UNIVERSITÉ PARIS-SACLAY

M2 Internship Report: Continuous Normalising Flow

Quentin Giton[1]

Orsay, France

11 septembre 2024

[1] quentin.giton@universite-paris-saclay.fr

Abstract

This report explores the application of Continuous Normalising Flow (CNF) within the framework of generative models, a crucial tool in statistics and machine learning for learning and sampling from data-generating distributions. Unlike traditional models such as Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs), CNFs offer a more flexible approach by employing continuous dynamics to transform data densities into target distributions. The study begins with an introduction to the problem of optimal transport, specifically Monge's problem, and proceeds to detail the implementation of the Jordan-Kinderlehrer-Otto (JKO) scheme. This approach is particularly novel as it leverages Kantorovich's dual potentials, allowing the problem to be reformulated into a series of convex sub-problems solvable via stochastic gradient descent. The report also proves convergence of the proposed stochastic gradient method. Numerical experiments are conducted to validate the theoretical results, although challenges such as the handling of large datasets and the occurrence of empty Laguerre cells are noted. The findings of this internship lay the groundwork for further research in CNF, paving the way for more robust and scalable generative models.

Contents

Introduction

Generative models have wide applications in statistics and machine-learning to infer data-generating distributions and to sample from the model distributions learned from the data. In addition to widely used deep generative models such as variational auto-encoders (VAE) [7, 8] and generative adversarial networks (GAN) [4, 5], normalising flow models [9] have been popular and with great potential. The flow models learns the data distribution through an invertible mapping between the data density μ on $\mathbb{R}^d$ and the target standard multivariate Gaussian density $\nu \sim \mathcal{N}(0, I_d)$. According to the pioneering article [13], these flow models are characterised by a function $F(\cdot, \theta)$ obtained by composition of simple invertible functions $f_t(\cdot, \theta)$ as follows

$$F(\cdot, \theta) = f_N(\cdot, \theta) \circ \cdots \circ f_1(\cdot, \theta). \tag{0.1}$$

The parameter θ is learned by a stochastic gradient method seeking to minimise a notion of error between the standard multivariate Gaussian distribution ν and the empirical measure derived from $(F(x_k, \theta))_{1 \leq k \leq n}$.

Among various flow models, continuous normalising flows (CNFs) transport the data density to a target distribution through continuous dynamics. More precisely, one way of transporting μ to ν is to follow the flow of the PDE, known as the Fokker-Planck equation

$$\begin{cases} \partial_t \rho + \nabla \cdot (\rho v) = 0, \\ v = -\nabla \ln(\rho) - \nabla V, \end{cases} \tag{0.2}$$

where V is a potential satisfying a certain growth condition. We are interested by the flow map Φ of the associated ODE

$$\dot{X}_t = v(X_t).$$

The Jordan-Kinderlehrer-Otto (JKO) approach [6] allows us to see (0.2) as the gradient flow of the free energy

$$E(\rho) := \int_{\mathbb{R}^d} V \mathrm{d}\rho + H(\rho)$$

with respect to the Wasserstein-Monge-Kantorovich metric. Moreover, Jordan's, Kinderlehrer's and Otto's article gives a way to discretise this gradient flow over time by solving several optimisation problems one after another

$$\rho_\tau^{k+1} = \arg\min_\rho \; E(\rho) + \frac{1}{\tau} W_2^2(\rho_\tau^k, \rho).$$

Indeed, letting $\tau \to 0$, one should expect to recover a weak solution to the gradient flow

$$\dot{X}_t = -\mathrm{grad}_{W_2} E(X_t).$$

We are then able to construct a transport map T that sends ρ_τ^k to ρ_τ^{k+1} and we can recover

$$X_t = T_\tau^k \circ \cdots \circ T_\tau^0(X_0)$$

for $k \simeq \frac{t}{\tau}$, which is the analogous of (0.1).

Such an approach has already been exploited in the literature [14, 16], but only, to our knowledge, using the Benamou-Brenier dynamic approach to optimal transport [1, 15], that is writing the optimal transport problem under the Benamou-Brenier formulation

$$W_2(\mu,\nu) = \inf_{(\rho,v)\in V(\mu,\nu)} \int_0^1 \left(\int_{\mathbb{R}^d} \rho_t(x)\|v_t(x)\|^2 \mathrm{d}x \right) \mathrm{d}t$$

and trying to solve the resulting optimisation problem. The constraints set $V(\mu,\nu)$ will not be specified here, as it is not what we are interested in.

We will study a different approach based on Kantorovich's potentials. With a sufficiently fine temporal discretisation, it is possible to reformulate the problem as a succession of convex sub-problems. It seems possible to solve each of these sub-problems using a stochastic gradient descent method. We expect to be able to prove the convergence of this stochastic gradient descent using a simple and appropriate parametrisation of the Kantorovich potentials [2]. The expressiveness of the parametrisation is ensured by the successive composition of such functions.

This internship is an introductory work to the thesis whose objective is to answer those questions.

1 Fitting an arbitrary measure onto a Gaussian

Let assume we have an unknown distribution μ that we want to push-forward to another distribution $\nu = \mathcal{N}(0, I_d)$, using a transport map T. Let us recall that it means finding a map T that solves the following optimal transport problem, called Monge's problem, for a cost function c

$$(\text{MP}) = \min_{T_{\#}\mu=\nu} \int c(x, T(x))\mathrm{d}\mu(x). \tag{1.1}$$

We intent to do it using the JKO scheme [6]. The JKO scheme was first introduced as a way to numerically solve the following *Fokker-Planck equation*

$$\begin{cases} \partial_t \rho = \mathrm{div}(\rho \nabla V + \rho \nabla \ln(\rho)), \\ \rho(0) = \rho_0. \end{cases} \tag{1.2}$$

Here, V is a function called *potential*. In our context we know that, under mild assumptions on the growth of the potential V, the Fokker-Planck equation (1.2) has a unique stationary solution ρ_s that has the form of a Gibbs distribution

$$\rho_s(y) = Z^{-1} \exp(-V(y))\,. \tag{1.3}$$

Thus, taking $\rho_0 = \mu$ and $V(y) = \frac{\|y\|^2}{2}$, we will end up with $\rho_s \simeq \nu$ in a sense that will be precised later. According to Jordan's, Kinderlehrer's and Otto's article [6], we are looking to construct a sequence of measures $(\mu_t)_{t\in[\![0,N]\!]}$ such that, defining the *free energy* as

$$E(\rho) := \int_{\mathbb{R}^d} V \mathrm{d}\rho + H(\rho), \text{ where } H(\rho) := \begin{cases} \int_{\mathbb{R}^d} \rho \ln(\rho) \text{ if } \rho << \mathcal{L}^d \\ +\infty \text{ otherwise,} \end{cases}$$

it minimises

$$\mu_{t+1} := \underset{\rho\in\mathcal{P}_2(\mathbb{R}^d)}{\arg\min}\ E(\rho) + \frac{1}{\tau} W_2^2(\mu_t, \rho)$$

for all $0 \leq t \leq N-1$. Here, τ is a positive real number and $W_2^2(\mu_k, \rho)$ refers to the transport cost from μ_t to ρ for the cost $c(x, y) = \frac{1}{2}\|x - y\|^2$. The functional W_2 is a distance over the set of probability measures with finite second moment $\mathcal{P}_2(\mathbb{R}^d)$ and is called the *Wasserstein* or *Monge-Kantorovich* distance.

In what follows, we will focus on one step: construct a transport map T_t that pushes forward μ_t to μ_{t+1}, for any $0 \leq t \leq N-1$.

2 Implementing a JKO step through Kantorovich's duality

Let us consider the following optimisation problem

$$\inf_{\rho\in\mathcal{P}_2(\mathbb{R}^d)} E(\rho) + \frac{1}{\tau} W_2^2(\mu, \rho). \tag{2.1}$$

Let us recall the following two statements coming from optimal transport theory. First, the problem (1.1) is equal to the problem

$$(\mathrm{KP}) = \min_{\gamma \in \Gamma(\mu,\nu)} \int c(x,y) \mathrm{d}\gamma(x,y), \tag{2.2}$$

which is called Kantorovich's problem. Here, γ is called a *transport plan*. Second, there's the Kantorovich's duality theorem that states the equality between (2.2) and

$$(\mathrm{DP}) = \max_{\phi \oplus \psi \leq c} \int \phi \mathrm{d}\mu + \int \psi \mathrm{d}\nu, \tag{2.3}$$

where the maximum is taken over all pairs (ϕ, ψ) of functions satisfying $\phi \oplus \psi \leq c$, meaning that $\phi(x) + \psi(y) \leq c(x,y)$ for all x and y. These functions are called *Kantorovich's potentials*. Equivalently, the dual problem can be written as the unconstrained maximisation problem

$$(\mathrm{DP}) = \max_{\phi} \mathcal{K}(\phi) \quad \text{where} \quad \mathcal{K}(\phi) = \int \phi \mathrm{d}\mu + \int \phi^c \mathrm{d}\nu \tag{2.4}$$

and $\phi^c(y) = \min_x c(x,y) - \phi(x)$ is the c-transform of ϕ, a notion closely related to the Legendre-Fenchel transform in convex analysis. The function $\mathcal{K}$ is called the *Kantorovich functional*. This theorem is valid under mild assumptions on $\mathcal{K}$.

Applying Kantorovich's duality theorem on (2.1) yields the following equivalent formulation

$$\sup_{\phi \in \mathcal{C}(\mathbb{R}^d)} \inf_{\rho \in \mathcal{P}_2(\mathbb{R}^d)} E(\rho) + \frac{1}{\tau} \left(\int_{\mathbb{R}^d} \phi \mathrm{d}\mu + \int_{\mathbb{R}^d} \phi^c \mathrm{d}\rho \right). \tag{2.5}$$

According to proposition (7.20) from [12], the first order optimality condition of the sub-problem is

$$\frac{\delta E}{\delta \rho}(\rho) + \frac{\phi^c}{\tau} = C,$$

where $\frac{\delta E}{\delta \rho}$ is the *first-order variation* of E, defined by

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon} E(\rho + \varepsilon\chi)\Big|_{\varepsilon=0} = \int \frac{\delta E}{\delta \rho} \mathrm{d}\chi.$$

Using the first-order Taylor expansion of the logarithm

$$\ln(\rho + \varepsilon\chi) \underset{\varepsilon \to 0}{\simeq} \ln(\rho) + \varepsilon \frac{\chi}{\rho},$$

we can show that

$$\frac{\delta E}{\delta \rho}(\rho) = V + 1 + \ln(\rho)\,.$$

We then deduce a solution of the sub-problem

$$\rho^\star = e^C \exp\left(-\frac{\phi^c}{\tau} - \frac{V}{2}\right), \quad e^C = \left(\int_{\mathbb{R}^d} \exp\left(-\frac{\phi^c(y)}{\tau} - \frac{V(y)}{2}\right) \mathrm{d}y\right)^{-1}. \tag{2.6}$$

Plugging this back to the previous minimisation problem, we end up rewriting (2.5) as

$$\sup_{\phi \in \mathcal{C}(\mathbb{R}^d)} -\ln\left(\int_{\mathbb{R}^d} \exp\left(-\frac{\phi^c(y)}{\tau} - \frac{V(y)}{2}\right) \mathrm{d}y\right) + \frac{1}{\tau}\int_{\mathbb{R}^d} \phi \mathrm{d}\mu. \tag{2.7}$$

3 The case of semi-discrete measures

In practice, we have access to discrete observations. Thus, we will first make the assumption that μ is finitely supported: $\mu = \frac{1}{n}\sum_{i=0}^n \delta_{x_i}$. With this assumption, we enter the theory of semi-discrete optimal transport. Let us recall that in this setting, the dual problem (2.4) amounts to maximising the Kantorovich functional given by

$$\mathcal{K}(\phi) = \sum_i \int_{L_i(\phi)} (c(x_i, y) - \phi(x_i)) \mathrm{d}\nu(y) + \sum_i \mu_i \phi(x_i), \tag{3.1}$$

where the *Laguerre cells* are defined by

$$L_i(\phi) = \{y \,|\, \forall j,\ c(x_i, y) - \phi(x_i) \leq c(x_j, y) - \phi(x_j)\}. \tag{3.2}$$

Moreover, in semi-discrete OT, the c-transform of functions are very easy to compute. The optimality condition for (3.1) is the following non-linear system of equations [11]

$$\forall i,\ \nu(L_i(\phi)) = \mu_i. \tag{3.3}$$

Thus, we can rewrite (2.7) as

$$\sup_{\phi \in \mathbb{R}^n} A(\phi) = -\ln\left(\sum_{i=1}^n \int_{L_i(\phi)} \exp\left(-\frac{\|x_i - y\|^2 - 2\phi_i}{2\tau} - \frac{\|y\|^2}{2}\right) \mathrm{d}y\right) + \frac{1}{\tau n}\sum_{i=1}^n \phi_i, \tag{3.4}$$

where ϕ_i denotes $\phi(x_i)$. This problem must be concave in order to guarantee the existence and uniqueness of a maximiser.

Theorem 3.1. *The function A is concave.*

Proof. Let us first remark that the function $\phi \in \mathbb{R}^n \mapsto -\frac{1}{\tau}\phi^c$ is convex as being a maximum of affine functions, thus $\phi \mapsto 2\phi^c + \tau\|y\|^2$ is concave. Let

$$a(\phi) = -\ln\left(\int_{\mathbb{R}^d} \exp\left(-\frac{\phi^c}{\tau} - \frac{\|y\|^2}{2}\right) \mathrm{d}y\right).$$

Let $\phi^0, \phi^1 \in \mathbb{R}^n$, let $t \in [0,1]$ and let $\phi^t = (1-t)\phi^0 + t\phi^1$. Next, define for all $y \in \mathbb{R}^d$,

$$u(y) := \exp\left(-\frac{\phi^{0,c}}{\tau} - \frac{\|y\|^2}{2}\right);$$
$$v(y) := \exp\left(-\frac{\phi^{1,c}}{\tau} - \frac{\|y\|^2}{2}\right);$$
$$w(y) := \exp\left(-\frac{\phi^{t,c}}{\tau} - \frac{\|y\|^2}{2}\right).$$

The function $\exp\left(-\frac{1}{\tau}\mathrm{id}\right)$ being non-increasing, the concavity of $2\phi^c + \tau\|y\|^2$ yields

$$w(y) \leq u(y)^{1-t} v(y)^t = m(y).$$

Integrating both sides with respect to y leads to

$$\begin{aligned}\int_{\mathbb{R}^d} w(y)\mathrm{d}y &\leq \int_{\mathbb{R}^d} m(y)\mathrm{d}y \\ &= \int_{\mathbb{R}^d} u(y)^{1-t} v(y)^t \mathrm{d}y \\ &\leq \left(\int_{\mathbb{R}^d} u(y)\mathrm{d}y\right)^{1-t} \left(\int_{\mathbb{R}^d} v(y)\mathrm{d}y\right)^t,\end{aligned}$$

where the last inequality comes from Hölder's inequality applied with $\frac{1}{p} = 1-t$ and $\frac{1}{q} = t$. Finally, composing by $-\ln$, we end up with the concavity of a, and thus A also is as the sum of two concave functions. □

Problem (3.4) is thus concave and its first order optimality condition is

$$\forall i, \quad -\frac{\int_{L_i(\phi)} \exp\left(-\frac{\|x_i - y\|^2 - 2\phi_i}{2\tau} - \frac{\|y\|^2}{2}\right)\mathrm{d}y}{\tau \sum_j \int_{L_j(\phi)} \exp\left(-\frac{\|x_j - y\|^2 - 2\phi_i}{2\tau} - \frac{\|y\|^2}{2}\right)\mathrm{d}y} = -\frac{1}{\tau n}, \tag{3.5}$$

which says that the mass of each Laguerre cell should be equal to $\frac{1}{n}$ with respect to the measure $\sigma = \exp\left(-\frac{\phi^c}{\tau} - V\right)\mathcal{L}^d$.

If we get rid of the logarithm in (3.4), we end up with a convex problem

$$\inf_{\phi \in \mathbb{R}^n} \int_{\mathbb{R}^d} \exp\left(-\frac{\phi^c(y)}{\tau} - \frac{\|y\|^2}{2}\right)\mathrm{d}y - \frac{1}{\tau n}\sum_i \phi_i \tag{3.6}$$

whose first order condition is

$$\forall i, \quad \frac{1}{\tau}\int_{L_i(\phi)} \exp\left(-\frac{\|x_i - y\|^2 - 2\phi_i}{2\tau} - \frac{\|y\|^2}{2}\right)\mathrm{d}y = \frac{1}{\tau n}, \tag{3.7}$$

which also says that the mass of each Laguerre cell should be equal to $\frac{1}{n}$ with respect to the measure $\sigma = \exp\left(-\frac{\phi^c}{\tau} - V\right)\mathcal{L}^d$. Thus, we will solve (3.6) in order to solve (3.4). One of the greatest advantage of problem (3.6) is that it can be written as an expectation

$$\inf_{\phi\in\mathbb{R}^n} \mathbb{E}_{y\sim\mathcal{N}(0,I_d)}\left[(2\pi)^{d/2}\exp\left(-\frac{\phi^c(y)}{\tau}\right) - \frac{1}{\tau n}\sum_i \phi_i\right], \tag{3.8}$$

which allows us to tackle it, even in high dimension, using stochastic gradient methods. The following theorem helps us proving the convergence of the stochastic gradient descent scheme that we will use to solve (3.8).

Theorem 3.2. *The optimal potential $\phi^\star$ solving (3.8) for the L^∞ norm. Said differently, there exist real numbers a, b such that $\phi^\star \in K = [a,b]^d$.*

We can now build a stochastic gradient descent algorithm in order to solve numerically (3.8). For all $k \geq 0$, define

$$\begin{cases} y_k \sim \mathcal{N}(0, I_d), \\ i_k := \underset{i}{\operatorname{argmin}}\ \frac{1}{2}\|x_i - y_k\|^2 - \phi_i^k, \\ h_k := \dfrac{1}{\sqrt{k}}, \\ \forall j,\ \psi_j^k = \phi_j^k - \dfrac{h_k}{2\tau}(2\pi)^{d/2}\left(\delta_{i_k,j}\exp\left(-\dfrac{\|x_{i_k} - y_k\|^2 - 2\phi_{i_k}^k}{2\tau}\right) - \dfrac{1}{n}\right), \\ \phi^{k+1} = \operatorname{proj}_K(\psi^k), \end{cases} \tag{3.9}$$

where proj_K is the projection onto the square K and δ is the Kronecker delta function.

Theorem 3.3. *The sequence $(\phi^k)_{k\in\mathbb{N}}$ defined by (3.9) converges towards a minimiser $\phi^\star$ of (3.8).*

To prove it, we will try to apply a the following theorem, proven in [3]. Let

$$L(y,\phi) = (2\pi)^{d/2}\exp\left(-\frac{\phi^c(y)}{\tau}\right) - \frac{1}{\tau n}\sum_{i=1}^n \phi_i$$

and

$$\mathcal{L}(\phi) = \mathbb{E}_{y\sim\mathcal{N}(0,I_d)}\left[L(y,\phi)\right].$$

Theorem 3.4. *Assume that*

1. *the mapping $\phi \mapsto L(y,\phi)$ is convex and differentiable for all y,*
2. *there exists a constant $c_0 > 0$ such that $\mathbb{E}_{y\sim\mathcal{N}(0,I_d)}\left[\|\nabla_\phi L(y,\phi)\|^2\right] \leq c_0$ for all ϕ,*

3. *there exists* $\phi^\star \in \arg\min_\phi \mathcal{L}(\phi)$,

4. *the sequence* $(h_k)_{k\in\mathbb{N}}$ *is deterministic.*

Then, the iterates of the stochastic gradient algorithm $(\phi^k)_{k\in\mathbb{N}}$ *satisfy the convergence guarantee*

$$\mathbb{E}\left[\mathcal{L}(\bar{\phi}_h^k) - \mathcal{L}(\phi^\star)\right] \leq \frac{\mathbb{E}\left[\|\phi^0 - \phi^\star\|^2\right] + c_0 \sum_{l=1}^k h_l^2}{2\sum_{l=1}^k h_l},$$

where $\bar{\phi}_h^k = \dfrac{\sum_{l=1}^k h_l \phi^l}{\sum_{l=1}^k h_l}$ *is a convex combination of all previous iterates.*

In order to apply theorem 3.4 to our case, we only need to show point 2. since the other ones are easily satisfied. Showing point 2. is not trivial and we need some tools. We first start with a lemma.

Lemma 3.5. *The d-dimensional error function*

$$\mathrm{erf}_d(x) = \frac{\displaystyle\int_0^x e^{-u^2} u^{d-1} \mathrm{d}u}{\displaystyle\int_0^\infty e^{-u^2} u^{d-1} \mathrm{d}u} \tag{3.10}$$

can be expanded as

$$\mathrm{erf}_d(x) = 1 - \frac{C}{2} e^{-x^2} \sum_{n=0}^\infty \left(\prod_{k=1}^n \frac{d}{2} - n\right) x^{d-2(n+1)},$$

where

$$C = \left(\int_0^\infty e^{-u^2} u^{d-1} \mathrm{d}u\right)^{-1}$$

Proof. Let us write

$$\mathrm{erf}_d(x) = C \int_0^x e^{-u^2} u^{d-1} \mathrm{d}u,$$

we first apply the change of variables $t = u^2$ so that

$$\mathrm{erf}_d(x) = \frac{C}{2} \int_{x^2}^\infty e^{-t} t^{d/2-1} \mathrm{d}t.$$

Then, a first step of integration by parts yields

$$\mathrm{erf}_d(x) = \frac{C}{2} e^{-x^2} x^{d-2} + \frac{C}{2}\left(\frac{d}{2} - 1\right) \int_{x^2}^\infty e^{-t} t^{d/2-2} \mathrm{d}t,$$

and another one yields

$$\begin{aligned}\mathrm{erf}_d(x) = \frac{C}{2}e^{-x^2}x^{d-2} + \frac{C}{2}\left(\frac{d}{2}-1\right)e^{-x^2}x^{d-4} \\ + \frac{C}{2}\left(\frac{d}{2}-1\right)\left(\frac{d}{2}-2\right)\int_{x^2}^{\infty} e^{-t}t^{d/2-3}\mathrm{d}t.\end{aligned}$$

By induction, we show the formula (3.10). □

This lemma will help us to prove the following theorem.

Theorem 3.6. *The optimal potential $\phi^\star$ solving (3.8) is bounded for the L^∞ norm. Said differently, there exist real numbers a, b such that $\phi^\star \in K = [a,b]^d$.*

Proof. Let ϕ be the optimal potential solving (3.8) and define ϕ_{i_0} and ϕ_{i_1} as the minimal and the maximal components of ϕ.

Step 1: Controlling the maximal component of ϕ. We have

$$\forall y \in \mathbb{R}^d,\ \phi^c(y) \leq \frac{\|y - x_{i_1}\|^2}{2} - \phi_{i_1},$$

from which we deduce

$$\exp\left(-\frac{\phi^c(y)}{\tau}\right) \geq \exp\left(-\frac{\|x_{i_1} - y\|^2 - 2\phi_{i_1}}{2\tau}\right),$$

finally we get

$$\begin{aligned}1 &= \int_{\mathbb{R}^d} \exp\left(-\frac{\phi^c(y)}{\tau}\right)\exp\left(-\frac{\|y\|^2}{2}\right)\mathrm{d}y \\ &\geq \exp\left(\frac{\phi_{i_1}}{\tau}\right)\int_{\mathbb{R}^d}\exp\left(-\frac{\|x_{i_1}-y\|^2}{2\tau}\right)\exp\left(-\frac{\|y\|^2}{2}\right)\mathrm{d}y \\ &= C_0 \cdot \exp\left(\frac{\phi_{i_1}}{\tau}\right),\end{aligned}$$

and hence ϕ_{i_1} can't be too large. More precisely, $\phi_{i_1} \leq -\tau\ln(C_0)$.

Step 2: The i_0-th Laguerre cell is far away. Let M be the oscillation of ϕ, *i.e.* $M = \max_i \phi_i - \min_i \phi_i = \phi_{i_1} - \phi_{i_0}$. Let assume that the points x_i are contained in a certain ball $\mathcal{B}(0,R)$ of radius $R > 0$. Recall the definition of a Laguerre cell

$$L_i(\phi) = \{y \,|\, \forall j,\ c(x_i,y) - \phi(x_i) \leq c(x_j,y) - \phi(x_j)\}.$$

The i_0-th Laguerre cell is contained in some other set

$$L_{i_0}(\phi) \subseteq \{y \,|\, \|x_{i_0} - y\|^2 + 2M \leq \|x_{i_1} - y\|^2\},$$

from which we deduce that

$$y \in L_{i_0}(\phi) \Longrightarrow \frac{2M + \|x_{i_0}\|^2 - \|x_{i_1}\|^2}{2} \leq \langle y | x_{i_0} - x_{i_1} \rangle.$$

In other words, the i_0-th Laguerre cell is contained in a closed half-space H_M that is at distance at least $\alpha(M) = \frac{|2M + \|x_{i_0}\|^2 - \|x_{i_1}\|^2|}{2\|x_{i_0} - x_{i_1}\|}$ from the origin.

Step 3: controlling the minimal component of ϕ. By definition of ϕ_{i_1} and since all the points x_i are in a ball of radius R, we have

$$\begin{aligned}\phi^c(y) &\geq \min_i \frac{\|x_i - y\|^2}{2} - \phi_{i_1} \\ &\geq \frac{\|y\|^2}{2} - \|y\|R - \phi_{i_1}.\end{aligned}$$

We then deduce another inequality on ϕ_{i_1}

$$\begin{aligned}\frac{1}{N} &= \int_{L_{i_0}(\phi)} \exp\left(-\frac{\phi^c(y)}{\tau}\right) \exp\left(-\frac{\|y\|^2}{2}\right) \mathrm{d}y \\ &\leq \int_{L_{i_0}(\phi)} \exp\left(-\frac{\|y\|^2 - 2\|y\|R - 2\phi_{i_1}}{2\tau}\right) \exp\left(-\frac{\|y\|^2}{2}\right) \mathrm{d}y \\ &\leq \exp\left(\frac{\phi_{i_1}}{\tau}\right) \exp\left(\frac{R^2 d^2}{2\tau(2+\tau)}\right) \int_{H_M} \exp\left(-\frac{1+\tau}{2\tau} \left\| y - \frac{R}{1+\tau}\mathrm{sign}(y)\right\|^2\right) \mathrm{d}y \\ &\leq C_1 \exp\left(\frac{\phi_{i_1}}{\tau}\right) \left[1 - \left(\frac{2\pi\tau}{1+\tau}\right)^{-d/2} \int_{\mathcal{B}(0,\alpha)} \exp\left(-\frac{1+\tau}{2\tau} \left\| y - \frac{R}{1+\tau}\mathrm{sign}(y)\right\|^2\right) \mathrm{d}y\right]\end{aligned}$$

where the last inequality comes from the fact that $H_M \subseteq \mathbb{R}^d \setminus \mathcal{B}(0,\alpha)$. Recognising the d-dimensional error function, we have

$$\frac{1}{N} \leq C_1 \exp\left(\frac{\phi_{i_1}}{\tau}\right) \left(1 - \mathrm{erf}_d\left(\alpha(M)\sqrt{\frac{1+\tau}{2\tau}}\right)\right).$$

According to lemma 3.5, when M goes to infinity,

$$\begin{aligned}\frac{1}{N} \leq & C_2 \exp\left(\frac{\phi_{i_1}}{\tau}\right) \left(\exp\left(-\frac{1+\tau}{2\tau}\alpha(M)^2\right) \left(\alpha(M)\sqrt{\frac{1+\tau}{2\tau}}\right)^{d-2}\right. \\ & \left. + o\left(\exp\left(-\frac{1+\tau}{2\tau}\alpha(M)^2\right) \left(\alpha(M)\sqrt{\frac{1+\tau}{2\tau}}\right)^{d-4}\right)\right),\end{aligned}$$

thus M can not be too large neither: $M \leq C_3$; hence ϕ_{i_0} can not be too small $\phi_{i_0} \geq \phi_{i_1} - C_3$, which concludes the proof. □

Now, we have all we need to show the following proposition.

Proposition 3.7. *There exists a constant $c_0 > 0$ such that for all $\phi \in K$,*

$$\mathbb{E}\left[\|\nabla_\phi L(y,\phi)\|^2\right] \leq c_0. \tag{3.11}$$

Proof. Assume first that y belongs to $L_i(\phi)$. Then, it can be shown that there exist constants $k_0, k_1 > 0$ such that

$$\begin{aligned}\|\nabla_\phi L(y,\phi)\|^2 &= k_0 + k_1\left(n\exp\left(\frac{2\phi_i}{\tau}\right) - 2\exp\left(\frac{\phi_i}{\tau}\right)\right)\\ &= k_0 + k_1\left(\exp\left(\frac{2\phi_i}{\tau} + \ln(n)\right) - \exp\left(\frac{\phi_i}{\tau} + \ln(2)\right)\right).\end{aligned}$$

Recall now that thanks to theorem 3.2, ϕ_i belongs to the interval $[a,b]$ on which exp is $\frac{e^b-e^a}{b-a}$-Lipschitz continuous. In other words,

$$\begin{aligned}\|\nabla_\phi L(y,\phi)\|^2 &\leq k_0 + k_1\frac{e^b-e^a}{b-a}\left(\frac{\phi_i}{\tau} + \ln\left(\frac{n}{2}\right)\right)\\ &\leq k_0 + k_1\frac{e^b-e^a}{b-a}\left(\frac{b}{\tau} + \ln\left(\frac{n}{2}\right)\right)\\ &=: c_0,\end{aligned}$$

which shows (3.11). □

Thus, theorem 3.3 is proven. Recalling (2.6), we define

$$\rho^\star = e^C\exp\left(-\frac{\phi^{\star,c}}{\tau} - \frac{V}{2}\right), \quad e^C = \left(\int_{\mathbb{R}^d}\exp\left(-\frac{\phi^{\star,c}(y)}{\tau} - \frac{V(y)}{2}\right)\mathrm{d}y\right)^{-1} \tag{3.12}$$

as being the optimal measure solving the initial problem (2.1). Finally, in order for this process to be iterated, we need to approximate $\rho^\star$ by a discrete measure. To do this, we refer to article [10]. We discretise $\rho^\star$ by minimising the functional

$$G_\varepsilon \,:\, X \in \mathbb{R}^{nd} \mapsto \frac{1}{\varepsilon}W_2^2(\rho^\star, \mu_X) + F(\rho^\star), \tag{3.13}$$

Where $\mu_X := \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$. This functional is $\frac{1}{n\varepsilon}$-semi concave and its gradient is

$$\nabla_{X_i} G_\varepsilon(X) = \frac{1}{n\varepsilon}(X_i - \beta_i(X)), \tag{3.14}$$

where

$$\beta_i(X) = \frac{\int_{L_i} y\mathrm{d}\rho^\star(y)}{\int_{L_i}\mathrm{d}\rho^\star(y)} = n\int_{L_i} y\mathrm{d}\rho^\star(y) \tag{3.15}$$

is the barycenter of the i-th Laguerre cell L_i for the $\rho^\star$ measure. We finally set

$$X^\star = (\beta_i(X))_{1\leq i\leq n} \tag{3.16}$$

as being the best discretisation of $\rho^\star$. Let us now implement this algorithm using python.

4 Numerical experiments

4.1 First experiment

We conducted a series of numerical experiments to validate the proposed algorithm, which performs multiple JKO steps on a given point cloud. The goal of these experiments is to observe the behaviour of the point cloud as it evolves over several iterations and to verify if it converges towards a Gaussian distribution, as predicted by the theory. The code can be found in appendix A. In our first experiment, we initialised the algorithm with 100 randomly distributed points and performed 10 JKO steps. The choice of 100 points provides a manageable starting point that allows us to clearly observe the transformation of the distribution across iterations. The results, shown in Figure 1, indicate a gradual transformation of the initial point cloud into a distribution that increasingly resembles a Gaussian distribution.

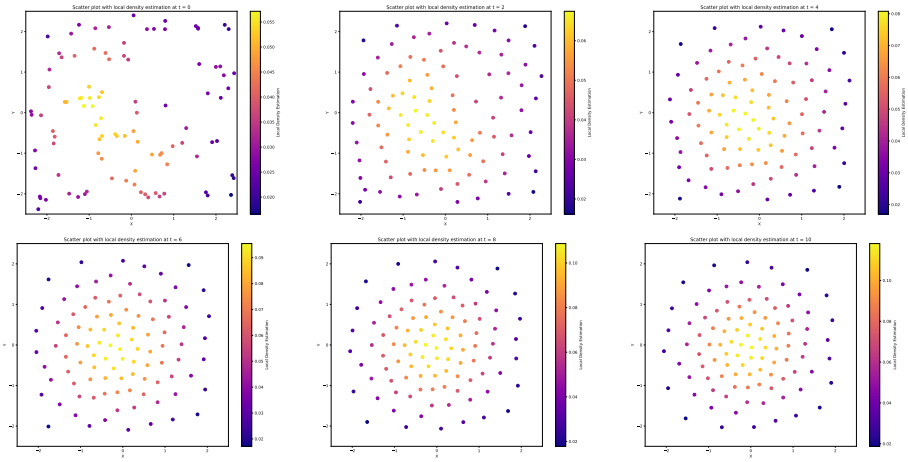

Figure 1: Resulting points cloud after JKO steps 0, 2, 4 (first row), 6, 8 and 10 (second row).

As we can see, it seems that the cloud slowly transforms into a Gaussian cloud. In order to check this hypothesis, we first plot the energy E of the cloud with respect to each iteration, see figure 2.

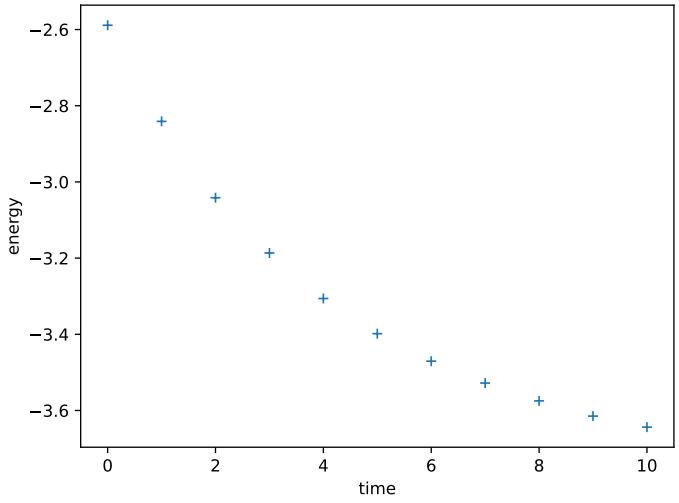

Figure 2: Plot of the energy with respect to each iteration

Each iteration makes the energy decrease, which was the goal of the whole scheme. To quantitatively assess whether the transformed point cloud resembles a Gaussian distribution, we employed the Henze-Zirkler multivariate normality test. As shown in Output 1, the test results indicate a progressive increase in the p-value with each iteration, confirming that by the 10th iteration, the point cloud can be statistically considered Gaussian with a high degree of confidence. This is a strong validation of our algorithm's effectiveness in this controlled setting.

```
At time t = 0 : HZResults(hz=1.7457692000376697,
                pval=0.0004064830415484236, normal=False)
At time t = 2 : HZResults(hz=0.9783629588048237,
                pval=0.04613593812439298, normal=False)
At time t = 4 : HZResults(hz=0.5381092286968481,
                pval=0.5140315041341157, normal=True)
At time t = 6 : HZResults(hz=0.3294587856123077,
                pval=0.9258649907507762, normal=True)
At time t = 8 : HZResults(hz=0.2200879517135601,
                pval=0.995413003774957, normal=True)
At time t = 10 : HZResults(hz=0.1729109893044617,
                 pval=0.9995150038716862, normal=True)
```

Outputs 1: Henze-Zirkler multivariate normality test

The p-value of the test is the probability that the points cloud could

have been an output to

```
np.random.multivariate_normal(np.zeros(2),np.eye(2),size = 100).
```

Thus, at time $t = 10$, we have less than 0.05% of chance of being wrong when saying that the cloud is Gaussian.

4.2 Second experiment

In the second experiment, we increased the number of points in the cloud to 500 to examine the algorithm's scalability and robustness. However, as the number of points increased, we encountered significant computational challenges, particularly related to the empty Laguerre cells, which led to division by zero errors as shown in Output 2. This experiment highlights the limitations of the current approach when applied to larger datasets.

```
RuntimeWarning: invalid value encountered in divide
                B[i,:] = np.sum(weighted_points, axis=0) / areas[i]
```

Outputs 2: Zero division error

Few experiments later, we concluded that this algorithm was only working when the number n of points is small.

4.3 Limitations

During the numerical experiments, we encountered significant challenges when scaling the algorithm to a large number of points. Specifically, some Laguerre cells remained empty throughout the process, resulting in zero-volume cells and causing computational errors, such as division by zero. This issue becomes more pronounced as the number of points increases.. It would take a lot more of computation time in order to avoid this issue. Consequently, this limitation hinders the algorithm's ability to handle detailed measures and may affect its applicability to large or high-dimensional datasets. To address this problem, we have conceived a new algorithm that could potentially avoid the issue of empty cells by using a different approach to compute Laguerre cells' area. However, this new method is still in the conceptual phase and has not yet been fully developed or tested. Additionally, we are exploring alternative parametrisation strategies that could provide more stable results in such scenarios.

5 Continuous case

In the beginning of section 2, we considered problem (2.1)

$$\inf_{\rho\in\mathcal{P}_2(\mathbb{R}^d)} E(\rho) + \frac{1}{\tau}W_2^2(\mu,\rho)$$

and we showed it is equivalent to (2.7)

$$\sup_{\phi\in\mathcal{C}(\mathbb{R}^d)} -\ln\left(\int_{\mathbb{R}^d}\exp\left(-\frac{\phi^c(y)}{\tau}-\frac{V(y)}{2}\right)\mathrm{d}y\right)+\frac{1}{\tau}\int_{\mathbb{R}^d}\phi\mathrm{d}\mu.$$

Notice that we can write $\phi^c(y)=\frac{1}{2}\|y\|^2-u^\star(y)$, where $u(x)=\frac{1}{2}\|x\|^2-\phi(x)$ is the Brenier potential associated to ϕ. Assuming ϕ is optimal, we know that it is c-convex. Thus, according to proposition 1.21 from [12], we know that u is convex. Since the Brenier potential is convex, we will parameterise it as

$$u_\theta(x)=\sum_{i=1}^n \theta_i\psi_i(x), \tag{5.1}$$

where the functions ψ_i are convex and the parameters θ_i are positive. It is a 1-layer convex neural network parametrisation [2], which works well and which will not be a problem since we are doing multiple JKO steps.

Plugging (5.1) into (2.7), we want to solve the following optimisation problem

$$\sup_{\theta\in\mathbb{R}^n} -\ln\left(\int_{\mathbb{R}^d}\exp\left(\frac{u_\theta^\star(y)}{2}\right)\exp\left(-\frac{1+\tau}{2\tau}\|y\|^2\right)\mathrm{d}y\right)-\frac{1}{\tau}\int_{\mathbb{R}^d}u_\theta(x)\mathrm{d}\mu(x). \tag{5.2}$$

Before tackling such a problem, we will first try to solve a simpler one.

5.1 A simplified version

The goal of this subsection is to solve the following transport problem

$$\inf_{\theta\in\mathbb{R}^n}\int_{\mathbb{R}^d}u_\theta(x)\mathrm{d}\mu(x)+\int_{\mathbb{R}^d}u_\theta^\star(y)\mathrm{d}\nu(y),$$

for $n=d$,

$$\mu\sim\mathcal{N}(0,I_d),\quad \nu\sim\mathcal{N}(0,\Sigma),$$

where Σ is diagonal with positive coefficients σ_i and

$$\forall i,\quad \psi_i(x):=\langle x,e_i\rangle^2=g(x_i).$$

Let us compute the Fenchel-Legendre transform of ψ_i:

$$\begin{aligned}\psi_i^\star(y) &= \sup_{x\in\mathbb{R}^d} \langle x, y\rangle - \psi_i(x) \\ &= \sup_{x\in\mathbb{R}^d} \sum_{j\neq i} x_j y_j + x_i y_i - x_i^2 \\ &= \begin{cases} \sup_{x_i\in\mathbb{R}} x_i y_i - x_i^2 \text{ if } y \in \mathbb{R}e_i \\ +\infty \text{ otherwise.}\end{cases}\end{aligned}$$

The function $x_i \mapsto x_i y_i - x_i^2$ is easy to maximise. The maximum is reached at $x_i = \frac{y_i}{2}$ and is equal to $\frac{y_i}{2} y_i - \left(\frac{y_i}{2}\right)^2 = \frac{y_i^2}{4}$. Thus, we have for all i

$$\psi_i^\star(y) = \begin{cases} \frac{y_i^2}{4} \text{ if } y \in \mathbb{R}e_i \\ +\infty \text{ otherwise,}\end{cases}$$

Now, let us compute the Fenchel-Legendre transform of $_\theta$. It is a sum of positive functions, so

$$\begin{aligned}u_\theta^\star(y) &= \inf_{y_1+\cdots+y_d=y} \sum_{i=1}^d \theta_i g^\star\left(\frac{y_i}{\theta_i}\right) \\ &= \inf_{y_1+\cdots+y_d=y} \sum_{i=1}^d \frac{\theta_i}{4}\frac{y_i^2}{\theta_i^2} \\ &= \sum_{i=1}^d \frac{y_i^2}{4\theta_i}.\end{aligned}$$

The initial problem can then be rewritten as

$$\inf_{\theta\in\mathbb{R}^d} G(\theta) := \sum_{i=1}^d \theta_i \int_{\mathbb{R}^d} \psi_i(x)\mathrm{d}\mu(x) + \sum_{i=1}^d \frac{1}{\theta_i}\int_{\mathbb{R}^d} \psi_i^\star(y)\mathrm{d}\nu(y).$$

Computing the gradient of G and solving for $\nabla_{\theta_i} G(\theta) = 0$ yields

$$\forall i, \quad \theta_i^\star = \sqrt{\frac{\int_{\mathbb{R}^d} \psi_i^\star(y)\mathrm{d}\nu(y)}{\int_{\mathbb{R}^d} \psi_i(x)\mathrm{d}\mu(x)}}.$$

Since $\mu \sim \mathcal{N}(0, I_d)$, each component x_i is an independent standard normal random variable with mean 0 and variance 1. Similarly, since $\nu \sim \mathcal{N}(0, \Sigma)$, each component y_i is an independent Gaussian random variable with mean 0 and variance σ_i. By definition of the variance, we have

$$\mathbb{E}_{x\sim\mu}[x_i^2] = \mathrm{Var}(x_i) + (\mathbb{E}_{x\sim\mu}[x_i])^2 = 1$$

and

$$\mathbb{E}_{y\sim\nu}[y_i^2] = \mathrm{Var}(y_i) + (\mathbb{E}_{y\sim\nu}[y_i])^2 = \sigma_i.$$

Finally, since

$$\forall i, \quad \theta_i^\star = \sqrt{\frac{\mathbb{E}_{y\sim\nu}[y_i^2]}{4\mathbb{E}_{x\sim\mu}[x_i^2]}},$$

we end up with

$$\forall i, \quad \theta_i^\star = \frac{\sqrt{\sigma_i}}{2}.$$

According to Brenier's theorem, the optimal transport map T that pushes μ onto ν is

$$T(x) = \nabla u_{\theta^\star}(x) = (\sqrt{\sigma_1}x_1, \dots, \sqrt{\sigma_d}x_d),$$

which is exactly what we obtain using classical optimal transport theory. It is very promising to see parametrisation (5.1) working on a toy example.

6 Conclusion

The primary aim of this work was to explore the application of the Jordan-Kinderlehrer-Otto (JKO) scheme within the framework of Continuous Normalising Flows (CNFs) by leveraging Kantorovich's potentials—a novel approach compared to the traditional Benamou-Brenier dynamic formulation of optimal transport. Although this study represents an initial exploration, it provides important insights into the feasibility and potential of this method.

Throughout the course of this research, we demonstrated that the use of Kantorovich's dual potentials allows the JKO scheme to be reformulated into a series of convex sub-problems, which can be addressed using stochastic gradient descent. While the numerical experiments carried out on relatively small datasets confirmed the theoretical predictions, the scalability issues encountered highlight the challenges that need to be addressed in future work.

Given the limited scope of this initial study, the findings should be viewed as a foundational step rather than a conclusive result. The challenges, particularly those related to scalability and computational stability, indicate areas that require substantial further investigation. However, this work does contribute a fresh perspective by exploring an alternative approach to the JKO scheme, which may offer new directions for future research.

Looking forward, there is ample opportunity to build upon this early work. Future efforts will focus on overcoming the identified limitations, exploring alternative parametrisation strategies, and testing the approach on more complex and high-dimensional datasets. With these improvements, this line of research holds the potential to significantly advance the field of optimal transport and generative modelling.

In conclusion, while this work is an early step in a longer research journey, it provides a useful starting point for further exploration. The insights gained here will serve as a foundation for more extensive studies over the coming years, contributing to the broader understanding and application of advanced mathematical frameworks in machine learning and data science.

Bibliography

[1] Jean-David Benamou and Yann Brenier. "A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem". In: *Numerische Mathematik* 84.3 (2000), pp. 375–393.

[2] Yoshua Bengio et al. "Convex neural networks". In: *Advances in neural information processing systems* 18 (2005).

[3] Olivier Fercoq. *Stochastic Optimization*. Jan. 10, 2023. URL: https://leclere.github.io/files/teaching/Saclay/fercoq/poly_optsto_fercoq.pdf.

[4] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014).

[5] Ishaan Gulrajani et al. *Improved Training of Wasserstein GANs*. 2017. arXiv: 1704.00028 [cs.LG]. URL: https://arxiv.org/abs/1704.00028.

[6] Richard Jordan, David Kinderlehrer, and Felix Otto. "The variational formulation of the Fokker–Planck equation". In: *SIAM Journal on Mathematical Analysis* 29.1 (1998), pp. 1–17.

[7] Diederik P. Kingma and Max Welling. "An Introduction to Variational Autoencoders". In: *Foundations and Trends® in Machine Learning* 12.4 (2019), pp. 307–392. ISSN: 1935-8245. DOI: 10.1561/2200000056. URL: http://dx.doi.org/10.1561/2200000056.

[8] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. 2014. arXiv: http://arxiv.org/abs/1312.6114v10 [stat.ML].

[9] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. "Normalizing Flows: An Introduction and Review of Current Methods". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11 (2021), pp. 3964–3979. DOI: 10.1109/TPAMI.2020.2992934.

[10] Hugo Leclerc et al. "Lagrangian Discretization of Crowd Motion and Linear Diffusion". In: (Apr. 30, 2020). URL: http://arxiv.org/abs/1905.08507.

[11] Quentin Mérigot and Boris Thibert. *Optimal Transport: Discretization and Algorithms*. Mar. 2, 2020. URL: http://arxiv.org/abs/2003.00855.

[12] Filippo Santambrogio. *Optimal Transport for Applied Mathematicians*. Springer, 2015. DOI: 10.1007/978-3-319-20828-2.

[13] Esteban G Tabak and Cristina V Turner. "A family of nonparametric density estimation algorithms". In: *Communications on Pure and Applied Mathematics* 66.2 (2013), pp. 145–164.

[14] Alexander Vidal et al. “Taming hyperparameter tuning in continuous normalizing flows using the JKO scheme”. In: *Scientific Reports* 13.1 (2023), p. 4501.

[15] C. Villani and American Mathematical Society. *Topics in Optimal Transportation*. Graduate studies in mathematics. American Mathematical Society, 2003. ISBN: 9781470418045. URL: https://books.google.fr/books?id=MyPjjgEACAAJ.

[16] Chen Xu, Xiuyuan Cheng, and Yao Xie. “Normalizing flow neural networks by JKO scheme”. In: *Advances in Neural Information Processing Systems* 36 (2024).

A Python code

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import multivariate_normal, gaussian_kde
from scipy.sparse import coo_array
from time import time
import datetime
import pingouin as pg
```

Code 3: Necessary imports

```
def optimization(Y, bounds, psi, tau, niter, nbatch):

    a, b = bounds[0], bounds[1]

    for k in range(1, niter+1):
        ## Core of the optimisation process
        x = np.random.multivariate_normal(np.zeros(d),
            np.eye(d), size = nbatch)
        c_transform = np.linalg.norm(Y[:,np.newaxis,:]-x,axis=2)**2-psi
        indexes = np.argmin(c_transform, axis=0)
        e = np.array([np.exp(-c_transform[indexes[j],j]
            /(2*tau)) for j in range(nbatch)])
        aux = coo_array((e, (indexes, np.arange(nbatch))),
              shape=(N, nbatch))
        aux = aux.sum(1).reshape(N, 1)
        lr = 1. / np.sqrt(niter)
        psi -= (aux - np.full((N, 1), nbatch/N)) *
               (lr * (2*np.pi)**(d/2)) / (nbatch*2*tau)

        ## Validity check
        if k % (niter / 10) == 0:
            areas, cells = cells_areas(Y, bounds, psi, nbatch, nsamples)
            mean = np.mean(areas)
            sigma = np.sqrt(np.var(areas))
            print(fr"Cells area at step {k}: {format(mean,'.3e')}
                  +/- {format(sigma,'.3e')}")

    return psi, areas, cells
```

Code 4: Optimisation step

```python
def cells_areas(Y, bounds, psi, nbatch, nsamples):

    N, d = Y.shape
    cells = [[] for _ in range(N)]
    areas = np.zeros((N,1))
    a, b = bounds[0], bounds[1]

    for _ in range(nsamples):
        x = np.random.uniform(low = a, high = b, size = (nbatch,d))
        c_transform = np.linalg.norm(Y[:,np.newaxis,:]-x,axis=2)**2-psi
        indexes = np.argmin(c_transform, axis=0)
        e = np.exp(np.array([-c_transform[indexes[j],j] for j in
           range(nbatch)]) /(2*tau)-(np.linalg.norm(x, axis=1)**2) / 2)
        aux = coo_array((e, (indexes, np.arange(nbatch))),
              shape=(N, nbatch))
        aux = aux.sum(1).reshape(N, 1)
        for i in range(nbatch):
            j = indexes[i]
            cells[j].append(x[i])
        areas += aux

    return (areas / (nsamples * nbatch), cells)
```

Code 5: Cell area computation

```python
def barycentres(psi, areas, cells, nbatch, nsamples):
    B = np.ones((N,d))

    for i in range(N):
        l = len(cells[i])
        points = np.array(cells[i]).reshape(l,d)
        c_transform = (np.linalg.norm(Y[i,:]-points, axis=1)**2
                       -psi[i]).reshape((l,1))
        e = np.exp(-c_transform/(2*tau)
                   -(np.linalg.norm(points, axis=1)**2).reshape(l,1)/2)
        weighted_points = np.multiply(points,e)
        B[i,:] = np.sum(weighted_points, axis=0) / areas[i]

    return np.array(B/(nsamples * nbatch))
```

Code 6: Barycenter computation

```
def transport(Y, bounds, psi, tau, t, niter, nbatch, nsamples):
    new_psi, areas, cells = optimization(Y, bounds, psi, tau,
                                niter, nbatch)
    B = barycentres(new_psi, areas, cells, nbatch, nsamples)
    new_Y = B

    return new_Y
```

Code 7: Transport function

```
def plot_positions(Y, t, test):
    kde = gaussian_kde(Y.T) # Kernel Density Estimation
    densities = kde(Y.T) # Local density estimation of the points cloud Y

    plt.figure(figsize=(10,8))

    sc = plt.scatter(Y[:, 0], Y[:, 1], c=densities, cmap='plasma', s=5)
    plt.colorbar(sc, label='Local Density Estimation')
    plt.title(f'Scatter plot with local density estimation at t = {t}')
    plt.xlim(-2.5, 2.5)
    plt.ylim(-2.5, 2.5)
    plt.xlabel('X')
    plt.ylabel('Y')

    plt.tight_layout()
    plt.savefig(f'plots/JKO-U-vers-N/essai_{test}/JKO_{t}.pdf')
    plt.show()
```

Code 8: Plot function

```
def potentiel(Y):
    return (np.linalg.norm(Y, axis=1)**2)/2

def energie(Y):
    return np.mean(potentiel(Y)-np.log(N))
```

Code 9: Metrics

```
np.random.seed(2214) # For reproductibility
test = 1
N = 100
d = 2
tau = 0.1
niter = 1000
nbatch = 1000
nsamples = 1000

loss = []
bounds = [-2.5, 2.5]
Y = np.random.uniform(low = bounds[0], high = bounds[1], size = (N, d))

energy = energie(Y)
loss.append(energy)
plot_positions(Y, 0, test)
print(f"Energy at time t = 0 : {energy}")
print(f"At time t = 0 : {pg.multivariate_normality(Y)}")

T = 10
for t in range(1,T+1):
    psi = np.zeros(N).reshape(N,1)
    Y = transport(Y, bounds, psi, tau, t, niter, nbatch, nsamples)
    if t%1 == 0:
        plot_positions(Y, t, test)
        energy = energie(Y)
        loss.append(energy)
        print(f"Energy at time t = {t} : {energy}")
        print(f"At time step t = {t} : {pg.multivariate_normality(Y)}")
```

Code 10: JKO steps algorithm