Proximal Optimal Transport Modeling of Population Dynamics

Charlotte Bunne ETH Zurich
Laetitia Meng-Papaxanthos Google Research
Andreas Krause ETH Zurich
Marco Cuturi Google Research[‡]

Abstract

We propose a new approach to model the collective dynamics of a population of particles evolving with time. As is often the case in challenging scientific applications, notably single-cell genomics, measuring features for these particles requires destroying them. As a result, the population can only be monitored with periodic snapshots, obtained by sampling a few particles that are sacrificed in exchange for measurements. Given only access to these snapshots, can we reconstruct likely individual trajectories for all other particles? We propose to model these trajectories as collective realizations of a causal Jordan-Kinderlehrer-Otto (JKO) flow of measures: The JKO scheme posits that the new configuration taken by a population at time $t+1$ is one that trades off an improvement, in the sense that it decreases an *energy*, while remaining close (in Wasserstein distance) to the previous configuration observed at t. In order to learn such an energy using only snapshots, we propose JKONET, a neural architecture that computes (in end-to-end differentiable fashion) the JKO flow given a parametric energy and initial configuration of points. We demonstrate the good performance and robustness of the JKONET fitting procedure, compared to a more direct forward method.

1 Introduction

Population Dynamics ... Many fields in science carry out experiments by monitoring complex systems composed of evolving particles. That monitoring consists in sampling, every now and then, a few representative particles in the system, and measure their features.

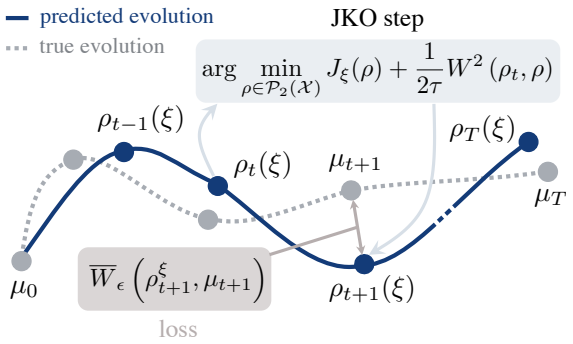


Figure 1: Given an observed trajectory $(\mu_0,\dots,\mu_T)$ of point clouds (gray), we seek parameters ξ for the energy J_ξ such that the predictions $\rho_1,\dots,\rho_T$ (blue) following a JKO flow from $\rho_0=\mu_0$ are close the observed trajectory (gray), by minimizing (as a function of ξ) the sum of Wasserstein distances between ρ_{t+1}, the JKO step from ρ_{t-1} using J_ξ, and data μ_{t+1}.

As a result, the observer has access to a collection of time-resolved point-clouds describing partially the dynamic of that population on aggregate. Such problems arise in many fields, when for instance, observing a population of cells in biology (Schiebinger et al., 2019; Moon et al., 2019), densities in meteorology (Fisher et al., 2009; Sigrist et al., 2015) or multi-target tracking (Luo et al., 2020; Sheldon et al., 2007; Sheldon and Dietterich, 2011; Haasler et al., 2019, 2021a,b).

... Without Individual Paths. While modeling and estimating parametric dynamics using datasets of point trajectories is the core subject of time series analysis (see Li et al. 2020; Krishnan et al. 2017 and references therein), the setting we consider makes it difficult to track the evolution of individual particles. Indeed, this would require tagging and measuring repeatedly the same particles, which can be costly or even impossible: For instance, measuring a cell's transcriptome requires splitting the cell o. With this constraint in mind, our goal is to better understand the evolution of single particles, using only the aggregate data described in point clouds.

[‡]Now at Apple. Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

Inferring Particle Paths from Cloud Trajectories. When the observer only seeks to reconstruct particles' paths given starting and ending point cloud configurations, the machinery of optimal transport (OT) (Schiebinger et al., 2019; Yang et al., 2020; Yang and Uhler, 2019) or likelihood-based normalizing flows (NF) (Rezende and Mohamed, 2015; Grathwohl et al., 2019) can be used, either separately, or even combined: Tong et al. (2020) use OT to motivate a regularizer (squared norm of displacements) in their NF estimation pipeline; Huang et al. (2021) restrict their attention to flows expressed as gradients of convex functions. This choice is motivated by OT because it agrees with the Brenier (1987) principle that displacements arising from convex potentials give rise to optimal flows. When the observer seeks instead a *causal model*, namely one that is able to explain/predict future configurations of the point cloud (and not only interpolate between configurations), the parameters of that model can also be fitted with OT, as proposed by Hashimoto et al. (2016). Their model assumes a Langevin dynamic for the particles, driven by the gradient flow of a (neural) energy function; They fit the parameters of that network by minimizing regularized OT distances (Cuturi, 2013) between their model's predictions and the corresponding ground truth snapshots.

Modeling Particle Dynamics as a JKO Scheme. In this paper, we draw inspiration from both approaches above—the intuition from the recent NF literature that flows should mimic an optimal transport (OT as prior), and be able, through training, to predict future configurations (OT as a loss)—to propose a causal model for population dynamics. Our approach relies on a powerful hammer: the Jordan-Kinderlehrer-Otto (JKO) flow (Jordan et al., 1998), widely regarded as one of the most influential mathematical breakthroughs in recent history. While the JKO flow was initially introduced as an alternative method to solve the Fokker-Planck partial differential equation (PDE), its flexibility can be showcased to handle more complex PDEs (Santambrogio, 2017, §4.7), or even describe the gradient flows of non-differentiable energies that have no PDE representation. On a purely mechanical level, a JKO step is to measures what the proximal step (Combettes and Pesquet, 2011) is to vectors: In a JKO step, particles move to decrease collectively an *energy* (a real-valued function defined on measures), yet remain close (in Wasserstein sense) to the previous configuration. Our goal in this paper is to treat JKO steps as parameterized modules, and fit their parameter (the energy function) so that its outputs agree repeatedly over time with observed data. This approach presents several challenges: While numerical approaches to solve JKO steps have been proposed in low dimensional settings (Burger et al., 2010; Carrillo et al., 2021; Peyré, 2015; Benamou et al., 2016a), scaling it to higher dimensions is an open problem. Moreover, minimizing a loss involving a JKO step w.r.t. energy requires not only solving the JKO problem, but also computing the (transpose) Jacobian of its output w.r.t. energy parameters.

Contributions. Our contributions are two-fold. First, we propose a method, given an input configuration and an energy function, to compute JKO steps using input convex neural networks (ICNN) (Amos et al., 2017; Makkuva et al., 2020) (see also concurrent works that have proposed similar approaches (Alvarez-Melis et al., 2021; Mokrov et al., 2021)). Second, we view the JKO step as an inner layer, a JKONET module parameterized by an energy function, which is tasked with moving the particles of an input configuration along an OT flow (the gradient of an optimal ICNN), trading off a lower energy with proximity to the previous configuration. We propose to estimate the parameters of the energy by minimizing a fitting loss computed between the outputs of the JKONET module (the prediction) and the ground truth displacements, as illustrated in Figure 1. We demonstrate JKONET's range of applications by applying in on synthetic potential- and trajectory-based population dynamics, as well as developmental trajectories of human embryonic stem cells based on single-cell genomics data.

2 Background

Optimal Transport. For two probability measures μ, ν in $\mathcal{P}(\mathbb{R}^d)$, their squared 2-Wasserstein distance is

$$W_2^2(\mu,\nu) = \inf_{\gamma\in\Gamma(\mu,\nu)} \iint \|x-y\|_2^2 \gamma(dx,dy), \qquad (1)$$

where $\Gamma(\mu,\nu)$ is the set of couplings on $\mathbb{R}^d \times \mathbb{R}^d$ with respective marginals μ, ν. When instantiated on finite discrete measures, such as $\mu = \sum_{i=1}^n a_i \delta_{x_i}$ and $\nu = \sum_{j=1}^m b_j \delta_{y_j}$, this problem translates to a linear program, which can be regularized using an entropy term (Cuturi, 2013; Peyré and Cuturi, 2019). For $\varepsilon \geq 0$, set

$$W_\varepsilon(\mu,\nu) := \min_{\mathbf{P}\in U(a,b)} \langle \mathbf{P}, [\|x_i - y_j\|^2]_{ij} \rangle - \varepsilon H(\mathbf{P}), \quad (2)$$

where $H(\mathbf{P}) := -\sum_{ij} \mathbf{P}_{ij}(\log \mathbf{P}_{ij} - 1)$ and the polytope $U(a,b)$ is the set of $n \times m$ matrices $\{\mathbf{P} \in \mathbb{R}_+^{n\times m}, \mathbf{P}\mathbf{1}_m = a, \mathbf{P}^\top \mathbf{1}_n = b\}$. Notice that the definition above reduces to the usual (squared) 2-Wasserstein distance when $\varepsilon = 0$. Setting $\varepsilon > 0$ yields a faster and differentiable proxy to approximate W_0, but introduces a bias, since $W_\varepsilon(\mu,\mu) \neq 0$ in general. In the rest of this work, we therefore use the *Sinkhorn divergence* (Ramdas et al., 2017; Genevay et al., 2019; Salimans et al., 2018; Feydy et al., 2019) as a valid non-negative discrepancy,

$$\overline{W}_\varepsilon(\mu,\nu) := W_\varepsilon(\mu,\nu) - \frac{1}{2}\left(W_\varepsilon(\mu,\mu) + W_\varepsilon(\nu,\nu)\right). \quad (3)$$

OT and Convexity. An alternative formulation for OT is given by the Monge (1781) problem

$$W_2^2(\mu,\nu) = \inf_{T:T_{\#}\mu=\nu} \int_{\mathcal{X}} ||x - T(x)||^2 d\mu(x) \quad (4)$$

where # is the push-forward operator, and the optimal solution $T^\star$ is known as the Monge map between μ and ν. The Brenier theorem (1987) states that if μ has a density, the Monge map $T^\star$ between μ and ν can be recovered as the gradient of a unique (up to constants) convex function ψ whose gradient pushes forward μ to ν. Namely, if $\psi : \mathbb{R}^d \to \mathbb{R}$ is convex and $(\nabla\psi)_{\#}\mu = \nu$, then $T^\star(x) = \nabla\psi(x)$ and

$$W_2^2(\mu,\nu) = \int_{\mathcal{X}} ||x - \nabla\psi(x)||^2 d\mu(x)\,. \quad (5)$$

JKO Flows. In their seminal paper, Jordan et al. (1998) study diffusion processes under the lens of the OT metric (see also Ambrosio et al., 2006) and introduce a scheme that is now known as the JKO flow: Starting with ρ_0, and given a real-valued energy function $J : \mathcal{P}(\mathbb{R}^d) \to \mathbb{R}$ driving the evolution of the system, they define iteratively for $t \geq 0$, :

$$\rho_{t+1} = \arg\min_{\rho\in\mathcal{P}_2(\mathbb{R}^d)} J(\rho) + \frac{1}{2\tau} W^2(\rho,\rho_t)\,, \quad (6)$$

where τ is a time step parameter. These successive minimization problems result in a sequence of probability measures in $\mathcal{P}(\mathbb{R}^d)$. The JKO flow can thus be seen as the analogy of the usual proximal descent scheme, tailored for probability measures (Santambrogio, 2015, p.285). Jordan et al. (1998) show that as step size $\tau \to 0$, and for a specific energy J that is the sum of a linear term and the negentropy, the measures describing the JKO flow recover solutions to a Fokker-Planck equation. In this work, following in the footsteps of more general applications of the JKO scheme (Santambrogio, 2017, §4.8), we model dynamics without necessarily having in mind PDE solutions in mind, to interpret instead the JKO step as a more general parametric type of dynamic for probability measures, exclusively parameterized by the energy J itself.

Convex Neural Architectures. Input convex neural networks are neural networks $\psi_\theta(x)$ with specific constraints on the architecture and parameters θ, such that their output is a convex function of some (or all) elements of the input x (Amos et al., 2017). We consider in this work *fully* input convex neural networks (ICNNs), such that the output is a convex function of the entire input x. A typical ICNN is a L-layer, fully connected network such that, for $l = 0,\dots,L-1$:

$$z_{l+1} = a_l(W_l^x x + W_l^z z_l + b_l) \text{ and } \psi_\theta(x) = z_L, \quad (7)$$

where by convention, z_0 and W_0^z are 0, a_l are convex non-decreasing (non-linear) activation functions, $\theta = \{b_l, W_l^z, W_l^x\}_{l=0}^{L-1}$ are the weights and biases of the neural network, with weight matrices W_l^z associated to latent representations z that have non-negative entries. Since Amos et al. (2017)'s work, convex neural architectures have been further extended and shown to capture relevant models despite these constraints (Amos et al., 2017; Makkuva et al., 2020; Huang et al., 2021). In particular, Chen et al. (2019) provide a theoretical analysis that any convex function over a convex domain can be approximated in sup norm by an ICNN.

3 Proximal Optimal Transport Model

Given T discrete measures $\mu_0,\dots,\mu_T$ describing the time evolution of a population, we posit that such an evolution follows a JKO flow for the free energy functional J, and assume that energy does not change throughout the dynamic. We parameterize the energy J as a neural network with parameters ξ, and fit ξ so that the JKO flow model matches the observed data.

Fitting parameter ξ with a reconstruction loss requires, using the chain rule, being able to differentiate the JKO step's output w.r.t. ξ (see Fig. 1), and more precisely provide a way to apply that transpose Jacobian to an arbitrary vector when using reverse-mode differentiation. To achieve this, we introduce a novel approach to numerically solve JKO flows using ICNNs (§ 3.1), resulting in a bilevel optimization problem targeting the energy J_ξ (§ 3.2).

3.1 Reformulation of JKO Flows via ICNNs

Given a starting condition ρ_t and energy functional J_ξ, the JKO step consists in producing a new measure ρ_{t+1} implicitly defined as the minimizer of (6). Solving directly (6) on the space of measures, involves substantial computational costs. Different numerical schemes have been developed, e.g., based notably on Eulerian discretization of measures (Carrillo et al., 2021; Benamou et al., 2016b), and/or entropy-regularized optimal transport (Peyré, 2015). However, these methods are limited to small dimensions since the cost of discretizing such spaces grows exponentially. Except for the Eulerian approach proposed in (Peyré, 2015), obtained as the fixed point of a Sinkhorn type iteration, the differentiation would also prove extremely challenging as a function of the energy parameter ξ.

To reach scalability and differentiability, we build upon the approach outlined in Benamou et al. (2016b) to reformulate the JKO scheme as a problem solved over convex functions, rather than on measures ρ. Effectively, this is equivalent to making a change of variables

in (6): Introduce a (variable) convex function ψ, and replace the variable ρ by the variable $\nabla\psi_{\#}\rho_t$. Writing

$$\mathcal{E}_J(\rho,\nu) := J(\rho) + \frac{1}{2\tau}W_2^2(\rho,\nu), \tag{8}$$

this identity states that, assuming μ and ν being absolutely continuous w.r.t. Lebesgue measure that

$$\min_{\rho} \mathcal{E}_J(\rho,\nu) = \min_{\psi \text{ convex}} \mathcal{F}_J(\psi,\nu) := \mathcal{E}_J(\nabla\psi_{\#}\nu,\nu)\,,$$

simplifying the Wasserstein term in (8), using the assumption that ψ is convex and Brenier's theorem (§ 1):

$$\mathcal{F}_J(\psi,\nu) = J(\nabla\psi_{\#}\nu) + \frac{1}{2\tau}\int \|x-\nabla\psi(x)\|^2 d\nu(x) \tag{9}$$

We pick an ICNN architecture to optimize over a restricted family of convex functions, $\{\psi_\theta\}$, and define, starting from $\rho_0(\xi) := \mu_0$, the recursive sequence for $t \geq 0$,

$$\rho_{t+1}(\xi) := \nabla\psi_{\theta^\star(\xi,\rho_t(\xi))\,\#}\,\rho_t(\xi)\,, \tag{10}$$

with $\theta^\star(\xi,\rho_t)$ defined implicitly using ξ and any ν as

$$\theta^\star(\xi,\nu) := \arg\min_{\theta} \mathcal{F}_J(\psi_\theta,\nu) \tag{11}$$

Strong Convexity of ψ_θ. The strong convexity and smoothness of a potential ψ impacts the regularity of the corresponding OT map $\nabla\psi$ (Caffarelli, 2000; Figalli, 2010), since one can show that for a ℓ-strongly convex, L-smooth ψ one has (Paty et al., 2020) that

$$\ell\|x-y\| \leq \|\nabla\psi(x) - \nabla\psi(y)\| \leq L\|x-y\|.$$

While it is more difficult to enforce the L-smoothness of a neural network, and more generally its Lipschitz constants (Scaman and Virmaux, 2018) it is easy to enforce its strong convexity, by simply adding a term $\ell\|x\|^2/2$ to the corresponding potential, or a residual rescaled term ℓx to the output $\nabla\psi(x)$. This approach can be used to enforce that the push-forward of the gradient of an ICNN does not collapse to a single point, maintaining spatial diversity.

3.2 Learning the Free Energy Functional

The energy function $J_\xi : \mathcal{P}(\mathbb{R}^d) \to \mathbb{R}$ can be any parameterized function taking a measures as an input. Since our model assumes that the observed dynamic is parameterized entirely by that energy (and the initial observation ρ_0), the more complex this dynamic, the more complex one would expect the energy J_ξ to be. We focus in this first attempt on linear functions in the space of measures, that is expectations over ρ of a vector-input neural network E_ξ

$$J_\xi(\rho) := \int E_\xi(x) d\rho(x), \tag{12}$$

where $E_\xi : \mathbb{R}^d \to \mathbb{R}$ is a multi-layer perceptron (MLP).

Algorithm 1 JKONET Algorithm.

Input: Dataset $\mathcal{D} = \{\{\mu_t^0\}_{t=0}^T, \ldots, \{\mu_t^N\}_{t=0}^T\}$ of N population trajectories, ξ^0 energy parameter initialization, θ^0 ICNN parameter initialization, learning rates lr_θ and lr_ξ, step τ, regularizer ε, tolerance α, `TeacherForcing` flag.

Output: Free energy J_ξ explaining underlying population dynamics of snapshot data.

$\xi \leftarrow \xi^0$
for $\{\mu_t\}_{t=0}^T \in \mathcal{D}$ **do**
  for $t \leftarrow 0$ **to** $T-1$ **do**
    $\theta \leftarrow \theta^0$
    if `TeacherForcing` **then**
      $\nu \leftarrow \mu_t$
    else
      $\nu \leftarrow \rho_t(\xi)$
    while $\frac{\sum_i \|\nabla_{\theta_i}\mathcal{F}_{J_\xi}(\theta)\|_2}{\sum_i \mathrm{count}(\theta_i)} \geq \alpha$ **do**
      $\theta \leftarrow \theta - \mathrm{lr}_\theta \times \nabla_\theta \mathcal{F}_{J_\xi,\nu}(\theta)$
    $\rho_{t+1}(\xi) \leftarrow \nabla\psi_{\theta\#}\nu$
    $\xi \leftarrow \xi - \mathrm{lr}_\xi \times \nabla_\xi \overline{W}_\varepsilon(\rho_{t+1}(\xi), \mu_{t+1})$
return J_ξ

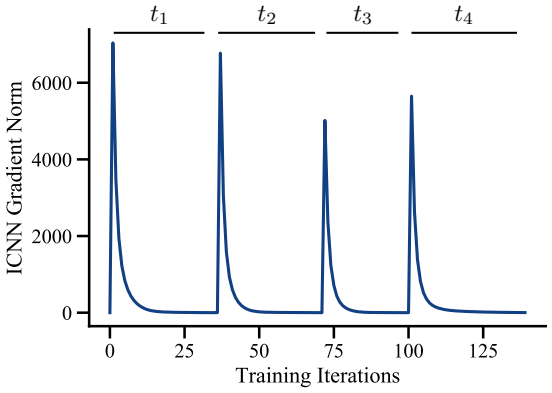


Figure 2: Optimization of the ICNN used in JKO steps. The bumps correspond to a change in the outer iteration, the smooth decrease in between correspond to a single minimization (11) of a time step t_i.

Inferring nonlinear energies accounting for population growth and decline, as well as interactions between points, using the formalism of (De Bie et al., 2019), transformers (Vaswani et al., 2017) or set pooling methods (Edwards and Storkey, 2017; Zaheer et al., 2017), is an exciting direction for future work.

To address slow convergence and instabilities for dynamics with many snapshots, we use teacher forcing (Williams and Zipser, 1989) to learn J_ξ through time. In those settings, during training, J_ξ uses the ground truth as input instead of predictions from the previous time step. At test time, we do not use teacher forcing.

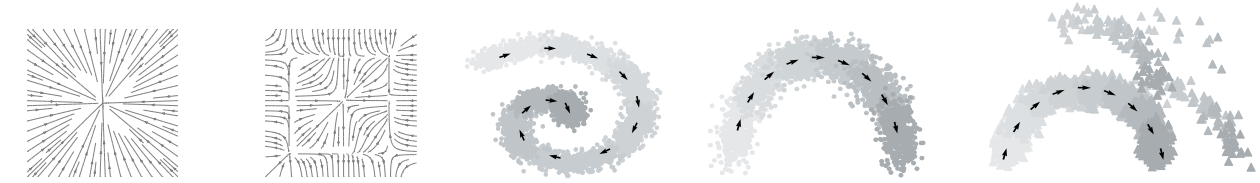
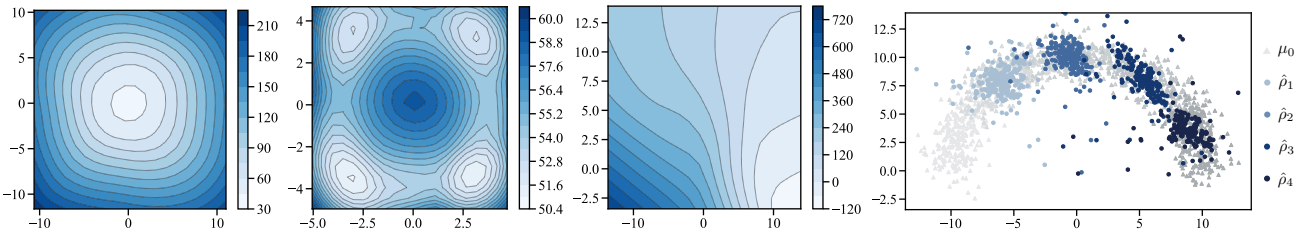


Figure 3: Overview on different tasks including trajectory- and potential-based dynamics.



(a) Quadratic Potential. (b) Styblinski Potential. (c) Semicircle Trajectory *with* teacher forcing. (d) Predicted Population Evolution on Semicircle Trajectory.

Figure 4: **Results of JKOnet on Potential- and Trajectory-based Dyanamics.** (a)-(c) Contour plots of the energy functionals J_ξ of JKONET on potential- and trajectory-based population dynamics in different training settings (i.e., trained with or without teacher forcing § 3.2), color gradients depict the magnitude of J_ξ. (d) Predicted population snapshots $(\hat{\rho}_1, \dots, \hat{\rho}_4)$ (blue) and data trajectory $(\mu_0, \dots, \mu_4)$ (gray).

3.3 Bilevel Formulation of JKONET

Learning the free energy functional J_ξ while solving each JKO step via an ICNN results in a challenging bilevel optimization problem. At each time step, the predicted dynamics are compared to the ground truth trajectory $(\mu_0, \mu_1, \dots, \mu_T)$ with a Sinkhorn loss (3),

$$\begin{aligned} \min_{\xi} & \sum_{t=0}^{T-1} \overline{W}_\varepsilon(\rho_{t+1}(\xi), \mu_{t+1}), \\ \text{s.t. } & \rho_0(\xi) := \mu_0, \\ & \rho_{t+1}(\xi) := \nabla\psi_{\theta^\star} \# \rho_t(\xi)\,, \\ & \theta^\star := \arg\min_{\theta} \mathcal{F}_{J_\xi}(\psi_\theta, \rho_t(\xi)) \end{aligned} \tag{13}$$

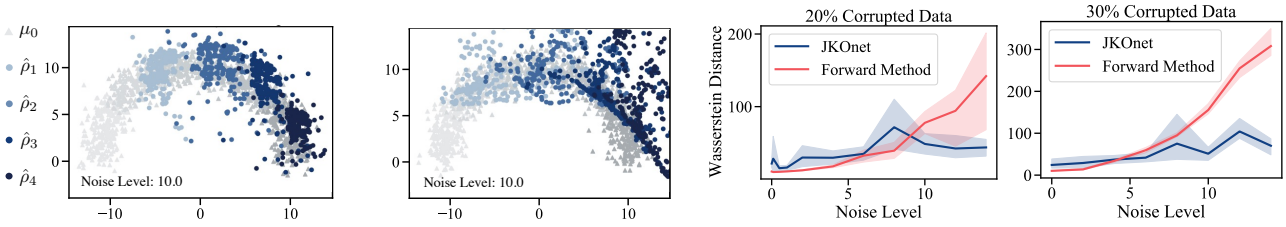
The dependence of the Sinkhorn divergence losses in (13) on ξ only appears in the fact that the predictions $\rho_{t+1}(\xi)$ are themselves implicitly defined as solving a JKO step parameterized with the energy J_ξ. Learning J_ξ through the exclusive supervision of data observations requires therefore to differentiate the arg-minimum of a JKO problem, down therefore through to the lower-level optimization of the ICNN. We achieve this by implementing a differentiable double loop in `JAX`, differentiating first the Sinkhorn divergence using the `OTT`[1] package (Cuturi et al., 2022), and then backpropagating through the ICNN optimization by unrolling Adam steps (Kingma and Ba, 2014; Metz et al., 2017; Lorraine et al., 2020).

[1] github.com/ott-jax/ott

Inner Loop Termination. A question that arises when defining $\rho_{t+1}(\xi)$ lies in the budget of gradient steps needed or allowed to optimize the parameters θ of the ICNN, before taking a new gradient step on ξ in the outer loss. A straightforward approach in `JAX` (Bradbury et al., 2018) would be to use a preset number of iterations with a `for` loop (`jax.lax.scan`). We do observe, however, that the number of iterations needed to converge in relevant scenarios can vary significantly with the ICNN architecture and/or the hardness of the underlying task. We propose to use instead a differentiable fixed-point loop to solve each JKO step up to a desired convergence threshold. We measure convergence of the optimization of the ICNN via the average norm of the gradient of the JKO objective w.r.t. the ICNN parameters θ, i.e., $\sum_i \left\|\nabla_{\theta_i} \mathcal{F}_{J_\xi}(\theta_i, \xi)\right\|_2 / \sum_i \text{count}(\theta_i)$. We observe that this approach is robust across datasets and architectures of the ICNN. An exemplary training curve for the ICNNs updated successively along a time sequence is shown in Figure 2.

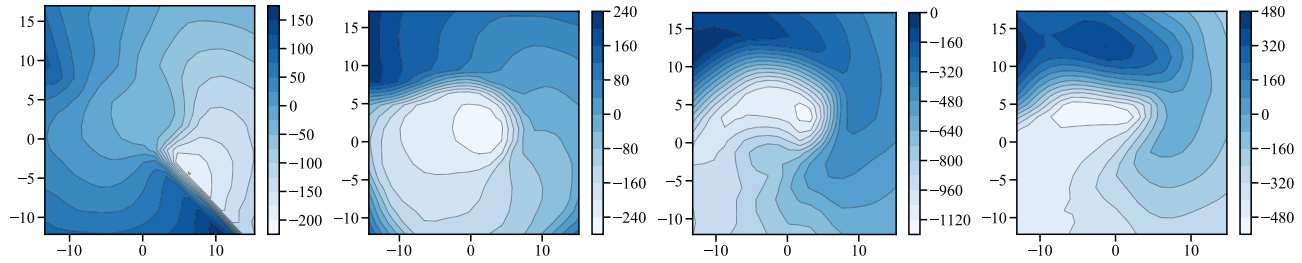
Reverse-Mode Differentiation. The Jacobian $\partial\rho_{t+1}/\partial\xi$ arising when computing the gradient $\nabla_\xi \overline{W}_\varepsilon(\rho_{t+1}(\xi), \mu_{t+1})$ is obtained by unrolling the while loop above. The gradient term of the Sinkhorn divergence w.r.t the first argument is given by the Danskin envelope theorem (Danskin, 1967).

Setting τ in (9). In usual JKO applications, τ needs to be tuned manually. In this work, the energy J_ξ is not fixed, but trained to fit data. Since we put



(a) JKONET on 30% corrupted data. (b) Forward method on 30% corrupted data. (c) W_ϵ (2) vs. noise level on 20% corrupted data. (d) W_ϵ (2) vs. noise level on 30% corrupted data.

Figure 5: Comparison between JKONET and the forward method in settings of increasing noise on corrupted data on the semicircle trajectory task.



(a) Forward method, *with* teacher forcing. (b) Forward method, *no* teacher forcing. (c) JKONET, *with* teacher forcing. (d) JKONET, *no* teacher forcing.

Figure 6: Comparison between energy functionals J_ξ of the spiral trajectory task (see 3) between the forward method and JKONET, trained with or without teacher forcing § 3.2). When using teacher forcing, the forward method overfits a gap on the lower-right corner of the spiral, outputting a highly irregular energy. When taking into account the entire trajectory recursively, the Forward method does better overall, but is unable to recover an energy as precise as that returned by JKONET.

no constraints on the scaling of J_ξ, τ can be set to 1 without loss of generality, as the parameter ξ will automatically adjust so that the scale of J_ξ induces steps of a relevant length to fit data. This only holds (as with a usual JKO step) if the trajectories are sampled regularly. For irregularly spaced time series, τ can be adapted at train and test time to the spacing of timestamps (shorter steps requiring larger τ).

4 Evaluation

In the following, we evaluate our method empirically on a variety of tasks. This includes recovering synthetic potential- and trajectory-based population dynamics (see Fig. 3), as well as the evolution of high-dimensional single-cell populations during a developmental process.

4.1 Synthetic Population Dynamics

Energy-Driven Trajectories. The first task involves evolutions of partial differential equations with known potential. We hereby consider both convex (e.g., the quadratic function $J(x) = \|x\|_2^2$) and nonconvex potentials (e.g., Styblinski function) (see Fig. 3). These two-dimensional synthetic flows are generated using the Euler-Maruyama method (Kloeden and Platen, 1992). For details, see § B.1. To recover the true potential via JKONET, we parameterize both energy J_ξ and ICNN ψ_θ with linear layers ($\epsilon = 1.0$, $\tau = 1.0$, § C.3). More details on the architectures can be found in § C.2. Figure 4a-b demonstrate JKONET's ability to recover convex and nonconvex potentials via energy J_ξ.

Arbitrary Trajectories. As a sanity check, we evaluate if JKONET can recover an energy functional J_ξ from trajectories that are not necessarily arising from the gradient of an energy. Here, a 2-dimensional Gaussian moves along a predefined trajectory with nonconstant speed. For details on the data generation, see § B.2. We consider a line, a spiral, and movement along a semicircle (Fig. 3). As visible in Figure 4c (5 snapshots), Figure 10b (2 snapshots), and Figure 6c-d (10 snapshots), JKONET learns energy functionals J_ξ that can then model the ground truth trajectories. These trajectory-based dynamics are learned using the strong convexity regularizer ($\ell = 0.8$, see § 3.1).

Table 1: Evaluation of predictive performance w.r.t. the entropy-regularized Wasserstein distance W_ε (2) of JKONET and the forward method on the embryoid body scRNA-seq data per time step (using 3 runs).

Method	**Prediction Loss** (W_ε)			
	Day 6 to 9	Day 12 to 15	Day 18 to 21	Day 24 to 27
One-Step Ahead				
Forward Method	0.187 ± 0.001	0.162 ± 0.010	0.185 ± 0.020	0.203 ± 0.004
JKONET	$\mathbf{0.133 \pm 0.020}$	$\mathbf{0.133 \pm 0.008}$	$\mathbf{0.172 \pm 0.0130}$	$\mathbf{0.169 \pm 0.004}$
All-Steps Ahead				
Forward Method	0.225 ± 0.023	0.160 ± 0.001	0.171 ± 0.016	0.183 ± 0.007
JKONET	$\mathbf{0.148 \pm 0.015}$	$\mathbf{0.144 \pm 0.013}$	$\mathbf{0.154 \pm 0.024}$	$\mathbf{0.138 \pm 0.034}$

Comparison to Forward Methods. Instead of parameterizing the next iteration $\rho_{t+1}(\xi)$ as we do in the JKONET formulation (6), the *forward* scheme states that the prediction at time $t+1$, η_{t+1}, can be obtained as $(\nabla F_\xi)_\# \eta_t(\xi)$, where F_ξ is any arbitrary neural network, as considered in Hashimoto et al. (2016), namely $\eta_0 := \mu_0$ and subsequently $\eta_{t+1}(\xi) := (\nabla F_\xi)_\# \eta_t(\xi)$. Although OT still plays an important role in that paper, since the potential F is estimated by minimizing a Sinkhorn loss $\overline{W}_\varepsilon(\eta_{t+1}, \mu_{t+1})$, as we do in (13), the forward displacement operator $(\nabla F_\xi)_\#$ has no spatial regularity. Because of that, we observe that the forward method can get more easily trapped in local minima, and, in particular, overfits the training data (see § A.2) as shown by a substantial decrease in performance in the presence of noise. We demonstrate this in different scenarios: First, we compare the robustness of both JKONET and the forward method to noise. For this, we corrupt 20% or 30% of the training data on the example of the semicircle trajectory with different levels of noise (see Fig. 3). We insist that noise is only added at training time, as random shifts on both feature dimensions, while we test on the original semicircle trajectory. In low noise regimes, where train and test data are similar, the forward method overfits and performs marginally better than JKONET (see Fig. 5c,d). As noise increases, the performance of the forward method deteriorates (Fig. 5b), while JKONET, constrained to move points with OT maps, is robust (Fig. 5a).

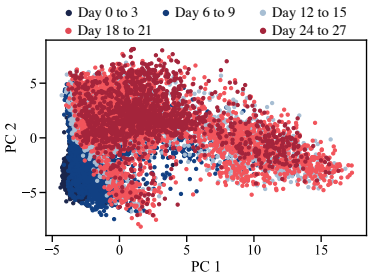
In a second experiment, we evaluate the capacity of JKONET and the forward method to extrapolate and generalize the learned trajectories, e.g., when vertically translating a line during test time (Fig. 11). Due to the less constrained energy, the *forward* method perfectly resembles the seen trajectory during training, but fails to extrapolate to shifted test data (Table 3 in § A.2).

Lastly, we compare the resulting energy functionals F_ξ and J_ξ of the forward method and JKONET, respectively, on the spiral trajectory (see Fig. 6). When learning long and complex population dynamics, teacher forcing improves training (see additional results in Fig. 8c-d as well as Fig. 4c-d). While facilitating training of the forward method in some settings, it likewise results in wrong energy functionals F_ξ (Fig. 6a). JKONET, on the other hand, is able to globally learn the energy functional J_ξ, despite being only exposed to a one-step history of snapshots during training with teacher forcing (see Fig. 6c).

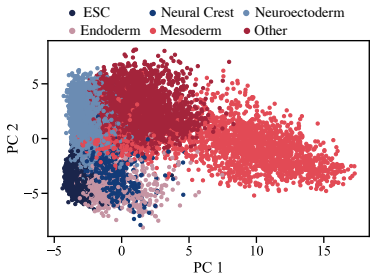
4.2 Single-Cell Population Dynamics

We investigate the ability of JKONET to predict the evolution of cellular and molecular processes through time. The advent of single cell profiling technologies has enabled the generation of high-resolution single-cell data, making it possible to profile individual cells at different states in the development. A key difficulty in learning the evolution of cell populations is that a cell is (usually) destroyed during a measurement. Thus, although one is able to collect features at the level of individual cells, the same cell cannot be measured twice. Instead, we collect independent samples at each snapshot, resulting in *unaligned* distributions across snapshots, without access to ground-truth single-cell trajectories. The goal of learning individual dynamics is to identify ancestor and descendant cells, and get a better understanding of biological differentiation or reprogramming mechanisms.

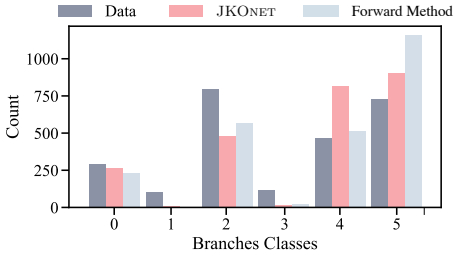
We apply JKONET to embryoid body single-cell RNA sequencing (scRNA-seq) data (Moon et al., 2019), describing the differentiation of human embryonic stem cells grown as embryoid bodies into diverse cell lineages over a period of 27 days. During this time, cells are collected at 5 different snapshots (day 1 to 3, day 6 to 9, day 12 to 15, day 18 to 21, day 24 to 27) and measured via scRNA-seq (resulting in 15,150 cells). For details on the dataset and data preprocessing see § B.3. We run JKONET as well as the baseline on the first 20 components of a principal component analysis (PCA) of the 4000 highly differentiable genes (see Fig. 12). We split the dataset into train and test data ($\sim 15\%$) and parameterize both energy J_ξ and ICNN ψ_θ with linear layers ($\epsilon = 1.0$, $\tau = 1.0$, § C.3).



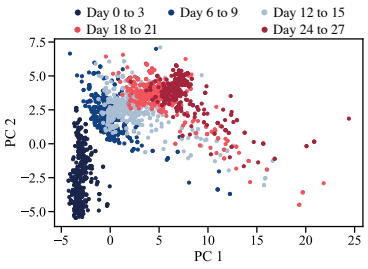
(a) PCA embedding of the embryoid body scRNA-seq data colored by the snapshot time.



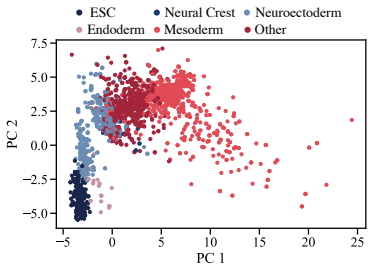
(b) PCA embedding of the embryoid body scRNA-seq data colored by the lineage branch class.



(c) Distribution of cell lineage branch classes in the data or predicted by JKOnet or the forward method.



(d) PCA embedding of JKOnet predictions colored by the snapshot time.



(e) PCA embedding of JKOnet predictions colored by the lineage branch class.

Figure 7: Analysis of population dynamics predictions of JKOnet on the embryoid body scRNA-seq data.

Table 2: Evaluation of cell lineage branch classification performance of JKOnet and the forward method on the embryoid body scRNA-seq data based on the ℓ_1-distance of the histograms and the Hellinger distance H^2 (14) of the predicted branch class distributions (using 3 runs).

Method	Cell Lineage Classification	
	ℓ_1	H^2
One-Step Ahead		
Forward Method	132.27 ± 5.00	0.026 ± 0.002
JKOnet	$\mathbf{88.80 \pm 0.57}$	$\mathbf{0.016 \pm 0.001}$
All-Steps Ahead		
Forward Method	$\mathbf{185.47 \pm 12.18}$	0.033 ± 0.002
JKOnet	215.60 ± 12.53	0.034 ± 0.004

Capturing Spatio-Temporal Dynamics. Given the samples from the cell population at day 1 to 3 (μ_0), JKOnet learns the underlying spatio-temporal dynamics giving rise to the developmental evolution of embryonic stem cells. As no ground truth trajectories are available in the data, we use distributional distances, i.e., the entropy-regularized Wasserstein distance W_ε (2) (Flamary et al., 2021), to measure the correctness of the predictions at each time step. We hereby measure the W_ε discrepancy between data and predictions for one-step ahead as well as inference of the entire evolution (all-steps ahead) for each time step t_i, see results in Table 1. JKOnet outperforms the forward method in terms of W_ε (2) distance for both one-step ahead and all-steps ahead predictions for all time steps. The performance of both methods is relatively stable even until day 24 to 27, i.e., the W_ε distance does not significantly grow for future snapshots. We further visualize the first two principal components of the entire dataset (Fig. 7a) and of JKOnet's predictions on the test dataset ($\sim$ 500 cells per snapshot, Fig. 7d). Visualization of predictions of the forward method can be found in the Appendix (Fig. 9a).

Capturing Biological Heterogeneity. Besides measuring the ability of JKOnet to model and predict the spatio-temporal dynamics of embryonic stem cells, we would like to guarantee, at a more macroscopic level, that JKOnet is also able to learn the cell's differentiation into various cell lineages. Embryoid bodies differentiation covers key aspects of early embryogenesis and thus captures the development of embryonic stem cells (ESC) into the mesoderm, endoderm, neuroectoderm, neural crest and others.

Following Moon et al. (2019, Fig. 6, Suppl. Note 4), we compute lineage branch classes (Fig. 13c) for all cells based on an initial k-means clustering ($k = 30$) in a 10-dimensional embedding space using PHATE, a non-linear dimensionality reduction method capturing a denoised representation of both local and global structure of a dataset (Fig. 13b). For details, see § B.3.2. We then train a k-nearest neighbor (k-NN) classifier ($k = 5$) to infer the lineage branch class based on a 20-dimensional PCA embedding of a cell (classes: ESC: 0, neural crest: 1, neuroectoderm: 2, endoderm: 3, mesoderm: 4, other: 5).

We analyze the captured lineage branch heterogeneity of the population predicted by JKOnet and the forward method by estimating the lineage branch class of each cell using the trained k-NN classifier. The predicted populations colored by the estimated lin-

eage branch as well as the data with the true lineage branch labels are visualized in Figure 7e and Figure 7b, respectively. The corresponding predicted and true distributions of lineage branch classes are shown in Figure 7c. To quantify how well JKONET and the forward method capture different cell lineage branches, we compute the ℓ_1 distance between the predicted and true histograms as well as the Hellinger distance

$$H^2(a,b) = \frac{1}{2}\sum_{i=1}^{k}\left(\sqrt{a_i/\|a\|_1} - \sqrt{b_i/\|b\|_1}\right)^2 \qquad (14)$$

between both true and predicted class discrete distributions a and b. Figure 7c and Table 2 demonstrate that both, JKONET and the forward method, capture most lineage branches during the differentiation of embryonic stem cells. Both methods, however, have difficulties recovering cells of the neural crest (class 1) and the endoderm (class 3), lineage branches which are scarcely represented in the original data. The analysis further suggests that both methods reduce in performance w.r.t. biological heterogeneity when predicting the entire trajectory (all-steps ahead), instead of inferring the next snapshot only (one-step ahead).

5 Conclusion

We proposed JKONET, a model to infer and predict the evolution of population dynamics using a proximal optimal transport scheme, the JKO flow. JKONET solves local JKO steps using ICNNs and learns the energy that parameterizes these steps by fitting JKO flow predictions to observed trajectories using a fully differentiable bilevel optimization problem. We validate its effectiveness through experiments on synthetic potential- and trajectory-based population dynamics, and observe that it is far more robust to noise than a more direct Forward approach. We use JKONET to infer the developmental trajectories of human embryonic stem cells captured via high-dimensional and time-resolved single-cell RNAseq. Our analysis also shows that JKONET captures diverse cell fates during the incremental differentiation of embryonic cells into multiple lineage branches. Using proximal optimal transport to model real complex population dynamics thus makes for an exciting avenue of future work. Extensions could include modeling higher-order interactions among population particles in the energy function, e.g., cell-cell communication.

Acknowledgments

This project received funding from the Swiss National Science Foundation under the National Center of Competence in Research (NCCR) Catalysis under grant agreement 51NF40 180544, and was supported by Google Cloud for Higher Education.

References

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.

L. Ambrosio, N. Gigli, and G. Savaré. *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Springer, 2006.

B. Amos, L. Xu, and J. Z. Kolter. Input Convex Neural Networks. In *International Conference on Machine Learning (ICML)*, volume 34, 2017.

J.-D. Benamou, G. Carlier, and M. Laborde. An augmented lagrangian approach to wasserstein gradient flows and applications. *ESAIM: Proceedings and surveys*, 54:1–17, 2016a.

J.-D. Benamou, G. Carlier, Q. Mérigot, and E. Oudet. Discretization of functionals involving the Monge–Ampére operator. *Numerische Mathematik*, 134(3), 2016b.

J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Y. Brenier. Décomposition polaire et réarrangement monotone des champs de vecteurs. *CR Acad. Sci. Paris Sér. I Math.*, 305, 1987.

M. Burger, J. A. Carrillo, and M.-T. Wolfram. A mixed finite element method for nonlinear diffusion equations. *Kinetic & Related Models*, 3(1), 2010.

L. A. Caffarelli. Monotonicity Properties of Optimal Transportation and the FKG and Related Inequalities. *Communications in Mathematical Physics*, 214 (3), 2000.

J. A. Carrillo, K. Craig, L. Wang, and C. Wei. Primal Dual Methods for Wasserstein Gradient Flows. *Foundations of Computational Mathematics*, 2021.

Y. Chen, Y. Shi, and B. Zhang. Optimal Control Via Neural Networks: A Convex Approach. In *International Conference on Learning Representations (ICLR)*, 2019.

P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011.

M. Cuturi. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 26, 2013.

M. Cuturi, L. Meng-Papaxanthos, Y. Tian, C. Bunne, G. Davis, and O. Teboul. Optimal Transport Tools (OTT): A JAX Toolbox for all things Wasserstein. *arXiv preprint arXiv:2201.12324*, 2022.

J. M. Danskin. *The Theory of Max-Min and its Applications to Weapons Allocation Problems*, volume 5. Springer, 1967.

G. De Bie, G. Peyré, and M. Cuturi. Stochastic Deep Networks. In *International Conference on Machine Learning (ICML)*, volume 36, 2019.

H. Edwards and A. Storkey. Towards a Neural Statistician. In *International Conference on Learning Representations (ICLR)*, volume 5, 2017.

J. Feydy, T. Séjourné, F.-X. Vialard, S.-I. Amari, A. Trouvé, and G. Peyré. Interpolating between Optimal Transport and MMD using Sinkhorn Divergences. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 22, 2019.

A. Figalli. The Optimal Partial Transport Problem. *Archive for Rational Mechanics and Analysis*, 195 (2), 2010.

M. Fisher, J. Nocedal, Y. Trémolet, and S. J. Wright. Data assimilation in weather forecasting: a case study in pde-constrained optimization. *Optimization and Engineering*, 10(3):409–426, 2009.

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

A. Genevay, L. Chizat, F. Bach, M. Cuturi, and G. Peyré. Sample Complexity of Sinkhorn Divergences. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 22, 2019.

W. Grathwohl, R. T. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud. FFJORD: Free-Form Continuous Dynamics for Scalable Reversible Generative Models. In *International Conference on Learning Representations (ICLR)*, 2019.

I. Haasler, A. Ringh, Y. Chen, and J. Karlsson. Estimating ensemble flows on a hidden Markov chain. In *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE, 2019.

I. Haasler, A. Ringh, Y. Chen, and J. Karlsson. Multimarginal Optimal Transport with a Tree-Structured Cost and the Schrödinger Bridge Problem. *SIAM Journal on Control and Optimization*, 59(4), 2021a.

I. Haasler, R. Singh, Q. Zhang, J. Karlsson, and Y. Chen. Multi-marginal optimal transport and probabilistic graphical models. *IEEE Transactions on Information Theory*, 2021b.

T. Hashimoto, D. Gifford, and T. Jaakkola. Learning Population-Level Diffusions with Generative Recurrent Networks. In *International Conference on Machine Learning (ICML)*, volume 33, 2016.

K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

C.-W. Huang, R. T. Q. Chen, C. Tsirigotis, and A. Courville. Convex Potential Flows: Universal Probability Distributions with Optimal Transport and Convex Optimization. In *International Conference on Learning Representations (ICLR)*, 2021.

R. Jordan, D. Kinderlehrer, and F. Otto. The Variational Formulation of the Fokker–Planck Equation. *SIAM Journal on Mathematical Analysis*, 29 (1), 1998.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2014.

P. E. Kloeden and E. Platen. Stochastic Differential Equations. In *Numerical Solution of Stochastic Differential Equations*. Springer, 1992.

R. Krishnan, U. Shalit, and D. Sontag. Structured Inference Networks for Nonlinear State Space Models. In *AAAI Conference on Artificial Intelligence*, volume 31, 2017.

Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient Backprop. In *Neural Networks: Tricks of the Trade*. Springer, 2012.

J. Lee, Y. Lee, J. Kim, A. Kosiorek, S. Choi, and Y. W. Teh. Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks. In *International Conference on Machine Learning (ICML)*, 2019.

X. Li, T.-K. L. Wong, R. T. Chen, and D. K. Duvenaud. Scalable Gradients and Variational Inference for Stochastic Differential Equations. In *Symposium on Advances in Approximate Bayesian Inference*. PMLR, 2020.

J. Lorraine, P. Vicol, and D. Duvenaud. Optimizing Millions of Hyperparameters by Implicit Differentiation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

M. D. Luecken and F. J. Theis. Current best practices in single-cell RNA-seqanalysis: a tutorial. *Molecular Systems Biology*, 15(6), 2019.

W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim. Multiple object tracking: A literature review. *Artificial Intelligence*, page 103448, 2020.

A. Makkuva, A. Taghvaei, S. Oh, and J. Lee. Optimal transport mapping via input convex neural networks. In *International Conference on Machine Learning (ICML)*, volume 37, 2020.

G. R. Martin and M. J. Evans. Differentiation of Clonal Lines of Teratocarcinoma Cells: Formation of Embryoid Bodies In Vitro. *Proceedings of the National Academy of Sciences*, 72(4), 1975.

L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled Generative Adversarial Networks. In *International Conference on Learning Representations (ICLR)*, 2017.

P. Mokrov, A. Korotin, L. Li, A. Genevay, J. Solomon, and E. Burnaev. Large-Scale Wasserstein Gradient Flows. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

G. Monge. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences*, pages 666–704, 1781.

K. R. Moon, D. van Dijk, Z. Wang, S. Gigante, D. B. Burkhardt, W. S. Chen, K. Yim, A. van den Elzen, M. J. Hirn, R. R. Coifman, et al. Visualizing structure and transitions in high-dimensional biological data. *Nature Biotechnology*, 37(12), 2019.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training Recurrent Neural Networks. In *International Conference on Machine Learning (ICML)*, volume 28, 2013.

F.-P. Paty, A. d'Aspremont, and M. Cuturi. Regularity as Regularization: Smooth and Strongly Convex Brenier Potentials in Optimal Transport. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020.

G. Peyré. Entropic Approximation of Wasserstein Gradient Flows. *SIAM Journal on Imaging Sciences*, 8 (4), 2015.

G. Peyré and M. Cuturi. Computational optimal transport. *Foundations and Trends in Machine Learning*, 11(5-6), 2019. ISSN 1935-8245.

A. Ramdas, N. G. Trillos, and M. Cuturi. On Wasserstein Two Sample Testing and Related Families of Nonparametric Tests. *Entropy*, 19(2):47, 2017.

D. Rezende and S. Mohamed. Variational Inference with Normalizing Flows. In *International Conference on Machine Learning (ICML)*, 2015.

T. Salimans, H. Zhang, A. Radford, and D. Metaxas. Improving GANs Using Optimal Transport. In *International Conference on Learning Representations (ICLR)*, 2018.

F. Santambrogio. Optimal Transport for Applied Mathematicians. *Birkäuser, NY*, 55(58-63):94, 2015.

F. Santambrogio. {Euclidean, metric, and Wasserstein} gradient flows: an overview. *Bulletin of Mathematical Sciences*, 7(1), 2017.

K. Scaman and A. Virmaux. Lipschitz regularity of deep neural networks: analysis and efficient estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

G. Schiebinger, J. Shu, M. Tabaka, B. Cleary, V. Subramanian, A. Solomon, J. Gould, S. Liu, S. Lin, P. Berube, et al. Optimal-Transport Analysis of Single-Cell Gene Expression Identifies Developmental Trajectories in Reprogramming. *Cell*, 176(4), 2019.

M. J. Shamblott, C. L. Kerr, J. Axelman, J. W. Littlefield, G. O. Clark, E. S. Patterson, R. C. Addis, J. N. Kraszewski, K. C. Kent, and J. D. Gearhart. Derivation and Differentiation of Human Embryonic Germ Cells. In *Essentials of Stem Cell Biology*. Elsevier, 2009.

D. Sheldon, M. Elmohamed, and D. Kozen. Collective Inference on Markov Models for Modeling Bird Migration. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 20, 2007.

D. R. Sheldon and T. G. Dietterich. Collective Graphical Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

F. Sigrist, H. R. Künsch, and W. A. Stahel. Stochastic partial differential equation based modelling of large space–time data sets. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, pages 3–33, 2015.

A. Tong, J. Huang, G. Wolf, D. Van Dijk, and S. Krishnaswamy. Trajectorynet: A dynamic optimal transport network for modeling cellular dynamics. In *International Conference on Machine Learning (ICML)*, 2020.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

R. J. Williams and D. Zipser. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2), 1989.

F. A. Wolf, P. Angerer, and F. J. Theis. SCANPY: large-scale single-cell gene expression data analysis. *Genome biology*, 19(1), 2018.

K. D. Yang and C. Uhler. Scalable Unbalanced Optimal Transport using Generative Adversarial Networks. *International Conference on Learning Representations (ICLR)*, 2019.

K. D. Yang, K. Damodaran, S. Venkatachalapathy, A. C. Soylemezoglu, G. Shivashankar, and C. Uhler. Predicting cell lineages using autoencoders and optimal transport. *PLoS Computational Biology*, 16(4), 2020.

M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep Sets. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017.

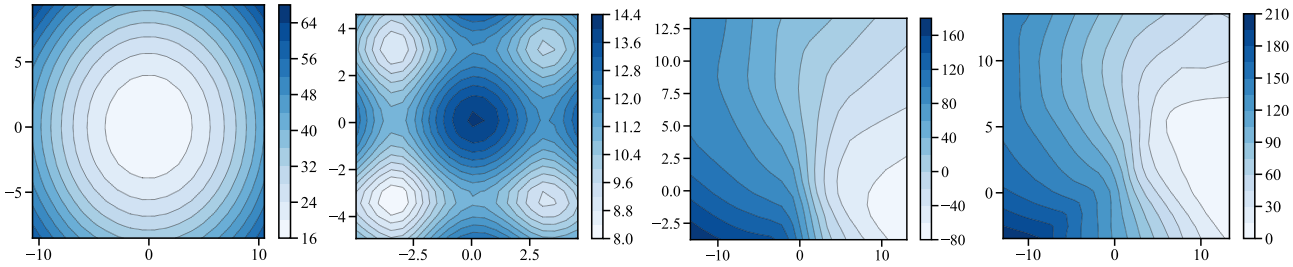
G. X. Zheng, J. M. Terry, P. Belgrader, P. Ryvkin, Z. W. Bent, R. Wilson, S. B. Ziraldo, T. D. Wheeler, G. P. McDermott, J. Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8(1), 2017.

Appendix

A Additional Evaluation

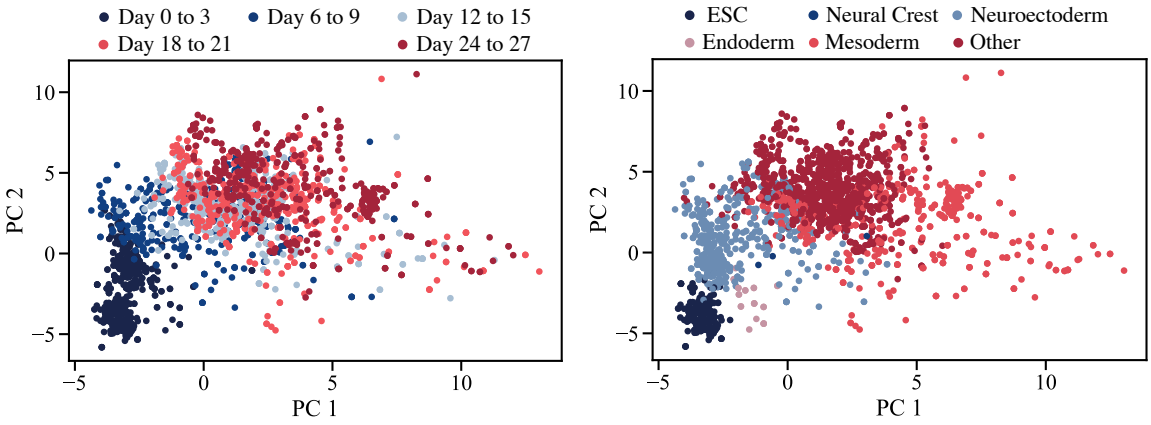
A.1 Synthetic Population Dynamics

JKOnet provides a model to understand complex population dynamics, by inferring the mechanism driving the population's time evolution. This is achieved via solving a proximal gradient descent step in the Wasserstein space, which in our case is approximated using ICNNs. *Forward* methods, on the other hand, estimate the population at the next time step $t+1$ by directly moving along the gradient direction. Thus, η_{t+1} is inferred via $(\nabla F_\xi)_\# \eta_t$, where F_ξ is any arbitrary neural network (Hashimoto et al., 2016) and η_t the predicted population at time point t. In Figure 8 we further evaluate the forward method on convex (9a) and non-convex (9b) potential-based dynamics, as well as trajectory-based dynamics (9c and d). Similarly as in the JKOnet setting, teacher forcing generally stabilizes and improves training of the energy functional F_ξ (see Fig. 8c vs. 8d). Figure 9 further shows the performance of the forward method on predicting embryoid body developmental trajectories. For further discussion of the results, see § 4.2.



(a) Quadratic Potential. (b) Styblinski Potential. (c) Semicircle Trajectory *with* teacher forcing. (d) Semicircle Trajectory.

Figure 8: **Results of the Forward Method on Potential- and Trajectory-based Dyanamics.** (a)-(d) Contour plots of the energy functionals F_ξ of the forward method on potential- and trajectory-based population dynamics in different training settings (i.e., trained with or without teacher forcing § 3.2), color gradients depict the magnitude of F_ξ.

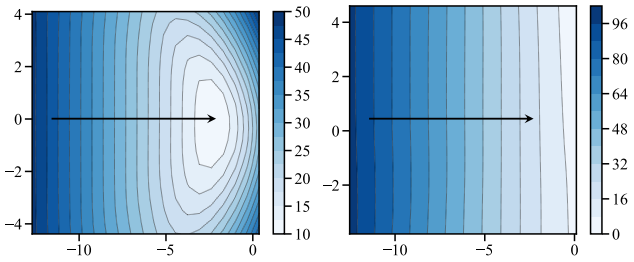


(a) PCA embedding of predictions of the forward method colored by the snapshot time.

(b) PCA embedding of predictions of the forward method colored by the lineage branch class.

Figure 9: Predictions of the forward method on time-resolved embryoid body scRNA-seq data.

A.2 Comparison to *Forward* Methods



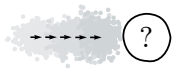
(a) Forward method. (b) JKONET.

Figure 10: Comparison between energy functionals J_ξ of the line trajectory task between the forward method and JKONET.

Table 3: Comparison of JKONET to the forward method for predicting and extrapolating linear translations (see Figure 10) (using 3 runs).

Method	**Sinkhorn Distance** $(\overline{W}_\varepsilon)$	
	Validation	Test
Forward Method	**1.94 ± 0.06**	26.10 ± 1.76
JKONET	2.90 ± 0.37	**20.30 ± 0.65**

In the following, we extend the comparison of JKONET to the forward method (see also § 4.1) and further demonstrates, that in the absence of any prior, we observe that the forward method can get more easily trapped in local minima, and overfit the training data. Figure 10 shows a simple experiment, in which we want to learn a population evolution along a line. During evaluation, we shift the line (see Fig. 11) and evaluate the prediction performance w.r.t. the Sinkhorn distance (3). Due to the less constrained energy, the *forward* method perfectly resembles the seen trajectory during training, but fails to generalize and extrapolate on shifted test data (see Table 3).



Line Trajectory

Figure 11: Out-of-Sample Predictions along a Line.

B Datasets

To evaluate JKONET, we use multiple datasets comprising different examples of population dynamics. This includes synthetic population dynamics (potential- and trajectory-based dynamics), whose results are described in § 4.1, as well as single-cell dynamics of a human developmental process, which we cover in § 4.2.

B.1 Potential-Based Dynamics

In the following, we assume a random diffusion process evolving according to an Îto stochastic difference equation (SDE) across time

$$dX_t = -\nabla\Phi(X_t)dt + \sqrt{2\sigma^2}dB_t,$$

where $B(t)$ is the unit Brownian motion (standard Wiener process with magnitude $\sigma > 0$) and the drift is defined via a potential function $\Phi(x) : \mathbb{R}^d \to \mathbb{R}$. The population-level inference problem on X_t at each t then satisfies the Fokker-Planck equation with fixed diffusion coefficient

$$\frac{\partial \rho_t}{\partial t} = \operatorname{div}\left(\nabla\Phi(x)\rho_t\right) + \sigma^{-1}\Delta\rho_t$$

with given initial condition $\rho_0 = \rho^0$. We generate the potential-based data by approximating trajectories X_t via the Euler-Maruyama method (Kloeden and Platen, 1992, § 9.2). Then given a `drift` (i.e., $\nabla\Phi$), one step of the Euler-Maruyama method is defined as

```
X = X + drift(X) * dt + np.random.normal(scale=sd, size=X.shape) * np.sqrt(dt).
```

In our experiments, we consider examples of convex, i.e., the quadratic potential $\Psi(x) = \|x\|_2^2$, and nonconvex potentials, i.e., Styblinski flow $\Psi(x) = \|3x^3 - 32x + 5\|_2^2$. For the convex potential, we simulate the trajectories using the Euler-Maruyama method with `dt` $= 0.25$ and `sd` $= 0.2$ for `n` $=$ `t/dt` iterations, where `t` $= 1.0$. Trajectories of the nonconvex potential are generated with `dt` $= 0.06$ and `sd` $= 0.4$ for `n` $=$ `t/dt` iterations, where `t` $= 0.5$.

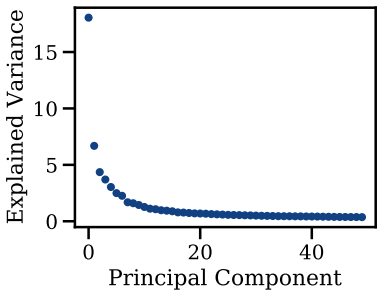


Figure 12: Proportion of explained variance per principal component of the embryoid body scRNA-seq data after preprocessing.

B.2 Trajectory-Based Dynamics

Besides population dynamics evolving according to a potential Ψ, we consider population dynamics following trajectories in space. To achieve this, we generate data by moving a 2-dimensional Gaussian distribution along a pre-defined trajectory. We compute 2-dimensional trajectories along the coordinates x and y via

```
x = r * np.cos(theta)
y = r * np.sin(theta)
```

with radius `r` and angles `theta`. The semicircle trajectory is computed using `r = 10` and `theta = np.linspace(2 * np.pi, 0, 100)`. For the spiral trajectory, `r = np.linspace(10, 1, 100)` and `theta = np.linspace(2.75 * np.pi, 0, 100)` is used. The line trajectory is generated using `x = np.linspace(-10, -2.5, 100)` and `y = np.zeros(100)`, where at test time, `x` is shifted to `x = np.linspace(-5, 7.5, 100)`. Trajectory-based dynamics are then simulated by moving a 2-dimensional Gaussian distribution along these trajectories. For the semicircle trajectories, this results in $T = 5$ snapshots, the spiral-based population dynamics contain $T = 10$ snapshots, and the line $T = 2$ snapshots.

B.3 Single-Cell Dynamics

Developmental processes in biology involve tissue and organ development, body axis formation, cell division, and cell differentiation, e.g., the development of stem cells into functional cell types. An example of such a process is the differentiation of embryonic stem cells (ESCs) into hematopoietic, cardiac, neural, pancreatic, hepatocytic and germ lineages. This development can be approximated *in vitro* using embryoid bodies (EBs) (Martin and Evans, 1975), three-dimensional aggregates of pluripotent stem cells, including ESCs (Shamblott et al., 2009). Recently, Moon et al. (2019) conducted a scRNA-seq analysis to unveil the developmental trajectories, as well as cellular and molecular identities through which early lineage precursors emerge from human ESCs. The dataset is available via Mendeley Data (V6N743H5NG)[2]. In the following, we describe the preprocessing of the raw scRNA-seq data as well as the lineage branch analysis extracting the functional cell types emerging in this developmental process.

B.3.1 Data Preprocessing

To preprocess the data, we follow the analysis of Moon et al. (2019) as well as Luecken and Theis (2019). For the analysis, we use the Python package `scanpy` (Wolf et al., 2018).

Moon et al. (2019) originally measure approximately 31,000 cells over a 27 days differentiation time course, comprising gene expression matrices and barcodes, i.e., DNA tags used to identify reads originating from the same cell. The measured cells are then filtered in a quality control stage, their gene expression levels normalized and further processed in a feature selection step, where only highly-differentiated genes are selected. The resulting data is then visualized using standard PCA as well as the dimensionality reduction method PHATE (Moon et al., 2019), in order to extract biological labels.

The data quality control is based on the number of counts per barcode (count depth), the number of genes per barcode, and the fraction of counts from mitochondrial genes per barcode. We only keep cells with at least 4000

[2]Dataset available via https://data.mendeley.com/datasets/v6n743h5ng.

and at most 10000 counts, as well as more than 550 expressed genes and less than 20% of mitochondrial counts, as a high fraction is indicative of cells whose cytoplasmic mRNA has leaked out through a broken membrane (Luecken and Theis, 2019). For the subsequent analysis, we further only keep genes which are expressed in at least 10 genes. After quality control, the dataset consists of 15150 cells and 17945 genes. We normalize each cell by total counts over all genes and logarithmize the data matrix. We extract 4000 highly variable genes (HVG) using the 10X genomics preprocessing software `Cell Ranger` (Zheng et al., 2017) to further reduce the dimensionality of the dataset and include only the most informative genes. Given the resulting data matrix with 15150 cells and 4000 genes across 5 different time points, we compute a corresponding low-dimensional embedding using PCA. Figure 12 shows the proportion of explained variance of each principal component (PC). We use the first 20 PCs for predicting population dynamics using JKONET and the forward method. This is in alignment with previous analysis of developmental trajectories, which use 5 (Tong et al., 2020) and 30 PCs (Schiebinger et al., 2019), respectively.

B.3.2 Lineage Branch Analysis of the Embryoid Body scRNA-Seq Data

To annotate the developmental process and detect lineage branches originating from the differentiation of embryonic stem cells, we follow the analysis of Moon et al. (2019). Using a 10-dimensional PHATE embedding of the embryoid body scRNA-seq data (see the first two PHATE components in Fig. 13a), we segment the dataset into 30 clusters using k-means. PHATE is a non-linear dimensionality reduction method capturing a denoised representation of both local and global structure of a dataset (Moon et al., 2019). We then assign the resulting cluster to a lineage subbranch (i - x), using the following assignment of subbranch to cluster identification (see Fig. 13b):

i. 2, 20	iv. 3, 6, 8, 13, 15, 21, 24	vii. 4, 10, 12, 17, 22	x. 29.
ii. 5, 19	v. 0, 7, 14, 25, 28	viii. 1	
iii. 9, 11, 23	vi. 16, 18, 27	ix. 26	

Then, subbranches are summarized to lineage branches using the assignment in Moon et al. (2019, Suppl. Note 4):

ESC.	i, ii	Neuroectoderm.	iv	Mesoderm.	vi, vii
Neural Crest.	iii	Endoderm.	v	Other.	viii, ix, x.

The resulting lineage branch annotation of the embryoid body scRNA-seq data can be found in Figure 13c.

C Experimental Details

In the following, we describe the baselines considered, as well as provide details on network architectures and hyperparameters used.

C.1 Baselines

We compare JKONET with *explicit* integration schemes (forward methods) such as Hashimoto et al. (2016). In our proximal method, the prediction of the population ρ_t at the next time step $t+1$ is parameterized via a separate function (ψ_θ (10)) and is thus decoupled from the free energy functional J_ξ driving the underlying dynamics. When learning *forward* methods, however, the prediction is based on the gradient of an energy functional F_ξ. Given a distribution ρ_t at time t and energy F_ξ, the population particles at time $t+1$ are thus predicted via

$$\rho_{t+1} := (\nabla F_\xi)_\# \rho_t.$$

We parameterize $F_\xi(x)$ with a MLP similar as in JKONET (see C.2.2 for more details). In this work we only consider linear functions *in the space of measures*, i.e., expectations over ρ of a vector-input neural network E_ξ (12). In these cases, we can compare JKONET to the forward methods described above. Considering energies which take particle interactions into account, however, is not straightforward when using *forward* methods.

C.2 Network Architectures

In the following, we describe network architectures used in JKONET to parameterize the Brenier map ψ_θ (Section C.2.1) as well as the free energy functional J_ξ (Section C.2.2).

C.2.1 Parameterization of Brenier Map

In the following, we describe the architectural details of the ICNN, parametrizing the Brenier map ψ_θ. We set the hidden layer size of W_l^x and W_l^z (7) to 64 and use 3 hidden layers before the final output layer ($L = 4$ layer). Similar to (Makkuva et al., 2020), we use a squared leaky ReLU function with a small positive constant β as *convex* activation function for the first layer, i.e., $a_0(x) = \max(\beta x, x)^2$, and leaky ReLU $a_l(x) = \max(\beta x, x)$, $l = 1, \ldots, L-1$ as *monotonically non-decreasing* and *convex* activation functions the remaining layers. Crucial for the stability of training ICNNs is the choice of weight initialization. We initialize W_l^x and W_l^z (7) from the standard normal distribution with standard deviation of 0.1, significantly improving in performance over the initialization strategies for standard MLPs (He et al., 2016; LeCun et al., 2012).

We further tested the performance of the *vanilla* ICNN to advanced formulations such as input-augmented ICNNs (Huang et al., 2021), whereby no difference in performance is evident. In addition, we evaluated the performance of JKONET when relaxing the convexity constraints of ψ_θ by adding a penalty

$$R(\theta) = \lambda \sum_{W_l^z \in \theta} \|\max(-W_l^z, 0)\|_F^2 ,$$

instead of enforcing its weights W_l^z to only take values > 0 as suggested in Makkuva et al. (2020). This, however, did not increase performance of our method.

C.2.2 Parameterization of Energy Functional

The free energy functional J_ξ can take various forms, accounting for diffusion as well as potentials of interaction. In this work, we concentrate on linear functions in the space of measures (12). We parametrize E_ξ as a MLP with 2 hidden layers of size 64 with softplus activation functions, followed by a one-dimensional output layer. Future work will involve an extension of the framework to energy functionals covering higher-level interactions and population growth and decline, i.e., via deep sets (Zaheer et al., 2017) or set transformers (Lee et al., 2019).

C.3 Hyperparameters and Training

For all experiments, we use a batch size of 250. For training the ICNN ψ_θ, we use the Adam optimizer (Kingma and Ba, 2014) with learning rate $\text{lr}_\theta = 0.01$ ($\beta_1 = 0.5$, $\beta_2 = 0.9$). The fixed-point loop runs for minimally 50 and maximally 100 iterations with $\alpha = 1$. When using a static number of iterations, we set the number of iterations to 100. We again use the Adam optimizer for learning the energy functional J_ξ with learning rate ranging from $\text{lr}_\xi = 0.001$ to 0.0001 ($\beta_1 = 0.5$, $\beta_2 = 0.9$). In our experiments, we use a constant JKO step size $\tau = 1.0$. For all experiments, we use $\varepsilon = 1.0$ for the Sinkhorn loss (13). Trajectory-based dynamics are trained with an additional strong convexity regularizer using $\ell = 0.8$. Both, JKONET and the forward method, are trained with gradient clipping with maximum global norm for an update of 10 (Pascanu et al., 2013).

D Reproducability

An implementation of JKONET can be found on github.com/bunnech/jkonet.

(a) PHATE embedding hued by time of snapshot.

(b) PHATE embedding hued by k-Means clustering ($k = 30$).

(c) PHATE embedding hued by predicted lineage branch.

Figure 13: Analysis of embryoid body scRNA-seq data based on PHATE embedding (Moon et al., 2019). Lineage branches are determined based on contiguous k-means clusters.