# Sliced-Wasserstein Gradient Flows

**Clément Bonet** [1] **Nicolas Courty** [1] **François Septier** [1] **Lucas Drumetz** [2]

## Abstract

Minimizing functionals in the space of probability distributions can be done with Wasserstein gradient flows. To solve them numerically, a possible approach is to rely on the Jordan–Kinderlehrer–Otto (JKO) scheme which is analogous to the proximal scheme in Euclidean spaces. However, it requires solving a nested optimization problem at each iteration, and is known for its computational challenges, especially in high dimension. To alleviate it, very recent works propose to approximate the JKO scheme leveraging Brenier's theorem, and using gradients of Input Convex Neural Networks to parameterize the density (JKO-ICNN). However, this method comes with a high computational cost and stability issues. Instead, this work proposes to use gradient flows in the space of probability measures endowed with the sliced-Wasserstein (SW) distance. We argue that this method is more flexible than JKO-ICNN, since SW enjoys a closed-form differentiable approximation. Thus, the density at each step can be parameterized by any generative model which alleviates the computational burden and makes it tractable in higher dimensions.

## 1. Introduction

Minimizing functionals with respect to probability measures is an ubiquitous problem in machine learning. Important examples are generative models such as GANs (Goodfellow et al., 2014; Arjovsky et al., 2017), VAEs (Kingma & Welling, 2013) or normalizing flows (Papamakarios et al., 2019).

To that aim, one can rely on Wasserstein gradient flows (WGF) (Ambrosio et al., 2008) which are curves decreasing the functional as fast as possible (Santambrogio, 2017). For particular functionals, these curves are known to be characterized by the solution of some partial differential equation (PDE) (Jordan et al., 1998). Hence, to solve Wasserstein gradient flows numerically, we can solve the related PDE when it is available. Another way to solve it numerically is to approximate the curve in discrete time by the so-called JKO scheme introduced in (Jordan et al., 1998), which is analogous in probability space to the well-known proximal operator (Parikh & Boyd, 2014) in Hilbertian space. It has for example been used by Lin et al. (2021) in order to train GANs.

However, as a nested minimization problem, it is a difficult problem to handle numerically. Some works use a discretization in space (*e.g.* a grid) and the entropic regularization of the Wasserstein distance (Peyré, 2015; Carlier et al., 2017), which benefits from specific resolution strategies. However, those approaches do not scale to high dimensions, as the discretization of space scales exponentially with the dimension. Another line of work proposes to use particle schemes which are derived from the stochastic differential equation (SDE) related to the PDE followed by the gradient flow or from the forward Euler scheme. Such schemes have been derived for diverse functionals such as the Kullback-Leibler divergence (Feng et al., 2021; Wang et al., 2021), the Sliced-Wasserstein distance (Liutkus et al., 2019), the maximum mean discrepancy (Arbel et al., 2019), the kernel Stein discrepancy (Korba et al., 2021) or KALE (Glaser et al., 2021). Salim et al. (2020) propose instead a forward-backward discretization scheme analogously to the proximal gradient algorithm (Bauschke et al., 2011). Yet, these methods only provide samples approximately following the gradient flow, but without any information about the underlying density. Very recently, it was proposed in several concomitant works (Alvarez-Melis et al., 2021; Mokrov et al., 2021; Bunne et al., 2021) to take advantage of Brenier's theorem (Brenier, 1991) and model the optimal transport map (Monge map) as the gradient of a convex function with Input Convex Neural Networks (ICNN) (Amos et al., 2017). These models are called JKO-ICNN and handle well higher dimension problems.

Yet, a drawback of JKO-ICNN is the training time due to a number of evaluations of the gradient of each ICNN that is quadratic in the number of JKO iterations. It also requires to backpropagate through the gradient which is challenging in high dimension, even though stochastic methods were proposed in (Huang et al., 2020) to alleviate it. Moreover, it has also been observed in several works that ICNNs have a poor expressiveness (Rout et al., 2021; Korotin et al.,

[1]Univ. Bretagne Sud [2]IMT Atlantique.

Preliminary work. Under review.

2019; 2021) and that we should rather directly estimate the gradient of convex functions by neural networks (Saremi, 2019; Richter-Powell et al., 2021).

Other recent works proposed to use the JKO scheme by either exploiting variational formulations of functionals in order to avoid the evaluation of densities and allowing to use more general neural networks in (Fan et al., 2021), or by learning directly the density in (Hwang et al., 2021).

**Contributions.** In the following, we propose to study the JKO scheme in the space of probability distributions endowed with the sliced-Wasserstein (SW) distance (Rabin et al., 2011). This novel and simple modification of the original problem comes with several benefits, mostly linked to the fact that this distance is easily differentiable and computationally more tractable than the Wasserstein distance. We first derive some properties of this new class of flows. Then, we show that it is possible to minimize functionals and learn the stationary distributions in high dimensions, on toy datasets as well as real image datasets, using *e.g.* neural networks. In particular, we propose to use normalizing flows for functionals which involve the density, such as the negative entropy. Finally, we exhibit several examples for which our strategy performs better than state-of-the-art methods, either *w.r.t.* to computation times and/or *w.r.t.* the quality of the final solutions.

## 2. Background

The goal of this paper is to minimize a functional with respect to probability measures. Let $\mathcal{P}(\mathbb{R}^d)$ bet the set of probability measures on $\mathbb{R}^d$ and $\mathcal{F}$ be such functional, then the problem can be formalized as

$$\min_{\mu \in \mathcal{P}(\mathbb{R}^d)} \mathcal{F}(\mu). \tag{1}$$

To solve this problem, we can rely on Wasserstein gradient flows (WGF) that we introduce in this section.

### 2.1. Gradient Flows in Euclidean Spaces

Let $F : \mathbb{R}^d \to \mathbb{R}$ be a functional. A gradient flow of $F$ is a curve (*i.e.* a continuous function from $\mathbb{R}_+$ to $\mathbb{R}^d$) which decreases $F$ as much as possible along it. If $F$ is differentiable, then a gradient flow $x : [0, T] \to \mathbb{R}^d$ solves the following Cauchy problem (Santambrogio, 2017)

$$\begin{cases} \frac{\mathrm{d}x(t)}{\mathrm{d}t} = -\nabla F(x(t)), \\ x(0) = x_0. \end{cases} \tag{2}$$

Under conditions on $F$ (*e.g.* $\nabla F$ Lipschitz continuous, $F$ convex or semi-convex), this problem admits a unique solution which can be approximated using numerical schemes for ordinary differential equations such as the explicit or the implicit Euler scheme. For the former, we recover the regular gradient descent, and for the latter, we recover the proximal point algorithm (Parikh & Boyd, 2014): let $\tau > 0$,

$$x_{k+1}^\tau \in \operatorname*{argmin}_x \frac{\|x - x_k^\tau\|_2^2}{2\tau} + F(x) = \operatorname{prox}_{\tau F}(x_k^\tau). \tag{3}$$

This formulation does not use any gradient, and can therefore be used in any metric space by replacing $\|\cdot\|_2$ with the right distance.

### 2.2. Gradient Flows in Probability Spaces

To define gradient flows in the space of probability measures, we first need a metric. We restrict our analysis to probability measures with moments of order 2: $\mathcal{P}_2(\mathbb{R}^d) = \{\mu \in \mathcal{P}(\mathbb{R}^d),\ \int \|x\|^2 \mathrm{d}\mu(x) < +\infty\}$. Then, a possible distance on $\mathcal{P}_2(\mathbb{R}^d)$ is the Wasserstein distance (Villani, 2008), let $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$,

$$W_2^2(\mu, \nu) = \min_{\gamma \in \Pi(\mu,\nu)} \int \|x - y\|_2^2 \, \mathrm{d}\gamma(x, y), \tag{4}$$

where $\Pi(\mu, \nu)$ is the set of probability measures on $\mathbb{R}^d \times \mathbb{R}^d$ with marginals $\mu$ and $\nu$.

Now, by endowing the space of measures with $W_2$, we can define the Wasserstein gradient flow of a functional $\mathcal{F} : \mathcal{P}_2(\mathbb{R}^d) \to \mathbb{R}$ by plugging $W_2$ in (3) which becomes

$$\mu_{k+1}^\tau \in \operatorname*{argmin}_{\mu \in \mathcal{P}_2(\mathbb{R}^d)} \frac{W_2^2(\mu, \mu_k^\tau)}{2\tau} + \mathcal{F}(\mu). \tag{5}$$

The gradient flow is then the limit of the sequence of minimizers when $\tau \to 0$. This scheme was introduced in the seminal work of Jordan et al. (1998) and is therefore referred to as the JKO scheme. In this work, Authors showed that gradient flows are linked to PDEs, and in particular with the Fokker-Planck equation when the functional $\mathcal{F}$ is of the form

$$\mathcal{F}(\mu) = \int V \mathrm{d}\mu + \mathcal{H}(\mu) \tag{6}$$

where $V$ is some potential function and $\mathcal{H}$ is the negative entropy: let $\sigma$ denote the Lebesgue measure,

$$\mathcal{H}(\mu) = \begin{cases} \int \rho(x) \log(\rho(x)) \, \mathrm{d}x & \text{if } \mathrm{d}\mu = \rho \mathrm{d}\sigma \\ +\infty \text{ otherwise.} \end{cases} \tag{7}$$

Then, the limit of $(\mu^\tau)_\tau$ when $\tau \to 0$ is a curve $t \mapsto \mu_t$ such that for all $t > 0$, $\mu_t$ has a density $\rho_t$. The curve $\rho$ satisfies (weakly) the Fokker-Planck PDE

$$\frac{\partial \rho}{\partial t} = \operatorname{div}(\rho \nabla V) + \Delta \rho. \tag{8}$$

For more details on gradient flows in metric space and in Wasserstein space, we refer to (Ambrosio et al., 2008).

### 2.3. Numerical Methods to solve the JKO Scheme

Solving (5) is not an easy problem as it requires to solve an optimal transport problem at each step. Moreover, it is composed of two nested minimization problems.

There are several strategies which were used to tackle this problem. For example, Laborde (2016) rewrites (5) as a convex minimization problem using the Benamou-Brenier dynamic formulation of the Wasserstein distance (Benamou & Brenier, 2000). Peyré (2015) approximates the JKO scheme by using the entropic regularization and rewrite the problem with respect to the Kullback-Leibler proximal operator.

The problem becomes easier to solve using Dykstra's algorithm (Dykstra, 1985). This scheme was proved to converge to the right PDE in (Carlier et al., 2017). It was proposed to use the dual formulation in other works such as (Caluya & Halder, 2019) or (Frogner & Poggio, 2020).

More recently, Alvarez-Melis et al. (2021) and Mokrov et al. (2021) proposed to exploit Brenier's theorem by rewriting the JKO scheme as

$$u_{k+1}^\tau \in \underset{u \text{ convex}}{\operatorname{argmin}} \frac{1}{2\tau} \int \|\nabla u(x) - x\|_2^2 \, \mathrm{d}\mu_k^\tau(x) + \mathcal{F}\big((\nabla u)_\# \mu_k^\tau\big) \tag{9}$$

and model the probability measures as $\mu_{k+1}^\tau = (\nabla u_{k+1}^\tau)_\# \mu_k^\tau$ where $\#$ is the push forward operator, defined as $(\nabla u)_\# \mu_k^\tau(A) = \mu_k^\tau((\nabla u)^{-1}(A))$ for all $A \in \mathcal{B}(\mathbb{R}^d)$. Then, to solve it numerically, they model convex functions using ICNNs (Amos et al., 2017):

$$\theta_{k+1}^\tau \in \underset{\theta \in \{\theta, u_\theta \in \text{ICNN}\}}{\operatorname{argmin}} \frac{1}{2\tau} \int \|\nabla_x u_\theta(x) - x\|_2^2 \, \mathrm{d}\mu_k^\tau(x) + \mathcal{F}\big((\nabla_x u_\theta))_\# \mu_k^\tau\big). \tag{10}$$

In the remainder, this method is denoted as JKO-ICNN. Bunne et al. (2021) also proposed to use ICNNs into the JKO scheme, but with a different objective of learning the functional from samples trajectories along the timesteps. Lastly, Fan et al. (2021) proposed to learn directly the Monge map $T$ by solving at each step the following problem:

$$T_{k+1}^\tau \in \underset{T}{\operatorname{argmin}} \frac{1}{2\tau} \int \|T(x) - x\|_2^2 \, \mathrm{d}\mu_k^\tau(x) + \mathcal{F}(T_\# \mu_k^\tau) \tag{11}$$

and using variational formulations of functionals involving the density. This formulation requires only to use samples from the measure. However, it needs to be derived for each functional, and involves minimax optimization problems which are notoriously hard to train (Arjovsky & Bottou, 2017; Bond-Taylor et al., 2021).

## 3. Sliced-Wasserstein Gradient Flows

As seen in the previous section, solving numerically (5) is a challenging problem. To tackle high-dimensional settings, one could benefit from neural networks, such as generative models, that are known to model accurately high-dimensional distributions. The problem being not directly differentiable, previous works relied on Brenier's theorem and modeled convex functions through ICNNs, which results in JKO-ICNN. However, this method is very costly to train. For a diffusion of length $k$, it requires $O(k^2)$ evaluations of gradients (Mokrov et al., 2021) which can be a huge price to pay when the diffusion is very long. Moreover, it requires to backpropagate through gradients, and to compute the determinant of the Jacobian when we need to evaluate the likelihood (assuming the ICNN is strictly convex). The method of Fan et al. (2021) also requires $O(k^2)$ evaluations of neural networks, as well as well as to solve a minimax optimization problem at each step.

Here, we propose instead to use the space of probability measures endowed with the sliced-Wasserstein (SW) distance by modifying adequately the JKO scheme. Surprisingly enough, this class of gradient flows, which are very easy to compute, has never been considered numerically in the literature. Close to our work, **Wasserstein** gradient flows using SW as a functional (called Sliced-Wasserstein flows) have been considered in (Liutkus et al., 2019). Our method differs significantly from this work, since we propose to compute **sliced-Wasserstein** gradient flows of different functionals.

We first introduce SW along with motivations to use this distance. We then study some properties of the scheme and discuss links with Wasserstein gradient flows. Since this metric is known in closed-form, the JKO scheme is more tractable numerically and can be approximated in several ways that we describe in Section 3.3.

### 3.1. Motivations

**Sliced-Wasserstein Distance.** The Wasserstein distance (4) is generally intractable. However, in one dimension, for $\mu, \nu \in \mathcal{P}_2(\mathbb{R})$, we have the following closed-form (Peyré et al., 2019)[Remark 2.30]

$$W_2^2(\mu, \nu) = \int_0^1 \left(F_\mu^{-1}(u) - F_\nu^{-1}(u)\right)^2 \mathrm{d}u \tag{12}$$

where $F_\mu^{-1}$ (resp. $F_\nu^{-1}$) is the quantile function of $\mu$ (resp. $\nu$). It motivated the construction of the sliced-Wasserstein distance (Rabin et al., 2011; Bonnotte, 2013), as for $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$,

$$SW_2^2(\mu, \nu) = \int_{S^{d-1}} W_2^2(P_\#^\theta \mu, P_\#^\theta \nu) \mathrm{d}\lambda(\theta) \tag{13}$$

where $P^\theta(x) = \langle x, \theta \rangle$ and $\lambda$ is the uniform distribution on $S^{d-1} = \{\theta \in \mathbb{R}^d, \ \|\theta\|_2 = 1\}$.

**Computational Properties.** Firstly, $SW_2$ is very easy to compute by a Monte-Carlo approximation (see Appendix B.1). It is also differentiable, and hence using *e.g.* the Python Optimal Transport (POT) library (Flamary et al., 2021), we can backpropagate *w.r.t.* parameters or weights parametrizing the distributions (see Section 3.3). Moreover, contrary to $W_2$, its sample complexity does not depend on the dimension (Nadjahi et al., 2020) which is important to overcome the curse of dimensionality. However, it is known to be hard to approximate in high-dimension (Deshpande et al., 2019) since the error of the Monte-Carlo estimates is impacted by the number of projections in practice (Nadjahi et al., 2020). Nevertheless, there exist several variants which could also be used. Moreover, a deterministic approach using a concentration of measure phenomenon (and hence being more accurate in high dimension) was recently proposed by Nadjahi et al. (2021) to approximate $SW_2$.

**Link with Wasserstein.** The sliced-Wasserstein distance has also many properties related to the Wasserstein distance. First, they actually induce the same topology (Nadjahi et al., 2019; Bayraktar & Guoï, 2021). Moreover, as showed in Chapter 5 of (Bonnotte, 2013), they can be related on compact sets by the following inequalities, let $R > 0$, for all $\mu, \nu \in \mathcal{P}(B(0,R))$,

$$SW_2^2(\mu,\nu) \leq c_d^2 W_2^2(\mu,\nu) \leq C_d^2 SW_2^{\frac{1}{d+1}}(\mu,\nu), \quad (14)$$

with $c_d^2 = \frac{1}{d}$ and $C_d$ some constant.

Hence, from these properties, we can wonder whether their gradient flows are related or not. Some previous works started to gather some hints on this question. For example, Candau-Tilh (2020) showed that, while $(\mathcal{P}_2(\mathbb{R}^d), SW_2)$ is not a geodesic space, the minimal length (in metric space, Definition 2.4 in (Santambrogio, 2017)) connecting two measures is $W_2$ up to a constant (which is actually $c_d$).

### 3.2. Definition and Properties of Sliced-Wasserstein Gradient Flows

Instead of solving the regular JKO scheme (5), we propose to introduce a SW-JKO scheme, let $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$,

$$\forall k \geq 0,\ \mu_{k+1}^\tau \in \operatorname*{argmin}_{\mu \in \mathcal{P}_2(\mathbb{R}^d)} \frac{SW_2^2(\mu, \mu_k^\tau)}{2\tau} + \mathcal{F}(\mu) \quad (15)$$

in which we replaced the Wasserstein distance by $SW_2$.

To study gradient flows and show that they are well defined, we first have to check that discrete solutions of the problem (15) indeed exist. Then, we have to check that we can pass to the limit $\tau \to 0$ and that the limit satisfies gradient flows properties. These limit curves will be called Sliced-Wasserstein gradient flows (SWGFs).

In the following, we restrain ourselves to measures on $\mathcal{P}_2(K)$ where $K \subset \mathbb{R}^d$ is a compact set. We report some properties of the scheme (15) such as existence and uniqueness of the minimizer, and refer to Appendix A for the proofs.

**Proposition 3.1.** *Let $\mathcal{F} : \mathcal{P}_2(K) \to \mathbb{R}$ be a lower semi continuous functional, then the scheme (15) admits a minimizer. Moreover, it is unique if $\mu_k^\tau$ is absolutely continuous and $\mathcal{F}$ convex or if $\mathcal{F}$ is strictly convex.*

This proposition shows that the problem is well defined for convex lower semi continuous functionals since we can find at least a minimizer at each step. The assumptions on $\mathcal{F}$ are fairly standard and will apply for diverse functionals such as for example (6).

**Proposition 3.2.** *The functional $\mathcal{F}$ is non increasing along the sequence of minimizers $(\mu_k^\tau)_k$.*

As the ultimate goal is to find the minimizer of the functional, this proposition assures us that the solution will decrease $\mathcal{F}$ along it at each step. If $\mathcal{F}$ is bounded below, then the sequence $(\mathcal{F}(\mu_k^\tau))_k$ will converge (since it is non increasing).

More generally, by defining the piecewise constant interpolation as $\mu^\tau(0) = \mu_0$ and for all $k \geq 0$, $t \in ]k\tau, (k+1)\tau]$, $\mu^\tau(t) = \mu_{k+1}^\tau$, we can show that for all $t < s$, $SW_2(\mu^\tau(t), \mu^\tau(s)) \leq C\left(|t-s|^{\frac{1}{2}} + \tau^{\frac{1}{2}}\right)$ (Santambrogio, 2017). By the Ascoli-Arzelà theorem (Santambrogio, 2015)[Box 1.7], at least informally, we can extract a converging subsequence. However, the limit is possibly not unique and has a priori no relation with $\mathcal{F}$. Since $(\mathcal{P}_2(\mathbb{R}^d), SW_2)$ is not a geodesic space, but rather a "pseudo-geodesic" space whose true geodesics are $c_d W_2$ (Candau-Tilh, 2020), we cannot directly apply the theory introduced in (Ambrosio et al., 2008). We leave for future works the study of the theoretical properties of the limit. Nevertheless, we conjecture that in the limit $t \to \infty$, SWGFs converge toward the same measure as for WGFs. We will study it empirically in Section 4 by showing that we are able to find as good minima as WGFs for different functionals.

**Limit PDE.** We discuss here some possible links between SWGFs and WGFs. Candau-Tilh (2020) shows that the Euler-Lagrange equation of the functional (6) has a similar form (up to the first variation of the distance). Hence, he conjectures that there is a correlation between the two gradient flows.

We first notice that for one dimensional supported measures, $W_2$ and $SW_2$ are the same up to a constant $\sqrt{d}$, *i.e.* let $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$ be supported on the same line, then $SW_2^2(\mu,\nu) = W_2^2(\mu,\nu)/d$. Interestingly enough, this is the same constant as between geodesics. This property is actually still true in any dimension for isotropic Gaussians

sharing the same variance. Therefore, we argue that for these class of measures, provided that the minimum at each step stays in the same class, we would have a dilation of factor $d$ between the WGF and the SWGF. Hence, by correcting the SW-JKO scheme as

$$\mu_{k+1}^\tau \in \operatorname*{argmin}_{\mu \in \mathcal{P}_2(\mathbb{R}^d)} \frac{d}{2\tau} SW_2^2(\mu, \mu_k^\tau) + \mathcal{F}(\mu), \tag{16}$$

we would have the same dynamic. For more general measures, it is not the case anymore. But, by rewriting $SW_2^2$ and $W_2^2$ *w.r.t.* the centered measures $\bar{\mu}$ and $\bar{\nu}$, as well as the means $m_\mu = \int x \, \mathrm{d}\mu(x)$ and $m_\nu = \int x \, \mathrm{d}\nu(x)$, we have:

$$\begin{cases} W_2^2(\mu, \nu) = \|m_\mu - m_\nu\|_2^2 + W_2^2(\bar{\mu}, \bar{\nu}) \\ SW_2^2(\mu, \nu) = \frac{\|m_\mu - m_\nu\|_2^2}{d} + SW_2^2(\bar{\mu}, \bar{\nu}). \end{cases} \tag{17}$$

Hence, for measures characterized by their mean and variance (*e.g.* Gaussians), there will be a constant $d$ between the optimal mean of the SWGF and of the WGF. However, such a direct relation is not available between variances, even on simple cases like Gaussians. We report in Appendix A.4 the details of the calculations. We also report in Appendix C.1 evolutions along the approximated WGF and SWGF of different functionals. In particular, for the interaction functional studied in Section 4.2, we do not observe the dilation of factor $d$, which tends to suggest that the relation between the limit PDEs, if any, is more subtle and complex.

### 3.3. Solving the SW-JKO Scheme in Practice

As a Monte-Carlo approximate of SW can be computed in closed-form, (15) is not a nested minimization problem anymore and is differentiable. We present here a few possible parameterizations of probability distributions which we can use in practice through SW-JKO to approximate the gradient flow. We further state how to approximate the Fokker-Planck functional (6). Then, from these parameterizations, we can apply gradient-based optimization algorithms by using backpropagation over the loss at each step.

**Discretized Grid.** A first proposition is to model the distribution on a regular fixed grid, as it is done *e.g.* in (Peyré, 2015). If we approximate the distribution by a discrete distribution with a fixed grid on which the different samples are located, then we only have to learn the weights. Let us denote $\mu_k^\tau = \sum_{i=1}^N \rho_i^{(k)} \delta_{x_i}$ where we use $N$ samples located at $(x_i)_{i=1}^N$, and $\sum_{i=1}^N \rho_i = 1$. Let $\Sigma_N$ denote the simplex, then the optimization problem (15) becomes:

$$\min_{(\rho_i)_i \in \Sigma_N} \frac{SW_2^2(\sum_{i=1}^N \rho_i \delta_{x_i}, \mu_k^\tau)}{2\tau} + \mathcal{F}(\sum_{i=1}^N \rho_i \delta_{x_i}). \tag{18}$$

The entropy is only defined for absolutely continuous distributions. However, following (Carlier et al., 2017; Peyré, 2015), we can approximate the Lebesgue measure as: $L = l \sum_{i=1}^N \delta_{x_i}$ where $l$ represents a volume of each grid point (we assume that each grid point represents a volume element of uniform size). In that case, the Lebesgue density can be approximated by $(\frac{\rho_i}{l})_i$. Hence, for the Fokker-Planck (6) example, we approximate the potential and internal energies as

$$\begin{aligned} \mathcal{V}(\mu) &= \int V(x)\rho(x)\mathrm{d}x \approx \sum_{i=1}^N V(x_i)\rho_i \\ \mathcal{H}(\mu) &= \int \log(\rho(x))\rho(x)\mathrm{d}x \approx \sum_{i=1}^N \log\left(\frac{\rho_i}{l}\right)\rho_i. \end{aligned} \tag{19}$$

To stay on the simplex, we use a projected gradient descent (Condat, 2016). A drawback of discretizing the grid is that it becomes intractable in high dimension.

**With Particles.** We can also optimize over the position of a set of particles, assigning them uniform weights: $\mu_k^\tau = \frac{1}{N}\sum_{i=1}^N \delta_{x_i^{(k)}}$. The problem (15) becomes:

$$\min_{(x_i)_i} \frac{SW_2^2(\frac{1}{N}\sum_{i=1}^N \delta_{x_i}, \mu_k^\tau)}{2\tau} + \mathcal{F}(\frac{1}{N}\sum_{i=1}^N \delta_{x_i}). \tag{20}$$

In that case however, we do not have access to the density and cannot directly approximate $\mathcal{H}$ (or more generally internal energies). A workaround is to use nonparametric estimators (Beirlant et al., 1997), which is however impractical in high dimension.

**Generative Models.** Another solution to model the distribution is to use generative models. Let us denote $g_\theta : \mathcal{Z} \to \mathcal{X}$ such a model, with $\mathcal{Z}$ a latent space, $\theta$ the parameters of the model that will be learned, and let $p_Z$ be a simple distribution (*e.g.* Gaussian). Then, we will denote $\mu_{k+1}^\tau = (g_\theta^{k+1})_\# p_Z$. The SW-JKO scheme (15) will become in this case

$$\min_\theta \frac{SW_2^2\big((g_\theta^{k+1})_\# p_Z, \mu_k^\tau\big)}{2\tau} + \mathcal{F}\big((g_\theta^{k+1})_\# p_Z\big). \tag{21}$$

To approximate the negative entropy, we have to be able to evaluate the density. A straightforward choice that we use in our experiments is to use invertible neural networks with tractable density such as normalizing flows (Papamakarios et al., 2019; Kobyzev et al., 2020). Another solution could be to use the variational formulation as in (Fan et al., 2021) as we only need samples in that case, but at the cost of solving a minimax problem.

To perform the optimization, we can sample points of the different distributions at each step and use a Monte-Carlo

**Algorithm 1** SW-JKO with Generative Models

**Input:** $\mu_0$ the initial distribution, $K$ the number of SW-JKO steps, $\tau$ the step size, $\mathcal{F}$ the functional, $N_e$ the number of epochs to solve each SW-JKO step, $N$ the batch size
**for** $k = 1$ **to** $K$ **do**
  Initialize a neural network $g_\theta^{k+1}$ *e.g.* with $g_\theta^k$
  **for** $i = 1$ **to** $N_e$ **do**
    Sample $z_j^{(k)}, z_j^{(k+1)} \sim p_Z$ i.i.d
    $x_j^{(k)} = g_\theta^k(z_j^{(k)})$, $x_j^{(k+1)} = g_\theta^{k+1}(z_j^{(k+1)})$
    // Denote $\hat{\mu}_k^\tau = \frac{1}{N}\sum_{j=1}^N \delta_{x_j^{(k)}}$
    // Denote $\hat{\mu}_{k+1}^\tau = \frac{1}{N}\sum_{j=1}^N \delta_{x_j^{(k+1)}}$
    $J(\hat{\mu}_{k+1}^\tau) = \frac{1}{2\tau} SW_2^2(\hat{\mu}_k^\tau, \hat{\mu}_{k+1}^\tau) + \mathcal{F}(\hat{\mu}_{k+1}^\tau)$
    Backpropagate through $J$ *w.r.t.* $\theta$
    Perform a gradient step using *e.g.* Adam
  **end for**
**end for**

approximation in order to approximate the integrals. Let $z_i \sim p_Z$ i.i.d, then $g_\theta(z_i) \sim (g_\theta)_\# p_Z = \mu$ and

$$\mathcal{V}(\mu) \approx \frac{1}{N}\sum_{i=1}^N V(g_\theta(z_i))$$
$$\mathcal{H}(\mu) \approx \frac{1}{N}\sum_{i=1}^N \left(\log(p_Z(z_i)) - \log\left|\det(J_{g_\theta}(z_i))\right|\right) \tag{22}$$

using the change of variable formula in $\mathcal{H}$.

We sum up the procedure when modeling distributions with generative models in Algorithm 1. We provide the algorithms for the discretized grid and for the particles in Appendix B.2.

## 4. Experiments

In this section, we show that by approximating sliced-Wasserstein gradient flows using the SW-JKO scheme (15), we are able to minimize functionals as well as Wasserstein gradient flows approximated by the JKO-ICNN scheme and with a better computational complexity. We first evaluate the ability to learn the stationary density for the Fokker-Planck equation (8) in the Gaussian case, and in the context of Bayesian Logistic Regression. Then, we evaluate it on an Aggregation equation. Finally, we use SW as a functional with image datasets as target, and compare the results with Sliced-Wasserstein flows introduced in (Liutkus et al., 2019).

For these experiments, we mainly use generative models. When it is required to evaluate the density (*e.g.* to estimate $\mathcal{H}$), we use Real Non Volume Preserving (RealNVP) normalizing flows (Dinh et al., 2016). Our experiments were conducted using PyTorch (Paszke et al., 2019).

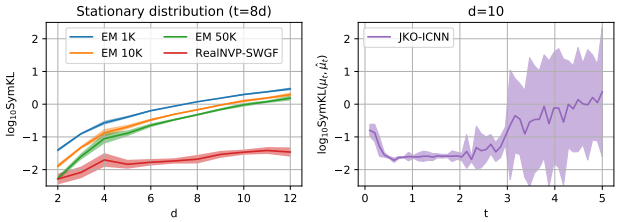



Figure 1: On the left, SymKL divergence between solutions at time $t = 8d$ (using $\tau = 0.1$ and 80 steps in (15)) and stationary measure. On the right, SymKL between the true WGF $\mu_t$ and the approximation with JKO-ICNN $\hat{\mu}_t$, run through 3 Gaussians with $\tau = 0.1$. We observe unstabilities at some point.

### 4.1. Convergence to Stationary Distribution for the Fokker-Planck Equation

**Gaussian Case.** The solution of a PDE of the form of (8) converges toward a unique stationary measure $\mu^* \propto e^{-V}$ (Risken, 1996). Taking $V$ of the form

$$\forall x \in \mathbb{R}^d,\ V(x) = \frac{1}{2}(x-m)^T A(x-b) \tag{23}$$

with $A$ a symmetric positive definite matrix, and $m \in \mathbb{R}^d$, then the stationary distribution is $\mu^* = \mathcal{N}(m, A^{-1})$. We plot in Figure 1 the symmetric Kullback-Leibler (SymKL) divergence over dimensions between approximated distributions and the true stationary distribution. We choose $\tau = 0.1$ and performed 80 SW-JKO steps. We take the mean over 15 random gaussians for dimensions $d \in \{2, \dots, 12\}$ for randomly generated positive semi-definite matrices $A$ using "make_spd_matrix" from scikit-learn (Pedregosa et al., 2011). Moreover, we use RealNVPs in SW-JKO. We see that the results are better than the particle schemes obtained with the Unadjusted Langevin Algorithm (ULA) (Roberts & Tweedie, 1996), called Euler-Maruyama (EM) since it is the EM scheme of the Langevin equation, with a step size of $10^{-3}$ and with either $10^3$, $10^4$ or $5 \cdot 10^4$ particles in dimension higher than 2. We do not plot the results for JKO-ICNN as we observe unstabilities and very long training time (right plot in Figure 1). We add more details in Appendix C.2.

**Curse of Dimensionality.** Even though the sliced-Wasserstein distance sample complexity does not suffer from the curse of dimensionality, it appears through the Monte-Carlo approximation (Nadjahi et al., 2020). Here, since SW plays a regularizer role, the objective is not necessarily to approximate it well but rather to minimize the given functional. Nevertheless, the number of projections

can still have an impact on the minimization, and we report on Figure 2 the evolution of the found minima *w.r.t.* the number of projections, averaged over 15 random Gaussians. We observe that we do not need much projections to have fairly good results, even in higher dimension.

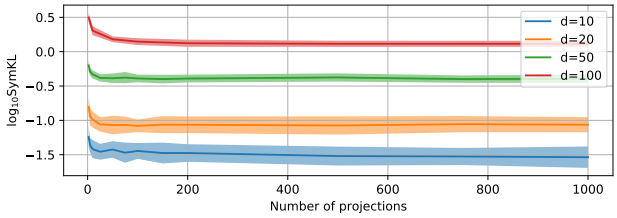



Figure 2: Impact of the number of projections for a fixed number of epochs.

**Bayesian Logistic Regression.** Following the experiment of (Mokrov et al., 2021) in Section 4.3, we propose to tackle the Bayesian Logistic Regression problem using SWGFs. For this task, we want to sample from $p(x|D)$ where $D$ represent data and $x = (w, \log \alpha)$ with $w$ the regression weights on which we apply a Gaussian prior $p_0(w|\alpha) = \mathcal{N}(w; 0, \alpha^{-1})$ and with $p_0(\alpha) = \Gamma(\alpha; 1, 0.01)$. In that case, we use $V(x) = -\log p(x|D)$ to learn $p(x|D)$. We refer to Appendix C.2 for more details on the experiments, as well as hyperparameters. We report in Table 1 the accuracy results obtained on different datasets with SWGFs and compared with JKO-ICNN. We also report the training time and see that SWGFs allow to obtain results as good as with JKO-ICNN but for shorter training times which underlines the better complexity of our scheme.

Table 1: Accuracy and Training Time for Bayesian Logistic Regression over 5 runs

| | JKO-ICNN | | SWGF+RealNVP | |
|---|---|---|---|---|
| Dataset | Acc | t | Acc | t |
| covtype | $0.755 \pm 5 \cdot 10^{-4}$ | 33702s | $0.755 \pm 3 \cdot 10^{-3}$ | 103s |
| german | $0.679 \pm 5 \cdot 10^{-3}$ | 2123s | **0.68** $\pm 5 \cdot 10^{-3}$ | 82s |
| diabetis | $0.777 \pm 7 \cdot 10^{-3}$ | 4913s | **0.778** $\pm 2 \cdot 10^{-3}$ | 122s |
| twonorm | $0.981 \pm 2 \cdot 10^{-4}$ | 6551s | $0.981 \pm 6 \cdot 10^{-4}$ | 301s |
| ringnorm | $0.736 \pm 10^{-3}$ | 1228s | **0.741** $\pm 6 \cdot 10^{-4}$ | 82s |
| banana | $0.55 \pm 10^{-2}$ | 1229s | **0.559** $\pm 10^{-2}$ | 66s |
| splice | $0.847 \pm 2 \cdot 10^{-3}$ | 2290s | **0.85** $\pm 2 \cdot 10^{-3}$ | 113s |
| waveform | **0.782** $\pm 8 \cdot 10^{-4}$ | 856s | $0.776 \pm 8 \cdot 10^{-4}$ | 120s |
| image | **0.822** $\pm 10^{-3}$ | 1947s | $0.821 \pm 3 \cdot 10^{-3}$ | 72s |

### 4.2. Convergence to Stationary Distribution for an Aggregation Equation

We also show the possibility to solve different PDEs than Fokker-Planck. For example, using an interaction functional of the form

$$\mathcal{W}(\mu) = \frac{1}{2} \iint W(x-y) \mathrm{d}\mu(x) \mathrm{d}\mu(y). \tag{24}$$

In that case, the Wasserstein gradient flow is the aggregation equation (Santambrogio, 2015)[Chapter 8]

$$\frac{\partial \rho}{\partial t} = \mathrm{div}(\rho(\nabla W * \rho)), \tag{25}$$

where $*$ denotes the convolution operation. We notice here that we do not need to evaluate the density. Therefore, we can apply any neural network. For example, in the following, we will use a simple fully connected neural network (FCNN) and compare the results obtained with JKO-ICNN. We also show the results when learning directly over the particles and when learning weights over a grid.

Carrillo et al. (2021) use a repulsive-attractive interaction potential $W(x) = \frac{\|x\|_2^4}{4} - \frac{\|x\|_2^2}{2}$. In this case, they showed empirically that the solution is a Dirac ring with radius 0.5 and centered at the origin. Starting from $\mu_0 = \mathcal{N}(0, 0.25^2 I_2)$, with $\tau = 0.05$, we show on Figure 3 that we recover this result with SWGFs for different parametrizations of the probabilities. More precisely, we first use a discretized grid of $50 \times 50$ samples of $[-1, 1]^2$. Then, we show the results when directly learning the particles and when using a FCNN. We also compare with the results obtained with JKO-ICNN. The densities reported for the last three methods are obtained through a kernel density estimator (KDE) with a bandwidth manually chosen since we either do not have access to the density, or we observed for JKO-ICNN that the likelihood exploded (see Appendix C.4). It may be due to the fact that the stationary solution does not admit a density with respect to the Lebesgue measure. For JKO-ICNN, we observe that the ring shape is recovered, but the samples are not evenly distributed on it.

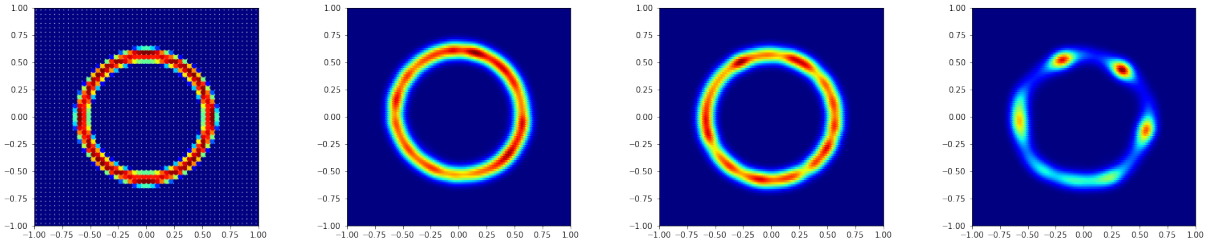
We report the solution at time $t = 10$, and used $\tau = 0.05$ for SW-JKO and $\tau = 0.1$ for JKO-ICNN. As JKO-ICNN requires $O(k^2)$ evaluations of gradients of ICNNs, the training is very long for such a diffusion. Here, the training took around 5 hours on a RTX 2080 TI (for 100 steps), versus 20 minutes for the FCNN and 10 minutes for 1000 particles (for 200 steps).

To sum up, an advantage of the SW-JKO scheme is to be able to use easier, yet powerful enough, architectures to learn the diffusion. This is cheaper in training time and less memory costly. Furthermore, we can tune the architecture with respect to the characteristics of the problem and add inductive biases (*e.g.* using CNN for images) or learn directly over the particles.

### 4.3. Application on Real Data

In what follows, we show that the SW-JKO scheme can generate real data, and perform better than the associated particle scheme. We focus on the functional

$$\mathcal{F}(\mu) = \frac{1}{2} SW_2^2(\mu, \nu) + \lambda \mathcal{H}(\mu) \tag{26}$$



(a) Steady state on the discretized grid

(b) Steady state for the fully connected neural network

(c) Steady state for particles

(d) Steady state for JKO-ICNN (with $\tau = 0.1$)

Figure 3: Steady state of the aggregation equation for $a = 4$, $b = 2$. From left to right, we plot it for the discretized grid, for the FCNN, for particles and for JKO-ICNN. We observe that JKO-ICNN does not recover well the ring as the particles are not evenly distributed on it.

where $\nu$ is some target distribution, for which we should have access to samples. This functional was introduced in (Bonnotte, 2013) for $\lambda = 0$, and in (Liutkus et al., 2019) with the negative entropy term. Liutkus et al. (2019) showcased a particle scheme called SWF (Sliced Wasserstein Flow) to approximate the WGF of (26). Applied on images such as MNIST (LeCun & Cortes, 2010), FashionMNIST (Xiao et al., 2017) or CelebA (Liu et al., 2015), SWFs need a very long convergence due to the curse of dimensionality and the trouble approximating SW. Hence, they used instead a pretrained autoencoder (AE) and applied the particle scheme in the latent space. Likewise, we use the AE proposed by Liutkus et al. (2019) with a latent space of dimension $d = 48$, and we perform SW-JKO steps on thoses images. We report on Figure 4 samples obtained with RealNVPs and on Table 2 the Fréchet Inception distance (FID) (Heusel et al., 2017) obtained. We denote "golden score" the FID obtained with the pretrained autoencoder. Hence, we cannot obtain better results than this. We compared the results in the latent and in the ambient space with SWFs and see that we obtain fairly better results using generative models within the SW-JKO scheme, especially in the ambient space, although the results are not really competitive with state-of-the-art methods. This may be due more to the curse of dimensionality in approximating the objective SW than in approximating the regularizer SW.

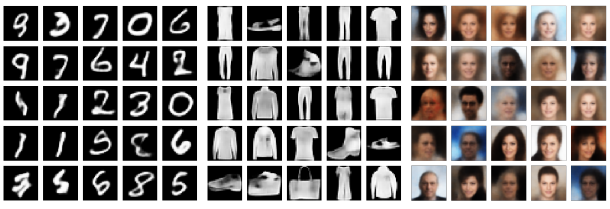


Figure 4: Generated sample obtained through a pretrained decoder + RealNVP.

Table 2: FID scores on some datasets (lower is better)

| | Methods | MNIST | Fashion | CelebA |
|---|---|---|---|---|
| Ambient Space | SWF (Liutkus et al., 2019) | 225.1 | 207.6 | - |
| | SWGF + RealNVP | 88.1 | **95.5** | - |
| | SWGF + CNN | **69.3** | 102.3 | - |
| Latent Space | AE (golden score) | 15.55 | 31 | 77 |
| | SWGF + AE + RealNVP | **17.8** | **40.6** | 90.9 |
| | SWGF + AE + FCNN | 18.3 | 41.7 | **88** |
| | SWF | 22.5 | 56.4 | 91.2 |

## 5. Conclusion

In this work, we derive a new class of gradient flows in the space of probability measures endowed with the sliced-Wasserstein metric, and the corresponding algorithms. To the best of our knowledge, and despite its simplicity, this is the first time that this class of flows is proposed in a machine learning context. We showed that it has several advantages over state-of-the-art approaches such as the recent JKO-ICNN. Aside from being less computationally intensive, it is more versatile *w.r.t.* the different practical solutions for modeling probability distributions, such as normalizing flows, generative models or sets of evolving particles.

Regarding the theoretical aspects, several challenges remain ahead: First, its connections with Wasserstein gradient flows are still unclear. Second, one needs to understand if, regarding the optimization task, convergence speeds or guarantees are changed with this novel formulation, revealing potentially interesting practical properties. Lastly, it is natural to study if popular variants of the sliced-Wasserstein distance such as Max-sliced (Deshpande et al., 2019), Distributional sliced (Nguyen et al., 2021), Subspace robust (Paty & Cuturi, 2019), generalized sliced (Kolouri et al., 2019) or projection Wasserstein distances (Rowland et al., 2019) can also be used in similar gradient flow schemes. The study of higher-order approximation schemes such as BDF2 (Matthes & Plazotta, 2019; Plazotta, 2018) could also be of interest.

## References


Alvarez-Melis, D., Schiff, Y., and Mroueh, Y. Optimizing functionals on the space of probabilities with input convex neural networks, 2021.

Ambrosio, L., Gigli, N., and Savaré, G. *Gradient flows: in metric spaces and in the space of probability measures*. Springer Science & Business Media, 2008.

Amos, B., Xu, L., and Kolter, J. Z. Input convex neural networks, 2017.

Arbel, M., Korba, A., Salim, A., and Gretton, A. Maximum mean discrepancy gradient flow. *arXiv preprint arXiv:1906.04370*, 2019.

Arjovsky, M. and Bottou, L. Towards principled methods for training generative adversarial networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Hk4_qw5xe.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Balagué, D., Carrillo, J. A., Laurent, T., and Raoul, G. Nonlocal interactions by repulsive–attractive potentials: radial ins/stability. *Physica D: Nonlinear Phenomena*, 260:5–25, 2013.

Bauschke, H. H., Combettes, P. L., et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer, 2011.

Bayraktar, E. and Guoï, G. Strong equivalence between metrics of wasserstein type. *Electronic Communications in Probability*, 26:1–13, 2021.

Beirlant, J., Dudewicz, E. J., Györfi, L., Van der Meulen, E. C., et al. Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences*, 6(1):17–39, 1997.

Benamou, J.-D. and Brenier, Y. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000.

Bond-Taylor, S., Leach, A., Long, Y., and Willcocks, C. G. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *arXiv preprint arXiv:2103.04922*, 2021.

Bonnotte, N. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Paris 11, 2013.

Brenier, Y. Polar factorization and monotone rearrangement of vector-valued functions. *Communications on pure and applied mathematics*, 44(4):375–417, 1991.

Bunne, C., Meng-Papaxanthos, L., Krause, A., and Cuturi, M. Jkonet: Proximal optimal transport modeling of population dynamics, 2021.

Caluya, K. F. and Halder, A. Proximal recursion for solving the fokker-planck equation. In *2019 American Control Conference (ACC)*, pp. 4098–4103. IEEE, 2019.

Candau-Tilh, J. Wasserstein and sliced-wasserstein distances. Master's thesis, Université Pierre et Marie Curie, 2020.

Carlier, G., Duval, V., Peyré, G., and Schmitzer, B. Convergence of entropic schemes for optimal transport and gradient flows. *SIAM Journal on Mathematical Analysis*, 49(2):1385–1418, 2017.

Carrillo, J. A., Chertock, A., and Huang, Y. A finite-volume method for nonlinear nonlocal equations with a gradient flow structure. *Communications in Computational Physics*, 17(1):233–258, 2015.

Carrillo, J. A., Craig, K., Wang, L., and Wei, C. Primal dual methods for wasserstein gradient flows. *Foundations of Computational Mathematics*, pp. 1–55, 2021.

Chen, Y. and Kolokolnikov, T. A minimal model of predator–swarm interactions. *Journal of The Royal Society Interface*, 11(94):20131208, 2014.

Condat, L. Fast projection onto the simplex and the l1 ball. *Mathematical Programming*, 158(1):575–585, 2016.

Cornish, R., Caterini, A., Deligiannidis, G., and Doucet, A. Relaxing bijectivity constraints with continuously indexed normalising flows. In *International Conference on Machine Learning*, pp. 2133–2143. PMLR, 2020.

Dai, B. and Seljak, U. Sliced iterative normalizing flows. In *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021.

Deshpande, I., Hu, Y.-T., Sun, R., Pyrros, A., Siddiqui, N., Koyejo, S., Zhao, Z., Forsyth, D., and Schwing, A. G. Max-sliced wasserstein distance and its use for gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10648–10656, 2019.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Dykstra, R. L. An iterative procedure for obtaining i-projections onto the intersection of convex sets. *The annals of Probability*, pp. 975–984, 1985.

Fan, J., Taghvaei, A., and Chen, Y. Variational wasserstein gradient flow. *arXiv preprint arXiv:2112.02424*, 2021.

Fang, K.-T., Kotz, S., and Ng, K. W. *Symmetric multivariate and related distributions*. Chapman and Hall/CRC, 1992.

Feng, X., Gao, Y., Huang, J., Jiao, Y., and Liu, X. Relative entropy gradient sampler for unnormalized distributions. *arXiv preprint arXiv:2110.02787*, 2021.

Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N. T., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021. URL http://jmlr.org/papers/v22/20-451.html.

Frogner, C. and Poggio, T. Approximate inference with wasserstein gradient flows. In *International Conference on Artificial Intelligence and Statistics*, pp. 2581–2590. PMLR, 2020.

Givens, C. R. and Shortt, R. M. A class of wasserstein metrics for probability distributions. *Michigan Mathematical Journal*, 31(2):231–240, 1984.

Glaser, P., Arbel, M., and Gretton, A. Kale flow: A relaxed kl gradient flow for probabilities with disjoint support, 2021.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Ho, J., Chen, X., Srinivas, A., Duan, Y., and Abbeel, P. Flow++: Improving flow-based generative models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pp. 2722–2730. PMLR, 2019.

Huang, C.-W., Chen, R. T., Tsirigotis, C., and Courville, A. Convex potential flows: Universal probability distributions with optimal transport and convex optimization. *arXiv preprint arXiv:2012.05942*, 2020.

Hwang, H. J., Kim, C., Park, M. S., and Son, H. The deep minimizing movement scheme. *arXiv preprint arXiv:2109.14851*, 2021.

Jordan, R., Kinderlehrer, D., and Otto, F. The variational formulation of the fokker–planck equation. *SIAM journal on mathematical analysis*, 29(1):1–17, 1998.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kobyzev, I., Prince, S., and Brubaker, M. Normalizing flows: An introduction and review of current methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., and Rohde, G. K. Generalized sliced wasserstein distances. *arXiv preprint arXiv:1902.00434*, 2019.

Korba, A., Aubin-Frankowski, P., Majewski, S., and Ablin, P. Kernel stein discrepancy descent. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5719–5730. PMLR, 2021.

Korotin, A., Egiazarian, V., Asadulaev, A., Safin, A., and Burnaev, E. Wasserstein-2 generative networks. *arXiv preprint arXiv:1909.13082*, 2019.

Korotin, A., Li, L., Genevay, A., Solomon, J., Filippov, A., and Burnaev, E. Do neural optimal transport solvers work? a continuous wasserstein-2 benchmark. *arXiv preprint arXiv:2106.01954*, 2021.

Laborde, M. *Interacting particles systems, Wasserstein gradient flow approach*. PhD thesis, PSL Research University, 2016.

Le Gall, J.-F. *Brownian motion, martingales, and stochastic calculus*. Springer, 2016.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Lin, A. T., Li, W., Osher, S., and Montúfar, G. Wasserstein proximal of gans. *arXiv preprint arXiv:2102.06862*, 2021.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. *arXiv preprint arXiv:1608.04471*, 2016.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738, 2015.

Liutkus, A., Simsekli, U., Majewski, S., Durmus, A., and Stöter, F.-R. Sliced-wasserstein flows: Nonparametric generative modeling via optimal transport and diffusions. In *International Conference on Machine Learning*, pp. 4104–4113. PMLR, 2019.

Mackey, M. C. *Time's arrow: The origins of thermodynamic behavior*. Courier Corporation, 1992.

Matthes, D. and Plazotta, S. A variational formulation of the bdf2 method for metric gradient flows. *ESAIM: Mathematical Modelling and Numerical Analysis*, 53(1): 145–172, 2019.

Mika, S., Ratsch, G., Weston, J., Scholkopf, B., and Mullers, K.-R. Fisher discriminant analysis with kernels. In *Neural networks for signal processing IX: Proceedings of the 1999 IEEE signal processing society workshop (cat. no. 98th8468)*, pp. 41–48. Ieee, 1999.

Mokrov, P., Korotin, A., Li, L., Genevay, A., Solomon, J., and Burnaev, E. Large-scale wasserstein gradient flows. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

Nadjahi, K., Durmus, A., Simsekli, U., and Badeau, R. Asymptotic guarantees for learning generative models with the sliced-wasserstein distance. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Nadjahi, K., Durmus, A., Chizat, L., Kolouri, S., Shahrampour, S., and Simsekli, U. Statistical and topological properties of sliced probability divergences. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 20802–20812. Curran Associates, Inc., 2020.

Nadjahi, K., Durmus, A., Jacob, P. E., Badeau, R., and Şimşekli, U. Fast approximation of the sliced-wasserstein distance using concentration of random projections. *arXiv preprint arXiv:2106.15427*, 2021.

Nguyen, K., Ho, N., Pham, T., and Bui, H. Distributional sliced-wasserstein and applications to generative modeling. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Papamakarios, G., Pavlakou, T., and Murray, I. Masked autoregressive flow for density estimation. *arXiv preprint arXiv:1705.07057*, 2017.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

Parikh, N. and Boyd, S. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):127–239, 2014.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Paty, F.-P. and Cuturi, M. Subspace robust wasserstein distances. In *International Conference on Machine Learning*, pp. 5072–5081. PMLR, 2019.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Peyré, G. Entropic approximation of wasserstein gradient flows. *SIAM Journal on Imaging Sciences*, 8(4):2323–2351, 2015.

Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Plazotta, S. A bdf2-approach for the non-linear fokker-planck equation. *arXiv preprint arXiv:1801.09603*, 2018.

Rabin, J., Peyré, G., Delon, J., and Bernot, M. Wasserstein barycenter and its application to texture mixing. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pp. 435–446. Springer, 2011.

Richter-Powell, J., Lorraine, J., and Amos, B. Input convex gradient networks, 2021.

Risken, H. Fokker-planck equation. In *The Fokker-Planck Equation*, pp. 63–95. Springer, 1996.

Roberts, G. O. and Tweedie, R. L. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, pp. 341–363, 1996.

Rout, L., Korotin, A., and Burnaev, E. Generative modeling with optimal transport maps. *arXiv preprint arXiv:2110.02999*, 2021.

Rowland, M., Hron, J., Tang, Y., Choromanski, K., Sarlos, T., and Weller, A. Orthogonal estimation of wasserstein distances. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 186–195. PMLR, 2019.

Salim, A., Korba, A., and Luise, G. The wasserstein proximal gradient algorithm. *arXiv e-prints*, pp. arXiv–2002, 2020.

Santambrogio, F. Optimal transport for applied mathematicians. *Birkäuser, NY*, 55(58-63):94, 2015.

Santambrogio, F. {Euclidean, metric, and Wasserstein} gradient flows: an overview. *Bulletin of Mathematical Sciences*, 7(1):87–154, 2017.

Saremi, S. On approximating $\nabla f$ with neural networks. *arXiv preprint arXiv:1910.12744*, 2019.

Vatiwutipong, P. and Phewchean, N. Alternative way to derive the distribution of the multivariate ornstein–uhlenbeck process. *Advances in Difference Equations*, 2019(1):1–7, 2019.

Villani, C. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Wang, Y., Chen, P., and Li, W. Projected wasserstein gradient descent for high-dimensional bayesian inference. *arXiv preprint arXiv:2102.06350*, 2021.

Wibisono, A. 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.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

# A. PROOFS

## A.1. Proof of Proposition 3.1

We refer to the proposition 3.7 in (Candau-Tilh, 2020).

Let $\tau > 0$, $k \in \mathbb{N}$, $\mu_k^\tau \in \mathcal{P}_2(K)$. Let's note $J(\mu) = \frac{SW_2^2(\mu, \mu_k^\tau)}{2\tau} + \mathcal{F}(\mu)$.

According to Proposition 3.4 in (Candau-Tilh, 2020), $\mu \mapsto SW_2^2(\mu, \mu_k^\tau)$ is continuous with respect to the weak convergence. Indeed, let $\mu \in \mathcal{P}_2(K)$ and let $(\mu_n)_n$ converging weakly to $\mu$, *i.e.* $\mu_n \xrightarrow[n\to\infty]{\mathcal{L}} \mu$. Then, by the reverse triangular inequality, we have

$$|SW_2^2(\mu_n, \mu_k^\tau) - SW_2^2(\mu, \mu_k^\tau)| \leq SW_2^2(\mu_n, \mu) \leq W_2^2(\mu_n, \mu). \tag{27}$$

Since the Wasserstein distance metrizes the weak convergence, we have that $W_2^2(\mu_n, \mu) \to 0$. And therefore, $\mu \mapsto SW_2^2(\mu, \mu_k^\tau)$ is continuous *w.r.t.* the weak convergence.

By hypothesis, $\mathcal{F}$ is lower semi continuous, hence $\mu \mapsto J(\mu)$ is lower semi continuous. Moreover, $\mathcal{P}_2(K)$ is compact for the weak convergence, thus we can apply the Weierstrass theorem (Box 1.1 in (Santambrogio, 2015)) and there exists a minimizer $\mu_{k+1}^\tau$ of $J$.

By Proposition 3.5 in (Candau-Tilh, 2020), $\mu \mapsto SW_2^2(\mu, \nu)$ is convex and strictly convex whenever $\nu$ is absolutely continuous *w.r.t.* the Lebesgue measure. Hence, for the uniqueness, if $\mathcal{F}$ is strictly convex then $\mu \mapsto J(\mu)$ is also strictly convex and the minimizer is unique. And if $\rho_k^\tau$ is absolutely continuous, then according to Proposition 3.5 in (Candau-Tilh, 2020), $\mu \mapsto SW_2^2(\mu, \mu_k^\tau)$ is strictly convex, and hence $\mu \mapsto J(\mu)$ is also strictly convex since $\mathcal{F}$ was taken convex by hypothesis.

## A.2. Proof of Proposition 3.2

Let $k \in \mathbb{N}$, then since $\mu_{k+1}^\tau$ is the minimizer of (15),

$$\mathcal{F}(\mu_{k+1}^\tau) + \frac{SW_2^2(\mu_{k+1}^\tau, \mu_k^\tau)}{2\tau} \leq \mathcal{F}(\mu_k^\tau) + \frac{SW_2^2(\mu_k^\tau, \mu_k^\tau)}{2\tau} = \mathcal{F}(\mu_k^\tau). \tag{28}$$

Hence, as $SW_2^2(\mu_{k+1}^\tau, \mu_k^\tau) \geq 0$,

$$\mathcal{F}(\mu_{k+1}^\tau) \leq \mathcal{F}(\mu_k^\tau). \tag{29}$$

## A.3. Upper Bound on the Errors

Following (Hwang et al., 2021), we can also derive an upper bound on the error made at each step.

**Proposition A.1.** *Let $k \in \mathbb{N}$, $\mu_0 \in \mathcal{P}_2(\mathbb{R}^d)$, $C$ some constant, and assume that $\mathcal{F}$ admits a lower bound, then*

$$\begin{aligned} SW_2\big((g_\theta^{k+1,\tau})_\# p_Z, \mu_{k+1}^\tau\big) &\leq SW_2\big((g_\theta^{k,\tau})_\# p_Z, \mu_k^\tau\big) + C\tau^{\frac{1}{2}} \\ &\leq (k+1)C\tau^{\frac{1}{2}} + SW_2(\mu_0, \mu_1^\tau). \end{aligned} \tag{30}$$

*Proof.* The bound can be found by applying Theorem 3.1 in (Hwang et al., 2021) with $X = (\mathcal{P}_2(\mathbb{R}^d, SW_2)$, and then by applying a straightforward induction. We report here the proof for the sake of completeness.

Let $k \in \mathbb{N}$, then since $\mu_{k+1}^\tau$ is the minimizer of (15),

$$\mathcal{F}(\mu_{k+1}^\tau) + \frac{SW_2^2(\mu_{k+1}^\tau, \mu_k^\tau)}{2\tau} \leq \mathcal{F}(\mu_k^\tau) + \frac{SW_2^2(\mu_k^\tau, \mu_k^\tau)}{2\tau} = \mathcal{F}(\mu_k^\tau). \tag{31}$$

Therefore, using that $\mathcal{F}$ is non increasing along $(\mu_k^\tau)_k$ (Proposition 3.2), we have $\mathcal{F}(\mu_k^\tau) \leq \mathcal{F}(\mu_0)$. Moreover, using that $\mathcal{F}$ admits an infimum, we find

$$SW_2^2(\mu_k^\tau, \mu_{k+1}^\tau) \leq 2\tau\big(\mathcal{F}(\mu_k^\tau) - \mathcal{F}(\mu_{k+1}^\tau)\big) \leq 2\tau\big(\mathcal{F}(\mu_0) - \inf_\mu \mathcal{F}(\mu)\big). \tag{32}$$

Let $A = \mathcal{F}(\mu_0) - \inf_\mu \mathcal{F}(\mu)$. By the same reasoning, we have that

$$SW_2^2\big((g_\theta^{k+1,\tau})_\# p_Z, (g_\theta^{k,\tau})_\# p_Z\big) \leq 2\tau A. \tag{33}$$

Now, using the triangular inequality, we have that

$$\begin{aligned} SW_2\big((g_\theta^{k+1,\tau})_\# p_Z, \mu_{k+1}^\tau\big) &\leq SW_2\big((g_\theta^{k+1,\tau})_\# p_Z, (g_\theta^k)_\# p_Z\big) + SW_2\big((g_\theta^k)_\# p_Z, \mu_k^\tau\big) + SW_2(\mu_k^\tau, \mu_{k+1}^\tau) \\ &= 2\sqrt{2\tau A} + SW_2\big((g_\theta^{k,\tau})_\# p_Z, \mu_k^\tau\big). \end{aligned} \tag{34}$$

Let $C = 2\sqrt{2A}$, then by induction we find

$$SW_2\big((g_\theta^{k+1,\tau})_\# p_Z, \mu_{k+1}^\tau\big) \leq (k+1)C\sqrt{\tau} + SW_2(\mu_0, \mu_1^\tau). \tag{35}$$

□

### A.4. Sliced-Wasserstein Results

**Link for 1D Supported Measures.** Let $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ supported on a line. For simplicity, we suppose that the measures are supported on an axis, *i.e.* $\mu(x) = \mu_1(x_1)\prod_{i=2}^d \delta_0(x_i)$ and $\nu(x) = \nu_1(x_1)\prod_{i=2}^d \delta_0(x_i)$.

In this case, we have that

$$W_2^2(\mu, \nu) = W_2^2(P^{e_1}_\# \mu, P^{e_1}_\# \nu) = \int_0^1 |F^{-1}_{P^{e_1}_\# \mu}(x) - F^{-1}_{P^{e_1}_\# \nu}(x)|^2 \, \mathrm{d}x. \tag{36}$$

On the other hand, let $\theta \in S^{d-1}$, then we have

$$\begin{aligned} \forall y \in \mathbb{R},\ F_{P^\theta_\# \mu}(y) &= \int_{\mathbb{R}} \mathbb{1}_{]-\infty, y]}(x)\ P^\theta_\# \mu(\mathrm{d}x) \\ &= \int_{\mathbb{R}^d} \mathbb{1}_{]-\infty, y]}(\langle \theta, x\rangle)\ \mu(\mathrm{d}x) \\ &= \int_{\mathbb{R}} \mathbb{1}_{]-\infty, y]}(x_1\theta_1)\ \mu_1(\mathrm{d}x_1) \\ &= \int_{\mathbb{R}} \mathbb{1}_{]-\infty, \frac{y}{\theta_1}]}\ \mu_1(\mathrm{d}x_i) \\ &= F_{P^{e_1}_\# \mu}(\frac{y}{\theta_1}). \end{aligned} \tag{37}$$

Therefore, $F^{-1}_{P^\theta_\# \mu}(z) = \theta_1 F^{-1}_{P^{e_1}_\# \mu}(z)$ and

$$\begin{aligned} W_2^2(P^\theta_\# \mu, P^\theta_\# \nu) &= \int_0^1 |\theta_1 F^{-1}_{P^{e_1}_\# \mu}(z) - \theta_1 F^{-1}_{P^{e_1}_\# \nu}(z)|^2 \, \mathrm{d}z \\ &= \theta_1^2 \int_0^1 |F^{-1}_{P^{e_1}_\# \mu}(z) - F^{-1}_{P^{e_1}_\# \nu}(z)|^2 \, \mathrm{d}z \\ &= \theta_1^2 W_2^2(\mu, \nu). \end{aligned} \tag{38}$$

Finally, using that $\int_{S^{d-1}} \theta\theta^T \mathrm{d}\lambda(\theta) = \frac{1}{d} I_d$, we can conclude that

$$SW_2^2(\mu, \nu) = \int_{S^{d-1}} \theta_1^2 W_2^2(\mu, \nu) \mathrm{d}\theta = \frac{W_2^2(\mu, \nu)}{d}.$$

**Closed-form between Gaussians.** It is well known that there is a closed-form for the Wasserstein distance between Gaussians (Givens & Shortt, 1984). If we take $\alpha = \mathcal{N}(\mu, \Sigma)$ and $\beta = \mathcal{N}(m, \Lambda)$ with $m, \mu \in \mathbb{R}^d$ and $\Sigma, \Lambda \in \mathbb{R}^{d\times d}$ two symmetric positive definite matrices, then

$$W_2^2(\alpha, \beta) = \|m - \mu\|_2^2 + \mathrm{Tr}\big(\Sigma + \Lambda - 2(\Sigma^{\frac{1}{2}}\Lambda\Sigma^{\frac{1}{2}})^{\frac{1}{2}}\big). \tag{39}$$

Let $\alpha = \mathcal{N}(\mu, \sigma^2 I_d)$ and $\beta = \mathcal{N}(m, s^2 I_d)$ two isotropic Gaussians. Here, we have

$$\begin{aligned} W_2^2(\alpha, \beta) &= \|\mu - m\|_2^2 + \operatorname{Tr}(\sigma^2 I_d + s^2 I_d - 2(\sigma s^2 \sigma I_d)^{\frac{1}{2}}) \\ &= \|\mu - m\|_2^2 + (\sigma - s)^2 \operatorname{Tr}(I_d) \\ &= \|\mu - m\|_2^2 + d(\sigma - s)^2. \end{aligned} \tag{40}$$

On the other hand, Nadjahi et al. (2021) showed (Equation 73) that

$$SW_2^2(\alpha, \beta) = \frac{\|\mu - m\|_2^2}{d} + (\sigma - s)^2 = \frac{W_2^2(\alpha, \beta)}{d}. \tag{41}$$

In that case, the dilation of factor $d$ between WGF and SWGF clearly appears.

For more complicated gaussians, we may not have this equality. For example, let $\alpha = \mathcal{N}(\mu, D)$, $\beta = \mathcal{N}(m, \Delta)$ with $D$ and $\Delta$ diagonal. Then, $P^\theta_\# \alpha = \mathcal{N}(\langle \mu, \theta \rangle, \sum_{i=1}^N \theta_i^2 D_i)$, $P^\theta_\# \beta = \mathcal{N}(\langle m, \theta \rangle, \sum_{i=1}^N \theta_i^2 \Delta_i)$ and

$$W_2^2(P^\theta_\# \bar{\alpha}, P^\theta_\# \bar{\beta}) = \left( \sqrt{\sum_i \theta_i^2 D_i} - \sqrt{\sum_i \theta_i^2 \Delta_i} \right)^2 \tag{42}$$

with $\bar{\alpha}$, $\bar{\beta}$ the centered measures (noting $T^\alpha : x \mapsto x - \mu$, then $\bar{\alpha} = T^\alpha_\# \alpha$). Hence, we have

$$\begin{aligned} SW_2^2(\alpha, \beta) &= \frac{\|\mu - m\|_2^2}{d} + SW_2^2(\bar{\alpha}, \bar{\beta}) \\ &= \frac{\|\mu - m\|_2^2}{d} + \int_{S^{d-1}} \left( \sqrt{\sum_i \theta_i^2 D_i} - \sqrt{\sum_i \theta_i^2 \Delta_i} \right)^2 \mathrm{d}\lambda(\theta) \\ &= \frac{\|\mu - m\|_2^2}{d} + \int_{S^{d-1}} \left( \sum_{i=1}^d \theta_i^2 D_i + \sum_{i=1}^d \theta_i^2 \Delta_i - 2 \sqrt{\sum_{i,j} \theta_i^2 \theta_j^2 D_i \Delta_j} \right)^2 \mathrm{d}\lambda(\theta) \\ &= \frac{\|\mu - m\|_2^2}{d} + \sum_{i=1}^d D_i \int_{S^{d-1}} \theta_i^2 \mathrm{d}\lambda(\theta) + \sum_{i=1}^D \Delta_i \int_{S^{d-1}} \theta_i^2 \mathrm{d}\lambda(\theta) - 2 \int_{S^{d-1}} \sqrt{\sum_{i,j} \theta_i^2 \theta_j^2 D_i \Delta_j} \mathrm{d}\lambda(\theta) \\ &= \frac{\|\mu - m\|_2^2}{d} + \frac{1}{d} \sum_{i=1}^d (D_i + \Delta_i) - 2 \int_{S^{d-1}} \sqrt{\sum_{i,j} \theta_i^2 \theta_j^2 D_i \Delta_j} \mathrm{d}\lambda(\theta), \end{aligned} \tag{43}$$

using that $\int_{S^{d-1}} \theta \theta^T \mathrm{d}\lambda(\theta) = \frac{1}{d} I_d$ and by applying Proposition 2 in (Nadjahi et al., 2021) to decompose $SW_2^2(\alpha, \beta) = \frac{\|\mu - m\|_2^2}{d} + SW_2^2(\bar{\alpha}, \bar{\beta})$.

On the other hand, we have

$$\begin{aligned} W_2^2(\alpha, \beta) &= \|\mu - m\|_2^2 + \operatorname{Tr}\big(D + \Delta - 2(D^{\frac{1}{2}} \Delta D^{\frac{1}{2}})^{\frac{1}{2}}\big) \\ &= \|\mu - m\|_2^2 + \operatorname{Tr}(D + \Delta - 2(D\Delta)^{\frac{1}{2}}) \\ &= \|\mu - m\|_2^2 + \sum_{i=1}^d (D_i + \Delta_i - 2 D_i^{\frac{1}{2}} \Delta_i^{\frac{1}{2}}) \\ &= \|\mu - m\|_2^2 + \sum_{i=1}^d (D_i^{\frac{1}{2}} - \Delta_i^{\frac{1}{2}})^2. \end{aligned} \tag{44}$$

Since $SW_2^2(\alpha, \beta) \leq \frac{1}{d} W_2^2(\alpha, \beta)$, we have $\sum_{i=1}^d \sqrt{D_i \Delta_i} \leq d \int_{S^{d-1}} \sqrt{\sum_{i,j} \theta_i^2 \theta_j^2 D_i \Delta_j} \mathrm{d}\lambda(\theta)$.

Let $d = 2$, $\sigma, s > 0$ and $D = \mathrm{diag}(\sigma^2, \frac{\sigma^2}{2})$, $\Delta = \mathrm{diag}(\frac{s^2}{2}, s^2)$. In this case, on the one hand, we have

$$\sum_{i=1}^{2} \sqrt{D_i \Delta_i} = \sqrt{2}\sigma s. \tag{45}$$

On the other hand,

$$\begin{aligned} 2\int_{S^1} \sqrt{\sum_{i,j} \theta_i^2 \theta_j^2 D_i \Delta_j} \mathrm{d}\lambda(\theta) &= \sqrt{2}\sigma s \int_{S^1} \sqrt{(\theta_1^2+\theta_2^2)^2 + \frac{1}{2}\theta_1^2\theta_2^2}\, \mathrm{d}\lambda(\theta) \\ &= \sqrt{2}\sigma s \int_{S^1 \cap \{\theta_1 \neq 0, \theta_2 \neq 0\}} \sqrt{(\theta_1^2+\theta_2^2)^2 + \frac{1}{2}\theta_1^2\theta_2^2}\, \mathrm{d}\lambda(\theta) \\ &> \sqrt{2}\sigma s \int_{S^1} (\theta_1^2+\theta_2^2)\, \mathrm{d}\lambda(\theta) \\ &= \sqrt{2}\sigma s, \end{aligned} \tag{46}$$

using that $\lambda$ is absolutely continuous with respect to the Lebesgue measure and hence $\lambda(\{\theta_1 = 0\} \cup \{\theta_2 = 0\}) = 0$ and the fact that for every $\theta \in S^1 \cap \{\theta_1 \neq 0, \theta_2 \neq 0\}$, $\sqrt{(\theta_1^2+\theta_2^2)^2 + \frac{1}{2}\theta_1^2\theta_2^2} > \sqrt{(\theta_1^2+\theta_2^2)^2} = \theta_1^2+\theta_2^2$. From this strict inequality, we deduce that $W_2$ and $d \cdot SW_2$ are not always equal, even in this restricted case.

# B. COMPUTATION OF THE SW-JKO SCHEME IN PRACTICE

## B.1. Approximation of SW

For each inner optimization problem

$$\mu_{k+1}^\tau \in \operatorname*{argmin}_{\mu \in \mathcal{P}_2(\mathbb{R}^d)} \frac{SW_2^2(\mu, \mu_k^\tau)}{2\tau} + \mathcal{F}(\mu), \tag{47}$$

we need to approximate the sliced-Wasserstein distance. To do that, we used Monte-Carlo approximate by sampling $n_\theta$ directions $(\theta_i)_{i=1}^{n_\theta}$ following the uniform distribution on the hypersphere $S^{d-1}$ (which can be done by using the stochastic representation, *i.e.* let $Z \sim \mathcal{N}(0, I_d)$, then $\theta = \frac{Z}{\|Z\|_2} \sim \mathrm{Unif}(S^{d-1})$ (Fang et al., 1992)). Let $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$, we approximate the sliced-Wasserstein distance as

$$\widehat{SW}_2^2(\mu, \nu) = \frac{1}{n_\theta} \sum_{i=1}^{n_\theta} W_2^2(P_\#^{\theta_i}\mu, P_\#^{\theta_i}\nu). \tag{48}$$

In practice, we compute it for empirical distributions $\hat{\mu}_n$ and $\hat{\nu}_m$, and we approximate the one dimensional Wasserstein distance

$$W_2^2(P_\#^\theta \hat{\mu}_n, P_\#^\theta \hat{\nu}_m) = \int_0^1 |F^{-1}_{P_\#^\theta \hat{\mu}_n}(u) - F^{-1}_{P_\#^\theta \hat{\nu}_m}(u)|^2 \, \mathrm{d}u \tag{49}$$

by the rectangle method.

## B.2. Algorithms to Solve the SW-JKO Scheme

We provide here the algorithms used to solve the SW-JKO scheme (15) for the discrete grid (Section 3.3) and for the particles (Section 3.3).

**Discrete Grid.** We recall that in that case, we model the distributions as $\mu_k^\tau = \sum_{i=1}^N \rho_i^{(k)} \delta_{x_i}$ where we use $N$ samples located at $(x_i)_{i=1}^N$ and $(\rho_i^{(k)})_{i=1}^N$ belongs to the simplex $\Sigma_n$. Hence, the SW-JKO scheme at step $k+1$ rewrites

$$\min_{(\rho_i)_i \in \Sigma_N} \frac{SW_2^2(\sum_{i=1}^N \rho_i \delta_{x_i}, \mu_k^\tau)}{2\tau} + \mathcal{F}(\sum_{i=1}^N \rho_i \delta_{x_i}). \tag{50}$$

We report in Algorithm 2 the whole procedure.

**Algorithm 2** SW-JKO with Discrete Grid

**Input:** $\mu_0$ the initial distribution with density $\rho_0$, $K$ the number of SW-JKO steps, $\tau$ the step size, $\mathcal{F}$ the functional, $N_e$ the number of epochs to solve each SW-JKO step, $(x_j)_{j=1}^N$ the grid
Let $\rho^{(0)} = \left(\frac{\rho_0(x_j)}{\sum_{\ell=1}^N \rho_0(x_\ell)}\right)_{j=1}^N$
**for** $k = 1$ **to** $K$ **do**
  Initialize the weights $\rho^{(k+1)}$ (with for example a copy of $\rho^{(k)}$)
  // Denote $\mu_{k+1}^\tau = \sum_{j=1}^N \rho_j^{(k+1)} \delta_{x_j}$ and $\mu_k^\tau = \sum_{j=1}^N \rho_j^{(k)} \delta_{x_j}$
  **for** $i = 1$ **to** $N_e$ **do**
    Compute $J(\mu_{k+1}^\tau) = \frac{1}{2\tau} SW_2^2(\mu_k^\tau, \mu_{k+1}^\tau) + \mathcal{F}(\mu_{k+1}^\tau)$
    Backpropagate through $J$ with respect to $\rho^{(k+1)}$
    Perform a gradient step
    Project on the simplex $\rho^{(k+1)}$ using the algorithm of Condat (2016)
  **end for**
**end for**

**Particle Scheme.** In this case, we model the distributions as empirical distributions and we try to optimize the positions of the particles. Hence, we have $\mu_k^\tau = \frac{1}{N}\sum_{i=1}^N \delta_{x_i^{(k)}}$ and the problem (15) becomes

$$\min_{(x_i)_i} \frac{SW_2^2(\frac{1}{N}\sum_{i=1}^N \delta_{x_i}, \mu_k^\tau)}{2\tau} + \mathcal{F}(\frac{1}{N}\sum_{i=1}^N \delta_{x_i}). \tag{51}$$

In this case, we provide the procedure in Algorithm 3.

**Algorithm 3** SW-JKO with Particles

**Input:** $\mu_0$ the initial distribution, $K$ the number of SW-JKO steps, $\tau$ the step size, $\mathcal{F}$ the functional, $N_e$ the number of epochs to solve each SW-JKO step, $N$ the batch size
Sample $(x_j^{(0)})_{j=1}^N \sim \mu_0$ i.i.d
**for** $k = 1$ **to** $K$ **do**
  Initialize $N$ particles $(x_j^{(k+1)})_{j=1}^N$ (with for example a copy of $(x_j^{(k)})_{j=1}^N$)
  // Denote $\mu_{k+1}^\tau = \frac{1}{N}\sum_{j=1}^N \delta_{x_j^{(k+1)}}$ and $\mu_k^\tau = \frac{1}{N}\sum_{j=1}^N \delta_{x_j^{(k)}}$
  **for** $i = 1$ **to** $N_e$ **do**
    Compute $J(\mu_{k+1}^\tau) = \frac{1}{2\tau} SW_2^2(\mu_k^\tau, \mu_{k+1}^\tau) + \mathcal{F}(\mu_{k+1}^\tau)$
    Backpropagate through $J$ with respect to $(x_j^{(k+1)})_{j=1}^N$
    Perform a gradient step
  **end for**
**end for**

# C. ADDITIONAL EXPERIMENTS

## C.1. Dynamic of Sliced-Wasserstein Gradient Flows

The Fokker-Planck equation (8) is the Wasserstein gradient flow of the functional (6). Moreover, it is well-known to have a counterpart stochastic differential equation (SDE) (see *e.g.* (Mackey, 1992)[Chapter 11]) of the form

$$\mathrm{d}X_t = -\nabla V(X_t)\mathrm{d}t + \sqrt{2\beta}\,\mathrm{d}W_t \tag{52}$$

with $(W_t)_t$ a Wiener process. This SDE is actually the well-known Langevin equation. Hence, by approximating it using the Euler-Maruyama scheme, we recover the Unadjusted Langevin Algorithm (ULA) (Roberts & Tweedie, 1996; Wibisono, 2018).

For

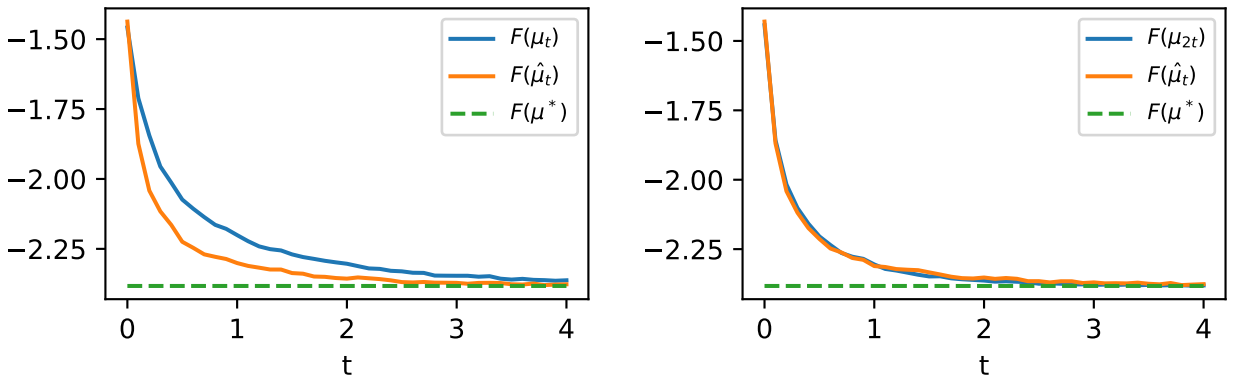
$$V(x) = \frac{1}{2}(x-m)^T A(x-m), \tag{53}$$

with $A$ symmetric and definite positive, we obtain an Ornstein-Uhlenbeck process (Le Gall, 2016)[Chapter 8]. If we choose $\mu_0$ as a Gaussian $\mathcal{N}(m_0, \Sigma_0)$, then we know the Wasserstein gradient flow $\mu_t$ in closed form (Wibisono, 2018; Vatiwutipong & Phewchean, 2019), for all $t > 0$, $\mu_t = \mathcal{N}(m_t, \Sigma_t)$ with

$$\begin{cases} m_t = m + e^{-tA}(m_0 - m) \\ \Sigma_t = e^{-tA}\Sigma_0(e^{-tA})^T + A^{-\frac{1}{2}}(I - e^{-2tA})(A^{-\frac{1}{2}})^T. \end{cases} \tag{54}$$

**Comparison of the Evolution of the Diffusion between SWGFs and WGFs.** For this experiment, we model the density using RealNVPs (Dinh et al., 2016). More precisely, we use RealNVPs with 5 affine coupling layers, using FCNN for the scaling and shifting networks with 100 hidden units and 5 layers. In both experiments, we always start the scheme with $\mu_0 = \mathcal{N}(0, I)$ and take $n_\theta = 1000$ projections to approximate the sliced-Wasserstein distance. We randomly generate a target Gaussian (using "make_spd_matrix" from scikit-learn (Pedregosa et al., 2011) to generate a random covariance with 42 as seed).

We look at the evolution of the distributions learned between $t = 0$ and $t = 4$ with a time step of $\tau = 0.1$. We compare it with the true Wasserstein gradient flow. On Figure 5a, we observe that they do not seem to match. However, they do converge to the same stationary value. On Figure 5b, we plot the functional along the true WGF dilated of a factor $d = 2$. We see here that the two curves are matching and we observed the same behaviour in higher dimension. Even though we cannot conclude on the PDE followed by SWGFs, this reinforces the conjecture that the SWGF obtained with a step size of $\frac{\tau}{d}$ (*i.e.* using the scheme (16)) is very close to the WGF obtained with a step size of $\tau$. We also report here the evolution of the mean (Fig. 6) and of the variance (Fig. 7). For the mean, it follows as expected the same diffusion. For the variance, it is less clear but it is hard to conclude since there are potentially optimization errors.




(a) Without dilation

(b) With dilation

Figure 5: Evolution of the functional (6) along the WGF $\mu_t$ and the learned SWGF $\hat{\mu}_t$. We observe a dilation of parameter 2 between the WGF and the SWGF.

**Comparison between JKO-ICNN and SW-JKO.** Following the experiment conducted by (Mokrov et al., 2021) in section 4.2, we plot in Figure 8 the symmetric Kullback-Leibler (SymKL) divergence over dimensions between approximated distributions and the true WGF at times $t = 0.5$ and $t = 0.9$. We take the mean over 15 random gaussians (generated using the scikit-learn function (Pedregosa et al., 2011) "make_spd_matrix" for the covariance matrices, and generating the means with a standard normal distribution) for dimensions $d \in \{2, \dots, 12\}$.

For each target Gaussian, we run the SW-JKO dilated scheme (16) with $\tau = 0.05$ for a RealNVP normalizing flow. We compare it with JKO-ICNN with also $\tau = 0.05$ and with Euler-Maruyama with $10^3$, $10^4$ and $5 \cdot 10^4$ particles and a step size of $10^{-3}$. For JKO-ICNN, we use, as (Mokrov et al., 2021), DenseICNN with convex quadratic layers introduced in (Korotin

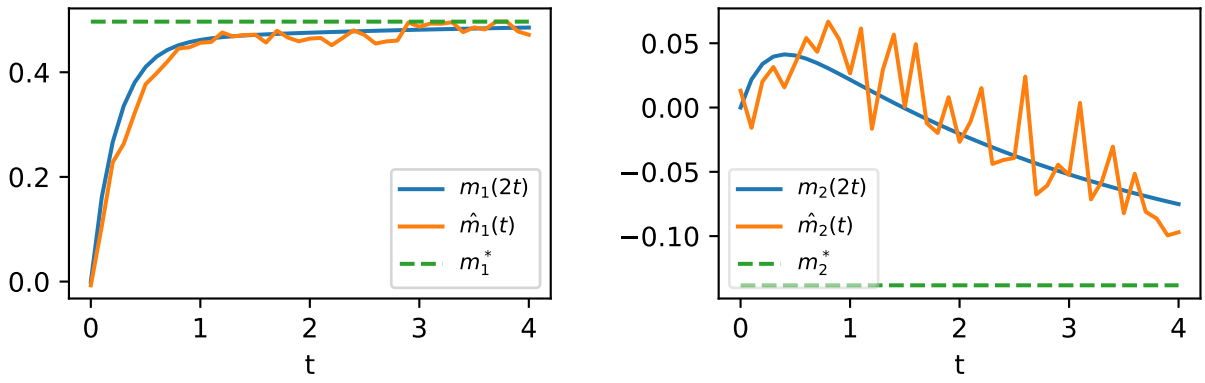



Figure 6: Evolution of the mean taking into account the dilation parameter. $\mu$ denotes the true mean of WGF, $\hat{\mu}$ the mean obtained through SW-JKO (15) with $\tau = 0.05$ and $\mu_*$ the mean of the stationary measure. We observe that the mean of approximated measure obtained through SW-JKO seems to follow the one of the WGF.

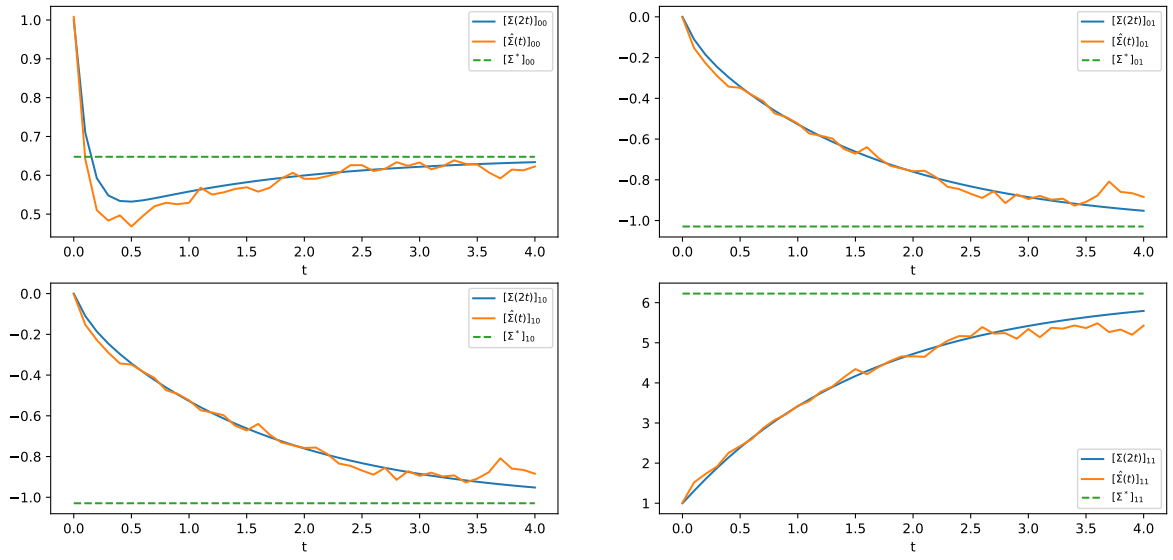



Figure 7: Evolution of the components of the covariance matrix taking into account the dilation parameter. $\Sigma$ denotes the true covariance matrix of WGF, $\hat{\Sigma}$ the covariance matrix obtained through SW-JKO (15) with $\tau = 0.05$ and $\Sigma^*$ the covariance matrix of the stationary distribution. We observe some difference between WGF and SWGF.

et al., 2019) and available at https://github.com/iamalexkorotin/Wasserstein2Barycenters. For the JKO-ICNN scheme, we use our own implementation.

We compute the symmetric Kullback-Leibler divergence between the ground truth of WGF $\mu^*$ and the distribution $\hat{\mu}$ approximated by the different schemes at times $t = 0.5$ and $t = 0.9$. The symmetric Kullback-Leibler divergence is obtained as

$$\text{SymKL}(\mu^*, \hat{\mu}) = \text{KL}(\mu^* || \hat{\mu}) + \text{KL}(\hat{\mu} || \mu^*). \tag{55}$$

To approximate it, we generate $10^4$ samples of each distribution and evaluate the density at those samples.

If we note $g_\theta$ a normalizing flows, $p_Z$ the distribution in the latent space and $\rho = (g_\theta)_\# p_Z$, then we can evaluate the log density of $\rho$ by using the change of variable formula. Let $x = g_\theta(z)$, then

$$\log(\rho(x)) = \log(p_Z(z)) - \log |\det J_{g_\theta}(z)|. \tag{56}$$

We choose RealNVPs (Dinh et al., 2016) for the simplicity of the transformations and the fact that we can compute efficiently the determinant of the Jacobian (since we have a closed-form). A RealNVP flow is a composition of transformations $T$ of the form

$$\forall z \in \mathbb{R}^d,\ x = T(z) = (z^1, \exp(s(z^1)) \odot z^2 + t(z^1)) \tag{57}$$

where we write $z = (z^1, z^2)$ and with $s$ and $t$ some neural networks. To modify all the components, we use also swap transformations (*i.e.* $(z^1, z^2) \mapsto (z^2, z^1)$). This transformation is invertible with $\log\det J_T(z) = \sum_i s(z_i^1)$.

For JKO-ICNN, we choose strictly convex ICNNs, and can hence invert them as well as compute the density. In this case, we do not have access to a closed-form for the Jacobian. Therefore, we used backpropagation to compute it. As this experiment is in low dimension, the computational cost is not too heavy. However, there exist stochastic methods to approximate it in greater dimension. We refer to (Huang et al., 2020) and (Alvarez-Melis et al., 2021) for more explanations.

We approximate the functional by using Monte-Carlo approximation as in Section 3.3.

For Euler-Maruyama, as in (Mokrov et al., 2021), we use kernel density estimation in order to approximate the density. We use the scipy implementation (Virtanen et al., 2020) "gaussian_kde" with the Scott's rule to choose the bandwidth.

Finally, we report on the Figure 8 the mean of the log of the symmetric Kullback-Leibler divergence over 15 Gaussians in each dimension and the 95% confidence interval.

For the training of the neural networks, we use an Adam optimizer (Kingma & Ba, 2014) with a learning rate of $10^{-4}$ for RealNVP (except for the 1st iteration where we take a learning rate of $5 \cdot 10^{-3}$) and of $5 \cdot 10^{-3}$ for JKO-ICNN. At each inner optimization step, we start from a deep copy of the last neural network, and optimize RealNVP for 200 epochs and ICNNs for 500 epochs, with a batch size of 1024.

We see on Figure 8 that the results are better than the particle schemes obtained with Euler-Maruyama (EM) with a step size of $10^{-3}$ and with either $10^3$, $10^4$ or $5 \cdot 10^4$ particles in dimension higher than 2. However, JKO-ICNN obtained better results. We hypothesize that it is due to a slightly different dynamic as discussed in Section 3.2.

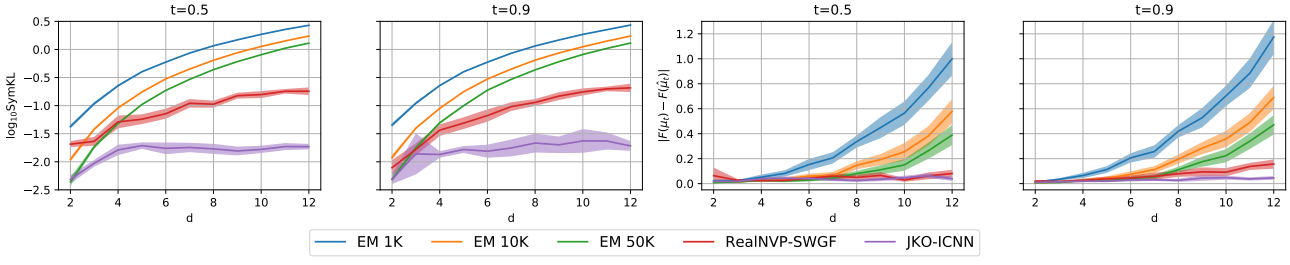



Figure 8: On the left: SymKL divergence at time $0.5$ and $0.9$ between the groundtruth of the Fokker-Planck equation at time $t$ and the solution of the SW Gradient Flow at time $t$. On the right: Absolute error between the functionals evaluated for WGF and SWGF at time $t$.

**Comparison between JKO-ICNN and SW-JKO for the Aggregation Equation.** In the same setting that Section 4.2, we report on Figure 9 the evolution of the diffusion for the interaction functional (24), that we recall here:

$$\mathcal{W}(\mu) = \frac{1}{2} \iint W(x-y) \mathrm{d}\mu(x) \mathrm{d}\mu(y) \tag{58}$$

with $W(x) = \frac{\|x\|_2^4}{4} - \frac{\|x|_2^2}{2}$. We see that we seem to lose the dilation factor of 2.

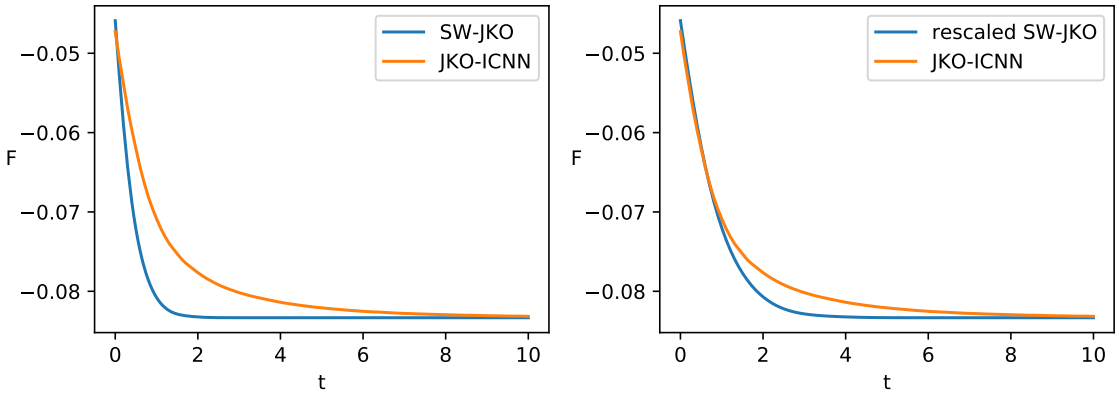



Figure 9: On the left: $\mathcal{W}$ along the dynamic returned by SW-JKO and JKO-ICNN. On the right, $\mathcal{W}$ along the dynamic returned by JKO-ICNN and SW-JKO dilated by a factor $d = 2$.

### C.2. Convergence to Stationary Distribution

Here, we want to demonstrate that, through the SW-JKO scheme, we are able to find good minima of functionals using simple generative models.

**Gaussian.** For this experiment, we place ourselves in the same setting of Section 4.1. We start from $\mu_0 = \mathcal{N}(0, I)$ and use a step size of $\tau = 0.1$ for 80 iterations in order to match the stationary distribution. In this case, the functional is

$$\mathcal{F}(\mu) = \int V(x) \mathrm{d}\mu(x) + \mathcal{H}(\mu) \tag{59}$$

with $V(x) = -\frac{1}{2}(x-b)^T A(x-b)$, and the stationary distribution is $\rho^*(x) \propto e^{-V(x)}$, hence $\rho^* = \mathcal{N}(b, A^{-1})$.

We generate 15 Gaussians for $d$ between 2 and 12, and $d \in \{20, 30, 40, 50, 75, 100\}$. Due to the length of the diffusion, and to numerical unstabilities, we do not report results obtained with JKO-ICNN. In Figure 1, we showed the results in low dimension (for $d \in \{2, \dots, 12\}$) and the unstability of JKO-ICNN. We report on Figure 10 the SymKL also in higher dimension.

We use 200 epochs of each inner optimization and an Adam optimizer with a learning rate of $5 \cdot 10^{-3}$ for the first iteration and $10^{-3}$ for the rest. We also use a batch size of 1000 sample.

**Bayesian Logistic Regression.** For the Bayesian logistic regression, we have access to covariates $s_1, \dots, s_n \in \mathbb{R}^d$ with their associated labels $y_1, \dots, y_n \in \{-1, 1\}$. Following (Liu & Wang, 2016; Mokrov et al., 2021), we put as prior on the regression weights $w$, $p_0(w|\alpha) = \mathcal{N}(w; 0, \frac{1}{\alpha})$ with $p_0(\alpha) = \Gamma(\alpha; 1, 0.01)$. Therefore, we aim at learning the posterior $p(w, \alpha|y)$:

$$p(w, \alpha|y) \propto p(y|w, \alpha) p_0(w|\alpha) p_0(\alpha) = p_0(\alpha) p_0(w|\alpha) \prod_{i=1}^{n} p(y_i|w, \alpha)$$

where $p(y_i|w, \alpha) = \sigma(w^T s_i)^{\frac{1+y_i}{2}} (1 - \sigma(w^T s_i))^{\frac{1-y}{2}}$ with $\sigma$ the sigmoid. To evaluate $\mathcal{V}(\mu) = \int V(x) \, \mathrm{d}\mu(x)$, we resample data uniformly.

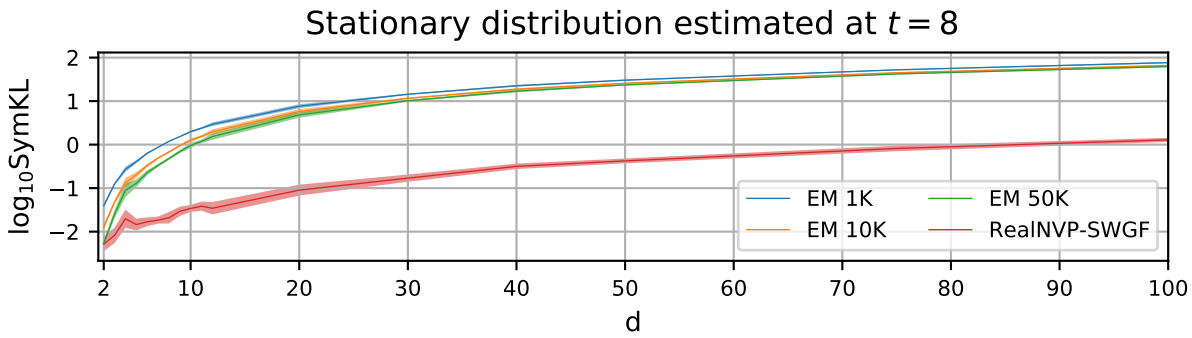



Figure 10: Symmetric KL divergence between the learned distribution at time $t = 8$ and the true stationary solution on Gaussians

In our context, let $V(x) = -\log\big(p_0(\alpha)p_0(w|\alpha)p(y|w,\alpha)\big)$, then using $\mathcal{F}(\mu) = \int V \mathrm{d}\mu + \mathcal{H}(\mu)$ as functional, we know that the limit of the stationary solution of Fokker-Planck is proportional to $e^{-V} = p(w, \alpha|y)$.

Following (Mokrov et al., 2021; Liu & Wang, 2016), we use the 8 datasets of (Mika et al., 1999) and the covertype dataset (https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html).

We report in Table 3 the characteristics of the different datasets. The datasets are loaded using the code of (Mokrov et al., 2021) (https://github.com/PetrMokrov/Large-Scale-Wasserstein-Gradient-Flows). We split the dataset between train set and test set with a 4:1 ratio.

| | covtype | german | diabetis | twonorm | ringnorm | banana | splice | waveform | image |
|---|---|---|---|---|---|---|---|---|---|
| features | 54 | 20 | 8 | 20 | 20 | 2 | 60 | 21 | 18 |
| samples | 581012 | 1000 | 768 | 7400 | 7400 | 5300 | 2991 | 5000 | 2086 |
| batch size | 512 | 800 | 614 | 1024 | 1024 | 1024 | 512 | 512 | 1024 |

Table 3: Number of features, of samples and batch size of each Dataset

We report in Table 4 the hyperparameters used for the results reported in Table 1. We also tuned the time step $\tau$ since for too big $\tau$, we observed bad results, which the SW-JKO scheme should be a good approximation of the SWGF only for small enough $\tau$.

Moreover, we reported in Table 1 the mean over 5 training. For the results obtained with JKO-ICNN, we used the same hyperparameters as Mokrov et al. (2021).

| | covtype | german | diabetis | twonorm | ringnorm | banana | splice | waveform | image |
|---|---|---|---|---|---|---|---|---|---|
| nl | 2 | 2 | 2 | 2 | 2 | 2 | 5 | 5 | 2 |
| nh | 512 | 512 | 512 | 512 | 512 | 512 | 128 | 128 | 512 |
| lr | $2e^{-5}$ | $1e^{-4}$ | $5e^{-4}$ | $1e^{-4}$ | $5e^{-5}$ | $1e^{-4}$ | $5e^{-4}$ | $1e^{-4}$ | $5e^{-5}$ |
| JKO steps | 5 | 5 | 10 | 20 | 5 | 5 | 5 | 5 | 5 |
| Iters by step | 1000 | 500 | 500 | 500 | 1000 | 500 | 500 | 500 | 500 |
| $\tau$ | 0.1 | $10^{-6}$ | $5 \cdot 10^{-6}$ | $10^{-8}$ | $10^{-6}$ | 0.1 | $10^{-6}$ | $10^{-8}$ | 0.1 |
| batch size | 1024 | 1024 | 1024 | 1024 | 1024 | 1024 | 1024 | 512 | 1024 |

Table 4: Hyperparameters for SWGFs with RealNVPs. nl: number of coupling layers in RealNVP, nh: number of hidden units of conditioner neural networks, lr: learning rate using Adam, JKO steps: number of SW-JKO steps, Iters by step: number of epochs for each SW-JKO step, $\tau$: the time step, batch size: number of samples taken to approximate the functional.

### C.3. Influence of the Number of Projections

It is well known that the approximation of Sliced-Wasserstein is subject to the curse of dimensionaly through the Monte-Carlo approximation (Nadjahi et al., 2020). We provide here some experiment to quantify this influence. However, first note that the goal is not to minimize the Sliced-Wasserstein distance, but rather the functional, SW playing mostly a regularizer role. Experiments on the influence of the number of experiments to approximate the SW have already been conducted (see *e.g.* Figure 2 in (Nadjahi et al., 2020) or Figure 1 in (Deshpande et al., 2019)).

Here, we take the same setting of Section 4.1, *i.e.* we generate 15 random Gaussians, and then vary the number of projections and report the Symmetric Kullback-Leibler divergence on Figure 11. We observe that the results seem to improve with the number of projections until it reach a certain plateau. The plateau seems to be attained for a bigger number of dimension in high dimension.

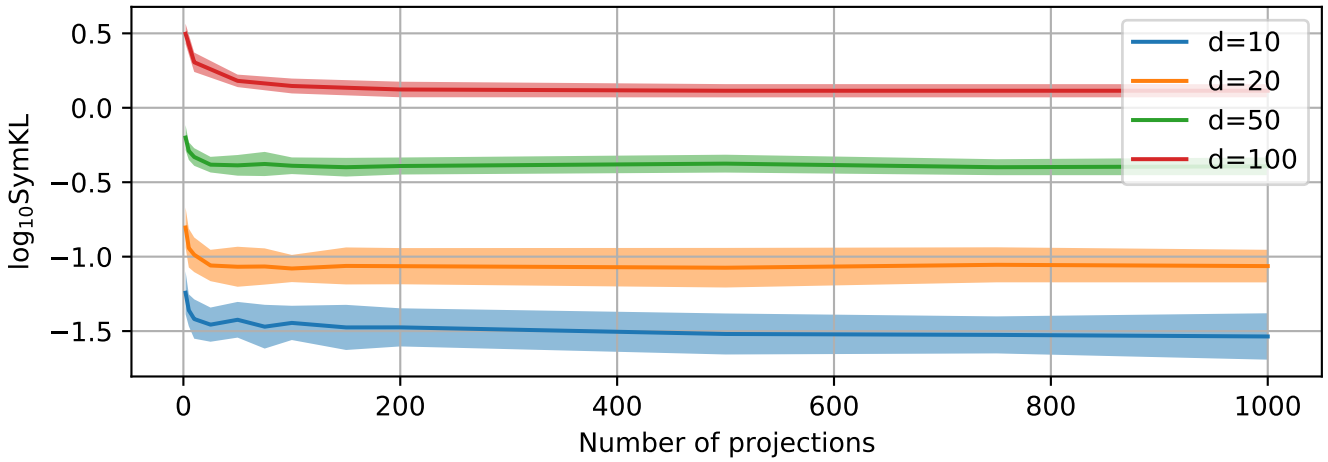



Figure 11: Impact of the number of projections for a fixed number of epochs.

### C.4. Aggregation Equations

Here, we use as functional

$$\mathcal{W}(\mu) = \iint W(x-y)\mathrm{d}\mu(x)\mathrm{d}\mu(y). \tag{60}$$

Carrillo et al. (2021) use a repulsive-attractive interaction potential, for $a > b \geq 0$,

$$W(x) = \frac{\|x\|^a}{a} - \frac{\|x\|^b}{b} \tag{61}$$

using the convention $\frac{\|x\|^0}{0} = \ln(\|x\|)$. For some values of $a$ and $b$, there is existence of stable equilibrium state (Balagué et al., 2013).

**Dirac Ring.** First, for the Dirac ring example, we take $a = 4$ and $b = 2$. Then, $W$ is a repulsive-attractive interaction potential (repulsive in the short range, and attractive in the long range because $b < a$ (Balagué et al., 2013)), we show the densities over time obtained on a discrete grid (Figure 12), by learning the position of the particles (Figure 13) and with the FCNN (Figure 14). For the last two, the density reported is obtained with kernel density estimation where we chose by hand the bandwidth to match well the sampled points. We also report the evolution of the density for the JKO-ICNN scheme. In Figure 15, we show the evolution of the density obtained by the change of variable formula. We observe that values seem to explode which may due to the fact that the stationary solution may not have a density or to numerical unstabilities. We report in Figure 16 the densities obtained with a kernel density estimation. We also report the particles generated through a FCNN, the particle evolution and JKO-ICNN in Figure 17. We observe that the particles match perfectly the ring. For the FCNN, there seem to be some noise. JKO-ICNN recover also well the ring but particles seem to not be uniformly distributed over the ring.

For the SW-JKO scheme, we take $\tau = 0.05$ and run it for 200 steps (from $t = 0$ to $t = 10$) starting from $\mu_0 = \mathcal{N}(0, I)$. For JKO-ICNN, we choose $\tau = 0.1$ and run it for 100 steps as the diffusion was really long.

We take a grid of $50 \times 50$ samples on $[-1, 1]^2$. To optimize the weights, we used an SGD optimizer with a momentum of 0.9 with a learning rate of $10^{-4}$ for 300 epochs by inner optimization scheme. For particles, we optimized 1000 particles (sampled initially from $\mu_0$) with an SGD optimizer with a momentum of 0.9, a learning rate of 1 and 500 epochs by JKO step. We take a FCNN composed of 5 hidden layers with 64 units and leaky relu activation functions. They are optimized with an Adam optimizer with a learning rate of $10^{-4}$ (except for the first iteration where we take a learning rate of $5 \cdot 10^{-3}$) for 400 epochs by JKO step. For JKO-ICNN, we choose the same parameters as for the previous experiments. In any case, we take $n_\theta = 10^4$ projections to approximate $SW_2$ and a batch size of 1000.

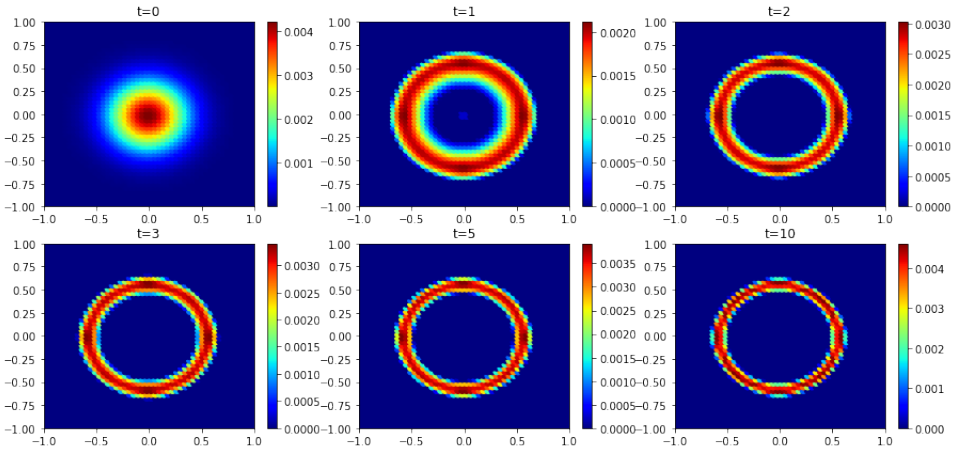



Figure 12: Density over time of the solution of the aggregation equation learned over the discre grid.

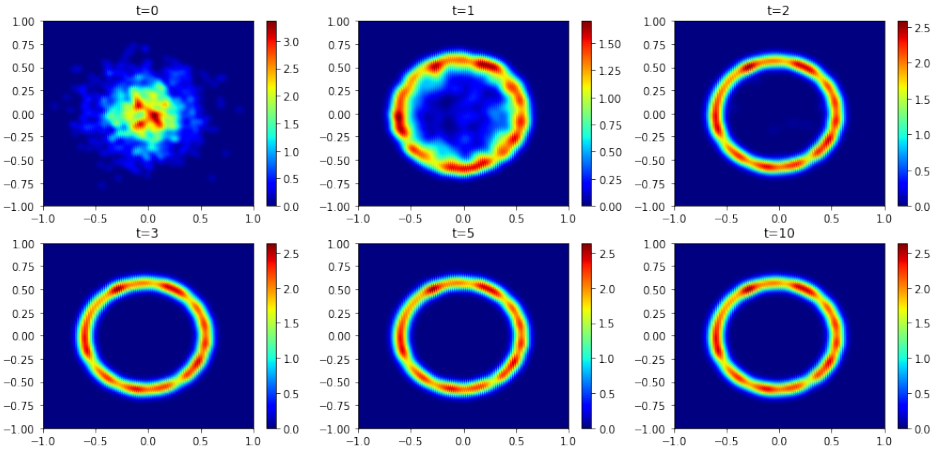



Figure 13: Density over time of the solution of the aggregation equation by learning particles.

On Figure 18, we plot on the two first columns the stationary density learned by the different methods. On the third column, we plot the evolution of the functional along the flows.

The functionals seem to converge towards the same value. However, JKO-ICNN seems to not be able to capture the right distribution. We also note that the curve for the FCNN is not very smooth, which can probably be explained by the fact that we take independent samples at each step.

On Figure 19, we show the evolution of the functionals along different learned flows. We observe that for the discretized grid, we obtain the worse results which is understandable since we have discretization error. The particle is the most stable as particle's position do not move anymore once the stationary state is reached. For the FCNN, we observe oscillations

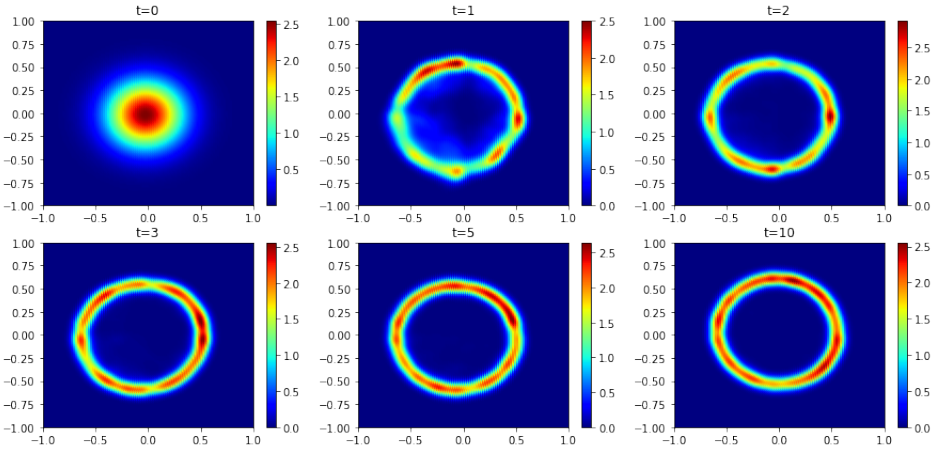



Figure 14: Density over time of the solution of the aggregation equation approximated with a FCNN.

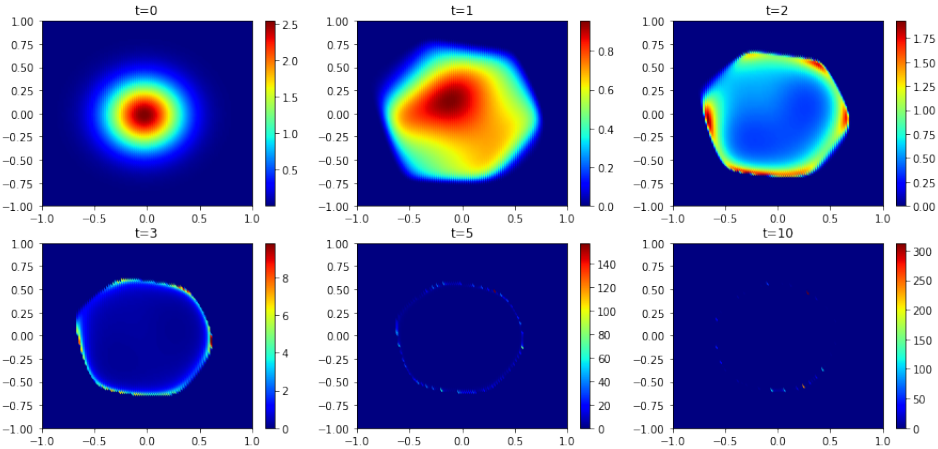



Figure 15: Density over time of the solution of the aggregation equation approximated with JKO-ICNN.

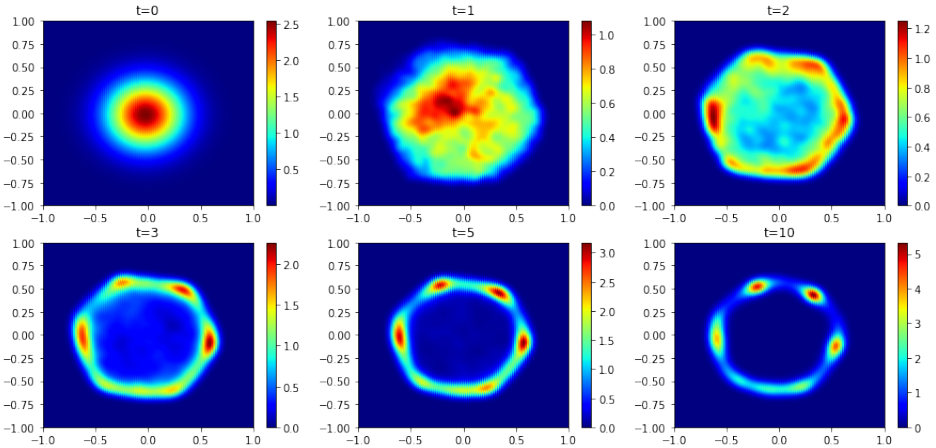
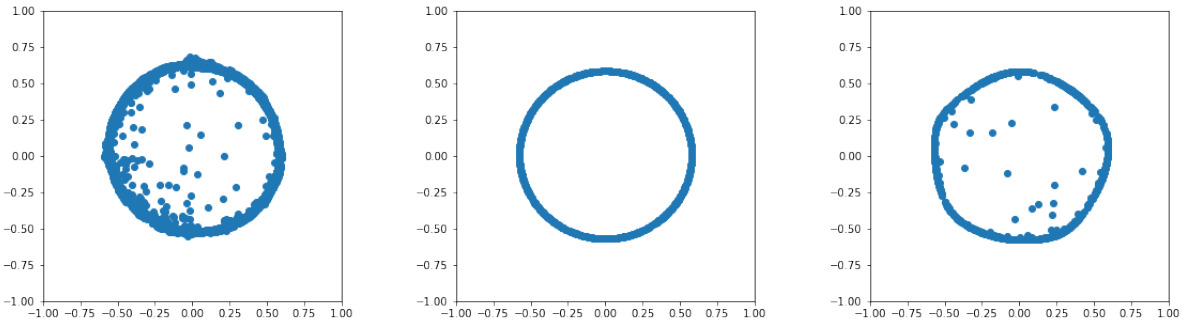



Figure 16: Density over time of the solution of the aggregation equation approximated with JKO-ICNN and kernel density estimator.



(a) Samples from the FCNN

(b) Samples learned by optimizing over particles

(c) Samples from JKO-ICNN

Figure 17: Samples of the stationary distribution

which are due to the fact that we take independent samples at each time t. Finally, the JKO-ICNN scheme seems to converge toward the same value as the SW-JKO scheme with FCNNs.

We also tried to use normalizing flows with this functional. We observed that the training seems harder than with the FCNN. Indeed, with simple flows such as RealNVP, the model has a lot of troubles of learning the Dirac ring, probably because of the hole. More generally, since normalizing flows are bijective transformations, they must preserve topological properties, and therefore do not perform well when the standard distribution and the target distribution do not share the same topology (*e.g.* do not have the same number of connected components or "holes" as it is explained in (Cornish et al., 2020)). For CPF, it worked slightly better, but was not able to fully recover the ring (at least at time $t = 5$) as we can see on Figure 20. Morover, the training time for CPF was really huge compared to the FCNN.

**Other Functionals.** As in section 3 of (Carrillo et al., 2021), we also tried to use

$$W(x) = \frac{\|x\|^2}{2} - \log(\|x\|) \tag{62}$$

as interaction potential with a FCNN. We find well that the steady state is an indicator function on the centered disk of radius 1 (Figure 21).

Another possible functional without using internal energies is to add a drift term

$$\int V(x)\rho(x)\mathrm{d}x. \tag{63}$$

Then, the Wasserstein gradient flow is solution to

$$\partial_t \rho_t = \mathrm{div}(\rho \nabla (W * \rho)) + \mathrm{div}(\rho \nabla V). \tag{64}$$

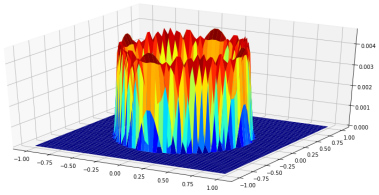
Carrillo et al. (2021) use $W(x) = \frac{\|x\|^2}{2} - \log(\|x\|)$ and $V(x) = -\frac{\alpha}{\beta}\log(\|x\|)$ with $\alpha = 1$ and $\beta = 4$. Then, it can be shown (see (Carrillo et al., 2021; Chen & Kolokolnikov, 2014; Carrillo et al., 2015)) that the steady state is an indicator function on a torus of inner radius $R_\mathrm{i} = \sqrt{\frac{\alpha}{\beta}}$ and outer radius $R_\mathrm{o} = \sqrt{\frac{\alpha}{\beta} + 1}$ which we observe on Figure 22 and Figure 23.

For this last experiment, we observed some unstabily issues in the training phase. It may be due to non-locality of the interaction potential $W$ as it is stated in (Carrillo et al., 2021).

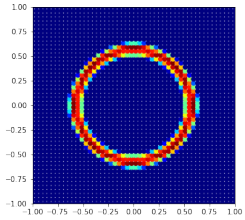
### C.5. Sliced-Wasserstein Flows

In this experiment, we aim at minimizing the following functional:

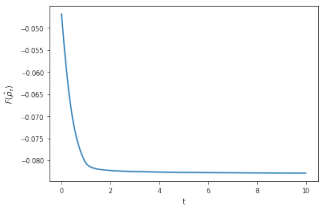
$$\mathcal{F}(\mu) = \frac{1}{2} SW_2^2(\mu, \nu) + \lambda \mathcal{H}(\mu) \tag{65}$$



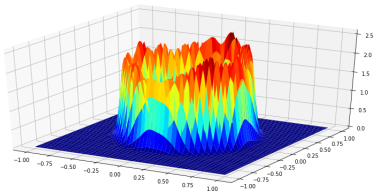
(a) Steady state on the discretized grid



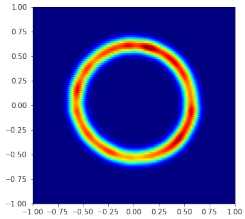
(b) Steady state on the discretized grid



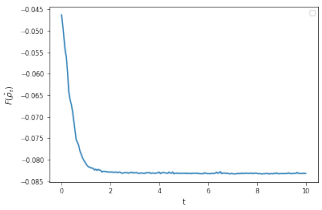
(c) Evolution of the functional



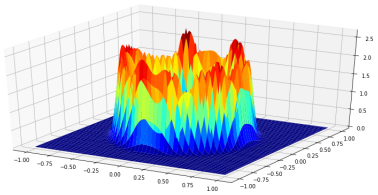
(d) Steady state for the fully connected neural network



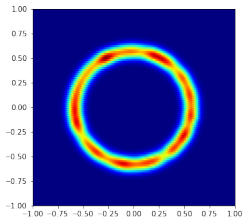
(e) Steady state for the fully connected neural network



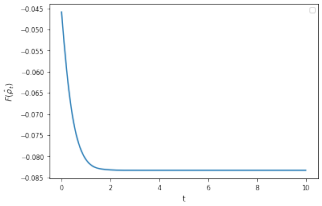
(f) Evolution of the functional



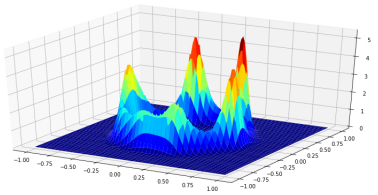
(g) Steady state for particles



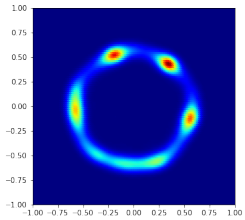
(h) Steady state for particles



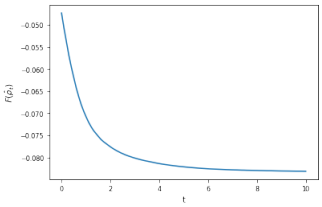
(i) Evolution of the functional



(j) Steady state for JKO-ICNN with KDE



(k) Steady state for JKO-ICNN with KDE



(l) Evolution of the functional

Figure 18: Steady state and evolution of the functional of the aggregation equation for $a = 4$, $b = 2$.

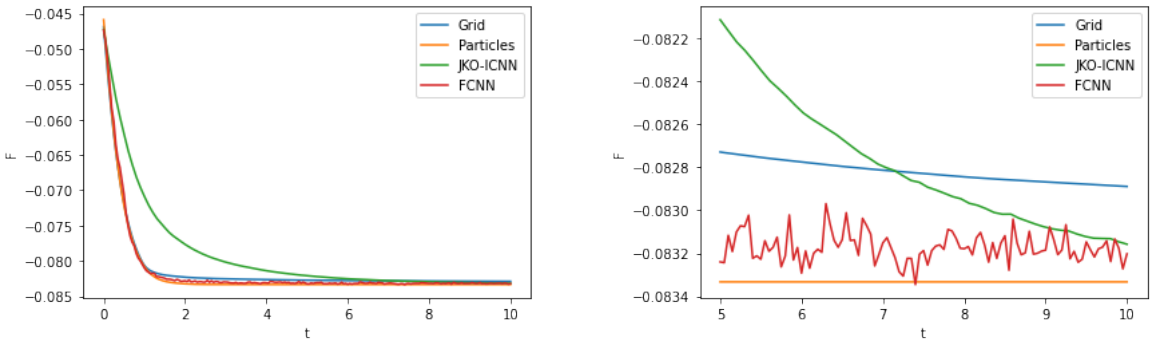
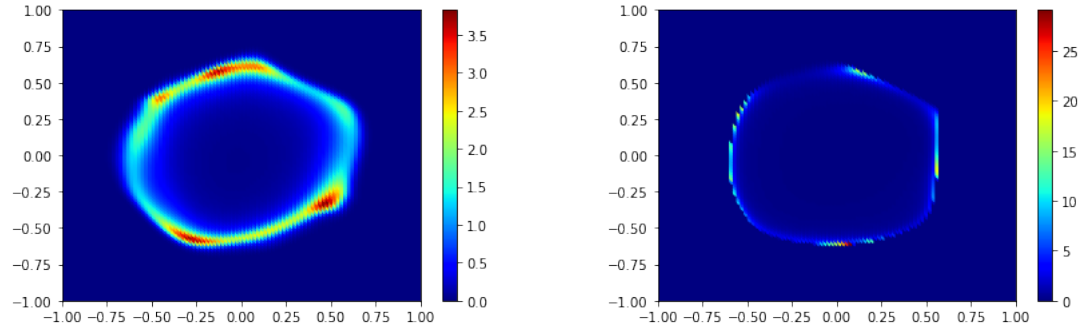


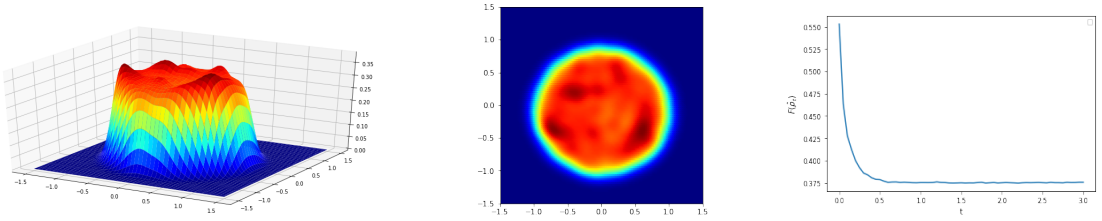

Figure 19: Evolution of the aggregation functional along different flows.



(a) Density learned by SWGF with CPF at time $t = 5$

(b) Density learned by JKO-ICNN at time $t = 4$

Figure 20: Density learned for the aggregation equation for CPF and JKO-ICNN.



(a) Density of the steady state

(b) Density of the steady state

(c) Evolution of the functional over time

Figure 21: Steady state and evolution of the functional for $W(x) = \frac{\|x\|^2}{2} - \log(\|x\|)$.

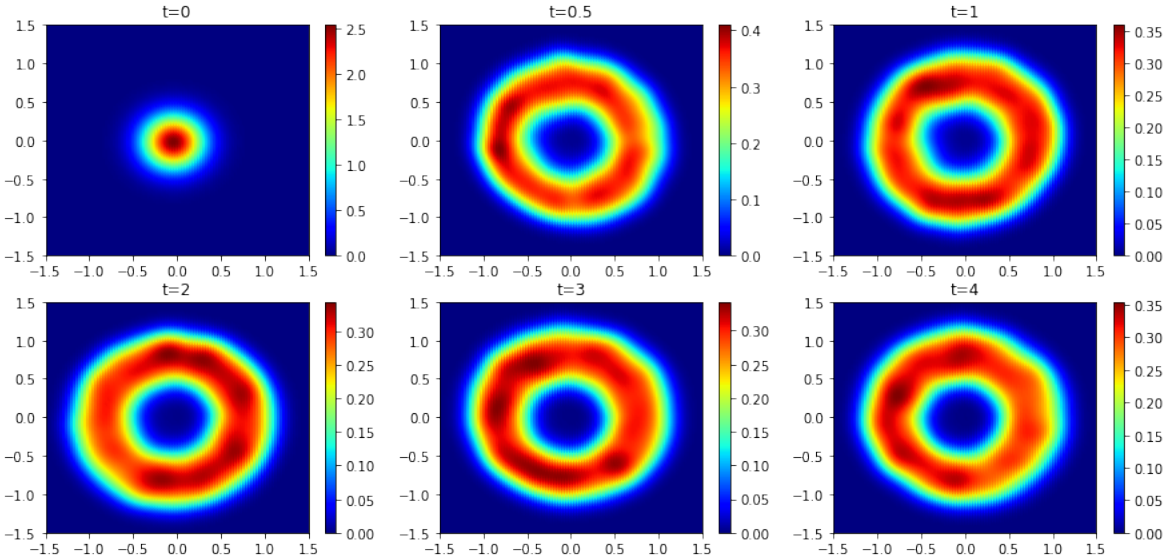
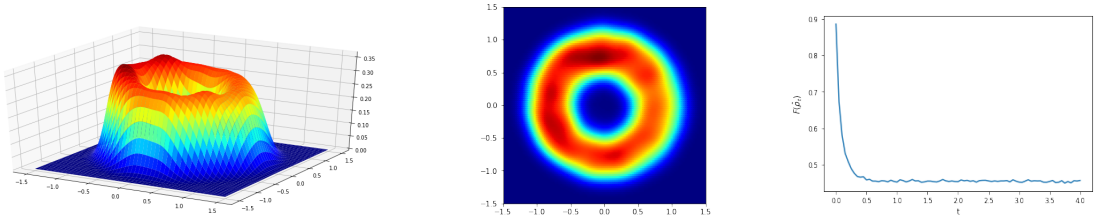



Figure 22: Density over time of the solution of the aggregation-drift equation approximated with a FCNN.



(a) Density of the steady state

(b) Density of the steady state

(c) Evolution of the functional over time

Figure 23: Steady state and evolution of the functional for the aggregation-drift equation.

where $\nu$ is some target distribution from which we have access to samples. In this section, we use MNIST (LeCun & Cortes, 2010), FashionMNIST (Xiao et al., 2017) and CelebA (Liu et al., 2015).

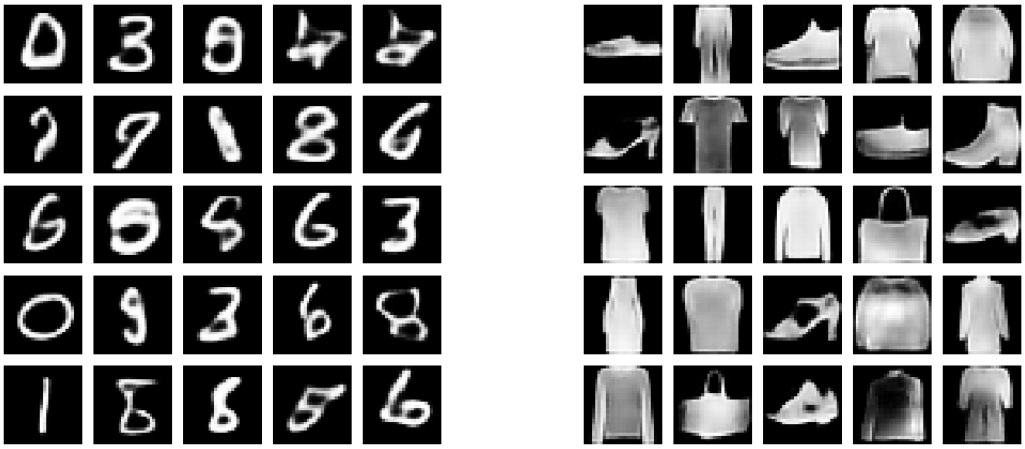
We report the Fréchet Inception Distance (FID) (Heusel et al., 2017) for MNIST and FashionMNIST between $10^4$ test samples and $10^4$ generated samples. As they are gray images, we duplicate the gray levels into 3 channels. Moreover, we use the code of Dai & Seljak (2021) (available at `https://github.com/biweidai/SINF`) and reported their result for SWF (in the ambient space) and SWAE in Table 2. For SWF in the latent space, we used our own implementation.

**With a Pretrained Autoencoder.** First, we optimize the functional in the latent space of a pretrained autoencoder (AE). We choose the same AE as Liutkus et al. (2019) which is available at `https://github.com/aliutkus/swf/blob/master/code/networks/autoencoder.py`. We report results for a latent space of dimension $d = 48$.

In this latent space, we applied a FCNN and a RealNVP. In either case, we used $n_\theta = 10^3$ projections to approximate $SW_2$. We chose a batch size of 128 samples. The FCNN was chosen with 5 hidden layers of 512 units with leaky relu activation function. The RealNVP was composed of 5 coupling layers, with FCNN as scaling and shifting networks (with 5 layers and 100 hidden units). We trained the networks at each inner optimization step with a learning rate of $5 \cdot 10^{-3}$ for RealNVP (and $10^{-2}$ for the first iteration) and of $10^{-3}$ for the FCNN (and $5 \cdot 10^{-3}$ for the first itration) during 1000 epochs.

Following Liutkus et al. (2019), we choose $\tau = 0.5$ and $\lambda = 0$. We run it for 10 outer iterations.

We report in Figure 24 the result obtained with FCNN on MNIST and FashionMNIST. Overall, the results seem quite comparable and our method seems to perform well compared to other methods applied in latent spaces.



(a) FID=19.3 (b) FID=41.7

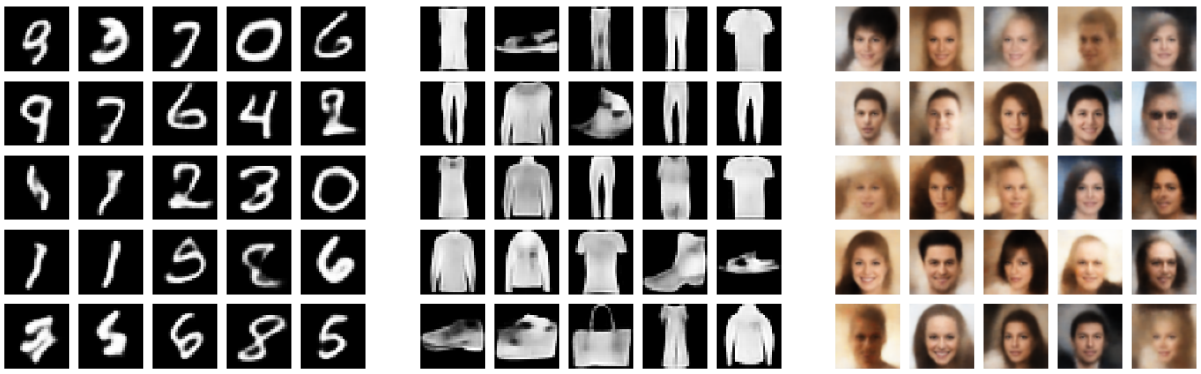
Figure 24: Generated sample obtained through a pretrained decoder + FCNN.

On Figure 25, we report samples obtained with RealNVP on MNIST, FashionMNIST and CelebA. For CelebA, we choose the same autoecoder with a latent space of dimension 48 and we ran the SW-JKO scheme for 20 steps with $\tau = 0.1$.

We can also optimize directly particles in the latent space and we show it on CelebA on Figure 26 for $\tau = 0.1$ and for 10 steps.

**In the Original Space.** We report here the results obtained by running the SW-JKO scheme in the original spaces of images, which are very high dimensional ($d = 784$ for MNIST and FashionMNIST). We obtained worse results than in the latent space. Notice that we used only 1000 projections, and ran it for 50 outer iterations with a step size of $\tau = 5$.

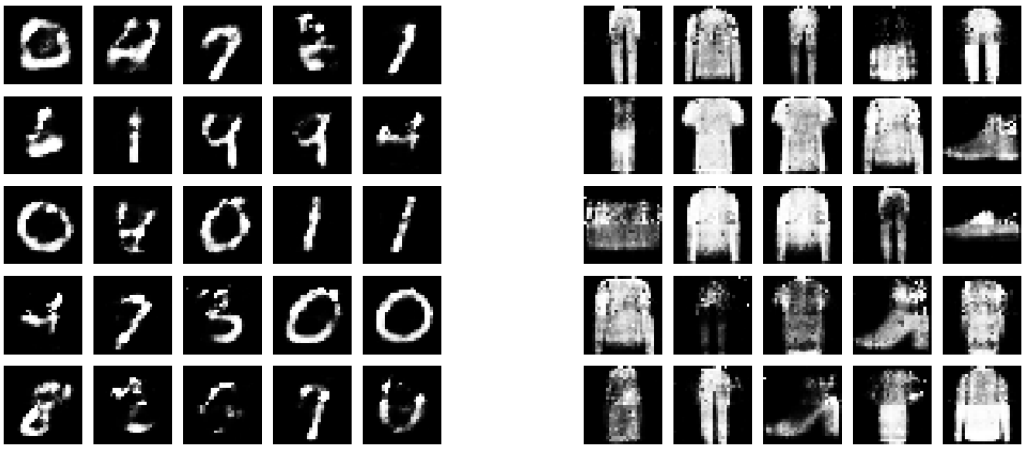
On Figure 27, we use a RealNVP and add a uniform dequantization (Ho et al., 2019) and learned it in the logit space as it is done in (Dinh et al., 2016; Papamakarios et al., 2017) because using normalizing flows need continuous data. The RealNVP is composed here of 2 coupling layers with FCNN also composed of 2 layers and 512 hidden units. We choose $\tau = 5$, and run it for 20 steps. At each inner optimization problem, the neural networks are optimized with an Adam optimizer with a learning rate of $10^{-2}$ and for 500 epochs.



(a) FID=17.8 (b) FID=40.6 (c) FID=90.8

Figure 25: Generated sample obtained through a pretrained decoder + RealNVP.

Figure 26: Particle through SW-JKO with $\tau = 0.1$ for 10 steps.

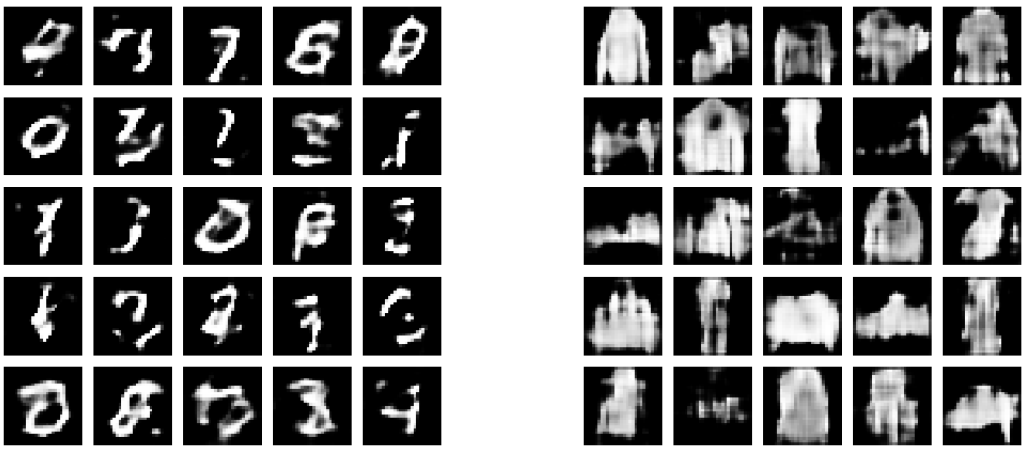


(a) FID=88.1 (b) FID=95.5

Figure 27: Generated sample obtained in the original space with RealNVP.

We also report results obtained using a convolutional neural network (CNN) in Figure 28. The idea here is that we can capture inductive bias, as it is well known that CNNs are efficient for image-related tasks. We obtained a slightly better FID on MNIST, but worse results on FashionMNIST. In term of quality of image, the generated samples do not seem better.

For the CNN, we choose a latent space of dimension 100, and we first apply a linear layer into a size of $128 \times 7 \times 7$. Then we apply 3 convolutions layers of (kernel_size, stride, padding) being respectively $(4, 2, 1)$, $(4, 2, 1)$, $(3, 1, 1)$. All layers are followed by a leaky ReLU activation, and a sigmoid is applied on the output.



(a) FID=69 (b) FID=102

Figure 28: Generated sample obtained in the original space with CNN.

Nevertheless, samples obtained in the space of image look better than those obtained through the particle scheme induced in (Liutkus et al., 2019).