] A. Blaquiere, "Controllability of a Fokker-Planck equation, the Schrödinger system, and a related stochastic optimal control (revised version)," *Dynamics and Control*, vol. 2, no. 3, pp. 235–253, 1992.

[133] A. Wakolbinger, "Schrödinger bridges from 1931 to 1991," in *4th Latin American Congress in Probability and Mathematical Statistics*, pp. 61–79, 1992.

[134] E. Hopf, "The partial differential equation $u_t + uu_x = \mu u_{xx}$," *Communications on Pure and Applied mathematics*, vol. 3, no. 3, pp. 201–230, 1950.

[135] J. D. Cole, "On a quasi-linear parabolic equation occurring in aerodynamics," *Quarterly of Applied Mathematics*, vol. 9, no. 3, pp. 225–236, 1951.

[136] Y. Chen, T. Georgiou, and M. Pavon, "Entropic and displacement interpolation: a computational approach using the Hilbert metric," *SIAM Journal on Applied Mathematics*, vol. 76, no. 6, pp. 2375–2396, 2016.

[137] S.-R. Yeh, M. Seul, and B. I. Shraiman, "Assembly of ordered colloidal aggregates by electric-field-induced fluid flow," *Nature*, vol. 386, no. 6620, pp. 57–59, 1997.

[138] G. M. Whitesides and B. Grzybowski, "Self-assembly at all scales," *Science*, vol. 295, no. 5564, pp. 2418–2421, 2002.

[139] J. A. Paulson, A. Mesbah, X. Zhu, M. C. Molaro, and R. D. Braatz, "Control of self-assembly in micro-and nano-scale systems," *Journal of Process Control*, vol. 27, pp. 38–49, 2015.

[140] J. J. Juárez and M. A. Bevan, "Feedback controlled colloidal self-assembly," *Advanced Functional Materials*, vol. 22, no. 18, pp. 3833–3839, 2012.

[141] D. T. Gillespie *et al.*, "Stochastic simulation of chemical kinetics," *Annual review of physical chemistry*, vol. 58, no. 1, pp. 35–55, 2007.

[142] D. T. Gillespie, A. Hellander, and L. R. Petzold, "Perspective: Stochastic algorithms for chemical kinetics," *The Journal of chemical physics*, vol. 138, no. 17, p. 05B201_1, 2013.

[143] E. M. Furst, "Directed self-assembly," *Soft Matter*, vol. 9, no. 38, pp. 9039–9045, 2013.

[144] J. A. Liddle and G. M. Gallatin, "Nanomanufacturing: a perspective," *ACS nano*, vol. 10, no. 3, pp. 2995–3014, 2016.

[145] X. Tang, B. Rupp, Y. Yang, T. D. Edwards, M. A. Grover, and M. A. Bevan, "Optimal feedback controlled assembly of perfect crystals," *ACS Nano*, vol. 10, no. 7, pp. 6791–6798, 2016.

[146] K. F. Caluya and A. Halder, "Finite horizon density control for static state feedback linearizable systems," *arXiv preprint arXiv:1904.02272*, 2019.

[147] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[148] C. P. Robert, "Simulation of truncated normal variables," *Statistics and computing*, vol. 5, no. 2, pp. 121–125, 1995.

[149] W. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, pp. 97–109, 1970.

[150] M. Fazlyab, M. Morari, and G. J. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Transactions on Automatic Control*, vol. 67, no. 1, pp. 1–15, 2020.

[151] M. Fazlyab, A. Robey, H. Hassani, M. Morari, and G. Pappas, "Efficient and accurate estimation of Lipschitz constants for deep neural networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[152] Y. Gao and R. Lakerveld, "Feedback control for defect-free alignment of colloidal particles," *Lab on a Chip*, vol. 18, no. 14, pp. 2099–2110, 2018.

[153] S. Whitelam and I. Tamblyn, "Learning to grow: Control of material self-assembly using evolutionary reinforcement learning," *Physical Review E*, vol. 101, no. 5, p. 052604, 2020.

[154] J. O'Leary, M. M. Khare, and A. Mesbah, "Novelty search for neuroevolutionary reinforcement learning of deceptive systems: An application to control of colloidal self-assembly," in *Proceedings of the American Control Conference*, (San Diego), pp. 2776–2781, 2023.

[155] M. Dijkstra and E. Luijten, "From predictive modelling to machine learning and reverse engineering of colloidal self-assembly," *Nature materials*, vol. 20, no. 6, pp. 762–773, 2021.

[156] J. O'Leary, J. A. Paulson, and A. Mesbah, "Stochastic physics-informed neural ordinary differential equations," *Journal of Computational Physics*, vol. 468, p. 111466, nov 2022.

[157] R. Mao, J. O'Leary, A. Mesbah, and J. Mittal, "A deep learning framework discovers compositional order and self-assembly pathways in binary colloidal mixtures," *JACS Au*, vol. 2, no. 8, pp. 1818–1828, 2022.

[158] A. Lizano and X. Tang, "Convolutional neural network-based colloidal self-assembly state classification," *Soft Matter*, vol. 19, no. 19, pp. 3450–3457, 2023.

[159] T. Koshizuka and I. Sato, "Neural Lagrangian Schrödinger bridge: Diffusion modeling for population dynamics," in *The Eleventh International Conference on Learning Representations*, 2022.

[160] B. Kim, G. Kwon, K. Kim, and J. C. Ye, "Unpaired image-to-image translation via neural Schrödinger bridge," *arXiv preprint arXiv:2305.15086*, 2023.

[161] R. Sinkhorn, "A relationship between arbitrary positive matrices and doubly stochastic matrices," *The annals of mathematical statistics*, vol. 35, no. 2, pp. 876–879, 1964.

[162] R. Sinkhorn, "Diagonal equivalence to matrices with prescribed row and column sums," *The American Mathematical Monthly*, vol. 74, no. 4, pp. 402–405, 1967.

[163] W. E. Deming and F. F. Stephan, "On a least squares adjustment of a sampled frequency table when the expected marginal totals are known," *The Annals of Mathematical Statistics*, vol. 11, no. 4, pp. 427–444, 1940.

[164] L. M. Bregman, "The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming," *USSR computational mathematics and mathematical physics*, vol. 7, no. 3, pp. 200–217, 1967.

[165] Y. T. Lee and A. Sidford, "Path finding methods for linear programming: Solving linear programs in $\widetilde{O}\left(\sqrt{\text{rank}}\right)$ iterations and faster algorithms for maximum flow," in *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pp. 424–433, IEEE, 2014.

[166] A. Genevay, G. Peyré, and M. Cuturi, "Learning generative models with Sinkhorn divergences," in *International Conference on Artificial Intelligence and Statistics*, pp. 1608–1617, PMLR, 2018.

[167] E. Pauwels and S. Vaiter, "The derivatives of Sinkhorn-Knopp converge," *arXiv preprint arXiv:2207.12717*, 2022.

[168] T. Viehmann, "Implementation of batched Sinkhorn iterations for entropy-regularized Wasserstein loss," *arXiv preprint arXiv:1907.01729*, 2019.

[169] D. McQuarrie, *Statistical Mechanics*. G - Reference,Information and Interdisciplinary Subjects Series, University Science Books, 2000.

[170] P. J. Steinhardt, D. R. Nelson, and M. Ronchetti, "Bond-orientational order in liquids and glasses," *Physical Review B*, vol. 28, no. 2, 1983.

[171] C. Rycroft, "Voro++: A three-dimensional voronoi cell library in C++," tech. rep., Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2009.

[172] J. Kovac, "Physical chemistry:: A molecular approach (McQuarrie, Donald A.; Simon, John D.)," *Journal of chemical education*, vol. 75, no. 5, pp. 545–545, 1998.

[173] M. Abramowitz and I. A. Stegun, *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, vol. 55. US Government printing office, 1968.

[174] V. Ramasubramani, B. D. Dice, E. S. Harper, M. P. Spellings, J. A. Anderson, and S. C. Glotzer, "freud: A software suite for high throughput analysis of particle simulation data," *Computer Physics Communications*, vol. 254, 2020.

[175] J. A. Anderson, J. Glaser, and S. C. Glotzer, "HOOMD-blue: A python package for high-performance molecular dynamics and hard particle monte carlo simulations," *Computational Materials Science*, vol. 173, p. 109363, 2020.

[176] X. Li, T.-K. L. Wong, R. T. Q. Chen, and D. Duvenaud, "Scalable gradients for stochastic differential equations," *International Conference on Artificial Intelligence and Statistics*, 2020.

[177] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[178] I. Matei, S. Nelaturi, J. P. Lu, J. A. Bert, L. S. Crawford, and E. Chow, "Towards printing as an electronics manufacturing method: Micro-scale chiplet position control," in *2017 American Control Conference (ACC)*, pp. 1549–1555, IEEE, 2017.

[179] I. Matei, S. Nelaturi, E. M. Chow, J. P. Lu, J. A. Bert, and L. S. Crawford, "Micro-scale chiplets position control," *Journal of Microelectromechanical Systems*, vol. 28, no. 4, pp. 643–655, 2019.

[180] J. Lu, J. Thompson, G. Whiting, D. Biegelsen, S. Raychaudhuri, R. Lujan, J. Veres, L. Lavery, A. Völkel, and E. Chow, "Open and closed loop manipulation of charged microchiplets in an electric field," *Applied Physics Letters*, vol. 105, no. 5, p. 054104, 2014.

[181] T. D. Edwards and M. A. Bevan, "Controlling colloidal particles with electric fields," *Langmuir*, vol. 30, no. 36, pp. 10793–10803, 2014.

[182] I. Matei, J. de Kleer, C. Somarakis, A. Plochowietz, and J. S. Baras, "Micro-scale 2d chiplet position control: a formal approach to policy design," in *2020 59th IEEE Conference on Decision and Control (CDC)*, pp. 5519–5524, IEEE, 2020.

[183] I. Matei, A. Plochowietz, J. de Kleer, and J. S. Baras, "Micro-scale chiplet assembly control with chiplet-to-chiplet potential interaction," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 623–628, IEEE, 2021.

[184] A. Lefevre, V. Gauthier, M. Gauthier, and A. Bolopion, "Closed-loop control of particles based on dielectrophoretic actuation," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 6, pp. 4764–4773, 2022.

[185] I. Matei, J. de Kleer, and M. Zhenirovskyy, "2D density control of micro-particles using kernel density estimation," *arXiv preprint arXiv:2209.03550*, 2022.

[186] I. Matei, A. Plochowietz, S. Nelaturi, J. De Kleer, J. P. Lu, L. S. Crawford, and E. M. Chow, "System and method for machine-learning enabled micro-assembly control with the aid of a digital computer," June 16 2022. US Patent App. 17/121,411.

[187] J. P. Lu, J. D. Thompson, G. L. Whiting, D. K. Biegelsen, S. Raychaudhuri, R. Lujan, J. Veres, L. L. Lavery, A. R. Völkel, and E. M. Chow, "Open and closed loop manipulation of charged microchiplets in an electric field," *Applied Physics Letters*, vol. 105, no. 5, pp. 054104–1–054104–4, 2014.

[188] COMSOL Multiphysics® v. 6.1., COMSOL AB, Stockholm, Sweden.

[189] H. P. McKean Jr, "A class of Markov processes associated with nonlinear parabolic equations," *Proceedings of the National Academy of Sciences*, vol. 56, no. 6, pp. 1907–1911, 1966.

[190] A.-S. Sznitman, "Topics in propagation of chaos," in *Ecole d'été de probabilités de Saint-Flour XIX—1989*, pp. 165–251, Springer, 1991.

[191] R. Carmona and F. Delarue, *Probabilistic theory of mean field games with applications I-II.* Springer, 2018.

[192] M. Kac, "Foundations of kinetic theory," in *Proceedings of The third Berkeley symposium on mathematical statistics and probability*, vol. 3, pp. 171–197, 1956.

[193] D. Lacker, "On a strong form of propagation of chaos for McKean-Vlasov equations," *Electronic Communications in Probability*, vol. 23, no. 45, pp. 1–11, 2018.

[194] M. Laborde, "On some nonlinear evolution systems which are perturbations of Wasserstein gradient flows," *Topological Optimization and Optimal Transport: In the Applied Sciences*, vol. 17, p. 304, 2017.

[195] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[196] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information theory*, vol. 39, no. 3, pp. 930–945, 1993.

[197] R. T. Rockafellar, "Augmented Lagrangians and applications of the proximal point algorithm in convex programming," *Mathematics of operations research*, vol. 1, no. 2, pp. 97–116, 1976.

[198] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM journal on control and optimization*, vol. 14, no. 5, pp. 877–898, 1976.

[199] H. H. Bauschke, P. L. Combettes, *et al.*, *Convex analysis and monotone operator theory in Hilbert spaces*, vol. 408. Springer, 2011.

[200] M. Teboulle, "Entropic proximal mappings with applications to nonlinear programming," *Mathematics of Operations Research*, vol. 17, no. 3, pp. 670–690, 1992.

[201] D. P. Bertsekas *et al.*, "Incremental gradient, subgradient, and proximal methods for convex optimization: A survey," *Optimization for Machine Learning*, vol. 2010, no. 1-38, p. 3, 2011.

[202] A. Halder, K. F. Caluya, P. Ojaghi, and X. Geng, "Stochastic uncertainty propagation in power system dynamics using measure-valued proximal recursions," *IEEE Transactions on Power Systems*, 2022.

[203] J.-D. Benamou, G. Carlier, and M. Laborde, "An augmented Lagrangian approach to Wasserstein gradient flows and applications," *ESAIM: Proceedings and surveys*, vol. 54, pp. 1–17, 2016.

[204] J. Karlsson and A. Ringh, "Generalized Sinkhorn iterations for regularizing inverse problems using optimal mass transport," *SIAM Journal on Imaging Sciences*, vol. 10, no. 4, pp. 1935–1962, 2017.

[205] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," ., 2017.

[206] D. Dua and C. Graff, "UCI machine learning machine learning repository," 2017.

[207] S. Visa, B. Ramsay, A. L. Ralescu, and E. Van Der Knaap, "Confusion matrix-based feature selection.," *MAICS*, vol. 710, pp. 120–127, 2011.

[208] S. Wojtowytsch and E. Weinan, "Can shallow neural networks beat the curse of dimensionality? a mean field training perspective," *IEEE Transactions on Artificial Intelligence*, vol. 1, no. 2, pp. 121–129, 2020.

[209] R. Novak, L. Xiao, J. Hron, J. Lee, A. A. Alemi, J. Sohl-Dickstein, and S. S. Schoenholz, "Neural tangents: Fast and easy infinite neural networks in python," in *International Conference on Learning Representations*, 2019.

[210] A. Wibisono, "Sampling as optimization in the space of measures: The Langevin dynamics as a composite optimization problem," in *Conference on Learning Theory*, pp. 2093–3027, PMLR, 2018.

[211] D. Alvarez-Melis, Y. Schiff, and Y. Mroueh, "Optimizing functionals on the space of probabilities with input convex neural networks," *arXiv preprint arXiv:2106.00774*, 2021.

[212] C. Kent, J. Li, J. Blanchet, and P. W. Glynn, "Modified Frank Wolfe in probability space," *Advances in Neural Information Processing Systems*, vol. 34, pp. 14448–14462, 2021.

[213] J. Fan, Q. Zhang, A. Taghvaei, and Y. Chen, "Variational Wasserstein gradient flow," in *International Conference on Machine Learning*, pp. 6185–6215, PMLR, 2022.

[214] Y. Wang and W. Li, "Accelerated information gradient flow," *Journal of Scientific Computing*, vol. 90, pp. 1–47, 2022.

[215] P. Dvurechenskii, D. Dvinskikh, A. Gasnikov, C. Uribe, and A. Nedich, "Decentralize and randomize: Faster algorithm for Wasserstein barycenters," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[216] F. Arqué, C. A. Uribe, and C. Ocampo-Martinez, "Approximate Wasserstein attraction flows for dynamic mass transport over networks," *Automatica*, vol. 143, p. 110432, 2022.

[217] M. Bowles and M. Agueh, "Weak solutions to a fractional Fokker–Planck equation via splitting and Wasserstein gradient flow," *Applied Mathematics Letters*, vol. 42, pp. 30–35, 2015.

[218] E. Bernton, "Langevin monte carlo and JKO splitting," in *Conference on learning theory*, pp. 1777–1798, PMLR, 2018.

[219] T. O. Gallouët and L. Monsaingeon, "A JKO splitting scheme for Kantorovich–Fisher–Rao gradient flows," *SIAM Journal on Mathematical Analysis*, vol. 49, no. 2, pp. 1100–1130, 2017.

[220] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics.* SIAM, 1989.

[221] R. Glowinski, T.-W. Pan, and X.-C. Tai, "Some facts about operator-splitting and alternating direction methods," *Splitting Methods in Communication, Imaging, Science, and Engineering*, pp. 19–94, 2016.

[222] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & mathematics with applications*, vol. 2, no. 1, pp. 17–40, 1976.

[223] K. Ito and K. Kunisch, "The augmented Lagrangian method for equality and inequality constraints in Hilbert spaces," *Mathematical programming*, vol. 46, no. 1-3, pp. 341–360, 1990.

[224] D. Butnariu and A. N. Iusem, *Totally convex functions for fixed points computation and infinite dimensional optimization*, vol. 40. Springer Science & Business Media, 2000.

[225] C. Kanzow, D. Steck, and D. Wachsmuth, "An augmented Lagrangian method for optimization problems in Banach spaces," *SIAM Journal on Control and Optimization*, vol. 56, no. 1, pp. 272–291, 2018.

[226] L. Yang, J. Li, D. Sun, and K.-C. Toh, "A fast globally linearly convergent algorithm for the computation of Wasserstein barycenters," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 984–1020, 2021.

[227] B. Lemmens and R. Nussbaum, *Nonlinear Perron-Frobenius Theory*, vol. 189. Cambridge University Press, 2012.

[228] F. Otto, "The geometry of dissipative evolution equations: the porous medium equation," 2001.

[229] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, vol. 87. Springer Science and Business Media, 2003.

[230] M. Hong, Z.-Q. Luo, and M. Razaviyayn, "Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 337–364, 2016.

[231] E. T. Bell, "The iterated exponential integers," *Annals of Mathematics*, pp. 539–557, 1938.

[232] R. Graham, D. Knuth, and O. Patashnik, "Concrete mathematics," 1988.

[233] M. Cuturi and A. Doucet, "Fast computation of wasserstein barycenters," in *International conference on machine learning*, pp. 685–693, PMLR, 2014.

[234] Y. Chen, "Density control of interacting agent systems," *arXiv preprint arXiv:2108.07342*, 2021.

[235] C. Sinigaglia, F. Braghin, and S. Berman, "Optimal control of velocity and nonlocal interactions in the mean-field Kuramoto model," in *2022 American Control Conference (ACC)*, pp. 290–295, IEEE, 2022.

[236] T. Zheng, Q. Han, and H. Lin, "Backstepping mean-field density control for large-scale heterogeneous nonlinear stochastic systems," in *2022 American Control Conference (ACC)*, pp. 4832–4837, IEEE, 2022.

[237] J. Sirignano and K. Spiliopoulos, "Mean field analysis of deep neural networks," *Mathematics of Operations Research*, vol. 47, no. 1, pp. 120–152, 2022.

[238] J. Yu and K. Spiliopoulos, "Normalization effects on deep neural networks," *Foundations of Data Science*, vol. 5, no. 3, pp. 389–465, 2023.

[239] D. Araújo, R. I. Oliveira, and D. Yukimura, "A mean-field limit for certain deep neural networks," *arXiv preprint arXiv:1906.00193*, 2019.

[240] P.-M. Nguyen, "Mean field limit of the learning dynamics of multilayer neural networks," *arXiv preprint arXiv:1902.02880*, 2019.

[241] C. Léonard, "From the schrödinger problem to the monge–kantorovich problem," *Journal of Functional Analysis*, vol. 262, no. 4, pp. 1879–1920, 2012.

[242] I. Haasler, A. Ringh, Y. Chen, and J. Karlsson, "Multimarginal optimal transport with a tree-structured cost and the schrodinger bridge problem," *SIAM Journal on Control and Optimization*, vol. 59, no. 4, pp. 2428–2453, 2021.

[243] G. Carlier, "On the linear convergence of the multimarginal sinkhorn algorithm," *SIAM Journal on Optimization*, vol. 32, no. 2, pp. 786–794, 2022.

[244] T. Chen, G.-H. Liu, M. Tao, and E. A. Theodorou, "Deep momentum multimarginal schr\" odinger bridge," *arXiv preprint arXiv:2303.01751*, 2023.

[245] D. Williams, *Probability with martingales.* Cambridge university press, 1991.

[246] K. Oelschläger, "A law of large numbers for moderately interacting diffusion processes," *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, vol. 69, no. 2, pp. 279–322, 1985.

[247] P. Billingsley, *Convergence of probability measures.* John Wiley & Sons, 2013.

[248] G. A. Pavliotis, *Stochastic processes and applications: diffusion processes, the Fokker-Planck and Langevin equations*, vol. 60. Springer, 2014.

[249] V. I. Bogachev, N. V. Krylov, M. Röckner, and S. V. Shaposhnikov, *Fokker-Planck-Kolmogorov Equations*, vol. 207. American Mathematical Soc., 2015.

[250] H. Bauschke and S. Kruk, "Reflection-projection method for convex feasibility problems with an obtuse cone," *Journal of Optimization Theory and Applications*, vol. 120, no. 3, pp. 503–531, 2004.