Importance Corrected Neural JKO Sampling

Johannes Hertrich [1 2] **Robert Gruhlke** [3]

Abstract

In order to sample from an unnormalized probability density function, we propose to combine continuous normalizing flows (CNFs) with rejection-resampling steps based on importance weights. We relate the iterative training of CNFs with regularized velocity fields to a JKO scheme and prove convergence of the involved velocity fields to the velocity field of the Wasserstein gradient flow (WGF). The alternation of local flow steps and non-local rejection-resampling steps allows to overcome local minima or slow convergence of the WGF for multimodal distributions. Since the proposal of the rejection step is generated by the model itself, they do not suffer from common drawbacks of classical rejection schemes. The arising model can be trained iteratively, reduces the reverse Kullback-Leibler (KL) loss function in each step, allows to generate *iid* samples and moreover allows for evaluations of the generated underlying density. Numerical examples show that our method yields accurate results on various test distributions including high-dimensional multimodal targets and outperforms the state of the art in almost all cases significantly.

1. Introduction

We consider the problem of sampling from an unnormalized probability density function. That is, we are given an integrable function $g\colon \mathbb{R}^d \to \mathbb{R}_{>0}$ and we aim to generate samples from the probability distribution ν given by the density $q(x) = g(x)/Z_g$, where the normalizing constant $Z_g = \int_{\mathbb{R}^d} g(x)\mathrm{d}x$ is unknown. Many classical sampling methods are based on Markov chain Monte Carlo (MCMC) methods like the overdamped Langevin sampling, see, e.g., (Welling & Teh, 2011). The generated probability path of the underlying stochastic differential equation follows the Wasserstein-2 gradient flow of the reverse KL divergence $\mathcal{F}(\mu) = \mathrm{KL}(\mu, \nu)$. Over the last years, generative models like normalizing flows (Rezende & Mohamed, 2015) or diffusion models (Ho et al., 2020; Song et al., 2021) became more popular for sampling, see, e.g., (Phillips et al., 2024; Vargas et al., 2023a). Also these methods are based on the reverse KL divergence as a loss function. While generative models have successfully been applied in data-driven setups, their application to the problem of sampling from arbitrary unnormalized densities is not straightforward. This difficulty arises from the significantly harder nature of the problem, even in moderate dimensions, particularly when dealing with target distributions that exhibit phenomena such as concentration effects, multimodalities, heavy tails, or other issues related to the curse of dimensionality.

In particular, the reverse KL is non-convex in the Wasserstein space as soon as the target density ν is not log-concave which is for example the case when ν consists of multiple modes. In this case generative models often collapse to one or a small number of modes. We observe that for continuous normalizing flows (CNFs, Chen et al., 2018; Grathwohl et al., 2019) this can be prevented by regularizing the L^2-norm of the velocity field as proposed under the name OT-flow by (Onken et al., 2021). In particular, this regularization converts the objective functional into a convex one. However, the minimizer of the regularized loss function is no longer given by the target measure ν but by the Wasserstein proximal mapping of the objective function applied onto the latent distribution. Considering that the Jordan-Kinderlehrer-Otto (JKO) scheme (Jordan et al., 1998) iteratively applies the Wasserstein proximal mapping and converges to the Wasserstein gradient flow, several papers proposed to approximate the steps of the scheme by generative models, see (Altekrüger et al., 2023; Alvarez-Melis et al., 2022; Fan et al., 2022; Lambert et al., 2022; Mokrov et al., 2021; Vidal et al., 2023; Xu et al., 2024). We will refer to this class of method by the name *neural JKO*. Even though this approximates the same gradient flow of the Langevin dynamics these approaches have usually the advantage of faster inference (once they are trained) and additional possibly allow for density evaluations.

However, already the time-continuous Wasserstein gradient flow suffers from the non-convexity of the reverse KL

[*]Equal contribution [1]Université Paris Dauphine - PSL [2]University College London [3]FU Berlin. Correspondence to: Johannes Hertrich <johannes.hertrich@dauphine.psl.eu>, Robert Gruhlke <r.gruhlke@fu-berlin.de>.

Proceedings of the 42nd International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

loss function by getting stuck in local minima or by very slow convergence times. In particular, it is well-known that Langevin-based sampling methods often do not distribute the mass correctly onto multimodal target distributions. As a remedy, (Neal, 2001) proposed importance sampling, i.e., to reweight the sample based on quotients of the target distribution and its current approximation which leads to an unbiased estimator of the Monte Carlo integral. However, this estimator may lead to highly imbalanced weights between these samples and the resulting estimator might have a large variance. Moreover, the density of the current approximation has to be known up to a possible multiplicative constant, which is not the case for many MCMC methods like Langevin sampling or Hamiltonian Monte Carlo. In (Del Moral et al., 2006) the authors propose a scheme for alternating importance sampling steps with local Monte Carlo steps. However, the corresponding importance sampling step generates non-iid samples, that de-correlate over time by construction.

Contributions In this paper, we propose a sampling method which combines neural JKO steps based on CNFs with importance based rejection steps. While the CNFs adjust the position of the generated samples *locally*, the rejection steps readjust the inferred distribution *non-locally* based on the quotient of generated and target distribution. Then, in each rejection step we resample the rejected points based on the current constructed generative model. We illustrate this procedure in Figure 1.

Our method generates independent samples and allows to evaluate the density of the generated distribution. In our numerical examples, we apply our method to common test distributions up to the dimension $d = 1600$ which are partially highly multimodal. We show that our *importance corrected neural JKO* sampling (neural JKO IC) achieves significantly better results than the comparisons[1].

From a theoretical side, we prove that the velocity fields from a sequence of neural JKO steps strongly converge to the velocity field of the corresponding Wasserstein gradient flow and that the reverse KL loss function decreases throughout the importance-based rejection steps.

Outline In Section 2, we recall the fundamental concepts which will be required. Afterwards, we consider neural JKO schemes more detail in Section 3. We introduce our importance-based rejection steps in Section 4. Finally, we evaluate our model numerically and compare it to existing methods in Section 5. Conclusions are drawn in Section 6. Additionally, proofs, further numerical and technical details are presented in Appendix A- E.

Related Work Common methods for sampling of unnormalized densities are often based on Markov Chain Monte Carlo (MCMC) methods, see e.g. (Gilks et al., 1995). In particular first order based variants, such as the Hamiltonian Monte Carlo (HMC) (Betancourt, 2017; Hoffman & Gelman, 2014) and the Metropolis Adjusted Langevin Algorithm (MALA Girolami & Calderhead, 2011; Rossky et al., 1978; Roberts & Tweedie, 1996) are heavily used in practice, see (Andrieu et al., 2003) for an overview. The viewpoint of these samplers as sample space description of gradient flows defined in a metricized probability space then allows for extensions such as interacting particle systems (Chen et al., 2023; Eigel et al., 2024; Garbuno-Inigo et al., 2020; Wang & Li, 2022). However, since these algorithms are based on local transformations of the samples, they are unable to distribute the mass correctly among different modes, which can partially be corrected by importance sampling (Neal, 2001) and sequential Monte Carlo samplers (SMC) (Del Moral et al., 2006) as described above. In contrast to our model, SMC approximates the density of the approximation by assigning "inverse Markov kernels" to certain MCMC kernels, which might lead to propagating errors. Furthermore, the generation of additional samples requires to rerun the whole procedure which can be very costly. Other approaches approximate the target density by combination of transport maps and low-rank models such as tensor trains (Eigel et al., 2022) yielding efficient access to posterior statistics.

In the last years, generative models became very popular, including VAEs (Kingma & Welling, 2014), normalizing flows (Rezende & Mohamed, 2015), diffusion models (Ho et al., 2020) or flow-matching (Lipman et al., 2022) which is also known as rectified flow (Liu et al., 2022). In contrast to our setting, they initially consider the *modeling* task, i.e., they assume that they are given samples from the target measure instead of an unnormalized density. However, there are several papers, which adapt these algorithms for the *sampling* task. For normalizing flows, this mostly amounts to changing the loss function (Hoffman et al., 2019; Marzouk et al., 2016; Qiu & Wang, 2024). Very recently, there appeared also a flow-matching variant for the sampling task (Woo & Ahn, 2024) based on (Akhound-Sadegh et al., 2024). For diffusion (and stochastic control) models this was done based on variational approaches (Blessing et al., 2024; Phillips et al., 2024; Vargas et al., 2023a;b; Zhang & Chen, 2021) or by computing the score by solving a PDE (Richter & Berner, 2024; Sommer et al., 2024). These methods usually provide much faster sampling times than MCMC methods and are often used in combination with some conditioning parameter for inverse problems, where a (generative) prior is combined with a known likelihood term (Ardizzone et al., 2019; Altekrüger & Hertrich, 2023; Andrle et al., 2021; Denker et al., 2024). Combinations of generative models with stochastic sampling steps were considered in the literature for generative modeling under the

[1]The code is available at https://github.com/johertrich/neural_JKO_ic

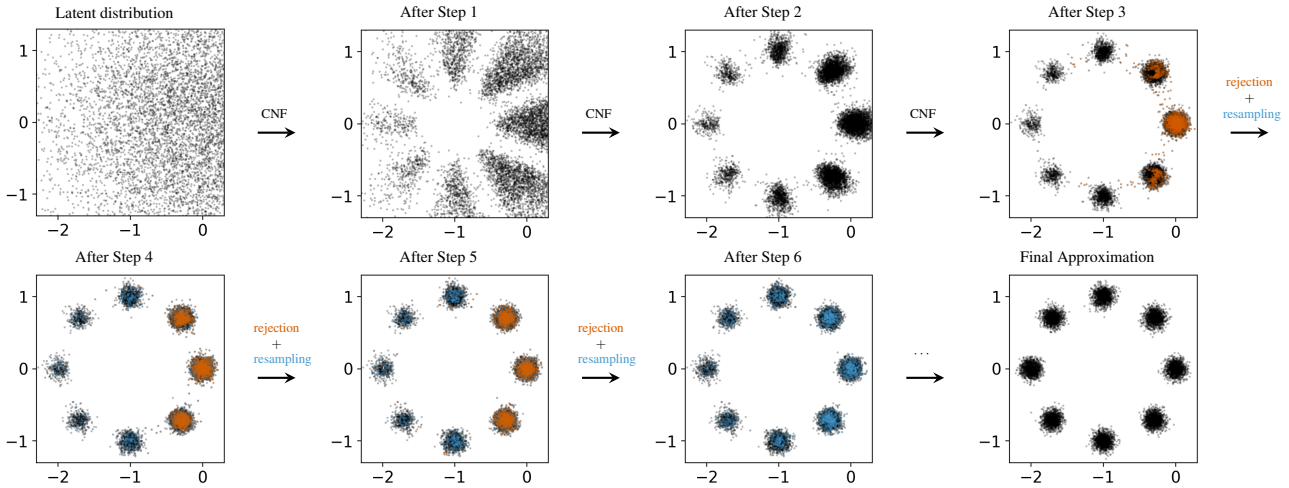

Figure 1: Iterative application of neural JKO steps and rejection steps for a shifted mixture target distribution. The orange samples are rejected in the next following rejection step and the blue samples are the resampled points. The latter approach enables for the correction of wrong mode weights introduced by the underlying WGF. See Figure 3 for more steps.

name stochastic normalizing flows (Hagemann et al., 2023; 2022; Noé et al., 2019; Wu et al., 2020) and for sampling under the name annealed flow transport Monte Carlo (Arbel et al., 2021; Matthews et al., 2022). (Gabrié et al., 2022) use normalizing flows to learn proposal distributions in an Metropolis-Hastings algorithm. These generative models can be adapted to follow a Wasserstein gradient flow, by mimicking a JKO scheme with generative models or directly following the velocity field of a kernel-based functional. Such approaches were proposed for generative modeling (Fan et al., 2022; Hagemann et al., 2024; Hertrich et al., 2024; Liutkus et al., 2019; Vidal et al., 2023; Xu et al., 2024), sampling (Fan et al., 2022; Lambert et al., 2022; Liu & Wang, 2016a; Mokrov et al., 2021) or other tasks (Altekrüger et al., 2023; Arbel et al., 2019; Alvarez-Melis et al., 2022). Gradient flows of the reverse KL divergence with respect to different metric were considered in (Liu & Wang, 2016b) under the name Stein variational gradient descent. Moreover, Lu et al. (2019) study gradient flows in the Wasserstein-Fisher-Rao metric which can be implemented via birth-death processes.

After the first version of our paper appeared, several related preprints were released, including (Albergo & Vanden-Eijnden, 2024; Chen et al., 2025; He et al., 2025; Wu & Xie, 2024).

2. Preliminaries

In this section, we provide a rough overview of the required concepts for this paper. To this end, we first revisit the basic definitions of Wasserstein gradient flows, e.g., based on (Ambrosio et al., 2005). Afterwards we recall continuous normalizing flows with OT-regularizations.

2.1. Curves in Wasserstein Spaces

Wasserstein Distance Let $\mathcal{P}(\mathbb{R}^d)$ be the space of probability measures on $\mathbb{R}^d$ and denote by $\mathcal{P}_2(\mathbb{R}^d) := \{\mu \in \mathcal{P}(\mathbb{R}^d) : \int_{\mathbb{R}^d} \|x\|^2 \mathrm{d}\mu(x) < \infty\}$ the subspace of probability measures with finite second moment. Let $\mathcal{P}_2^{\mathrm{ac}}(\mathbb{R}^d)$ be the subspace of absolutely continuous measures from $\mathcal{P}_2(\mathbb{R}^d)$. Moreover, we denote for $\mu, \nu \in \mathcal{P}_2(\mathbb{R}^d)$ by $\Gamma(\mu,\nu) := \{\boldsymbol{\pi} \in \mathcal{P}_2(\mathbb{R}^d \times \mathbb{R}^d) : P_{1\#}\boldsymbol{\pi} = \mu, P_{2\#}\boldsymbol{\pi} = \nu\}$ the set of all transport plans with marginals μ and ν, where $P_i \colon \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ defined by $P_i(x_1, x_2) = x_i$ is the projection onto the i-th component for $i = 1, 2$. Then, we equip $\mathcal{P}_2(\mathbb{R}^d)$ with the Wasserstein-2 metric defined by

$$W_2^2(\mu,\nu) = \inf_{\boldsymbol{\pi} \in \Gamma(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x-y\|^2 \mathrm{d}\boldsymbol{\pi}(x,y).$$

If $\mu \in \mathcal{P}_2^{\mathrm{ac}}(\mathbb{R}^d)$, the above problem has always a unique minimizer.

Absolutely Continuous Curves A curve $\gamma \colon I \to \mathcal{P}_2(\mathbb{R}^d)$ on the interval $I \subseteq \mathbb{R}$ is called *absolutely continuous* if there exists a Borel velocity field $v \colon \mathbb{R}^d \times I \to \mathbb{R}^d$ with $\int_I \|v(\cdot,t)\|_{L_2(\gamma(t),\mathbb{R}^d)} \mathrm{d}t < \infty$ such that the continuity equation

$$\partial_t \gamma(t) + \nabla \cdot (v(\cdot,t)\gamma(t)) = 0 \tag{1}$$

is fulfilled on $I \times \mathbb{R}^d$ in a weak sense. Then, any velocity field v solving the continuity equation (1) for fixed γ characterizes γ as $\gamma(t) = z(\cdot,t)_\# \gamma(t_0)$, where z is the solution of the ODE $\dot{z}(x,t) = v(z(x,t),t)$ with $z(x,t_0) = x$ and $t_0 \in I$. It can be shown that for an absolutely continuous curve there exists a unique solution of minimal norm which is equivalently characterized by the so-called regular tangent space $\mathrm{T}_{\gamma(t)}\mathcal{P}_2(\mathbb{R}^d)$, see Appendix A for details. An abso-

lutely continuous curve is a geodesic if there exists some $c > 0$ such that $W_2(\gamma(s), \gamma(t)) = c|s-t|$.

The following theorem formulates a dynamic version of the Wasserstein distance based minimal energy curves in the Wasserstein space.

Theorem 2.1 (Benamou & Brenier, 2000)**.** *Assume that $\mu, \nu \in \mathcal{P}_2^{\mathrm{ac}}(\mathbb{R}^d)$. Then, $W_2^2(\mu,\nu)$ is equal to*

$$\inf_{\substack{v\colon \mathbb{R}^d\times[0,1]\to\mathbb{R}^d,\\ \dot z(x,t)=v(z(x,t),t),\\ z(x,0)=x,\, z(\cdot,1)_\#\mu=\nu}} \int_0^1 \int_{\mathbb{R}^d} \|v(z(x,t),t)\|^2 \mathrm{d}\mu(x)\mathrm{d}t.$$

Moreover, there exists a unique minimizing velocity field v and the curve defined by $\gamma(t) = z(\cdot,t)_\#\mu$ with $t \in [0,1]$ and $\dot z(x,t) = v(z(x,t),t)$, $z(x,0) = x$ is a geodesic which fulfills the continuity equation $\partial_t\gamma(t) + \nabla\cdot(v(\cdot,t)\gamma(t)) = 0$.

Let $\tau > 0$. Then, by substitution of t by t/τ and rescaling v in the time variable, this is equal to

$$\inf_{\substack{v\colon \mathbb{R}^d\times[0,\tau]\to\mathbb{R}^d,\\ \dot z(x,t)=v(z(x,t),t),\\ z(x,0)=x,\, z(\cdot,\tau)_\#\mu=\nu}} \tau \int_0^\tau \int_{\mathbb{R}^d} \|v(z(x,t),t)\|^2 \mathrm{d}\mu(x)\mathrm{d}t. \tag{2}$$

Wasserstein Gradient Flows An absolutely continuous curve $\gamma\colon (0,\infty) \to \mathcal{P}_2(\mathbb{R}^d)$ with velocity field $v_t \in \mathrm{T}_{\gamma(t)}\mathcal{P}_2(\mathbb{R}^d)$ is a *Wasserstein gradient flow with respect to* $\mathcal{F}\colon \mathcal{P}_2(\mathbb{R}^d) \to (-\infty,\infty]$ if $v_t \in -\partial\mathcal{F}(\gamma(t))$, for a.e. $t > 0$, where $\partial\mathcal{F}(\mu)$ denotes the reduced Fréchet subdiffential at μ, see Appendix A for a definition.

To compute Wasserstein gradient flows numerically, we can use the generalized minimizing movements or Jordan-Kinderlehrer-Otto (JKO) scheme (Jordan et al., 1998). To this end, we consider the Wasserstein proximal mapping defined as

$$\mathrm{prox}_{\tau\mathcal{F}}(\hat\mu) = \operatorname*{arg\,min}_{\mu\in\mathcal{P}_2(\mathbb{R}^d)} \left\{ \tfrac{1}{2}W_2^2(\mu,\hat\mu) + \tau\mathcal{F}(\mu) \right\}.$$

Then, define as μ_τ^k for $k \in \mathbb{N}$ the steps of the minimizing movements scheme, i.e.,

$$\mu_\tau^0 = \mu^0, \qquad \mu_\tau^{k+1} = \mathrm{prox}_{\tau\mathcal{F}}(\mu_\tau^k). \tag{3}$$

We denote the piecewise constant interpolations $\tilde\gamma_\tau\colon [0,\infty) \to \mathcal{P}_2(\mathbb{R}^d)$ of the minimizing movement scheme by

$$\tilde\gamma_\tau(k\tau + t\tau) = \mu_\tau^k, \quad t \in [0,1). \tag{4}$$

Then, the following convergence result holds true. We recall the necessary definitions of coercivity and λ-convexity in Wasserstein spaces in Appendix A.

Theorem 2.2 (Ambrosio et al., 2005, Thm 11.2.1)**.** *Let $\mathcal{F}\colon \mathcal{P}_2(\mathbb{R}^d) \to (-\infty,+\infty]$ be proper, lsc, coercive, and λ-convex along generalized geodesics, and let $\mu^0 \in \overline{\mathrm{dom}\,\mathcal{F}}$. Then the curves $\tilde\gamma_\tau$ defined via the minimizing movement scheme (4) converge for $\tau \to 0$ locally uniformly to a locally Lipschitz curve $\gamma\colon (0,+\infty) \to \mathcal{P}_2(\mathbb{R}^d)$ which is the unique Wasserstein gradient flow of $\mathcal{F}$ with $\gamma(0+) = \mu^0$.*

2.2. Continuous Normalizing Flows and OT-Flows

The concept of normalizing flows first appeared in (Rezende & Mohamed, 2015). It follows the basic idea to approximate a probability distribution ν by considering a simple latent distribution μ_0 (usually a standard Gaussian) and to construct a diffeomorphism $\mathcal{T}_\theta\colon \mathbb{R}^d \to \mathbb{R}^d$ depending on some parameters θ such that $\nu \approx \mathcal{T}_{\theta\#}\mu_0$. In practice, the diffeomorphism can be approximated by coupling-based neural networks (Dinh et al., 2016; Kingma & Dhariwal, 2018), residual architectures (Behrmann et al., 2019; Chen et al., 2019; Hertrich, 2023) or autoregressive flows (De Cao et al., 2020; Durkan et al., 2019; Huang et al., 2018; Papamakarios et al., 2017). In this paper, we mainly focus on *continuous normalizing flows* proposed by (Chen et al., 2018; Grathwohl et al., 2019), see also (Ruthotto & Haber, 2021) for an overview. Here the diffeomorphism $\mathcal{T}_\theta$ is parameterized as neural ODE. To this end let $v_\theta\colon \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$ be a neural network with parameters θ and let $z_\theta\colon \mathbb{R}^d \times [0,\tau] \to \mathbb{R}^d$ for fixed $\tau > 0$ be the solution of $\dot z_\theta = v_\theta$, with initial condition $z_\theta(x,0) = x$. Then, we define $\mathcal{T}_\theta$ via the solution z_θ as $\mathcal{T}_\theta(x) = z_\theta(x,\tau)$. The density p_θ of $\mathcal{T}_{\theta\#}\mu_0$ can be described by the change-of-variables formula $p_\theta(x) = p_0(x)/|\det(\nabla\mathcal{T}_\theta(x))|$, where p_0 is the density of the latent distribution μ_0. In the case of continuous normalizing flows, it can be shown that the denominator can be computed as $\log(|\det(\nabla\mathcal{T}_\theta(x))|) = \ell_\theta(x,\tau)$ with $\partial_t\ell_\theta(x,t) = \mathrm{trace}(\nabla v_\theta(z_\theta(x,t),t))$ and $\ell_\theta(\cdot,0) = 0$. In order to train a normalizing flow, one usually uses the Kullback-Leibler divergence. If ν is given by a density $q(x) = g(x)/Z_g$, where Z_g is an unknown normalizing constant, then this amounts to the reverse KL loss function

$$\begin{aligned}\mathcal{L}(\theta) &= \mathrm{KL}(\mathcal{T}_{\theta\#}\mu_0, \nu)\\ &= \mathbb{E}_{x\sim\mu_0}[-\log(q(\mathcal{T}_\theta(x))) + \log(p_\theta(x))]\\ &= \mathbb{E}_{x\sim\mu_0}[-\log(g(\mathcal{T}_\theta(x))) - \ell_\theta(x,\tau)] + C,\end{aligned}$$

for some constant C independent of θ.

In order to stabilize and accelerate the training, (Onken et al., 2021) propose to regularize the velocity field v_θ by its expected squared norm. More precisely, they propose to add the regularizer $\mathcal{R}(\theta) = \tau\int_0^\tau \|v_\theta(z_\theta(x,t),t)\|^2\mathrm{d}t$ to the loss function. This leads to straight trajectories in the ODE such that adaptive solvers only require very few steps to solve them. Following Theorem 2.1, the authors of (Onken et al., 2021) note that for $\beta > 0$ the functional

$\mathcal{L}(\theta) + \beta\mathcal{R}(\theta)$ has the same minimizer as the functional $\mathcal{L}(\theta) + \beta W_2^2(\mu_0, \mathcal{T}_{\theta\#}\mu_0)$, which relates to the JKO scheme as pointed out by (Vidal et al., 2023).

3. Neural JKO Scheme

In the following, we learn the steps (3) of the JKO scheme by neural ODEs. While similar schemes were already suggested in several papers (Altekrüger et al., 2023; Alvarez-Melis et al., 2022; Fan et al., 2022; Lambert et al., 2022; Mokrov et al., 2021; Vidal et al., 2023; Xu et al., 2024), we are particularly interested in the convergence properties of the corresponding velocity fields. In Subsection 3.1, we introduce the general scheme and derive its properties. Afterwards, in Subsection 3.2, we describe the corresponding neural network approximation. Throughout this section, we consider the following assumptions on the objective functional $\mathcal{F}$ and our initialization μ_0.

Assumption 3.1. Let $\mathcal{F}\colon \mathcal{P}_2(\mathbb{R}^d) \to \mathbb{R} \cup \{\infty\}$ be proper, lower semi-continuous with respect to narrow convergence, coercive, λ-convex along generalized geodesics and bounded from below. Moreover, assume that $\mathrm{dom}(|\partial\mathcal{F}|) \subseteq \mathcal{P}_2^{\mathrm{ac}}(\mathbb{R}^d)$ and that $\mathcal{F}$ has finite metric derivative $|\partial\mathcal{F}|(\mu_0) < \infty$ at the initialization $\mu_0 \in \mathcal{P}_2^{\mathrm{ac}}(\mathbb{R}^d)$.

This assumption is fulfilled for many important divergences and loss functions $\mathcal{F}$. We list some examples in Appendix B.1. We will later pay particular attention to the reverse Kullback-Leibler divergence $\mathcal{F}(\mu) = \mathrm{KL}(\mu, \nu) = \int p(x) \log\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x$, where ν is a fixed target measure and p and q are the densities of μ and ν respectively. This functional fulfills Assumption 3.1 if $-\log(q)$ is λ-convex.

We stress the fact that the λ-convexity property is assumed for some $\lambda \in \mathbb{R}$, which explicitly includes negative values. Consequently, the theoretical results are also applicable for target densities which are not log-concave.

Additionally, (Ambrosio et al., 2005, Lem 9.2.7) states that the functional $\mathcal{G}(\mu) = \frac{1}{2\tau} W_2^2(\mu, \mu_\tau^k) + \mathcal{F}(\mu)$ is $(\lambda + \frac{1}{\tau})$-convex along geodesics. In particular, for $\tau < \frac{1}{\lambda}$, the functional $\mathcal{G}$ is strongly convex such that we expect that optimizing it with a generative model is much easier than optimizing $\mathcal{F}$, see also Appendix F.1 for a discussion how this can prevent mode collapse.

3.1. Piecewise Geodesic Interpolation

In order to represent the JKO scheme by neural ODEs, we first reformulate it based on Theorem 2.1. To this end, we insert the dynamic formulation of the Wasserstein distance in the Wasserstein proximal mapping defining the steps in (3). This leads to $\mu_\tau^{k+1} = z_\tau^k(\cdot, \tau)_\# \mu_\tau^k$, where

$$(v_{\tau,k}, z_{\tau,k}) \in \underset{\substack{v\colon \mathbb{R}^d \times [0,\tau] \to \mathbb{R}^d \\ \dot{z}(x,t) = v(z(x,t),t),\, z(x,0) = x}}{\arg\min} \mathcal{E}_{\mu_\tau^k}(z, v), \tag{5}$$

where $\mathcal{E}_{\mu_\tau^k}(z, v)$ is given by the dynamic formulation (2) as

$$\frac{1}{2}\int_0^\tau \int_{\mathbb{R}^d} \|v(z(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x)\mathrm{d}t + \mathcal{F}(z(\cdot,\tau)_\# \mu_\tau^k).$$

Finally, we concatenate the velocity fields of all steps by $v_\tau|_{(k\tau,(k+1)\tau]} = v_{\tau,k}$ and obtain the ODE

$$\dot{z}_\tau(x,t) = v_\tau(z_\tau(x,t),t), \qquad z_\tau(x,0) = x.$$

As a straightforward observation, we obtain that the curve defined by the velocity field v_τ is the geodesic interpolation between the points from JKO scheme (3). We state a proof in Appendix B.2.

Corollary 3.2. *Under Ass. 3.1 the following holds true.*

(i) It holds that

$$W_2^2(\mu_\tau^k, \mu_\tau^{k+1}) = \tau \int_0^\tau \int_{\mathbb{R}^d} \|v_{\tau,k}(z_{\tau,k}(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x)\mathrm{d}t,$$

i.e., $v_{\tau,k}$ is the optimal velocity field from Theorem 2.1.

(ii) The curve $\gamma_\tau(t) := z(\cdot,t)_\# \mu_0$ fulfills $\gamma_\tau(k\tau + t\tau) = ((1-t)I + t\mathcal{T}_\tau^k)_\# \mu_\tau^k$ for $t \in [0,1]$, where $\mathcal{T}_\tau^k$ is the optimal transport map between μ_τ^k and μ_τ^{k+1}.

(iii) v_τ and γ_τ solve the continuity equation

$$\partial_t \gamma_\tau(t) + \nabla \cdot (v_\tau(\cdot,t)\gamma(t)) = 0.$$

Analogously to Theorem 2.2 one can show that also the curves γ_τ are converging locally uniformly to the unique Wasserstein gradient flow (see, e.g., the proof of Ambrosio et al., 2005, Thm 11.1.6). In the next subsection, we will approximate the velocity fields $v_{\tau,k}$ by neural networks. In order to retain the stability of the resulting scheme, the next theorem states that also the velocity fields v_τ converge strongly towards the velocity field from the Wasserstein gradient flow. The proof is given in Appendix B.3.

Theorem 3.3. *Suppose that Assumption 3.1 is fulfilled and let $(\tau_l)_l \subseteq (0,\infty)$ with $\tau_l \to 0$. Then, $(v_{\tau_l})_l$ converges strongly to the velocity field $\hat{v} \in L_2(\gamma, \mathbb{R}^d \times [0,T])$ of the Wasserstein gradient flow $\gamma\colon (0,\infty) \to \mathcal{P}_2(\mathbb{R}^d)$ of $\mathcal{F}$ starting in μ^0.*

In some cases the limit velocity field v can be stated explicitly, even though its direct computation is intractable. For details, we refer to Appendix B.1.

3.2. Neural JKO Sampling

In the following, we learn the velocity fields $v_{\tau,k}$ as neural ODEs in order to sample from a target measure ν given by the density $q(x) = \frac{1}{Z_g} g(x)$ with unknown normalization constant $Z_g = \int_{\mathbb{R}^d} g(x)\mathrm{d}x$. To this end, we consider the Wasserstein gradient flow with respect to the reverse KL loss function $\mathcal{F}(\mu) = \mathrm{KL}(\mu, \nu)$ which has the unique minimizer $\mu = \nu$. Note that similar derivations were done in (Vidal et al., 2023; Xu et al., 2024) for the application of generative modeling instead of sampling.

Then, due to (5) the loss function $\mathcal{L}(\theta)$ from the JKO steps for the training of the velocity field v_θ reads as

$$\frac{1}{2}\int_0^\tau \int_{\mathbb{R}^d} \|v_\theta(z_\theta(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x)\mathrm{d}t + \mathcal{F}(z_\theta(\cdot,\tau)_\# \mu_\tau^k),$$

where z_θ is the solution of $\dot{z}_\theta(x,t) = v_\theta(z_\theta(x,t),t)$ with $z_\theta(x,0) = x$. Now, following the derivations of continuous normalizing flows, cf. Section 2.2, the second term of $\mathcal{L}$ can be rewritten (up to an additive constant) as

$$\mathcal{F}(z_\theta(\cdot,\tau)_\# \mu_\tau^k) \propto \mathbb{E}_{x\sim\mu_\tau^k}[-\log(g(z_\theta(x,\tau))) - \ell_\theta(x,\tau)],$$

where ℓ_θ solves $\dot{\ell}_\theta(x,t) = \mathrm{trace}(\nabla v_\theta(z_\theta(x,t),t))$ with $\ell_\theta(\cdot,0) = 0$. Moreover, we can rewrite the first term of $\mathcal{L}$ based on

$$\int_0^\tau \int_{\mathbb{R}^d} \|v_\theta(z_\theta(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x)\mathrm{d}t = \mathbb{E}_{x\sim\mu_\tau^k}[\omega_\theta(x,\tau)],$$

where ω_θ is the solution of $\dot{\omega}_\theta(x,t) = \|v_\theta(z_\theta(x,t),t)\|^2$. Hence, we can represent $\mathcal{L}$ up to an additive constant as

$$\mathcal{L}(\theta) = \mathbb{E}_{x\sim\mu_\tau^k}[-\log(g(z_\theta(x,\tau))) - \ell_\theta(x,\tau) + \omega_\theta(x,\tau)], \tag{6}$$

where $(z_\theta, \ell_\theta, \omega_\theta)$ solves the ODE system

$$\begin{pmatrix} \dot{z}_\theta(x,t) \\ \dot{\ell}_\theta(x,t) \\ \dot{\omega}_\theta(x,t) \end{pmatrix} = \begin{pmatrix} v_\theta(z_\theta(x,t),t) \\ \mathrm{trace}(\nabla v_\theta(z_\theta(x,t),t)) \\ \|v_\theta(z_\theta(x,t),t)\|^2 \end{pmatrix}, \tag{7}$$

with initial conditions $z_\theta(x,0) = x$, $\ell_\theta(x,0) = 0$ and $\omega_\theta(x,0) = 0$. In particular, the loss function $\mathcal{L}$ can be evaluated and differentiated based on samples from μ_τ^k. Once the parameters θ are optimized, we can evaluate the JKO steps in the same way as standard continuous normalizing flows. We summarize training and evaluation of the JKO steps in Algorithm 3 and 4 in Appendix D.1. In practice, the density values are computed and stored in log-space for numerical stability. Additionally, note that the continuous normalizing flows can be replaced by other normalizing flow architectures, see Appendix F.2 and Remark F.3 for details.

4. Importance-Based Rejection Steps

While a large number of existing sampling methods rely on Wasserstein gradient flows with respect to some divergences, it is well known that these loss functions are non-convex. This leads to very slow convergence or only convergence to suboptimal local minima. In particular, if the target distribution is multimodal the modes often do not have the right mass assigned. In this section, we derive a novel rejection step that corrects such imbalanced mass assignments. Remarkably, we will see in Theorem 4.2, that it is possible to access the density after these rejection steps. This observation is crucial, since it allows for the iterative combination of several rejection and neural JKO steps, leading to the proposed *importance corrected neural JKO sampling*.

Importance Sampling As a remedy, many sampling algorithms from the literature are based on importance weights, see, e.g., sequential Monte Carlo samplers (Del Moral et al., 2006) or annealed importance sampling (Neal, 2001). That is, we assign to each generated sample x_i a weight $w_i = \frac{q(x_i)}{p(x_i)}$, where p is some proposal density and q is the density of the target distribution ν. Then, for any ν-integrable function $f\colon \mathbb{R}^d \to \mathbb{R}$ it holds that $\sum_{i=1}^N w_i f(x_i)$ is an unbiased estimator of $\int_{\mathbb{R}^d} f(x)d\nu(x)$. Note that importance sampling is very sensitive with respect to the proposal p which needs to be designed carefully and problem adapted.

Rejection Steps Inspired by importance sampling, we propose to use importance-based rejection steps. More precisely, let μ be a proposal distribution where we can sample from with density $p(x) = f(x)/Z_f$ and denote by ν the target distribution with density $q(x) = g(x)/Z_g$. In the following, we assume that we have access to the unnormalized densities f and g, but not to the normalization constants Z_f and Z_g Then, for a random variable $X \sim \mu$, we now generate a new random variable $\tilde{X}$ by the following procedure: First, we compute the importance based acceptance probability $\alpha(X) = \min\left\{1, \frac{q(X)}{\tilde{c}p(X)}\right\} = \min\left\{1, \frac{g(X)}{cf(X)}\right\}$, where $c > 0$ is a positive hyperparameter and $\tilde{c} = cZ_f/Z_g$. Then, we set $\tilde{X} = X$ with probability $\alpha(X)$ and choose $\tilde{X} = X'$ otherwise, where $X' \sim \mu$ and X are independent.

Remark 4.1. This is a one-step approximation of the classical rejection sampling scheme (Von Neumann, 1951), see also (Andrieu et al., 2003) for an overview. More precisely, we arrive at the classical rejection sampling scheme by choosing $\tilde{c} > \sup_x q(x)/p(x)$ and redo the procedure when X is rejected instead of choosing $\tilde{X} = X'$.

Similarly to importance sampling, the rejection sampling algorithm is highly sensitive towards the proposal distribution p. In particular, it suffers from the curse of dimensionality, in case of a non-tailored proposal p. We will tackle this problem later in the section by choosing p already close to the target density q.

The following theorem describes the density of the distribution $\tilde{\mu}$ of $\tilde{X}$. Moreover, it ensures that the KL divergence to the target distribution decreases. We include the proof in Appendix C.1

Theorem 4.2. *Let $\tilde{\mu}$ be the distribution of $\tilde{X}$. Then, the following holds true.*

(i) $\tilde{\mu}$ admits the density $\tilde{p}$ given by $\tilde{p}(x) = p(x)(\alpha(x) + 1 - \mathbb{E}[\alpha(X)])$. In particular, we have $\tilde{p}(x) = \tilde{f}(x)/Z_{\tilde{f}}$ with $\tilde{f}(x) = f(x)(\alpha(x) + 1 - \mathbb{E}[\alpha(X)])$.

(ii) It holds that $\mathrm{KL}(\tilde{\mu}, \nu) \leq \mathrm{KL}(\mu, \nu)$.

The application of a single *importance-based rejection step* is summarized in Algorithm 1 and illustrated in Figure 2 in the appendix. Note that the value $\mathbb{E}[\alpha(X)]$ can easily be estimated based on samples during the training phase. Indeed, given N iid copies $X_1, ..., X_N$ of X, we obtain that $\mathbb{E}[\alpha(X)] \approx \frac{1}{N}\sum_{i=1}^{N} \alpha(X_N)$ is an unbiased estimator fulfilling the error estimate from the following corollary. The proof is a direct consequence of Hoeffding's inequality and given in Appendix C.2.

Corollary 4.3. *Let $X_1, ..., X_N$ be iid copies of X. Then, it holds*

$$\mathbb{E}\left[\left|\mathbb{E}[\alpha(X)] - \frac{1}{N}\sum_{i=1}^{N}\alpha(X_N)\right|\right] \leq \frac{\sqrt{2\pi}}{\sqrt{N}} \in O\left(\frac{1}{\sqrt{N}}\right).$$

Algorithm 1 Sampling and density propagation for importance-based rejection steps

Input: - samples $x_1^k, ..., x_N^k$ of μ^k with density p^k,
- hyperparameter c (see Remark 4.4).

Assume: - can draw samples from μ^k,
- can evaluate the unnormalized density g.

for $i = 1, ..., N$ **do**

1. Compute $\alpha_k(x_i^k) = \min\left\{1, \frac{g(x_i^k)}{cp^k(x_i^k)}\right\}$.
2. Draw u uniformly from $[0, 1]$ and x' from μ_τ^k.
3. Set $x_i^{k+1} = \begin{cases} x_i^k, & \text{if } u \leq \alpha(x_i^k), \\ x' & \text{if } u > \alpha(x_i^k). \end{cases}$
4. Compute $\alpha_k(x_i^{k+1}) = \min\left\{1, \frac{g(x_i^{k+1})}{cp^k(x_i^{k+1})}\right\}$.
5. Define the density value

$$p^{k+1}(x_i^{k+1}) = p^k(x_i^{k+1})(\alpha_k(x_i^{k+1}) + 1 - \mathbb{E}[\alpha_k(X_k)]).$$

end for

Output: - Samples $x_1^{k+1}, ..., x_N^{k+1}$ of μ_τ^{k+1}.
- Density values $\{p_\tau^{k+1}(x_i^{k+1})\}_{i=1}^N$.

Remark 4.4 (Choice of c). We choose the hyperparameter c such that a constant ratio $r > 0$ of the samples will be resampled, i.e., that $\mathbb{E}[\alpha(X)] \approx 1 - r$. To this end, we assume that we are given samples $x_1, ..., x_N$ from X and approximate

$$\mathbb{E}[\alpha(X)] \approx \frac{1}{N}\sum_{i=1}^{N}\alpha(x_i) = \frac{1}{N}\sum_{i=1}^{N}\min\left\{1, \frac{g(x_i)}{cp(x_i)}\right\}.$$

Note that the right side of this formula depends monotonically on c such that we can find $c > 0$ such that $\mathbb{E}[\alpha(X)] = 1 - r$ by a bisection search. In our numerical experiments, we set $r = 0.2$. We summarize the choice of c in Algorithm 5 in Appendix D.1.

Algorithm 2 Importance corrected neural JKO sampling

Input:
- unnormalized target density g,
- initial measure μ^0 with density p_0,
- Number $N \in \mathbb{N}$ of samples,
- Number $K \in \mathbb{N}$ of total steps.

Output: Sample generator $\{x^i\}_{i=1}^N \sim \hat{\nu} \approx \nu = g \cdot \lambda$.

Let $x_1^0, \ldots, x_N^0 \sim \mu^0$.

for $k = 1, \ldots, K$ **do**

Define μ^k with density p^k and draw samples $x_1^k, \ldots, x_N^k \sim \mu^k$ either by
- *neural JKO step* by solving the ODE (7) or
- *importance based rejection* by Alg. 1.

(details in Alg. 3, 4 resp. Alg. 1, 5)

end for

Set $\hat{\nu} = \mu^K$ with density p^K.

Neural JKO Sampling with Importance Correction Finally, we combine the neural JKO scheme from the previous section with our rejection steps to obtain a sampling algorithm. More precisely, we start with a simple latent distribution μ_0 with known density and which we can sample from. In our numerical experiments this will be a standard Gaussian distribution. Now, we iteratively generate distributions μ_k, $k = 1, ..., K$ by applying either neural JKO steps as described in Algorithm 3 and 4 or importance-based rejection steps as described in Algorithm 5 and 1. We call the resulting model an importance corrected neural JKO model, which we summarize in Algorithm 2. During the sampling process we can maintain the density values $p^k(x)$ of the density p^k of μ_k for the generated samples by the Algorithms 4 and 1. Moreover, we can also use Theorem 4.2 to evaluate the density p^k at some arbitrary point $x \in \mathbb{R}^d$. We outline this density evaluation process in Appendix D.2.

Remark 4.5 (Runtime Limitations). The sampling time of our importance corrected neural JKO sampling depends exponentially on the number of rejection steps since in each rejection step we resample a constant fraction of the samples.

Table 1: Energy distance. We run each method 5 times and state the average value and corresponding standard deviations. The rightmost column shows the reference sampling error, i.e., the lower bound magnitude of the average energy distance between two sets of different samples drawn from the ground truth. A smaller energy distance indicates a better result.

	Sampler						
Distribution	MALA	HMC	DDS	CRAFT	Neural JKO	Neural JKO IC (**ours**)	Sampling Error
Mustache	$4.6 \times 10^{-2} \pm 1.6 \times 10^{-3}$	$1.7 \times 10^{-2} \pm 4.3 \times 10^{-4}$	$6.9 \times 10^{-2} \pm 1.8 \times 10^{-3}$	$9.2 \times 10^{-2} \pm 9.9 \times 10^{-3}$	$1.8 \times 10^{-2} \pm 2.0 \times 10^{-3}$	$\mathbf{2.9 \times 10^{-3} \pm 4.4 \times 10^{-4}}$	8.6×10^{-5}
shifted 8 Modes	$5.3 \times 10^{-3} \pm 4.9 \times 10^{-4}$	$4.1 \times 10^{-5} \pm 3.3 \times 10^{-5}$	$1.2 \times 10^{-2} \pm 4.1 \times 10^{-3}$	$5.2 \times 10^{-2} \pm 1.1 \times 10^{-2}$	$1.3 \times 10^{-1} \pm 3.8 \times 10^{-3}$	$\mathbf{1.2 \times 10^{-5} \pm 5.1 \times 10^{-6}}$	2.6×10^{-5}
shifted 8 Peaky	$1.3 \times 10^{-1} \pm 3.2 \times 10^{-3}$	$1.2 \times 10^{-1} \pm 2.4 \times 10^{-3}$	$1.1 \times 10^{-2} \pm 3.4 \times 10^{-3}$	$5.2 \times 10^{-2} \pm 2.2 \times 10^{-2}$	$1.3 \times 10^{-1} \pm 2.2 \times 10^{-3}$	$\mathbf{3.4 \times 10^{-5} \pm 8.2 \times 10^{-6}}$	2.4×10^{-5}
Funnel	$1.2 \times 10^{-1} \pm 3.1 \times 10^{-3}$	$\mathbf{3.1 \times 10^{-3} \pm 3.2 \times 10^{-4}}$	$2.6 \times 10^{-1} \pm 2.6 \times 10^{-2}$	$7.4 \times 10^{-2} \pm 2.8 \times 10^{-3}$	$4.6 \times 10^{-2} \pm 1.6 \times 10^{-3}$	$1.4 \times 10^{-2} \pm 8.2 \times 10^{-4}$	3.4×10^{-4}
GMM-10	$1.2 \times 10^{-2} \pm 5.5 \times 10^{-3}$	$1.2 \times 10^{-2} \pm 5.2 \times 10^{-3}$	$3.7 \times 10^{-3} \pm 1.6 \times 10^{-3}$	$1.8 \times 10^{-1} \pm 6.6 \times 10^{-2}$	$1.1 \times 10^{-2} \pm 5.6 \times 10^{-3}$	$\mathbf{5.3 \times 10^{-5} \pm 1.7 \times 10^{-5}}$	4.6×10^{-5}
GMM-20	$9.1 \times 10^{-3} \pm 2.8 \times 10^{-3}$	$9.1 \times 10^{-3} \pm 2.7 \times 10^{-3}$	$5.0 \times 10^{-3} \pm 1.5 \times 10^{-3}$	$5.4 \times 10^{-1} \pm 1.4 \times 10^{-1}$	$1.0 \times 10^{-2} \pm 2.8 \times 10^{-3}$	$\mathbf{1.1 \times 10^{-4} \pm 3.4 \times 10^{-5}}$	6.4×10^{-5}
GMM-50	$2.4 \times 10^{-2} \pm 7.5 \times 10^{-3}$	$2.4 \times 10^{-2} \pm 7.5 \times 10^{-3}$	$2.3 \times 10^{-2} \pm 1.1 \times 10^{-2}$	$1.8 \times 10^{0} \pm 1.7 \times 10^{-1}$	$2.7 \times 10^{-2} \pm 7.8 \times 10^{-3}$	$\mathbf{1.0 \times 10^{-4} \pm 4.6 \times 10^{-5}}$	1.1×10^{-4}
GMM-100	$3.6 \times 10^{-2} \pm 1.6 \times 10^{-2}$	$3.7 \times 10^{-2} \pm 1.7 \times 10^{-2}$	$3.9 \times 10^{-2} \pm 2.1 \times 10^{-2}$	$2.8 \times 10^{+1} \pm 1.0 \times 10^{-1}$	$4.7 \times 10^{-2} \pm 2.2 \times 10^{-2}$	$\mathbf{6.0 \times 10^{-4} \pm 3.4 \times 10^{-4}}$	1.5×10^{-4}
GMM-200	$6.4 \times 10^{-2} \pm 2.1 \times 10^{-2}$	$6.6 \times 10^{-2} \pm 1.9 \times 10^{-2}$	$9.8 \times 10^{-2} \pm 3.1 \times 10^{-2}$	$3.9 \times 10^{0} \pm 1.6 \times 10^{-1}$	$8.9 \times 10^{-2} \pm 2.7 \times 10^{-2}$	$\mathbf{3.3 \times 10^{-3} \pm 1.9 \times 10^{-3}}$	2.0×10^{-4}

Table 2: Estimated $\log(Z)$. We run each method 5 times and state the average value and corresponding standard deviations. The DDS values for LGCP are from (Vargas et al., 2023a). Higher values of $\log(Z)$ estimates correspond to better results.

	Sampler				
Distribution	DDS	CRAFT	Neural JKO	Neural JKO IC (**ours**)	Ground Truth
Mustache	$-1.5 \times 10^{-1} \pm 2.7 \times 10^{-2}$	$-6.5 \times 10^{-2} \pm 5.5 \times 10^{-2}$	$-3.0 \times 10^{-2} \pm 2.6 \times 10^{-3}$	$\mathbf{-7.3 \times 10^{-3} \pm 8.2 \times 10^{-4}}$	0
shifted 8 Modes	$-5.7 \times 10^{-2} \pm 2.0 \times 10^{-2}$	$-1.2 \times 10^{-2} \pm 1.4 \times 10^{-3}$	$-3.4 \times 10^{-1} \pm 3.1 \times 10^{-3}$	$\mathbf{+5.1 \times 10^{-6} \pm 2.4 \times 10^{-3}}$	0
shifted 8 Peaky	$-1.2 \times 10^{-1} \pm 2.2 \times 10^{-2}$	$-1.8 \times 10^{-3} \pm 2.6 \times 10^{-3}$	$-3.5 \times 10^{-1} \pm 3.1 \times 10^{-3}$	$\mathbf{-2.1 \times 10^{-3} \pm 3.2 \times 10^{-3}}$	0
Funnel	$-1.8 \times 10^{-1} \pm 6.8 \times 10^{-2}$	$-1.2 \times 10^{-1} \pm 7.9 \times 10^{-3}$	$-1.4 \times 10^{-1} \pm 1.6 \times 10^{-3}$	$\mathbf{-7.1 \times 10^{-3} \pm 1.9 \times 10^{-3}}$	0
GMM-10	$-2.3 \times 10^{-1} \pm 1.0 \times 10^{-1}$	$-8.5 \times 10^{-1} \pm 1.7 \times 10^{-1}$	$-4.3 \times 10^{-1} \pm 5.1 \times 10^{-2}$	$\mathbf{+3.5 \times 10^{-3} \pm 2.0 \times 10^{-3}}$	0
GMM-20	$-5.1 \times 10^{-1} \pm 6.0 \times 10^{-2}$	$-1.5 \times 10^{0} \pm 1.7 \times 10^{-1}$	$-6.3 \times 10^{-1} \pm 2.7 \times 10^{-2}$	$\mathbf{+6.4 \times 10^{-3} \pm 3.8 \times 10^{-3}}$	0
GMM-50	$-1.3 \times 10^{0} \pm 3.3 \times 10^{-1}$	$-2.3 \times 10^{0} \pm 1.5 \times 10^{-3}$	$-9.3 \times 10^{-1} \pm 4.6 \times 10^{-2}$	$\mathbf{+1.1 \times 10^{-2} \pm 3.9 \times 10^{-3}}$	0
GMM-100	$-3.0 \times 10^{0} \pm 7.3 \times 10^{-1}$	$-2.3 \times 10^{0} \pm 8.6 \times 10^{-2}$	$-1.8 \times 10^{0} \pm 9.6 \times 10^{-2}$	$\mathbf{-3.9 \times 10^{-2} \pm 7.8 \times 10^{-3}}$	0
GMM-200	$-9.4 \times 10^{0} \pm 7.2 \times 10^{-1}$	$-6.3 \times 10^{0} \pm 1.5 \times 10^{-1}$	$-5.2 \times 10^{0} \pm 2.5 \times 10^{-1}$	$\mathbf{-5.6 \times 10^{-2} \pm 1.3 \times 10^{-2}}$	0
LGCP	$503.0 \pm 7.7 \times 10^{-1}$	$507.6 \pm 3.2 \times 10^{-1}$	$499.9 \pm 1.7 \times 10^{-1}$	$\mathbf{508.2 \pm 1.0 \times 10^{-1}}$	not available

However, due to the moderate exponential base of $1 + r$ we will see in the numerical part that we are able to perform a significant number of rejection steps in a tractable time.

In our numerics, we build our importance corrected neural JKO model by first applying $n_1 \in \mathbb{N}$ neural JKO steps followed by $n_2 \in \mathbb{N}$ blocks consisting out of one neural JKO step and three importance-based rejection steps, where n_1 and n_2 are hyperparameters given for each model separately in Table 6. For the neural JKO steps, we choose an initial step size $\tau_0 > 0$ as a hyper-parameter and then increase the step size exponentially by $\tau_{k+1} = 4\tau_k$. Note that one could alternatively use adaptive step sizes similar to (Xu et al., 2024). However, for our setting, we found that the simple step size rule is sufficient.

5. Numerical Results

We compare our method with classical Monte Carlo samplers like a Metropolis adjusted Langevin sampling (MALA) and Hamiltonian Monte Carlo (HMC), see e.g., (Betancourt, 2017; Roberts & Tweedie, 1996). Additionally, we compare with two recent deep-learning based sampling algorithms, namely denoising diffusion samplers (DDS, Vargas et al., 2023a) and continual repeated annealed flow transport Monte Carlo (CRAFT, Matthews et al., 2022). We evaluate all methods on a set of common test distributions which is described in detail in Appendix E.1. Moreover, we report the error measures for our *importance corrected neural JKO* sampler (neural JKO IC), see Appendix E.5 for implementation details. Additionally, we emphasize the importance of the rejection steps by reporting values for the same a neural JKO scheme without rejection steps (neural JKO).

For evaluating the quality of our results, we use two different metrics. First, we evaluate the energy distance (Székely, 2002). It is a kernel metric which can be evaluated purely based on two sets of samples from the model and the ground truth. Moreover it encodes the geometry of the space such that a slight perturbation of the samples only leads to a slight change in the energy distance. Second, we estimate the log-normalizing constant which is equivalent to approximating the reverse KL loss of the model. A higher estimate of the log-normalizing constant corresponds to a smaller KL divergence between generated and target distribution and therefore to a higher similarity of the two measures. Since this requires the density of the model, this approach is not applicable for MALA and HMC.

The results are given in Table 1 and 2. Our importance corrected neural JKO sampling significantly outperforms the comparison for all test distributions. In particular, we observe that for shifted 8 modes, shifted 8 peaky and GMM-d the energy distance between neural JKO IC and ground truth samples is in the same order of magnitude as the energy distance between to different sets of ground truth samples. This implies that the distribution generated by neural JKO IC is indistinguishable from the target distribution in the energy distance. For these examples, the $\log(Z)$ esitmate is sometimes slightly larger than the ground truth, which can be explained by numerical effects, see Remark E.1 for a detailed discussion. We point out to a precise description

of the metrics, the implementation details and additional experiments and figures in Appendix E.

6. Conclusions

Methodology We proposed a novel and expressive generative method that enables the efficient and accurate sampling from a prescribed unnormalized target density which is empirically confirmed in numerical examples. To this end, we combine *local sampling steps*, relying on piecewise geodesic interpolations of the JKO scheme realized by CNFs, and *non-local rejection and resampling steps* based on importance weights. Since the proposal of the rejection step is generated by the model itself, they do not suffer from the curse of dimensionality as opposed to classical variants of rejection sampling. The proposed approach provides the advantage that we can draw *independent* samples while correcting imbalanced mode weights, iteratively refine the current approximation and evaluate the density of the generated distribution. This is a consequence of the *density value propagation through CNFs* and Theorem 4.2 and enables possible further post-processing steps that require density evaluations of the approximated sample process.

Outlook Our method allows for the pointwise access to the approximated target density and the log normalization constant. These quantities can be used for the error monitoring during the training and hence provide guidelines for the adaptive design of the emulator in terms of CNF -or rejection/resampling steps and provide a straightforward stopping criterion. The importance-based rejection steps can also be used in other domains like fine-tuning of score-based diffusion models for downstream tasks, see Denker et al. (2025).

Limitations In the situation, when the emulator is realized through a stack of underlying rejection/resampling steps, the sample generation process time is negatively affected, see Remark 4.5. In order to resolve the drawback we plan to utilize diffusion models for the sample generation. This is part of ongoing and future work by the authors. Finally, the use of continuous normalizing flows comes with computational challenges, which we discuss in detail in Appendix F.2.

Impact Statement

This paper studies a general problem from the field of statistics and machine learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgements

JH acknowledges funding by the German Research Foundation (DFG) within the Walter-Benjamin Programme with project number 530824055 and by the EPSRC programme grant *The Mathematics of Deep Learning* with reference EP/V026259/1. RG acknowledges support by the DFG MATH+ project AA5-5 (was EF1-25) - *Wasserstein Gradient Flows for Generalised Transport in Bayesian Inversion*.

References

Akhound-Sadegh, T., Rector-Brooks, J., Joey Bose, A., Mittal, S., Lemos, P., Liu, C.-H., Sendera, M., Ravanbakhsh, S., Gidel, G., Bengio, Y., et al. Iterated denoising energy matching for sampling from boltzmann densities. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 760–786, 2024.

Albergo, M. S. and Vanden-Eijnden, E. Nets: A non-equilibrium transport sampler. *arXiv preprint arXiv:2410.02711*, 2024.

Altekrüger, F. and Hertrich, J. WPPNets and WPPFlows: The power of Wasserstein patch priors for superresolution. *SIAM Journal on Imaging Sciences*, 16(3):1033–1067, 2023.

Altekrüger, F., Hertrich, J., and Steidl, G. Neural Wasserstein gradient flows for maximum mean discrepancies with Riesz kernels. In *International Conference on Machine Learning*, pp. 664–690. PMLR, 2023.

Alvarez-Melis, D., Schiff, Y., and Mroueh, Y. Optimizing functionals on the space of probabilities with input convex neural networks. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856.

Ambrosio, L., Gigli, N., and Savare, G. *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Lectures in Mathematics ETH Zürich. Birkhäuser, Basel, 2005.

Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. An introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003.

Andrle, A., Farchmin, N., Hagemann, P., Heidenreich, S., Soltwisch, V., and Steidl, G. Invertible neural networks versus MCMC for posterior reconstruction in grazing incidence X-ray fluorescence. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pp. 528–539. Springer, 2021.

Arbel, M., Korba, A., Salim, A., and Gretton, A. Maximum mean discrepancy gradient flow. *Advances in Neural Information Processing Systems*, 32, 2019.

Arbel, M., Matthews, A., and Doucet, A. Annealed flow transport Monte Carlo. In *International Conference on Machine Learning*, pp. 318–330. PMLR, 2021.

Ardizzone, L., Kruse, J., Wirkert, S., Rahner, D., Pellegrini, E. W., Klessen, R. S., Maier-Hein, L., Rother, C., and Köthe, U. Analyzing inverse problems with invertible neural networks. In *International Conference on Learning Representations*, 2019.

Behrmann, J., Grathwohl, W., Chen, R. T. Q., Duvenaud, D., and Jacobsen, J.-H. Invertible residual networks. In *International Conference on Machine Learning*, pp. 573–582. PMLR, 2019.

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.

Betancourt, M. A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv:1701.02434*, 2017.

Blessing, D., Jia, X., Esslinger, J., Vargas, F., and Neumann, G. Beyond ELBOs: A large-scale evaluation of variational methods for sampling. In *International Conference on Machine Learning*, 2024.

Brenier, Y. Décomposition polaire et réarrangement monotone des champs de vecteurs. *Comptes Rendus de l'Académie des Sciences Paris Series I Mathematics*, 305 (19):805–808, 1987.

Chen, J., Richter, L., Berner, J., Blessing, D., Neumann, G., and Anandkumar, A. Sequential controlled Langevin diffusions. In *The Thirteenth International Conference on Learning Representations*, 2025.

Chen, R. T. Q. torchdiffeq, 2018. URL https://github.com/rtqichen/torchdiffeq.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 31, 2018.

Chen, R. T. Q., Behrmann, J., Duvenaud, D. K., and Jacobsen, J.-H. Residual flows for invertible generative modeling. *Advances in Neural Information Processing Systems*, 32, 2019.

Chen, Y., Huang, D. Z., Huang, J., Reich, S., and Stuart, A. M. Sampling via gradient flows in the space of probability measures. *arXiv preprint arXiv:2310.03597*, 2023.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems*, 26, 2013.

De Cao, N., Titov, I., and Aziz, W. Block neural autoregressive flow. In *Uncertainty in Artificial Intelligence*, pp. 1263–1273. PMLR, 2020.

Del Moral, P., Doucet, A., and Jasra, A. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 68(3):411–436, 2006.

Denker, A., Vargas, F., Padhy, S., Didi, K., Mathis, S., Dutordoir, V., Barbano, R., Mathieu, E., Komorowska, U. J., and Lio, P. DEFT: Efficient finetuning of conditional diffusion models by learning the generalised h-transform. *arXiv preprint arXiv:2406.01781*, 2024.

Denker, A., Padhy, S., Vargas, F., and Hertrich, J. Iterative importance fine-tuning of diffusion models. *arXiv preprint arXiv:2502.04468*, 2025.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. In *International Conference on Learning Representations*, 2016.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. Neural spline flows. *Advances in Neural Information Processing Systems*, 2019.

Eigel, M., Gruhlke, R., and Marschall, M. Low-rank tensor reconstruction of concentrated densities with application to bayesian inversion. *Statistics and Computing*, 32(2): 27, 2022.

Eigel, M., Gruhlke, R., and Sommer, D. Less interaction with forward models in Langevin dynamics: Enrichment and homotopy. *SIAM Journal on Applied Dynamical Systems*, 23(3):1870–1908, 2024.

Fan, J., Zhang, Q., Taghvaei, A., and Chen, Y. Variational Wasserstein gradient flow. In *International Conference on Machine Learning*, pp. 6185–6215. PMLR, 2022.

Flamary, R., Courty, N., Gramfort, A., Alaya, M., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., et al. POT: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

Gabrié, M., Rotskoff, G., and Vanden-Eijnden, E. Adaptive Monte Carlo augmented with normalizing flows. *Proceedings of the National Academy of Sciences*, 119(10): e2109420119, 2022.

Garbuno-Inigo, A., Nüsken, N., and Reich, S. Affine invariant interacting Langevin dynamics for Bayesian inference. *SIAM Journal on Applied Dynamical Systems*, 19 (3):1633–1658, 2020.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. *Markov chain Monte Carlo in practice*. CRC press, 1995.

Girolami, M. and Calderhead, B. Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, 2011.

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2019.

Hagemann, P., Hertrich, J., and Steidl, G. Stochastic normalizing flows for inverse problems: a Markov chains viewpoint. *SIAM/ASA Journal on Uncertainty Quantification*, 10(3):1162–1190, 2022.

Hagemann, P., Hertrich, J., and Steidl, G. *Generalized normalizing flows via Markov chains*. Cambridge University Press, 2023.

Hagemann, P., Hertrich, J., Altekrüger, F., Beinert, R., Chemseddine, J., and Steidl, G. Posterior sampling based on gradient flows of the MMD with negative distance kernel. In *International Conference on Learning Representations*, 2024.

He, J., Du, Y., Vargas, F., Zhang, D., Padhy, S., OuYang, R., Gomes, C. P., and Hernández-Lobato, J. M. No trick, no treat: Pursuits and challenges towards simulation-free training of neural samplers. In *Frontiers in Probabilistic Inference: Learning meets Sampling*, 2025.

Hertrich, J. Proximal residual flows for Bayesian inverse problems. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pp. 210–222. Springer, 2023.

Hertrich, J., Wald, C., Altekrüger, F., and Hagemann, P. Generative sliced MMD flows with Riesz kernels. In *International Conference on Learning Representations*, 2024.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020.

Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

Hoffman, M., Sountsov, P., Dillon, J., Langmore, I., Tran, D., and Vasudevan, S. Neutra-lizing bad geometry in Hamiltonian Monte Carlo using neural transport. *arXiv preprint arXiv:1903.03704*, 2019.

Hoffman, M. D. and Gelman, A. The no-u-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15(1): 1593–1623, 2014.

Huang, C.-W., Krueger, D., Lacoste, A., and Courville, A. Neural autoregressive flows. In *International Conference on Machine Learning*, pp. 2078–2087. PMLR, 2018.

Hutchinson, M. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

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 Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Lambert, M., Chewi, S., Bach, F., Bonnabel, S., and Rigollet, P. Variational inference via Wasserstein gradient flows. *Advances in Neural Information Processing Systems*, 35: 14434–14447, 2022.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose Bayesian inference algorithm. *Advances in Neural Information Processing Systems*, 29, 2016a.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose Bayesian inference algorithm. *Advances in Neural Information Processing Systems*, 29, 2016b.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

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.

Lu, Y., Lu, J., and Nolen, J. Accelerating Langevin sampling with birth-death. *arXiv preprint arXiv:1905.09863*, 2019.

Marzouk, Y., Moselhy, T., Parno, M., and Spantini, A. Sampling via measure transport: An introduction. *Handbook of uncertainty quantification*, 1:2, 2016.

Matthews, A., Arbel, M., Rezende, D. J., and Doucet, A. Continual repeated annealed flow transport Monte Carlo. In *International Conference on Machine Learning*, pp. 15196–15219. PMLR, 2022.

Midgley, L. I., Stimper, V., Simm, G. N. C., Schölkopf, B., and Hernández-Lobato, J. M. Flow annealed importance sampling bootstrap. In *The Eleventh International Conference on Learning Representations*, 2023.

Mokrov, P., Korotin, A., Li, L., Genevay, A., Solomon, J. M., and Burnaev, E. Large-scale Wasserstein gradient flows. *Advances in Neural Information Processing Systems*, 34: 15243–15256, 2021.

Møller, J., Syversveen, A. R., and Waagepetersen, R. P. Log gaussian cox processes. *Scandinavian Journal of Statistics*, 25(3):451–482, 1998.

Neal, R. M. Annealed importance sampling. *Statistics and computing*, 11:125–139, 2001.

Neal, R. M. Slice sampling. *The Annals of Statistics*, 31(3): 705–767, 2003.

Noé, F., Olsson, S., Köhler, J., and Wu, H. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

Onken, D., Fung, S. W., Li, X., and Ruthotto, L. OT-flow: Fast and accurate continuous normalizing flows via optimal transport. In *AAAI Conference on Artificial Intelligence*, volume 35, pp. 9223–9232, 2021.

Papamakarios, G., Pavlakou, T., , and Murray, I. Masked autoregressive flow for density estimation. *Advances in Neural Information Processing Systems*, pp. 2338–2347, 2017.

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

Phillips, A., Dau, H.-D., Hutchinson, M. J., De Bortoli, V., Deligiannidis, G., and Doucet, A. Particle denoising diffusion sampler. In *International Conference on Machine Learning*, 2024.

Qiu, Y. and Wang, X. Efficient multimodal sampling via tempered distribution flow. *Journal of the American Statistical Association*, 119(546):1446–1460, 2024.

Rezende, D. and Mohamed, S. Variational inference with normalizing flows. In *International Conference on Machine Learning*, pp. 1530–1538. PMLR, 2015.

Richter, L. and Berner, J. Improved sampling via learned diffusions. In *International Conference on Learning Representations*, 2024.

Roberts, G. O. and Tweedie, R. L. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

Rossky, P. J., Doll, J. D., and Friedman, H. L. Brownian dynamics as smart Monte Carlo simulation. *The Journal of Chemical Physics*, 69(10):4628–4633, 1978. doi: 10.1063/1.436415.

Ruthotto, L. and Haber, E. An introduction to deep generative modeling. *GAMM-Mitteilungen*, 44(2):e202100008, 2021.

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

Sejdinovic, D., Sriperumbudur, B., Gretton, A., and Fukumizu, K. Equivalence of distance-based and RKHS-based statistics in hypothesis testing. *The Annals of Statistics*, 41(5):2263 – 2291, 2013.

Sommer, D., Gruhlke, R., Kirstein, M., Eigel, M., and Schillings, C. Generative modelling with tensor train approximations of Hamilton–Jacobi–Bellman equations. *arXiv preprint arXiv:2402.15285*, 2024.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

Székely, G. E-statistics: The energy of statistical samples. *Techical Report, Bowling Green University*, 2002.

Vargas, F., Grathwohl, W., and Doucet, A. Denoising diffusion samplers. *International Conference on Learning Representations*, 2023a.

Vargas, F., Padhy, S., Blessing, D., and Nüsken, N. Transport meets variational inference: Controlled Monte Carlo diffusions. *arXiv preprint arXiv:2307.01050*, 2023b.

Vidal, A., Wu Fung, S., Tenorio, L., Osher, S., and Nurbekyan, L. Taming hyperparameter tuning in continuous normalizing flows using the JKO scheme. *Scientific Reports*, 13(1):4501, 2023.

Von Neumann, J. Various techniques used in connection with random digits. *Applied Math Series*, 12(36-38):1, 1951.

Wang, Y. and Li, W. Accelerated information gradient flow. *Journal of Scientific Computing*, 90:1–47, 2022.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient Langevin dynamics. In *International Conference on Machine Learning*, pp. 681–688, 2011.

Woo, D. and Ahn, S. Iterated energy-based flow matching for sampling from Boltzmann densities. *arXiv preprint arXiv:2408.16249*, 2024.

Wu, D. and Xie, Y. Annealing flow generative model towards sampling high-dimensional and multi-modal distributions. *arXiv preprint arXiv:2409.20547*, 2024.

Wu, H., Köhler, J., and Noé, F. Stochastic normalizing flows. *Advances in Neural Information Processing Systems*, 33: 5933–5944, 2020.

Xu, C., Cheng, X., and Xie, Y. Normalizing flow neural networks by JKO scheme. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhang, Q. and Chen, Y. Path integral sampler: a stochastic control approach for sampling. *arXiv preprint arXiv:2111.15141*, 2021.

A. Background on Wasserstein Spaces

We give some theoretical background on Wasserstein gradient flows extending Section 2. In what follows we refer to $L_2(\mu) := L_2(\mu, \mathbb{R}^d)$ as the set of square integrable measurable functions on $\mathbb{R}^d$ with respect to a given measure $\mu \in \mathcal{P}(\mathbb{R}^d)$.

For an absolutely continuous curve γ, there exists a unique minimal norm solution v_t of the continuity equation (1) in the sense that any solution $\tilde{v}$ of (1) fulfills $\|\tilde{v}(\cdot,t)\|_{L_2(\gamma(t))} \geq \|v(\cdot,t)\|_{L_2(\gamma(t))}$ for almost every t. This is the unique solution of (1) such that $v(\cdot,t)$ is contained in the regular tangent space

$$\mathrm{T}_\mu \mathcal{P}_2(\mathbb{R}^d) := \overline{\{\lambda(T - \mathrm{Id}) : (\mathrm{Id}, T)_\# \mu \in \Gamma^{\mathrm{opt}}(\mu, T_\# \mu),\ \lambda > 0\}}^{L_2(\mu)}.$$

An absolutely continuous curve $\gamma\colon (0,\infty) \to \mathcal{P}_2(\mathbb{R}^d)$ with velocity field $v_t \in \mathrm{T}_{\gamma(t)}\mathcal{P}_2(\mathbb{R}^d)$ is a *Wasserstein gradient flow with respect to* $\mathcal{F}\colon \mathcal{P}_2(\mathbb{R}^d) \to (-\infty, \infty]$ if

$$v_t \in -\partial\mathcal{F}(\gamma(t)), \quad \text{for a.e. } t > 0,$$

where $\partial\mathcal{F}(\mu)$ denotes the reduced Fréchet subdiffential at μ defined as

$$\partial\mathcal{F}(\mu) := \left\{ \xi \in L_2(\mu) : \mathcal{F}(\nu) - \mathcal{F}(\mu) \geq \inf_{\pi \in \Gamma^{\mathrm{opt}}(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \langle \xi(x), y - x \rangle \, \mathrm{d}\pi(x,y) + o(W_2(\mu,\nu))\ \forall \nu \in \mathcal{P}_2(\mathbb{R}^d) \right\}.$$

The norm $\|v_t\|_{L_2(\gamma(t))}$ of the velocity field of a Wasserstein gradient flow coincides for almost every t with the metric derivative

$$|\partial\mathcal{F}|(\mu) = \inf_{\nu \to \mu} \frac{\mathcal{F}(\mu) - \mathcal{F}(\nu)}{W_2(\mu,\nu)}.$$

For the convergence result from Theorem 2.2, we need two more definitions. First, we need some convexity assumption. For $\lambda \in \mathbb{R}$, $\mathcal{F}\colon \mathcal{P}_2(\mathbb{R}^d) \to \mathbb{R} \cup \{+\infty\}$ is called *λ-convex along geodesics* if, for every $\mu, \nu \in \operatorname{dom}\mathcal{F} := \{\mu \in \mathcal{P}_2(\mathbb{R}^d) : \mathcal{F}(\mu) < \infty\}$, there exists at least one geodesics $\gamma\colon [0,1] \to \mathcal{P}_2(\mathbb{R}^d)$ between μ and ν such that

$$\mathcal{F}(\gamma(t)) \leq (1-t)\,\mathcal{F}(\mu) + t\,\mathcal{F}(\nu) - \tfrac{\lambda}{2}\, t(1-t)\, W_2^2(\mu,\nu), \qquad t \in [0,1].$$

To ensure uniqueness and convergence of the JKO scheme, a slightly stronger condition, namely being *λ-convex along generalized geodesics* will be in general needed. Based on the set of three-plans with base $\sigma \in \mathcal{P}_2(\mathbb{R}^d)$ given by

$$\Gamma_\sigma(\mu,\nu) := \left\{ \boldsymbol{\alpha} \in \mathcal{P}_2(\mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d) : (\pi_1)_\# \boldsymbol{\alpha} = \sigma, (\pi_2)_\# \boldsymbol{\alpha} = \mu, (\pi_3)_\# \boldsymbol{\alpha} = \nu \right\},$$

the so-called *generalized geodesics* $\gamma\colon [0,\epsilon] \to \mathcal{P}_2(\mathbb{R}^d)$ joining μ and ν (with base σ) is defined as

$$\gamma(t) := \left((1 - \tfrac{t}{\epsilon})\pi_2 + \tfrac{t}{\epsilon}\pi_3\right)_\# \boldsymbol{\alpha}, \qquad t \in [0,\epsilon], \tag{8}$$

where $\boldsymbol{\alpha} \in \Gamma_\sigma(\mu,\nu)$ with $(\pi_{1,2})_\# \boldsymbol{\alpha} \in \Gamma^{\mathrm{opt}}(\sigma,\mu)$ and $(\pi_{1,3})_\# \boldsymbol{\alpha} \in \Gamma^{\mathrm{opt}}(\sigma,\nu)$, see Definition 9.2.2 in (Ambrosio et al., 2005). The plan $\boldsymbol{\alpha}$ may be interpreted as transport from μ to ν via σ. Then a function $\mathcal{F}\colon \mathcal{P}_2(\mathbb{R}^d) \to (-\infty,\infty]$ is called *λ-convex along generalized geodesics* (see Ambrosio et al., 2005, Definition 9.2.4), if for every $\sigma, \mu, \nu \in \operatorname{dom}\mathcal{F}$, there exists at least one generalized geodesics $\gamma\colon [0,1] \to \mathcal{P}_2(\mathbb{R}^d)$ related to some $\boldsymbol{\alpha}$ in (8) such that

$$\mathcal{F}(\gamma(t)) \leq (1-t)\,\mathcal{F}(\mu) + t\,\mathcal{F}(\nu) - \tfrac{\lambda}{2}\, t(1-t)\, W_{\boldsymbol{\alpha}}^2(\mu,\nu), \qquad t \in [0,1],$$

where

$$W_{\boldsymbol{\alpha}}^2(\mu,\nu) := \int_{\mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d} \|y - z\|_2^2 \, \mathrm{d}\boldsymbol{\alpha}(x,y,z).$$

Every function being λ-convex along generalized geodesics is also λ-convex along geodesics since generalized geodesics with base $\sigma = \mu$ are actual geodesics. Second, a λ-convex functional $\mathcal{F}\colon \mathcal{P}_2(\mathbb{R}^d) \to \mathbb{R} \cup \{+\infty\}$ is called coercive, if there exists some $r > 0$ such that

$$\inf\{\mathcal{F}(\mu) : \mu \in \mathcal{P}_2(\mathbb{R}^d), \int_{\mathbb{R}^d} \|x\|^2 \mathrm{d}\mu(x) \leq r\} > -\infty,$$

see (Ambrosio et al., 2005, eq. (11.2.1b)). In particular, any functional which is bounded from below is coercive.

If $\mathcal{F}$ is proper, lower semicontinuous, coercive and λ-convex along generalized geodesics, one can show that the $\operatorname{prox}_{\tau\mathcal{F}}(\mu)$ is non-empty and unique for τ small enough (see Ambrosio et al., 2005, page 295).

B. Proofs and Examples from Section 3

B.1. Examples fulfilling Assumption 3.1

Assumption 3.1 is fulfilled for many important divergences and loss functions $\mathcal{F}$. We list some examples below. While it is straightforward to check that they are proper, lower semicontinuous, coercive and bounded from below, the convexity is non-trivial. However, the conditions under which these functionals are λ-convex along generalized geodesics are well investigated in (Ambrosio et al., 2005, Section 9.3). In the following we denote the Lebesgue measure as μ_{Leb}. For a measure $\mu \in \mathcal{P}^{\text{ac}}(\mathbb{R}^d)$, we denote by $\mathrm{d}\mu/\mathrm{d}\mu_{\text{Leb}}$ the Lebesgue density of μ if it exists.

- Let $\nu \in \mathcal{P}_2^{\text{ac}}(\mathbb{R}^d)$ with Lebesgue density q and define the *forward Kullback-Leibler (KL) loss function*

$$\mathcal{F}(\mu) = \mathrm{KL}(\nu, \mu) := \begin{cases} \int_{\mathbb{R}^d} q(x) \log\left(\frac{q(x)}{p(x)}\right) \mathrm{d}x, & \text{if } \exists \mathrm{d}\mu/\mathrm{d}\mu_{\text{Leb}} = p \text{ and } \mathrm{d}\nu/\mathrm{d}\mu_{\text{Leb}} = q, \\ +\infty & \text{otherwise.} \end{cases}$$

  By (Ambrosio et al., 2005, Proposition 9.3.9), we obtain that $\mathcal{F}$ fulfills Assumption 3.1.

- We can also derive a functional, be reversing the arguments in the KL divergence. Then, we arrive at the *reverse KL loss function* given by

$$\mathcal{F}(\mu) = \mathrm{KL}(\mu, \nu) := \begin{cases} \int_{\mathbb{R}^d} p(x) \log\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x, & \text{if } \exists \mathrm{d}\mu/\mathrm{d}\mu_{\text{Leb}} = p \text{ and } \mathrm{d}\nu/\mathrm{d}\mu_{\text{Leb}} = q, \\ +\infty & \text{otherwise.} \end{cases}$$

  Given that $-\log(q)$ is λ-convex, we obtain that $\mathcal{F}$ fulfills Assumption 3.1, see (Ambrosio et al., 2005, Proposition 9.3.2).

- Finally, we can define $\mathcal{F}$ based on the Jensen-Shannon divergence. This results into the function

$$\mathcal{F}(\mu) = \mathrm{JS}(\mu, \nu) := \frac{1}{2}\left[\mathrm{KL}\left(\mu, \tfrac{1}{2}(\mu+\nu)\right) + \mathrm{KL}\left(\nu, \tfrac{1}{2}(\mu+\nu)\right)\right].$$

  Assume μ and ν admit Lebesgue densities p and q respectively. Then, combining the two previous statements, this fulfills Assumption 3.1 whenever $-\log(p)$ and $-\log(q)$ are λ-convex.

All of these functionals are integrals of a smooth Lagrangian functional, i.e., there exists some smooth $F\colon \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}$ such that

$$\mathcal{F}(\mu) = \begin{cases} \int_{\mathbb{R}^d} F(x, p(x), \nabla p(x)) \mathrm{d}x, & \text{if } \exists \mathrm{d}\mu/\mathrm{d}\mu_{\text{Leb}} = p, \\ \infty & \text{otherwise.} \end{cases}$$

In this case, the limit velocity field from Theorem 2.2 (which appears as a limit in Theorem 3.3) can be expressed analytically as the gradient of the so-called variational derivative of $\mathcal{F}$, which is given by

$$\frac{\delta \mathcal{F}}{\delta \gamma(t)}(x) = -\nabla \cdot \partial_3 F(x, p(x), \nabla p(x)) + \partial_2 F(x, p(x), \nabla p(x))$$

where $\partial_i F$ is the derivative of F with respect to the i-th argument and γ is the Wasserstein gradient flow, see (Ambrosio et al., 2005, Example 11.1.2). For the above divergence functionals, computing these terms lead to a (weighted) difference of the Stein scores of the input measure $\gamma(t)$ and the target measure ν which is a nice link to score-based methods. More precisely, denoting the density of $\gamma(t)$ by p_t, we obtain the following limiting velocity fields.

- For the forward KL loss function we have that $F(x, y, z) = q(x) \log\left(\frac{q(x)}{y}\right)$. Thus, we have that $\frac{\delta \mathcal{F}}{\delta \mu}(x) = -\frac{q(x)}{p(x)}$. Hence, the velocity field $v(\cdot, t) = \nabla \frac{\delta \mathcal{F}}{\delta \gamma(t)}$ is given by

$$v(x,t) = \frac{q(x)}{p_t(x)} \frac{\nabla p_t(x)}{p_t(x)} - \frac{q(x)}{p_t(x)} \frac{\nabla q(x)}{q(x)} = \frac{q(x)}{p_t(x)} \left(\nabla \log(p_t(x)) - \nabla \log(q(x))\right).$$

- For the reverse KL loss function the Lagrangian is given by $F(x, y, z) = y \log\left(\frac{y}{q(x)}\right)$. Thus, we have that $\frac{\delta \mathcal{F}}{\delta \mu}(x) = \log\left(\frac{p(x)}{q(x)}\right) + 1 = \log(p(x)) - \log(q(x)) + 1$. Hence, the velocity field $v(\cdot, t) = \nabla \frac{\delta \mathcal{F}}{\delta \gamma(t)}$ is given by

$$v(x, t) = \nabla(\log(p_t)(x)) - \nabla(\log(q)(x)).$$

- For the Jensen-Shannon divergence the Lagrangian is given by $F(x, y, z) = \frac{1}{2}\left(y \log\left(\frac{2y}{y+q(x)}\right) + q(x)\left(\frac{2q(x)}{y+q(x)}\right)\right)$. Thus, we have that $\frac{\delta \mathcal{F}}{\delta \mu}(x) = \frac{1}{2}\log\left(\frac{p(x)}{p(x)+q(x)}\right)$. Hence, the velocity field $v(\cdot, t) = \nabla \frac{\delta \mathcal{F}}{\delta \gamma(t)}$ is given by

$$\begin{aligned} v(x,t) &= \frac{1}{2}\frac{p_t(x)+q(x)}{p_t(x)}\frac{\nabla p_t(x)(p_t(x)+q(x)) - p_t(x)(\nabla p_t(x) + \nabla q(x))}{(p_t(x)+q(x))^2} \\ &= \frac{1}{2}\frac{\nabla p_t(x) q(x) - p_t(x)\nabla q(x)}{(p_t(x)+q(x))p_t(x)} \\ &= \frac{q(x)}{2(p_t(x)+q(x))}\left[\frac{\nabla p_t(x)}{p_t(x)} - \frac{\nabla q(x)}{q(x)}\right] \\ &= \frac{q(x)}{2(p_t(x)+q(x))}\left[\nabla(\log(p_t)(x)) - \nabla(\log(q)(x))\right]. \end{aligned}$$

Note that computing the score $\nabla \log(p_t)$ of the current approximation is usually intractable, such that these limits cannot be inserted into a neural ODE directly.

B.2. Proof of Corollary 3.2

Using similar arguments as in (Altekrüger et al., 2023; Mokrov et al., 2021; Onken et al., 2021; Xu et al., 2024), let $v\colon \mathbb{R}^d \times [0, \tau] \to \mathbb{R}^d$ such that the solution of

$$\dot{z}(x, t) = v(z(x, t), t), \quad z(x, 0) = x,$$

fulfills $z(\cdot, \tau)_\# \mu_\tau^k = \mu_\tau^{k+1}$. Since $(v_{\tau,k}, z_{\tau,k})$ is a minimizer of (5), we obtain that

$$\begin{aligned} &\frac{1}{2}\int_0^\tau \int_{\mathbb{R}^d} \|v_{\tau,k}(z_{\tau,k}(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x)\mathrm{d}t + \mathcal{F}(z_{\tau,k}(\cdot,\tau)_\#\mu_\tau^k) \\ \leq &\frac{1}{2}\int_0^\tau \int_{\mathbb{R}^d} \|v(z(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x)\mathrm{d}t + \mathcal{F}(z(\cdot,\tau)_\#\mu_\tau^k). \end{aligned}$$

Observing that $\mathcal{F}(z_{\tau,k}(\cdot,\tau)_\#\mu_\tau^k) = \mathcal{F}(z(\cdot,\tau)_\#\mu_\tau^k) = \mathcal{F}(\mu_\tau^{k+1})$, we obtain that

$$\tau \int_0^\tau \int_{\mathbb{R}^d} \|v_{\tau,k}(z_{\tau,k}(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x) \leq \tau \int_0^\tau \int_{\mathbb{R}^d} \|v(z(x,t),t)\|^2 \mathrm{d}\mu_\tau^k(x)\mathrm{d}t.$$

Since v was chose arbitrary, we obtain that $v_{\tau,k}$ is the optimal velocity field from the theorem of Benamou-Brenier, which now directly implies part (ii) and (iii). □

B.3. Proof of Theorem 3.3

In order to prove convergence of the velocity fields v_τ, we first introduce some notations. To this end, let $\mathcal{T}_\tau^k$ be the optimal transport maps between μ_τ^k and μ_τ^{k+1} and define by $v_\tau^k = (\mathcal{T}_\tau^k - I)/\tau$ the corresponding discrete velocity fields. Then, the velocity fields v_τ can be expressed as

$$v_\tau(x, k\tau + t\tau) = v_\tau^k(((1-t)I + t\mathcal{T}_\tau^k)^{-1}(x)). \tag{9}$$

Further, we denote the piece-wise constant concatenation of the discrete velocity fields by

$$\tilde{v}_\tau(x, k\tau + t\tau) = v_\tau^k, \quad t \in (0, 1).$$

Note that for any τ it holds that $v_\tau \in L_2(\gamma_\tau, \mathbb{R}^d \times [0,T])$ and $\tilde{v}_\tau \in L_2(\tilde{\gamma}_\tau, \mathbb{R}^d \times [0,T])$. To derive limits of these velocity fields, we recall the notion of convergence from (Ambrosio et al., 2005, Definition 5.4.3) allowing that the iterates are not defined on the same space. In this paper, we stick to square integrable measurable functions defined on finite dimensional domains, which slightly simplifies the definition.

Definition B.1. Let $\Omega \subseteq \mathbb{R}^d$ be a measurable domain, assume that $\mu_k \in \mathcal{P}_2(\Omega)$ converges weakly to $\mu \in \mathcal{P}_2(\Omega)$ and let $f_k \in L_2(\mu_k, \Omega)$ and $f \in L_2(\mu, \Omega)$. Then, we say that f_k converges weakly to f, if

$$\int_\Omega \langle \phi(x), f_k(x) \rangle \mathrm{d}\mu_k(x) \to \int_\Omega \langle \phi(x), f(x) \rangle \mathrm{d}\mu(x), \quad \text{as } k \to \infty$$

for all test functions $\phi \in C_c^\infty(\Omega)$. We say that f_k converges strongly to f if

$$\limsup_{k\to\infty} \|f_k\|_{L_2(\mu_k,\Omega)} \leq \|f\|_{L_2(\mu,\Omega)}. \tag{10}$$

Note that (Ambrosio et al., 2005, Theorem 5.4.4 (iii)) implies that formula (10) is fulfilled with equality for any strongly convergent sequence f_k to f. Moreover it is known from the literature that subsequences of the piece-wise constant velocity admit weak limits.

Theorem B.2 (Ambrosio et al., 2005, Theorem 11.1.6)**.** *Suppose that Assumption 3.1 is fulfilled for $\mathcal{F}$. Then, for any $\mu^0 \in \mathrm{dom}(\mathcal{F})$ and any sequence $(\tau_l)_l \subset (0,\infty)$, there exists a subsequence (again denoted by $(\tau_l)_l$) such that*

- *The piece-wise constant curve $\tilde{\gamma}_{\tau_l}(t)$ narrowly converges to some limit curve $\hat{\gamma}(t)$ for all $t \in [0,\infty)$.*
- *The velocity field $\tilde{v}_{\tau_l} \in L_2(\tilde{\gamma}_{\tau_l}(t), \mathbb{R}^d \times [0,T])$ weakly converges to some limit $\hat{v} \in L_2(\hat{\gamma}(t), \mathbb{R}^d \times [0,T])$ according to Definition B.1 for any $T > 0$.*
- *The limit $\hat{v}$ fulfills the continuity equation with respect to $\hat{\gamma}$, i.e.,*

$$\partial_t \hat{\gamma}(t) + \nabla \cdot (\hat{v}_{\tau_l}(\cdot, t)\hat{\gamma}(t)) = 0.$$

In order to show the desired result, there remain the following questions, which we answer in the rest of this section:

- Does Theorem B.2 also hold for the velocity fields v_{τ_l} defined in (9) belonging to the geodesic interpolations?
- Can we show strong convergence for the whole sequence v_{τ_l}?
- Does the limit $\hat{v}$ have the norm-minimizing property that $\|\hat{v}(\cdot,t)\|_{L_2(\hat{\gamma}(t))} = |\partial\mathcal{F}|(\hat{\gamma}(t))$?

To address the first of these questions, we show that weak limits of $\tilde{v}_\tau$ and v_τ coincide. The proof is a straightforward computation. A similar statement in a slightly different setting was proven in (Santambrogio, 2015, Section 8.3).

Lemma B.3. *Suppose that Assumption 3.1 is fulfilled and let $(\tau_l)_l \subseteq (0,\infty)$ be a sequence with $\tau_l \to 0$ as $l \to \infty$ such that $\tilde{v}_{\tau_l} \in L_2(\tilde{\gamma}_{\tau_l}, \mathbb{R}^d \times [0,T])$ converges weakly to some $\hat{v} \in L_2(\hat{\gamma}, \mathbb{R}^d \times [0,T])$. Then, also $v_{\tau_l} \in L_2(\gamma_{\tau_l}, \mathbb{R}^d \times [0,T])$ converges weakly to $\hat{v}$.*

Proof. Let $\phi \in C_c^\infty(\mathbb{R}^d \times [0,T])$. In particular, ϕ is Lipschitz continuous with some Lipschitz constant $L < \infty$. Then, it holds that

$$\begin{aligned}
&\left| \int_{\mathbb{R}^d \times [0,T]} \langle \phi, v_{\tau_l} \rangle \mathrm{d}\gamma_{\tau_l} - \int_{\mathbb{R}^d \times [0,T]} \langle \phi, v \rangle \mathrm{d}\hat{\gamma} \right| \\
\leq &\left| \int_{\mathbb{R}^d \times [0,T]} \langle \phi, v_{\tau_l} \rangle \mathrm{d}\gamma_{\tau_l} - \int_{\mathbb{R}^d \times [0,T]} \langle \phi, \tilde{v}_{\tau_l} \rangle \mathrm{d}\tilde{\gamma}_{\tau_l} \right| + \left| \int_{\mathbb{R}^d \times [0,T]} \langle \phi, \tilde{v}_{\tau_l} \rangle \mathrm{d}\tilde{\gamma}_{\tau_l} - \int_{\mathbb{R}^d \times [0,T]} \langle \phi, v \rangle \mathrm{d}\hat{\gamma} \right|.
\end{aligned}$$

Since $\tilde{v}_{\tau_l}$ converges weakly to $\hat{v}$, the second term converges to zero as $l \to \infty$. Thus in order to show that also v_{τ_l} converges weakly to $\hat{v}$, it remains to show that also the first term converges to zero. By using the notations $T_l = \min\{k\tau_l : k\tau_l \geq T, k \in \mathbb{Z}_{\geq 0}\}$ and $K_l = \lceil \frac{T}{\tau_l} \rceil = \frac{T_l}{\tau_l}$, we can estimate

$$
\begin{aligned}
&\left| \int_{\mathbb{R}^d \times [0,T]} \langle \phi, v_{\tau_l} \rangle \mathrm{d}\gamma_{\tau_l} - \int_{\mathbb{R}^d \times [0,T]} \langle \phi, \tilde{v}_{\tau_l} \rangle \mathrm{d}\tilde{\gamma}_{\tau_l} \right| \\
&= \left| \int_0^T \left(\int_{\mathbb{R}^d} \langle \phi(x,t), v_{\tau_l}(x,t) \rangle \mathrm{d}\gamma_{\tau_l}(t)(x) - \int_{\mathbb{R}^d} \langle \phi(x,t), \tilde{v}_{\tau_l}(x,t) \rangle \mathrm{d}\tilde{\gamma}_{\tau_l}(t)(x) \right) \mathrm{d}t \right| \\
&\leq \int_0^{T_l} \left| \int_{\mathbb{R}^d} \langle \phi(x,t), v_{\tau_l}(x,t) \rangle \mathrm{d}\gamma_{\tau_l}(t)(x) - \int_{\mathbb{R}^d} \langle \phi(x,t), \tilde{v}_{\tau_l}(x,t) \rangle \mathrm{d}\tilde{\gamma}_{\tau_l}(t)(x) \right| \mathrm{d}t \\
&\leq \sum_{k=0}^{K_l-1} \int_{k\tau_l}^{(k+1)\tau_l} \left| \int_{\mathbb{R}^d} \langle \phi(x,t), v_{\tau_l}(x,t) \rangle \mathrm{d}\gamma_{\tau_l}(t)(x) - \int_{\mathbb{R}^d} \langle \phi(x,t), \tilde{v}_{\tau_l}(x,t) \rangle \mathrm{d}\tilde{\gamma}_{\tau_l}(t)(x) \right| \mathrm{d}t \\
&= \tau_l \sum_{k=0}^{K_l-1} \int_0^1 \left| \int_{\mathbb{R}^d} \langle \phi(x, k\tau_l + t\tau_l), v_{\tau_l}(x, k\tau_l + t\tau_l) \rangle \mathrm{d}\gamma_{\tau_l}(k\tau_l + t\tau_l)(x) \right. \\
&\qquad \left. - \int_{\mathbb{R}^d} \langle \phi(x, k\tau_l + t\tau_l), \tilde{v}_{\tau_l}(x, k\tau_l + t\tau_l) \rangle \mathrm{d}\tilde{\gamma}_{\tau_l}(k\tau_l + t\tau_l)(x) \right| \mathrm{d}t.
\end{aligned}
$$

By denoting with $\mathcal{T}_{\tau_l}^k$ the optimal transport map from $\mu_{\tau_l}^k$ to $\mu_{\tau_l}^{k+1}$, we have for $t \in (0,1)$ that

$$
\gamma_{\tau_l}(k\tau_l + t\tau_l) = ((1-t)I + t\mathcal{T}_{\tau_l}^k)_{\#}\mu_{\tau_l}^k, \quad \tilde{\gamma}_{\tau_l}(k\tau_l + t\tau_l) = \mu_{\tau_l}^k,
$$

and the velocity fields satisfy

$$
v_{\tau_l}(x, k\tau_l + t\tau_l) = v_{\tau_l}^k(((1-t)I + t\mathcal{T}_{\tau_l}^k)^{-1}(x)), \quad \tilde{v}_{\tau_l}(x, k\tau_l + t\tau_l) = v_{\tau_l}^k(x).
$$

Then, the above term becomes

$$
\begin{aligned}
&\tau_l \sum_{k=0}^{K_l-1} \int_0^1 \left| \int_{\mathbb{R}^d} \langle \phi(x, k\tau_l + t\tau_l), v_{\tau_l}^k(((1-t)I + t\mathcal{T}_{\tau_l}^k)^{-1}(x)) \rangle \mathrm{d}((1-t)I + t\mathcal{T}_{\tau_l}^k)_{\#}\mu_{\tau_l}^k(x) \right. \\
&\qquad \left. - \int_{\mathbb{R}^d} \langle \phi(x, k\tau_l + t\tau_l), v_{\tau_l}^k(x) \rangle \mathrm{d}\mu_{\tau_l}^k(x) \right| \mathrm{d}t \\
&= \tau_l \sum_{k=0}^{K_l-1} \int_0^1 \left| \int_{\mathbb{R}^d} \langle \phi((1-t)x + t\mathcal{T}_{\tau_l}^k(x)), k\tau_l + t\tau_l), v_{\tau_l}^k(x) \rangle \mathrm{d}\mu_{\tau_l}^k(x) \right. \\
&\qquad \left. - \int_{\mathbb{R}^d} \langle \phi(x, k\tau_l + t\tau_l), v_{\tau_l}^k(x) \rangle \mathrm{d}\mu_{\tau_l}^k(x) \right| \mathrm{d}t \\
&\leq \tau_l \sum_{k=0}^{K_l-1} \int_0^1 \int_{\mathbb{R}^d} \left| \langle \phi((1-t)x + t\mathcal{T}_{\tau_l}^k(x)), k\tau_l + t\tau_l) - \phi(x, k\tau_l + t\tau_l), v_{\tau_l}^k(x) \rangle \right| \mathrm{d}\mu_{\tau_l}^k(x) \mathrm{d}t.
\end{aligned}
$$

Using Hölders' inequality and we obtain that this is smaller or equal than

$$
\tau_l \sum_{k=0}^{K_l-1} \int_0^1 \left(\int_{\mathbb{R}^d} \|\phi((1-t)x + t\mathcal{T}_{\tau_l}^k(x)), k\tau_l + t\tau_l) - \phi(x, k\tau_l + t\tau_l)\|^2 \mathrm{d}\mu_{\tau_l}^k(x) \int_{\mathbb{R}^d} \|v_{\tau_l}^k(x)\|^2 \mathrm{d}\mu_{\tau_l}^k(x) \right)^{1/2} \mathrm{d}t
$$

By the Lipschitz continuity of ϕ and inserting the definition of $v^k_{\tau_l} = \frac{\mathcal{T}^k_{\tau_l} - I}{\tau_l}$ this is smaller or equal than

$$\begin{aligned}
&\tau_l \sum_{k=0}^{K_l-1} \int_0^1 \left(t^2 \frac{L^2}{\tau_l^2} \int_{\mathbb{R}^d} \|\mathcal{T}^k_{\tau_l}(x) - x\|^2 \mathrm{d}\mu^k_{\tau_l}(x) \int_{\mathbb{R}^d} \|\mathcal{T}^k_{\tau_l}(x) - x\|^2 \mathrm{d}\mu^k_{\tau_l}(x) \right)^{1/2} \mathrm{d}t \\
&= L \sum_{k=0}^{K_l-1} \left(\int_0^1 t \mathrm{d}t \right) \left(\int_{\mathbb{R}^d} \|\mathcal{T}^k_{\tau_l}(x) - x\|^2 \mathrm{d}\mu^k_{\tau_l}(x) \right) \\
&= \frac{L}{2} \sum_{k=0}^{K_l-1} W_2^2(\mu^k_{\tau_l}, \mu^{k+1}_{\tau_l}) \\
&\leq \frac{L}{2} \sum_{k=0}^{\infty} W_2^2(\mu^k_{\tau_l}, \mu^{k+1}_{\tau_l})
\end{aligned}$$

Finally, we have by the definition of the minimizing movements scheme that

$$\frac{1}{2\tau_l} W_2^2(\mu^k_{\tau_l}, \mu^{k+1}_{\tau_l}) \leq \mathcal{F}(\mu^k_{\tau_l}) - \mathcal{F}(\mu^{k+1}_{\tau_l}).$$

Summing up for $k = 0, 1, ...$ we finally arrive at the bound

$$\left| \int_{\mathbb{R}^d \times [0,T]} \langle \phi, v_{\tau_l} \rangle \mathrm{d}\gamma_{\tau_l} - \int_{\mathbb{R}^d \times [0,T]} \langle \phi, v \rangle \mathrm{d}\hat{\gamma} \right| \leq \tau_l L \left(\mathcal{F}(\mu^0) - \inf_{\mu \in \mathcal{P}_2(\mathbb{R}^d)} \mathcal{F}(\mu) \right).$$

Since $\mathcal{F}$ is bounded from below the upper bound converges to zero as $l \to \infty$. This concludes the proof. □

Finally, we employ the previous results to show Theorem 3.3 from the main part of the paper. That is, we show that for any $(\tau_l)_l \subseteq (0, \infty)$ with $\tau_l \to 0$ the whole v_{τ_l} converges strongly to v.

Theorem B.4 (Theorem 3.3)**.** *Suppose that Assumption 3.1 is fulfilled and let $(\tau_l)_l \subseteq (0, \infty)$ with $\tau_l \to 0$. Then, $(v_{\tau_l})_l$ converges strongly to the velocity field $\hat{v} \in L_2(\gamma, \mathbb{R}^d \times [0, T])$ of the Wasserstein gradient flow $\gamma \colon (0, \infty) \to \mathcal{P}_2(\mathbb{R}^d)$ of $\mathcal{F}$ starting in μ^0.*

Proof. We show that any subsequence of τ_l admits a subsequence converging strongly to $\hat{v}$. Using the sub-subsequence criterion this yields the claim.

By Theorem B.2 and Lemma B.3 we know that any subsequence of τ_l admits a weakly convergent subsequence. In an abuse of notations, we denote it again by τ_l and its limit by $\tilde{v}$. Then, we prove that the convergence is indeed strong and that $\tilde{v} = \hat{v}$.

Step 1: Bounding $\limsup_{l \to \infty} \int_0^T \|v_{\tau_l}(\cdot, t)\|^2_{L_2(\gamma_{\tau_l}(t), \mathbb{R}^d)} \mathrm{d}t$ **from above.** Since λ-convexity with $\lambda \geq 0$ implies λ-convexity with $\lambda = -1$, we can assume without loss of generality that $\lambda < 0$. Then, by (Ambrosio et al., 2005, Lemma 9.2.7, Theorem 4.0.9), we know that for any $\tau > 0$ it holds

$$W_2(\tilde{\gamma}_\tau(t), \gamma(t)) \leq \tau C(\tau, t), \qquad C(\tau, t) = \frac{(1 + 2|\lambda| t_\tau) |\partial \mathcal{F}|(\mu_0)}{\sqrt{2}(1 + \lambda \tau)} \exp\left(-\frac{\log(1 + \lambda \tau)}{\tau} t \right),$$

where $t_\tau = \min\{k\tau : k\tau \geq t, k \in \mathbb{N}_0\}$. For simplicity, we use the notations $\tau_{\max} = \max\{\tau_l : l \in \mathbb{N}_0\}$, $K_l = \lceil \frac{T}{\tau_l} \rceil$ and $T_{\max} = T + \tau_{\max}$. Since $\lambda < 0$, we have that $C(\tau, t) \leq C(\tau, T_{\max}) =: C(\tau)$ for all $t \in [0, T_{\max}]$. Moreover, we have that $C(\tau) \to \frac{(1+2|\lambda| T_{\max} |\partial \mathcal{F}|(\mu_0)}{\sqrt{2}} \exp(-\lambda T_{\max})$ as $\tau \to 0$, such that the sequence $(C(\tau_l))_l$ is bounded. In particular, there exists a $C > 0$ such that

$$W_2(\tilde{\gamma}_{\tau_l}(t), \gamma(t)) \leq \tau_l C, \quad \text{for all} \quad t \in [0, T_{\max}].$$

Inserting $t = k\tau_l$ for $k = 0, ..., K_l$ gives

$$W_2(\mu^k_{\tau_l}, \gamma(k\tau_l)) \leq \tau_l C, \quad \text{for all} \quad k = 0, ..., K_l. \tag{11}$$

Now, we can conclude by the theorem of Benamou-Brenier that

$$\begin{aligned}\int_0^T \|v_{\tau_l}(\cdot,t)\|^2_{L_2(\gamma_{\tau_l}(t),\mathbb{R}^d)}\mathrm{d}t &\leq \int_0^{K_l\tau_l} \|v_{\tau_l}(\cdot,t)\|^2_{L_2(\gamma_{\tau_l}(t),\mathbb{R}^d)}\mathrm{d}t \\ &= \sum_{k=0}^{K_l-1}\int_{k\tau_l}^{(k+1)\tau_l} \|v_{\tau_l}(\cdot,t)\|^2_{L_2(\gamma_{\tau_l}(t),\mathbb{R}^d)}\mathrm{d}t \\ &= \sum_{k=1}^{K_l-1} W_2^2(\mu^k_{\tau_l},\mu^{k+1}_{\tau_l}).\end{aligned}$$

Now applying the triangular inequality for any $k = 1,\dots,K_l-1$

$$W_2^2(\mu^k_{\tau_l},\mu^{k+1}_{\tau_l}) \leq \left(W_2(\mu^k_{\tau_l},\gamma(\tau_l k)) + W_2(\gamma(\tau_l k),\gamma(\tau_l(k+1)) + W_2(\gamma(\tau_l(k+1)),\mu^{k+1}_{\tau_l})\right)^2$$

and the estimate from (11) yields

$$\int_0^T \|v_{\tau_l}(\cdot,t)\|^2_{L_2(\gamma_{\tau_l}(t),\mathbb{R}^d)}\mathrm{d}t \leq \sum_{k=0}^{K_l-1}\left(2\tau C + W_2(\gamma(\tau_l k),\gamma(\tau_l(k+1))\right)^2$$

By Jensens' inequality the right hand side can be bounded from above by

$$4K_l\tau^2C^2 + 2\tau K_l C\left(\sum_{k=0}^{K_l-1}\frac{1}{K_l}W_2^2(\gamma(\tau_l k),\gamma(\tau_l(k+1)))\right)^{1/2} + \sum_{k=0}^{K_l-1}W_2^2(\gamma(\tau_l k),\gamma(\tau_l(k+1)))$$

Finally, again Benamou-Brenier gives that $W_2^2(\gamma(\tau_l k),\gamma(\tau_l(k+1))) \leq \int_{k\tau_l}^{(k+1)\tau_l}\|\hat{v}(\cdot,t)\|^2_{L_2(\gamma(t))}\mathrm{d}t$ such that the above formula is smaller or equal than

$$4K_l\tau_l^2C^2 + 2\tau_l\sqrt{K_l}C\left(\int_0^{K_l\tau_l}\|\hat{v}(\cdot,t)\|^2_{L_2(\gamma(t))}\mathrm{d}t\right)^{1/2} + \int_0^{K_l\tau_l}\|\hat{v}(\cdot,t)\|^2_{L_2(\gamma(t))}\mathrm{d}t.$$

Since $K_l\tau_l \to T$ and $\tau_l \to 0$ as $l\to\infty$ this converges to $\int_0^T\|\hat{v}(\cdot,t)\|^2_{L_2(\gamma(t))}\mathrm{d}t$ such that we can conclude

$$\limsup_{l\to\infty}\int_0^T \|v_{\tau_l}(\cdot,t)\|^2_{L_2(\gamma_{\tau_l}(t),\mathbb{R}^d)}\mathrm{d}t \leq \int_0^T\|\hat{v}(\cdot,t)\|^2_{L_2(\gamma(t))}\mathrm{d}t.$$

Step 2: Strong convergence. By (Ambrosio et al., 2005, Theorem 8.3.1, Proposition 8.4.5) we know that for any v fulfilling the continuity equation

$$\partial_t\gamma(t) + \nabla\cdot(v(\cdot,t)\gamma(t)) = 0,$$

it holds that $\|v(\cdot,t)\|_{L_2(\gamma(t),\mathbb{R}^d} \geq \|\hat{v}(\cdot,t)\|_{L_2(\gamma(t),\mathbb{R}^d}$. Since $\tilde{v}$ fulfills the continuity equation by Theorem B.2, this implies that

$$\int_0^T\|\tilde{v}(\cdot,t)\|^2_{L_2(\gamma(t),\mathbb{R}^d}\mathrm{d}t \geq \int_0^T\|\hat{v}(\cdot,t)\|^2_{L_2(\gamma(t),\mathbb{R}^d}\mathrm{d}t = \limsup_{l\to\infty}\int_0^T\|v_{\tau_l}(\cdot,t)\|^2_{L_2(\gamma_{\tau_l}(t),\mathbb{R}^d)}\mathrm{d}t.$$

In particular, $v_{\tau_l} \in L_2(\gamma_{\tau_l},\mathbb{R}^d\times[0,T])$ converges strongly to $\tilde{v}$ such that by (Ambrosio et al., 2005, Theorem 5.4.4 (iii)) it holds equality in the above equation, i.e.,

$$\int_0^T\|\tilde{v}(\cdot,t)\|^2_{L_2(\gamma(t),\mathbb{R}^d}\mathrm{d}t = \int_0^T\|\hat{v}(\cdot,t)\|^2_{L_2(\gamma(t),\mathbb{R}^d}\mathrm{d}t$$

Using again (Ambrosio et al., 2005, Theorem 8.3.1, Proposition 8.4.5) this implies that $\tilde{v} = \hat{v}$. □

C. Proofs from Section 4

C.1. Proof of Theorem 4.2

i) Let $A \subset \mathbb{R}^d$ Lebesgue-measurable set. We denote by accept and reject the measurable set of accepted and rejected draws of X respectively. By the law of total probability, we have

$$\mathbb{P}(\tilde{X} \in A) = \mathbb{P}(\tilde{X} \in A \text{ and } X \text{ was accepted}) + \mathbb{P}(\tilde{X} \in A \text{ and } X \text{ was rejected}).$$

Since it holds $\tilde{X} = X$ if X is accepted and $\tilde{X} = X'$ if X is rejected this can be reformulated as

$$\mathbb{P}(\tilde{X} \in A) = \mathbb{P}(X \in A, \text{accept}) + \mathbb{P}(X' \in A, \text{reject})$$

Now it holds

$$\begin{aligned}
\mathbb{P}(X \in A, \text{accept}) &= \int_A \alpha(x) p(x)\, dx \\
\mathbb{P}(X' \in A, \text{reject}) &= \mathbb{P}(X' \in A) \cdot \mathbb{P}(\text{reject}) = \left(\int_A p(x)\, dx \right) (1 - \mathbb{E}[\alpha(X)]),
\end{aligned}$$

where we used the fact that $X, X' \sim \mu$ are independent. As a consequence, we obtain

$$\begin{aligned}
\mathbb{P}(\tilde{X} \in A) &= \int_A \alpha(x) p(x) dx + \left(\int_A p(x) dx \right) (1 - \mathbb{E}[\alpha(X)]) \\
&= \int_A p(x) \left(\alpha(x) + 1 - \mathbb{E}[\alpha(X)] \right) dx.
\end{aligned}$$

Thus, $\tilde{p}(x) = p(x)(\alpha(x) + 1 - \mathbb{E}[\alpha(X)])$ is the density of $\tilde{X}$.

(ii) Note, since $X \sim p$, we have

$$\int_{\mathbb{R}^d} \frac{p(x)\alpha(x)}{\mathbb{E}[\alpha(X)]} \mathrm{d}x = \int_{\mathbb{R}^d} \frac{p(x)\alpha(x)}{\int_{\mathbb{R}^d} p(y)\alpha(y) \mathrm{d}y} \mathrm{d}x = 1,$$

in particular $\frac{p(x)\alpha(x)}{\mathbb{E}[\alpha(X)]}$ defines a density. Now let $\eta \in \mathcal{P}_2(\mathbb{R}^d)$ be the corresponding probability measure. Then, it holds by part (i) that $\tilde{\mu} = \mathbb{E}[\alpha(X)]\eta + (1 - \mathbb{E}[\alpha(X)])\mu$. We will show that $\mathrm{KL}(\eta, \nu) \leq \mathrm{KL}(\mu, \nu)$. Due to the convexity of the KL divergence in the linear space of measures this implies the claim.

We denote $Z = \mathbb{E}[\alpha(X)] = \int_{\mathbb{R}^d} \min(p(x), \frac{q(x)}{\tilde{c}}) \mathrm{d}x$. Then it holds

$$\begin{aligned}
\mathrm{KL}(\eta, \nu) &= \int_{\mathbb{R}^d} \frac{p(x)\alpha(x)}{Z} \log \left(\frac{p(x)\alpha(x)}{Zq(x)} \right) \mathrm{d}x \\
&= \int_{\mathbb{R}^d} \frac{\min(p(x), \frac{q(x)}{\tilde{c}})}{Z} \log \left(\frac{\tilde{c}p(x)\alpha(x)}{q(x)} \right) \mathrm{d}x - \log(Z\tilde{c}) \int_{\mathbb{R}^d} \frac{\min(p(x), \frac{q(x)}{\tilde{c}})}{Z} \mathrm{d}x \\
&= \int_{\mathbb{R}^d} \frac{\min(p(x), \frac{q(x)}{\tilde{c}})}{Z} \log \left(\min \left(\frac{\tilde{c}p(x)}{q(x)}, 1 \right) \right) \mathrm{d}x - \log(Z) - \log(\tilde{c}) \\
&= \int_{\mathbb{R}^d} \min \left(\frac{\min(p(x), \frac{q(x)}{\tilde{c}})}{Z} \log \left(\frac{\tilde{c}p(x)}{q(x)} \right), 0 \right) \mathrm{d}x - \log(Z) - \log(\tilde{c}).
\end{aligned}$$

Since $\log \left(\frac{\tilde{c}p(x)}{q(x)} \right) \leq 0$ if and only if $p(x) \leq q(x)/\tilde{c}$ this can be reformulated as

$$\begin{aligned}
&\int_{\mathbb{R}^d} \min \left(\frac{p(x)}{Z} \log \left(\frac{\tilde{c}p(x)}{q(x)} \right), 0 \right) \mathrm{d}x - \log(Z) - \log(\tilde{c}) \\
\leq &\int_{\mathbb{R}^d} \min \left(p(x) \log \left(\frac{\tilde{c}p(x)}{q(x)} \right), 0 \right) \mathrm{d}x - \log(Z) - \log(\tilde{c}),
\end{aligned} \tag{12}$$

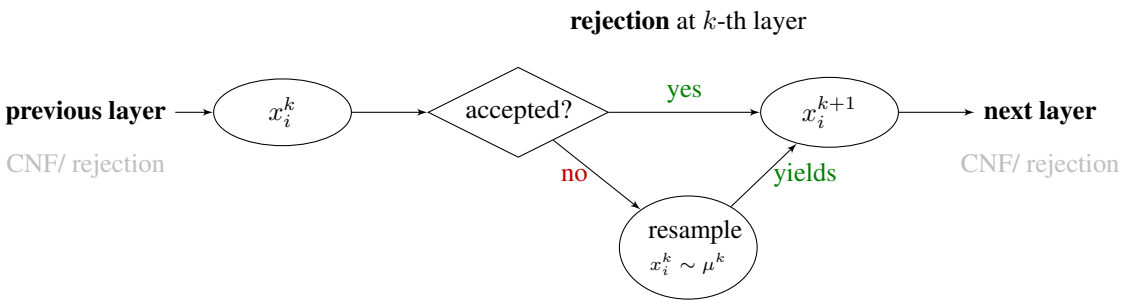

Figure 2: Illustration of a single sample within a rejection layer. The independent sample x_i^k is either accepted directly or else resampled once, and then used to define x_i^{k+1} as input for the next layer.

where the inequality comes from the fact that $Z = \mathbb{E}[\alpha(X)] \in [0, 1]$. Moreover, it holds by Jensen's inequality that

$$\begin{aligned}
-\log(Z) = -\log(\mathbb{E}[\alpha(X)]) &= -\log\left(\int_{\mathbb{R}^d} p(x) \min\left(\frac{q(x)}{\tilde{c}p(x)}, 1\right) \mathrm{d}x\right) \\
&\leq -\int_{\mathbb{R}^d} p(x) \log\left(\min\left(\frac{q(x)}{\tilde{c}p(x)}, 1\right)\right) \mathrm{d}x \\
&= -\int_{\mathbb{R}^d} p(x) \min\left(\log\left(\frac{q(x)}{\tilde{c}p(x)}\right), 0\right) \mathrm{d}x \\
&= \int_{\mathbb{R}^d} p(x) \max\left(\log\left(\frac{\tilde{c}p(x)}{q(x)}\right), 0\right) \mathrm{d}x \\
&= \int_{\mathbb{R}^d} \max\left(p(x) \log\left(\frac{\tilde{c}p(x)}{q(x)}\right), 0\right) \mathrm{d}x.
\end{aligned}$$

Thus, we obtain that (12) can be bounded from above by

$$\int_{\mathbb{R}^d} \min\left(p(x) \log\left(\frac{\tilde{c}p(x)}{q(x)}\right), 0\right) \mathrm{d}x + \int_{\mathbb{R}^d} \max\left(p(x) \log\left(\frac{\tilde{c}p(x)}{q(x)}\right), 0\right) \mathrm{d}x - \log(\tilde{c})$$

which equals

$$\int_{\mathbb{R}^d} p(x) \log\left(\frac{\tilde{c}p(x)}{q(x)}\right) \mathrm{d}x - \log(\tilde{c}) = \int_{\mathbb{R}^d} p(x) \log\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x = \mathrm{KL}(\mu, \nu).$$

In summary, we have $\mathrm{KL}(\eta, \nu) \leq \mathrm{KL}(\mu, \nu)$, which implies the assertion.

□

C.2. Proof of Corollary 4.3

Let $Y = \left|\mathbb{E}[\alpha(X)] - \frac{1}{N}\sum_{i=1}^N \alpha(X_N)\right|$ denote the random variable representing the error. Since $\alpha(X_N) \in [0, 1]$, Hoeffding's inequality (Hoeffding, 1963) yields that $P(Y > t) \leq 2\exp(-\frac{Nt^2}{2})$. Consequently, we have that

$$\mathbb{E}[Y] = \int_0^\infty P(Y > t)\mathrm{d}t \leq 2\int_0^\infty \exp\left(-\frac{Nt^2}{2}\right) \mathrm{d}t = \frac{\sqrt{2\pi}}{\sqrt{N}}.$$

□

D. Algorithms

D.1. Training and Evaluation Algorithms

We summarize the training and evaluation procedures for the neural JKO steps in Algorithm 3 and 4 and the parameter selection for the importance-based rejection steps in Algorithm 5. The evaluation of the importance-based rejection steps is summarized in Algorithm 1 in the main text.

Algorithm 3 Training of neural JKO steps

Input: Samples $x_1^k, ..., x_N^k$ of μ^k.
Minimize the loss function $\theta \mapsto \mathcal{L}(\theta)$ from (6) using the Adam optimizer.
Output: Parameters θ.

Algorithm 4 Sampling and density propagation for neural JKO steps

Input: $\begin{cases} \text{- Samples } x_1^k, ..., x_N^k \text{ of } \mu^k, \\ \text{- Density values } p^k(x_1^k), ..., p^k(x_N^k). \end{cases}$

for $i = 1, ..., N$ **do**
1. Solve the ODE (7) for $x = x_i^k$.
2. Set $x_i^{k+1} = z_\theta(x_i^k, \tau)$.
3. Set $p^{k+1}(x_i^{k+1}) = \frac{p^k(x_i^k)}{\exp(\ell_\theta(x_i^k, \tau))}$.

end for

Output: $\begin{cases} \text{- Samples } x_1^k, ..., x_N^k \text{ of } \mu^{k+1}, \\ \text{- Density values } p^{k+1}(x_1^{k+1}), ..., p^{k+1}(x_N^{k+1}). \end{cases}$

Algorithm 5 Parameter selection for importance-based rejection steps

Input: $\begin{cases} \text{- Samples } x_1^k, ..., x_N^k \text{ of } \mu^k \text{ with corresponding densities} \\ \text{- desired rejection rate } r\text{, unnormalized target density } g \end{cases}$

Choose c by bisection search such that

$$1 - r = \frac{1}{N} \sum_{i=1}^{N} \alpha_k(x_i^k), \qquad \alpha_k(x) = \min\left\{1, \frac{g(x)}{cp^k(x)}\right\}.$$

Output: Rejection parameter c and estimate of $\mathbb{E}[\alpha(X_\tau^k)] \approx 1 - r$.

Algorithm 6 Density Evaluation of Importance Corrected Neural JKO Models

Input: $x \in \mathbb{R}^d$, model $(X_0, ..., X_K)$
if $K = 0$ **then**
Return latent density $p^0(x)$.
else if the last step is a rejection step **then**
1. Evaluate $p^{K-1}(x)$ by applying this algorithm for $(X_0, ..., X_{K-1})$.
2. **Return** $p^K(x) = p^{K-1}(\alpha_k(x) + 1 - \mathbb{E}[\alpha_k(X_{k-1})])$

else

last step is a neural JKO step

1. Solve the ODE system (with v_θ from the neural JKO step)

$$\begin{pmatrix} \dot{z}_\theta(x,t) \\ \dot{\ell}_\theta(x,t) \end{pmatrix} = \begin{pmatrix} v_\theta(z_\theta(x,t),t) \\ \operatorname{trace}(\nabla v_\theta(z_\theta(x,t),t)) \end{pmatrix}, \qquad \begin{pmatrix} z_\theta(x,\tau) \\ \ell_\theta(x,\tau) \end{pmatrix} = \begin{pmatrix} x \\ 0 \end{pmatrix}.$$

2. Set $\tilde{x} = z_\theta(x, 0)$.
3. Evaluate $p^{K-1}(\tilde{x})$ by applying this algorithm for $(X_0, ..., X_{K-1})$.
4. **Return** $p^K(x) = p^{K-1}(\tilde{x}) \exp(\ell_\theta(x, 0))$.

end if
Output: Density $p^K(x)$

D.2. Density Evaluation of Importance Corrected Neural JKO Models

Let $(X_0, ..., X_K)$ be an importance corrected neural JKO model. We aim to evaluate the density p^K of X_K at some given point $x \in \mathbb{R}^d$. Using Theorem 4.2, this can be done recursively by Algorithm 6. Note that the algorithm always terminates since K is reduced by one in each call.

E. Additional Numerical Results and Implementation Details

E.1. Test Distributions

We evaluate our method on the following test distributions.

- **Mustache:** The two-dimensional log-density is given as $\log \mathcal{N}(0, \Sigma) \circ T$ with $\Sigma = [1, \sigma; \sigma, 1]$ and $T(x_1, x_2) = (x_1, (x_2 - (x_1^2 - 1)^2))$. Note that $\det(\nabla T(x)) = 1$ for all x. In particular, we obtain directly by the transformation formula that the normalization constant is one. Depending on $\sigma \in [0, 1)$ close to 1, this probability distribution has very long and narrow tails making it hard for classical MCMC methods to sample them. In our experiments we use $\sigma = 0.9$.

- **Shifted 8 Modes:** A two-dimensional Gaussian mixture model with 8 equal weighted modes and covariance matrix $1 \times 10^{-2} I$. The modes are placed in a circle with radius 1 and center $(-1, 0)$. Due to the shifted center classical MCMC methods have difficulties to distribute the mass correctly onto the modes.

- **Shifted 8 Peaky:** This is the same distribution as the shifted 8 Modes with the difference that we reduce the width of the modes to the covariance matrix $5 \times 10^{-3} I$. Since the modes are disconnected, it becomes harder to sample from them.

- **Funnel:** We consider the (normalized) probability density function given by $q(x) = \mathcal{N}(x_1|0, \sigma_f^2)\mathcal{N}(x_{2:10}|0, \exp(x_1)I)$, where $\sigma_f^2 = 9$. This example was introduced by (Neal, 2003). Similarly to the mustache example, this distribution has a narrow funnel for small values x_1 which can be hard to sample.

- **GMM-d:** A d-dimensional Gaussian mixture model with 10 equal weighted modes with covariance matrix $1 \times 10^{-2} I$ and means drawn randomly from a uniform distribution on $[-1, 1]^d$. This leads to a peaky high-dimensional and multimodal probability distribution which is hard to sample from.

- **GMM40-50D:** Another Gaussian mixture model, which was used as an example in (Blessing et al., 2024; Chen et al., 2025) based on (Midgley et al., 2023). It is a 50 dimensional GMM with 40 equally weighted modes with the identity as covariance matrix, where the means are drawn uniformly from $[-40, 40]^d$. We use the same seed for drawing the modes as in (Chen et al., 2025).

- **LGCP:** This is a high dimensional standard example taken from (Arbel et al., 2021; Matthews et al., 2022; Vargas et al., 2023a). It describes a Log-Gaussian Cox process on a 40×40 grid as arising from spatial statistics (Møller et al., 1998). This leads to a 1600-dimensional probability distribution with the unnormalized density function $q(x) \propto \mathcal{N}(x|\mu, K) \prod_{i \in \{1,...,40\}^2} \exp(x_i y_i - a \exp(x_i))$, where μ and K are a fixed mean and covariance kernel, y_i is some data and a is a hyperparameter. For details on this example and the specific parameter choice we refer to (Matthews et al., 2022).

E.2. Error Measures

We use the following error measures.

- The **energy distance** was proposed by (Székely, 2002) and is defined by

$$D(\mu, \nu) = -\frac{1}{2} \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \|x - y\| \mathrm{d}(\mu - \nu)(x) \mathrm{d}(\mu - \nu)(y).$$

  It is the maximum mean discrepancy with the negative distance kernel $K(x, y) = -\|x - y\|$ (see Sejdinovic et al., 2013) and can be estimated from below and above by the Wasserstein-1 distance (see Hertrich et al., 2024). It is a metric on the space of probability measures. Consequently, a smaller energy distance indicates a higher similarity of the

Table 3: We report the expected squared empirical Wasserstein-2 distance (W_2^2) and its standard deviation between generated and ground truth samples of size $N = 5000$ for the different methods and for all examples where the ground truth model is known. A smaller value of W_2^2 indicates a better result. Note that the curse of dimensionality present in the sample complexity, might limit the reliability of the results for the high-dimensional examples. In particular for the funnel distribution, we observe that the expected empirical Wasserstein-2 distance between two independent sets of ground truth samples is higher than the observed W_2^2 values.

	Sampler						
Distribution	MALA	HMC	DDS	CRAFT	Neural JKO	Neural JKO IC (**ours**)	Sampling Error
Mustache	$4.7 \times 10^{+1} \pm 1.3 \times 10^{+1}$	$2.8 \times 10^{+1} \pm 4.2 \times 10^{0}$	$5.2 \times 10^{+1} \pm 2.0 \times 10^{0}$	$5.4 \times 10^{+1} \pm 1.2 \times 10^{+1}$	$3.0 \times 10^{+1} \pm 1.3 \times 10^{+1}$	$\mathbf{1.7 \times 10^{+1} \pm 6.0 \times 10^{0}}$	$1.2 \times 10^{+1}$
shifted 8 Modes	$5.5 \times 10^{-2} \pm 6.9 \times 10^{-3}$	$4.7 \times 10^{-3} \pm 3.6 \times 10^{-3}$	$8.7 \times 10^{-2} \pm 3.1 \times 10^{-2}$	$2.4 \times 10^{-1} \pm 2.4 \times 10^{-3}$	$5.6 \times 10^{-1} \pm 1.6 \times 10^{-2}$	$\mathbf{6.5 \times 10^{-3} \pm 2.1 \times 10^{-3}}$	7.4×10^{-3}
shifted 8 Peaky	$5.8 \times 10^{-2} \pm 2.4 \times 10^{-2}$	$5.6 \times 10^{-1} \pm 1.4 \times 10^{-2}$	$1.0 \times 10^{-1} \pm 2.5 \times 10^{-2}$	$2.5 \times 10^{-1} \pm 1.1 \times 10^{-1}$	$5.9 \times 10^{-1} \pm 1.7 \times 10^{-2}$	$\mathbf{7.2 \times 10^{-3} \pm 1.3 \times 10^{-3}}$	5.6×10^{-3}
Funnel	$5.5 \times 10^{+2} \pm 1.2 \times 10^{+2}$	$7.4 \times 10^{+2} \pm 5.6 \times 10^{+2}$	$\mathbf{5.3 \times 10^{+2} \pm 7.5 \times 10^{+1}}$	$7.9 \times 10^{+2} \pm 4.4 \times 10^{+2}$	$9.4 \times 10^{+2} \pm 9.3 \times 10^{+2}$	$8.5 \times 10^{+2} \pm 3.9 \times 10^{+2}$	$1.0 \times 10^{+3}$
GMM-10	$3.8 \times 10^{0} \pm 4.2 \times 10^{-1}$	$3.8 \times 10^{0} \pm 3.9 \times 10^{-1}$	$3.8 \times 10^{-1} \pm 1.1 \times 10^{-1}$	$2.4 \times 10^{0} \pm 1.0 \times 10^{0}$	$6.3 \times 10^{-1} \pm 1.4 \times 10^{-1}$	$\mathbf{1.4 \times 10^{-1} \pm 2.3 \times 10^{-2}}$	1.4×10^{-1}
GMM-20	$8.9 \times 10^{0} \pm 4.0 \times 10^{-1}$	$9.0 \times 10^{0} \pm 3.9 \times 10^{-1}$	$8.8 \times 10^{-1} \pm 1.0 \times 10^{-1}$	$7.9 \times 10^{0} \pm 1.5 \times 10^{0}$	$1.2 \times 10^{0} \pm 2.0 \times 10^{-1}$	$\mathbf{3.7 \times 10^{-1} \pm 5.0 \times 10^{-2}}$	3.7×10^{-1}
GMM-50	$2.7 \times 10^{+1} \pm 1.0 \times 10^{0}$	$2.7 \times 10^{+1} \pm 9.8 \times 10^{-1}$	$3.6 \times 10^{0} \pm 8.9 \times 10^{-1}$	$2.6 \times 10^{+1} \pm 2.6 \times 10^{0}$	$4.4 \times 10^{0} \pm 7.9 \times 10^{-1}$	$\mathbf{1.2 \times 10^{0} \pm 1.3 \times 10^{-1}}$	1.2×10^{0}
GMM-100	$5.7 \times 10^{+1} \pm 1.2 \times 10^{0}$	$5.7 \times 10^{+1} \pm 1.3 \times 10^{0}$	$9.9 \times 10^{0} \pm 2.9 \times 10^{0}$	$5.6 \times 10^{+1} \pm 1.1 \times 10^{0}$	$1.1 \times 10^{+1} \pm 2.7 \times 10^{0}$	$\mathbf{2.9 \times 10^{0} \pm 4.7 \times 10^{-1}}$	2.8×10^{0}
GMM-200	$1.2 \times 10^{+2} \pm 2.8 \times 10^{0}$	$1.2 \times 10^{+2} \pm 2.8 \times 10^{0}$	$2.4 \times 10^{+1} \pm 4.0 \times 10^{+1}$	$1.1 \times 10^{+2} \pm 3.0 \times 10^{0}$	$2.4 \times 10^{+1} \pm 3.9 \times 10^{0}$	$\mathbf{7.8 \times 10^{0} \pm 6.9 \times 10^{-1}}$	5.9×10^{0}

input distributions. By discretizing the integrals it can be easily evaluated based on $N \in \mathbb{N}$ samples $\boldsymbol{x} = (x_i)_i \sim \mu^{\otimes N}$, $\boldsymbol{y} = (y_i)_i \sim \nu^{\otimes N}$ as

$$D(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i,j=1}^{N} \|x_i - y_j\| - \frac{1}{2} \sum_{i,j=1}^{N} \|x_i - x_j\| - \frac{1}{2} \sum_{i,j=1}^{N} \|y_i - y_j\|.$$

We use $N = 50000$ samples in Table 1.

- We also evaluate the squared Wasserstein-2 distance which is defined in Section 2. To this end, we use the Python Optimal Transport package (POT, Flamary et al., 2021). Note that computing the Wasserstein distance has complexity $O(n^3)$ where n is the number of points. Hence, we evaluate the Wasserstein distance based on fewers samples compared to the case of other metrics. We use $N = 5000$ samples in Table 3. In addition, we want to highlight that the expected Wasserstein distance evaluated on empirical measures instead of its continuous counterpart suffers from the curse of dimensionality. In particular, its sample complexity scales as $O(n^{-1/d})$ (Peyré & Cuturi, 2019, Chapter 8.4.1). Consequently, the sample-based Wasserstein distance in high dimensions can differ significantly from the true Wasserstein distance of the continuous distributions. Indeed, we can see in Table 3 that the sampling error has often the same order of magnitude as the reported errors. Overall we can draw similar conclusions from this evaluation as for the energy distance in Table 1.

- We **estimate the log normalizing constant** (short $\log(Z)$ estimation) which is used as a benchmark standard in various references (e.g., in Arbel et al., 2021; Matthews et al., 2022; Phillips et al., 2024; Vargas et al., 2023a). More precisely, for the generated distribution μ with normalized density p and target measure ν with density $q(x) = g(x)/Z_g$ we evaluate the term

$$\mathbb{E}_{x\sim\mu}\left[\log\left(\frac{g(x)}{p(x)}\right)\right] = \log(Z_g) - \mathbb{E}_{x\sim\mu}\left[\log\left(\frac{p(x)}{q(x)}\right)\right] = \log(Z_g) - \mathrm{KL}(\mu, \nu).$$

  Due to the properties of the KL divergence, a higher $\log(Z)$ estimate implies a lower KL divergence between μ and ν and therefore a higher similarity of generated and target distribution. In our experiments we compute the $\log(Z)$ estimate based on $N = 50000$ samples. The results are given in Table 2.

- To quantify how well the mass is distributed on different modes for the mixture model examples (shifted 8 Modes, shifted 8 Peaky, GMM-d), we compute the mode weights. That is, we generate $N = 50000$ samples and assign each generated samples to the closest mode of the GMM. Afterwards, we compute for each mode the fraction of samples which is assigned to each mode. To evaluate this distribution quantitatively, we compute the mean square error (MSE) between the mode weights of the generated samples and the ground truth weights from the GMM. We call this error metric the **mode MSE**, give the results are in Table 4.

Remark E.1 (Bias in $\log(Z)$ Computation). In the cases, where the importance corrected neural JKO sampling fits the target distribution almost perfectly, we sometimes report in Table 2 $\log(Z)$ estimates which are slightly larger than the ground truth. This can be explained by the fact that the density evaluation of the continuous normalizing flows uses the Hutchinson

Table 4: We report the mode MSEs for the different methods for all examples which can be represented as mixture model. A smaller mode MSE indicates a better result.

	Sampler					
Distribution	MALA	HMC	DDS	CRAFT	Neural JKO	Neural JKO IC (**ours**)
shifted 8 Modes	$3.2 \times 10^{-3} \pm 1.3 \times 10^{-4}$	$1.9 \times 10^{-5} \pm 1.1 \times 10^{-5}$	$8.8 \times 10^{-3} \pm 2.2 \times 10^{-3}$	$3.2 \times 10^{-2} \pm 6.1 \times 10^{-3}$	$8.3 \times 10^{-2} \pm 1.0 \times 10^{-3}$	$\mathbf{1.3 \times 10^{-5} \pm 4.4 \times 10^{-6}}$
shifted 8 Peaky	$8.3 \times 10^{-2} \pm 3.7 \times 10^{-4}$	$7.8 \times 10^{-2} \pm 9.9 \times 10^{-4}$	$7.9 \times 10^{-3} \pm 2.2 \times 10^{-3}$	$3.3 \times 10^{-2} \pm 1.3 \times 10^{-2}$	$8.4 \times 10^{-2} \pm 6.7 \times 10^{-4}$	$\mathbf{1.5 \times 10^{-5} \pm 3.2 \times 10^{-6}}$
GMM-10	$1.2 \times 10^{-2} \pm 5.5 \times 10^{-3}$	$1.0 \times 10^{-2} \pm 5.8 \times 10^{-3}$	$3.6 \times 10^{-3} \pm 1.9 \times 10^{-3}$	$1.4 \times 10^{-1} \pm 4.7 \times 10^{-2}$	$1.1 \times 10^{-2} \pm 5.6 \times 10^{-3}$	$\mathbf{2.1 \times 10^{-5} \pm 3.0 \times 10^{-6}}$
GMM-20	$6.6 \times 10^{-3} \pm 2.2 \times 10^{-3}$	$6.6 \times 10^{-3} \pm 2.4 \times 10^{-3}$	$3.6 \times 10^{-3} \pm 1.5 \times 10^{-3}$	$3.8 \times 10^{-1} \pm 3.9 \times 10^{-2}$	$7.3 \times 10^{-3} \pm 2.8 \times 10^{-3}$	$\mathbf{2.4 \times 10^{-5} \pm 9.8 \times 10^{-6}}$
GMM-50	$1.0 \times 10^{-2} \pm 3.8 \times 10^{-3}$	$9.8 \times 10^{-3} \pm 3.6 \times 10^{-3}$	$9.6 \times 10^{-3} \pm 4.9 \times 10^{-3}$	$9.0 \times 10^{-1} \pm 6.4 \times 10^{-5}$	$1.2 \times 10^{-2} \pm 4.4 \times 10^{-3}$	$\mathbf{2.6 \times 10^{-5} \pm 6.0 \times 10^{-6}}$
GMM-100	$1.1 \times 10^{-2} \pm 5.4 \times 10^{-3}$	$1.1 \times 10^{-2} \pm 5.4 \times 10^{-3}$	$1.1 \times 10^{-2} \pm 5.8 \times 10^{-3}$	$9.0 \times 10^{-1} \pm 4.8 \times 10^{-8}$	$1.4 \times 10^{-2} \pm 7.0 \times 10^{-3}$	$\mathbf{1.5 \times 10^{-4} \pm 8.5 \times 10^{-5}}$
GMM-200	$1.4 \times 10^{-2} \pm 5.1 \times 10^{-3}$	$1.4 \times 10^{-2} \pm 4.6 \times 10^{-3}$	$2.2 \times 10^{-2} \pm 8.6 \times 10^{-3}$	$9.0 \times 10^{-1} \pm 5.8 \times 10^{-8}$	$1.9 \times 10^{-2} \pm 6.3 \times 10^{-3}$	$\mathbf{6.8 \times 10^{-4} \pm 4.0 \times 10^{-4}}$

Table 5: We report the mode Sinkhorn distance for the different methods for the GMM40-50D example. A smaller mode Sinkhorn distance indicates a better result. The comparison values are taken from (Chen et al., 2025).

	Sampler						
	SMC	DDS	CRAFT	PIS	CMCD-LV	SCLD	Neural JKO IC (**ours**)
Sinkhorn distance	46370.34 ± 137.79	5435.18 ± 172.20	28960.70 ± 354.89	10405.75 ± 69.41	4258.57 ± 737.15	3787.73 ± 249.75	$\mathbf{3025.61 \pm 418.10}$

trace estimator for evaluating the divergence and a numerical method for solving the ODE. Therefore, we have a small error in the density propagation of the neural JKO steps. This error is amplified by the rejection steps since samples with underestimated density are more likely to be rejected than samples with overestimated density.

We would like to emphasize that this effect only appears for examples, where the energy distance between generated and ground truth samples is in the same order of magnitude like the average distance between two different sets of ground truth samples (see Table 1). This means that in the terms of the energy distance the generated and ground truth distribution are indistinguishable. At the same time the bias in the $\log(Z)$ estimate appears at the third or fourth relevant digit meaning that it is likely to be negligible.

E.3. Additional Comparison

We additionally compare our importance corrected neural JKO model with several other methods on the example of a 50 dimensional Gaussian mixture model as used in (Midgley et al., 2023). We follow the setting of Table 3 in Chen et al. (2025) and evaluate the Sinkhorn distance (Cuturi, 2013) with entropic regularization parameter 10^{-3} based on 2000 generated and ground truth samples. Moreover we average the results over 5 training runs with 25 evaluations from each run. We compare our results with sequential Monte Carlo (SMC, Del Moral et al., 2006), denoising diffusion sampler (DDS, Vargas et al., 2023a), continual repeated annealed flow transport Monte Carloe (CRAFT, Matthews et al., 2022), path integral sampler (PIS, Zhang & Chen, 2021), controlled Monte Carlo Diffusion (CMCD, Vargas et al., 2023b) and sequential controlled Langevin diffusion (SCLD, Chen et al. (2025)). For CMCD, we consider the log-variance (LV) variant considered in (Chen et al., 2025) since it gives better results. The hyperparameters used for our importance corrected neural JKO scheme are given in Table 6. Similar as most comparison methods, we use a normal distribution with a larger variance as initial distribution μ_0 and use a gradient flow on the negative log-density prior to the learned CNF and rejection steps, see Remark E.2 for details.

The results are given in Table 5, where the values for the comparison are taken from (Chen et al., 2025). We observe that neural JKO IC significantly outperforms the comparisons on this example.

E.4. Additional Figures and Evaluations

Additionally to the results from the main part of the paper, we provide the following evaluations.

Visualization of the Rejection Steps In Figure 1 and with involved steps in Figure 3, we visualize the different steps of our importance corrected neural JKO model on the shifted 8 Peaky example. Due to the shift of the modes, the modes on the right attract initially more mass than the ones on the left. In the rejection layers, we can see that samples are mainly rejected in oversampled regions such that the mode weights are equalized over time. This can also be seen in Figure 4, where we plot the weights of the different modes over time. We observe, that these weights are quite imbalanced for the latent distribution but are equalized over the rejection steps until they reach the ground truth value.

(Marginal) Plots of Generated Samples Despite the quantitative comparison of the methods in the Tables 1, 2 and 4, we also plot the first to coordinates of generated samples for the different test distributions for all methods for a visual comparison.

In Figure 5, we plot samples of the shifted 8 Modes example. We can see that all methods roughly cover the ground truth distribution even though CRAFT and DDS have slight and the uncorrected neural JKO scheme has a severe imbalance in the assigned mass for the different modes.

For the shifted 8 Peaky example in Figure 6, we see that this imbalance increases drastically for CRAFT, HMC, MALA and the uncorrected neural JKO. Also DDS has has a slight imbalance, while the importance corrected neural JKO scheme fits the ground truth almost perfectly.

The Funnel distribution in Figure 7 has two difficult parts, namely the narrow but high-density funnel on the left and the wide moderate density fan on the right. We can see that DDS does cover none of them very well. Also MALA, CRAFT and the uncorrected neural JKO do not cover the end of the funnel correctly and have also difficulties to cover the fan. HMC covers the fan well, but not the funnel. Only our importance corrected neural JKO scheme covers both parts in a reasonable way.

For the Mustache distribution in Figure 8, the critical parts are the two heavy but narrow tails. We observe that CRAFT and DDS are not able to cover them at all, while HMC and MALA and uncorrected neural JKO only cover them only partially. The importance corrected neural JKO covers them well.

Finally, for the GMM-d example we consider the dimensions $d = 10$ and $d = 200$ in the Figures 9 and 10. We can see that CRAFT mode collapses, i.e., for $d = 10$ it already finds some of the modes and for $d = 200$ it only finds one mode. DDS, HMC, MALA and the uncorrected neural JKO find all modes but do not distribute the mass correctly onto all modes. While this already appears for $d = 10$ it is more severe for $d = 200$. The importance corrected neural JKO sampler finds all modes and distributes the mass accurately.

Note that the distributions generated by MALA and the neural JKO are always similar. This is not surprising since both simulate a Wasserstein gradient flow with respect to the KL divergence and only differ by the time-discretization and the Metropolis correction in MALA.

Development of Error Measures over the Steps We plot how the quantities of interest decrease over the application of the steps of our model. The results are given in the Figure 11. It can be observed that the different steps may optimize different metrics. While the rejection steps improve the $\log(Z)$ estimate more significantly, the JKO steps focus more on the minimization of the energy distance. Overall, we see that in all figures the errors decrease over time.

E.5. Implementation Details

To build our importance corrected neural JKO model, we first apply $n_1 \in \mathbb{N}$ JKO steps followed by $n_2 \in \mathbb{N}$ blocks consisting out of a JKO step and three importance-based rejection steps. The velocity fields of the normalizing flows are parameterized by a dense three-layer feed-forward neural network. For the JKO steps, we choose an initial step size $\tau_0 > 0$ and then increase the step size exponentially by $\tau_{k+1} = 4\tau_k$. The choices of n_1, n_2, τ_0 and the number of hidden neurons from the networks is given in Table 6 together with the execution times for training and sampling. For evaluating the density propagation through the CNFs, we use the Hutchinson trace estimator with 5 Rademacher distributed random vectors whenever $d > 5$ and the exact trace evaluation otherwise. Note that we redraw the Rademacher vectors in every step of the ODE solver such that errors average out throughout the trajectory. We observed that the estimator of the divergence already has a sufficiently small variance, particularly, since the trajectories are almost straight due to the norm penalty of the velocity field, see also Appendix F.2. For implementing the CNFs, we use the code from Ffjord (Grathwohl et al., 2019) and the `torchdiffeq` library by (Chen, 2018). We provide the code online at `https://github.com/johertrich/neural_JKO_ic`.

For the MALA and HMC we run an independent chain for each generated sample and perform 50000 steps of the algorithm. For HMC we use 5 momentum steps and set the step size to 0.1 for 8Modes and Funnel and step size 0.01 for mustache. To stabilize the first iterations of MALA and HMC we run the first iterations with smaller step sizes (0.01 times the final step size in the first 1000 iterations and 0.1 times the final step size for the second 1000 iterations). For MALA we use step size 0.001 for all examples. For DDS and CRAFT we use the official implementations. In particular for the test distributions such as Funnel and LGCP we use the hyperparameters suggested by the original authors. For the other examples we optimized them via grid search.

Table 6: Parameters, training and sampling times for the different examples. For the sampling time we draw $N = 50000$ samples once the method is trained. The execution times are measured on a single NVIDIA RTX 4090 GPU with 24 GB memory.

	Distribution										
Parameter	Mustache	shifted 8 Modes	shifted 8 Peaky	Funnel	GMM-10	GMM-20	GMM-50	GMM-100	GMM-200	GMM40-50D	LGCP
Dimension	2	2	2	10	10	20	50	100	200	50	1600
Number n_1 of flow steps	6	2	2	6	4	4	4	4	5	4	3
Number n_2 of rejection blocks	6	4	4	6	6	6	7	8	8	7	6
Initial step size τ_0	0.05	0.01	0.01	5	0.0025	0.0025	0.0025	0.0025	0.001	1	5
Latent scaling c	1	1	1	1	1	1	1	1	1	40	1
Gradient Flow Initialization	-	-	-	-	-	-	-	-	-	3	-
Hidden neurons	54	54	54	256	70	90	150	250	512	256	1024
batch size	5000	5000	5000	5000	5000	5000	5000	5000	2000	5000	500
Required GPU memory	5 GB	5 GB	5 GB	5 GB	5 GB	5 GB	5 GB	5 GB	6 GB	6 GB	11 GB
Training time (min)	38	20	21	107	33	34	44	53	80	34	163
Sampling time (sec)	15	2	3	79	22	23	72	129	473	550	535

Remark E.2 (Initializations). To deal with target distributions living on different scales, we consider initial distributions $\mu_0 = \mathcal{N}(0, c^2\mathrm{Id})$, where c^2 is a scaling parameter. The choice of c for each target distribution is givein in Table 6 in the row "latent scaling". Moreover, it is beneficial in some cases to choose the velocity field in the first CNF not by learning it, but by fixing it to $v_t(x) = \nabla_x \log(g(x)) = \nabla_x \log(q(x))$. We report the final time T of this first gradient flow step in Table 6 in the row "gradient flow initialization" with value - if we do not use a gradient flow step. This initialization is closely related to Langevin preconditioning, which is used in most neural sampling methods, see (He et al., 2025) for a detailed study.

During the training of v_θ^{k+1}, we do not draw in each training step a new batch from μ_k. Instead, we maintain a dataset of $N = 50000$ samples of μ_k and draw batches from it. After the training is completed, we generate a new dataset of N samples by applying the learned CNF onto N new drawn samples from μ_k. To avoid overfitting effects, we redraw our dataset after each CNF step.

F. Computational Aspects of Normalizing Flows

In this appendix, we give some more details about the computational aspects of the normalizing flows in our model. First, we discuss the relation between multimodal target distributions, mode collapse and non-convex loss functions of normalizing flows. Afterwards, we discuss some computational aspects of continuous normalizing flows like ODE discretizations and density evaluations with trace estimators. Finally, we run some standard normalizing flow networks on our numerical example distributions and compare them with our importance corrected neural JKO sampling.

F.1. Multimodalities, Mode Collapse and Non-Convex Loss Functions

For the sampling application, normalizing flows are usually trained with the reverse KL loss function $\mathcal{F}(\mu) = \mathrm{KL}(\mu, \nu)$, see, e.g., (Marzouk et al., 2016). More precisely, a normalizing flow aims to learn the parameters θ of a family of diffeomorphisms $\mathcal{T}_\theta$ such that $\mathcal{T}_{\theta\#}\mu_0 \approx \nu$ by minimizing $\mathcal{F}(\mathcal{T}_{\theta\#}\mu_0)$. In the case that ν is multimodal it can be observed that this training mode collapses. That is, the approximation $\mathcal{T}_\#\mu_0$ covers not all of the modes of ν but instead neglects some of them. Examples of this phenomenon can be seen in the numerical comparison in Appendix F.3. One reason for this effect is that the functional $\mathcal{F}$ is convex if and only if ν is log-concave and therefore unimodal, see (Ambrosio et al., 2005, Prop. 9.3.2). In particular, for multimodal ν, the functional $\mathcal{F}$ is non-convex. In the latter case of, then the mode collapses can correspond to the convergence to a local minimum, as the following example shows.

Example F.1. *We consider the case $d = 1$ and the target distribution*

$$\nu = \tfrac{1}{2}\mathcal{N}(-\tfrac{1}{2}, 0.05^2) + \tfrac{1}{2}\mathcal{N}(\tfrac{1}{2}, 0.05^2).$$

As latent distribution μ_0 we choose a standard Gaussian. Then, we parametrize the normalizing flow $\mathcal{T}_\theta = (1-\theta)T_0 + \theta\mathcal{T}_1$ for $\theta \in [0, 1]$, where $\mathcal{T}_0$ is the optimal transport map between μ_0 and ν and $\mathcal{T}_1$ is the optimal transport map between μ_0 and $\mathcal{N}(\frac{1}{2}, 0.05^2)$. In particular, $\mathcal{T}_{\theta\#}\mu_0$ perfectly recovers the target distribution for $\theta = 0$ and produces a mode collapsed version of it for $\theta = 1$. Now, we plot the reverse KL loss function $\mathcal{L}(\theta) = \mathrm{KL}(\mathcal{T}_{\theta\#}\mu_0, \nu)$ and the densities of the generated distributions $\mathcal{T}_{\theta\#}\mu_0$ in Figure 12. We observe that it has two local minima for $\theta = 0$ and $\theta = 1$ (for θ outside of $[0, 1]$, $\mathcal{T}$ is no longer invertible), where $\theta = 0$ is the perfectly learned parameter and $\theta = 1$ is the mode collapsed version. At the same

time, we note that the curve $\theta \mapsto \mathcal{T}_{\theta\#}\mu_0$ is a geodesic in the Wasserstein space. In particular, the non-convexity of $\mathcal{L}$ is a direct consequence of the fact that $\mathcal{F}(\mu) = \mathrm{KL}(\mu, \nu)$ is geodesically non-convex.

Remark F.2 (Motivation of Wasserstein Regularization). This connection between the non-convexity of the loss function $\mathcal{F}$ and mode collapse also motivates the Wasserstein regularization from Section 3. By considering the loss function $\mathcal{G}(\mu) = \mathcal{F}(\mu) + \frac{1}{2\tau}W_2^2(\mu, \mu_\tau^k)$ for $\tau > 0$ small enough instead of $\mathcal{F}$, we obtain a loss function which is convex in the Wasserstein space, see (Ambrosio et al., 2005, Lem. 9.2.7). Even though this does not imply that the map $\theta \mapsto \mathcal{G}(\mathcal{T}_{\theta\#}\mu_0)$ is convex, we can expect that the training does not get stuck in local minima as long as the architecture of $\mathcal{T}_\theta$ is expressive enough.

Theoretically, we need for λ-convex $\mathcal{F}$ that $\tau \leq \frac{1}{\lambda}$ to ensure that $\mathcal{G}$ is geodesically convex. However, if μ^k is already close to a minimum of $\mathcal{F}$, then it can be sufficient that the functional $\mathcal{G}$ is convex locally around μ^k. In this case, the distribution generated by the CNF will stay in this neighborhood. Note that in practice the constant λ is unknown such that we start with a small step size τ_0 and increase it over time as outlined at the end of Section 4 and in Appendix E.5.

F.2. Computational Limitations of (Continuous) Normalizing Flows

Similar to the literature on neural JKO schemes (Vidal et al., 2023; Xu et al., 2024), our implementation of the neural JKO scheme relies on continuous normalizing flows (Chen et al., 2018). This comes with some limitations and challenges, which were extensively discussed in (Chen et al., 2018). Since they are relevant for our method, we give a synopsis below.

Derivatives of ODE Solutions For the training phase we need the derivative of the solution of an ODE. For this, we use the `torchdiffeq` package (Chen, 2018). In particular, this package does not rely on backpropagation through the steps from the forward solver. Instead, it overwrites the backward pass of the ODE by solving an *adjoint ODE*. This avoids expensive tracing in the automatic differentiation process within the forward pass and keeps the memory consumption low. Moreover, the quadratic regularization of the velocity field leads to straight paths such that the adaptive solvers only require a few steps for solving the ODE, see (Onken et al., 2021) for a detailed discussion. Indeed, we use in our numerical examples the `dopri5` solver from `torchdiffeq`, which is an adaptive Runge-Kutta method. However, we still have to solve two ODEs during the training time. This still can be computational costly, in particular when the underlying model is large.

Trace Estimation for Density Computations In order to evaluate the (log-)density of our model, we have to compute the divergence $\operatorname{div} v_\theta = \operatorname{trace}(\nabla v_\theta(z_\theta(x,t),t))$ of the learned vector field v_θ (and integrate it over time), see (7). Computing the trace of the Jacobian of v_θ becomes computationally costly in high dimensions. As a remedy, we consider the Hutchinson trace estimator (Hutchinson, 1989), which states that for any matrix A and a random vector z with mean zero and identity covariance matrix it holds that $\mathbb{E}[z^{\mathrm{T}}Az] = \operatorname{trace}(A)$. Applying this estimator to the divergence, we obtain that the divergence coincides with $\mathbb{E}[z^{\mathrm{T}}\nabla v_\theta(z_\theta(x,t),t)z]$. The integrand is now again a Jacobian-vector product which can be computed efficiently. Finally, we estimate the trace by empirically discretizing the expectation by finitely many realizations of z.

In our numerics, we choose z to be Rademacher random vectors, i.e. each entry is with probability $\frac{1}{2}$ either -1 or 1. For the training of the continuous normalizing flow, an unbiased low-precision estimator is sufficient, such that we discretize the expectation with one realization of z. During evaluation, we require a higher precision and use 5 realizations instead.

Initialization In order to find a stable initialization of the model, we initialize the last layer of the velocity field v_θ with zeros such that it holds $v_\theta(x,t) = 0$ for all $x \in \mathbb{R}^d$, $t \in [0,\tau]$. In this case, the solution z_θ of the ODE $\dot{z}_\theta = v_\theta$ with initial condition $z_\theta(x,0) = x$ is given by $z_\theta(x,t) = x$ for all t. In particular, we have that the transport map $\mathcal{T}_\theta = z_\theta(x,\tau)$ is the identity such that the generated distribution $\mathcal{T}_{\theta\#}\mu_0$ coincides with the latent distribution for the initial parameters.

Remark F.3 (Other Normalizing Flow Architectures). In general any architecture $\mathcal{T}_\theta$ of normalizing flows can be considered in the neural JKO scheme. To this end, we can replace the velocity field regularization in (5) by the penalizer $\int_{\mathbb{R}^d} \|\mathcal{T}_\theta(x) - x\|^2 \mathrm{d}\mu_\tau^k$. By Brenier's theorem (Brenier, 1987) this is again equivalent to minimizing the Wasserstein distance. However, while discrete-time normalizing flows like coupling-based networks (Dinh et al., 2016; Kingma & Dhariwal, 2018) and autoregressive flows (De Cao et al., 2020; Durkan et al., 2019; Huang et al., 2018; Papamakarios et al., 2017) are often faster than CNFs, this is not generally true in our setting, since their evaluation time does not benefit from the OT-regularization. Additionally, we observed numerically that the expressiveness of discrete-time architectures scales much worse to high dimensions and that these architectures are less stable to train. Nevertheless, their evaluation can be cheaper since these architectures do not have to deal with trace estimators within the density evaluation. Residual architectures (Behrmann

Table 7: Comparison of neural JKO IC with normalizing flows trained with the reverse KL loss function. We evaluate the energy distance (smaller values are better), the $\log(Z)$-estimation (larger values are better) and the expected squared empirical Wasserstein distance (smaller values are better).

	Energy distance			$\log(Z)$-estimation			squared Wasserstein-2		
Distribution	continuous NF	coupling NF	neural JKO IC	continuous NF	coupling NF	neural JKO IC	continuous NF	coupling NF	neural JKO IC
Mustache	2.1×10^{-2}	4.9×10^{-3}	$\mathbf{2.9 \times 10^{-3}}$	-7.1×10^{-2}	-2.2×10^{-2}	$\mathbf{-7.3 \times 10^{-3}}$	$3.8 \times 10^{+1}$	$3.1 \times 10^{+1}$	$\mathbf{1.7 \times 10^{+1}}$
shifted 8 Modes	4.1×10^{-1}	4.1×10^{-2}	$\mathbf{1.2 \times 10^{-5}}$	-1.4×10^{0}	-4.3×10^{-1}	$\mathbf{+5.1 \times 10^{-6}}$	1.4×10^{0}	2.2×10^{-1}	$\mathbf{6.5 \times 10^{-3}}$
shifted 8 Peaky	5.7×10^{-1}	5.8×10^{-1}	$\mathbf{3.4 \times 10^{-5}}$	-2.1×10^{0}	-2.1×10^{0}	$\mathbf{-2.1 \times 10^{-3}}$	1.8×10^{0}	1.8×10^{0}	$\mathbf{7.2 \times 10^{-3}}$
Funnel	1.9×10^{-1}	$\mathbf{2.2 \times 10^{-3}}$	1.4×10^{-2}	-9.4×10^{-1}	-2.8×10^{-2}	$\mathbf{-7.1 \times 10^{-3}}$	$\mathbf{3.8 \times 10^{+2}}$	$8.1 \times 10^{+2}$	$8.5 \times 10^{+2}$
GMM-10	5.4×10^{-1}	5.3×10^{-1}	$\mathbf{5.2 \times 10^{-5}}$	-2.3×10^{0}	-2.3×10^{0}	$\mathbf{+3.5 \times 10^{-3}}$	3.4×10^{0}	3.5×10^{0}	$\mathbf{1.4 \times 10^{-1}}$
GMM-20	1.1×10^{0}	1.1×10^{0}	$\mathbf{1.1 \times 10^{-4}}$	-2.4×10^{0}	-2.3×10^{0}	$\mathbf{+6.4 \times 10^{-3}}$	9.5×10^{0}	9.4×10^{0}	$\mathbf{3.7 \times 10^{-1}}$

et al., 2019; Chen et al., 2019) are at least similar to continuous normalizing flows, but they are very expensive to train and evaluate, and additionally rely on trace estimators for computing the density of the generated samples.

F.3. Numerical Comparison

Finally, we compare our neural JKO IC scheme with standard normalizing flows trained with the reverse KL loss function as proposed in (Marzouk et al., 2016). In particular, we aim to demonstrate the benefits of our neural JKO IC scheme to avoid mode collapse.

To this end, we train two architectures of normalizing flows for our examples using the reverse KL loss function as proposed in (Marzouk et al., 2016). First, we use a continuous normalizing flow with the same architecture as used in the neural JKO (IC). That is, we parameterize the velocity field v_θ by a dense neural network with three hidden layers and the same hidden dimensions as in Table 6. Second, we compare with a coupling network with 5 Glow-coupling blocks (Kingma & Dhariwal, 2018), where the coupling blocks have two hidden layers with three times the hidden dimension as in Table 6. We found numerically that choosing larger architectures does not bring significant advantages.

We plot some generated distributions of the continuous and coupling normalizing flows as well as the neural JKO IC scheme in Figure 13. As we have already seen in the previous examples, the neural JKO IC scheme is able to recover multimodal distributions almost perfectly. On the other hand, the normalizing flow architectures always collapse to one or a small number of modes. We additionally report the error measures in Table 7. We can see that neural JKO IC performs always better than the normalizing flow architectures (note that for the Funnel distribution all W_2^2-values are below the sampling error reported in Table 3). For multimodal distributions, the normalizing flow approximations are by several orders of magnitude worse than neural JKO IC, while they work quite well for unimodal distributions (Mustache and Funnel).

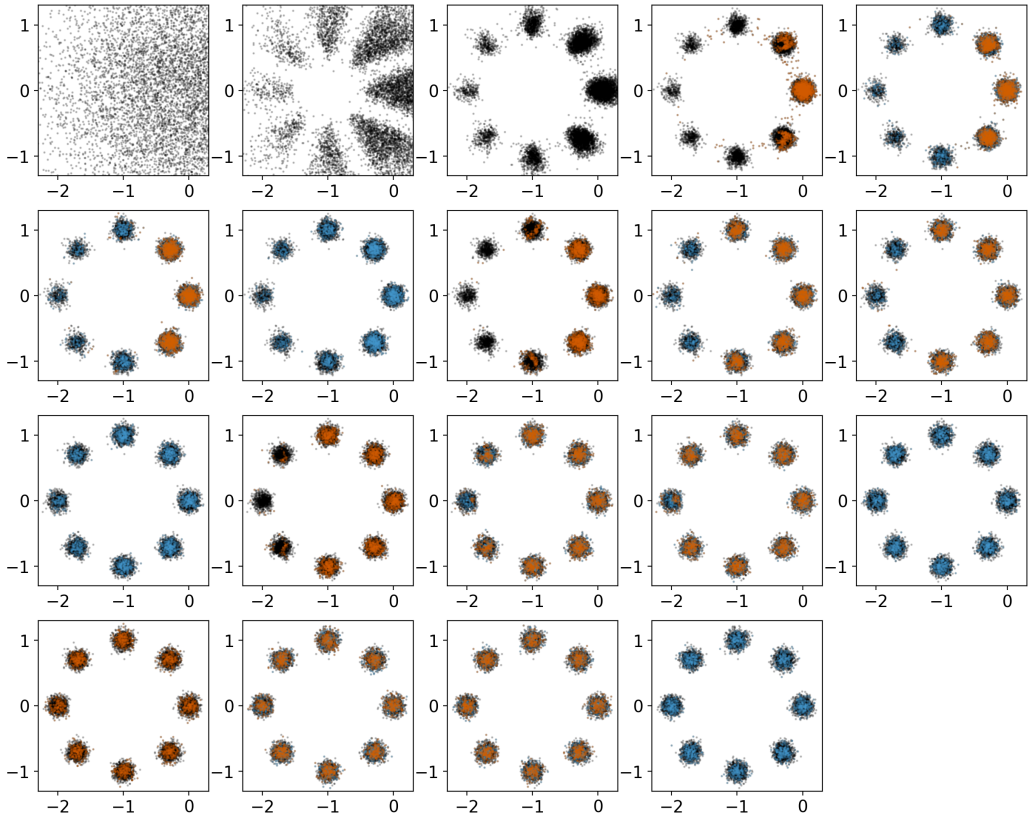

Figure 3: Visualization of the steps of the importance corrected neural JKO model for the shifted 8 Peaky example. We start at the top left with the latent distribution and apply in each image one step from the model. The orange color indicates samples which are rejected in the next step and the blue color marks the resampled points.

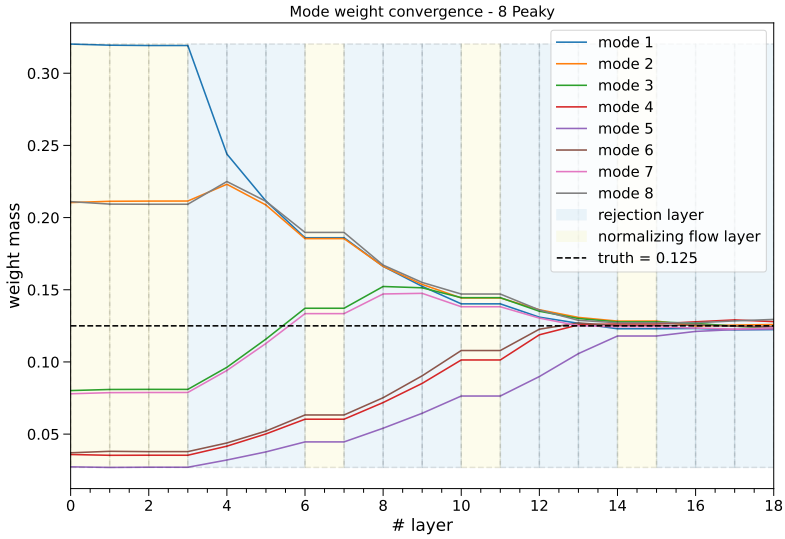

Figure 4: Plot of the mode weights for the 8 Peaky example over the different layers of the importance corrected neural JKO model. We observe that the weights are mainly changed by the rejection steps and not by the neural JKO steps.

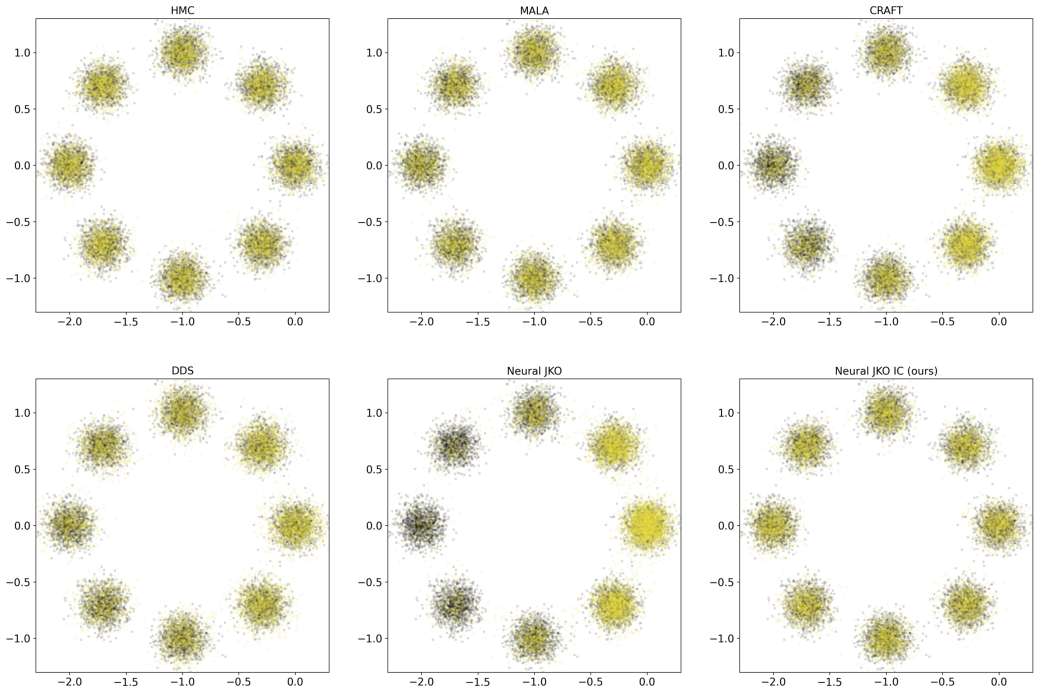

Figure 5: Sample generation with various methods for shifted 8 Modes example with *ground truth* samples and *generated* samples for each associated method. While most methods recover the distribution well, we can see a imbalance in the modes for the uncorrected neural JKO, CRAFT and DDS.

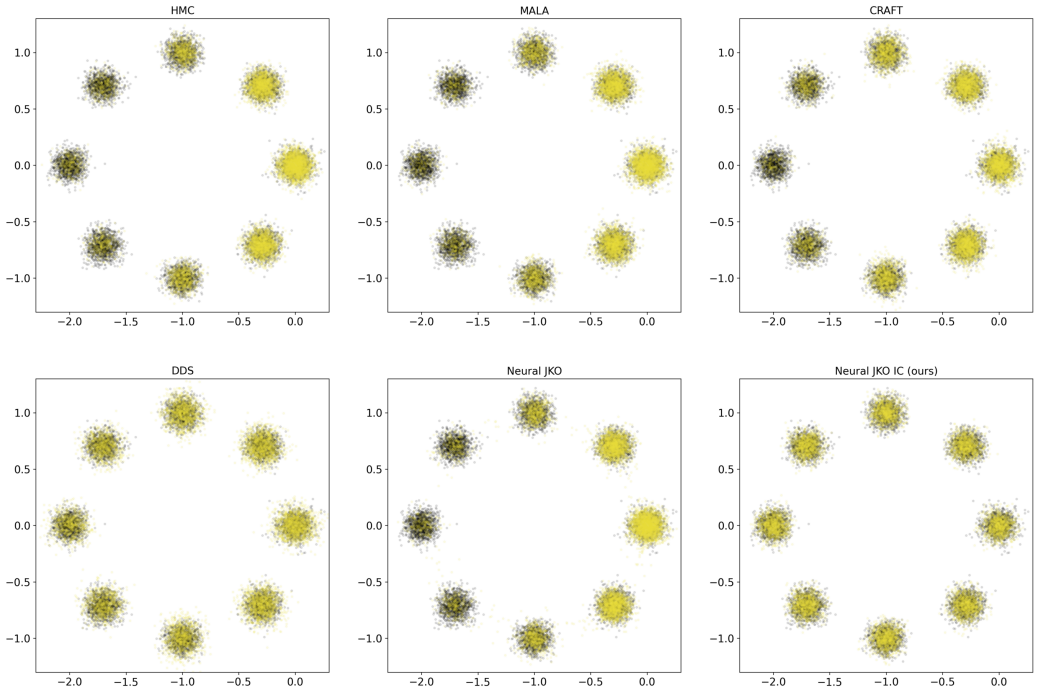

Figure 6: Sample generation with various methods for shifted 8 Peaky mixtures with *ground truth* samples and *generated* samples for each associated method. We can see a severe imbalance among the modes for HMC, MALA, CRAFT and neural JKO. Also DDS has a slight imbalance between the modes.

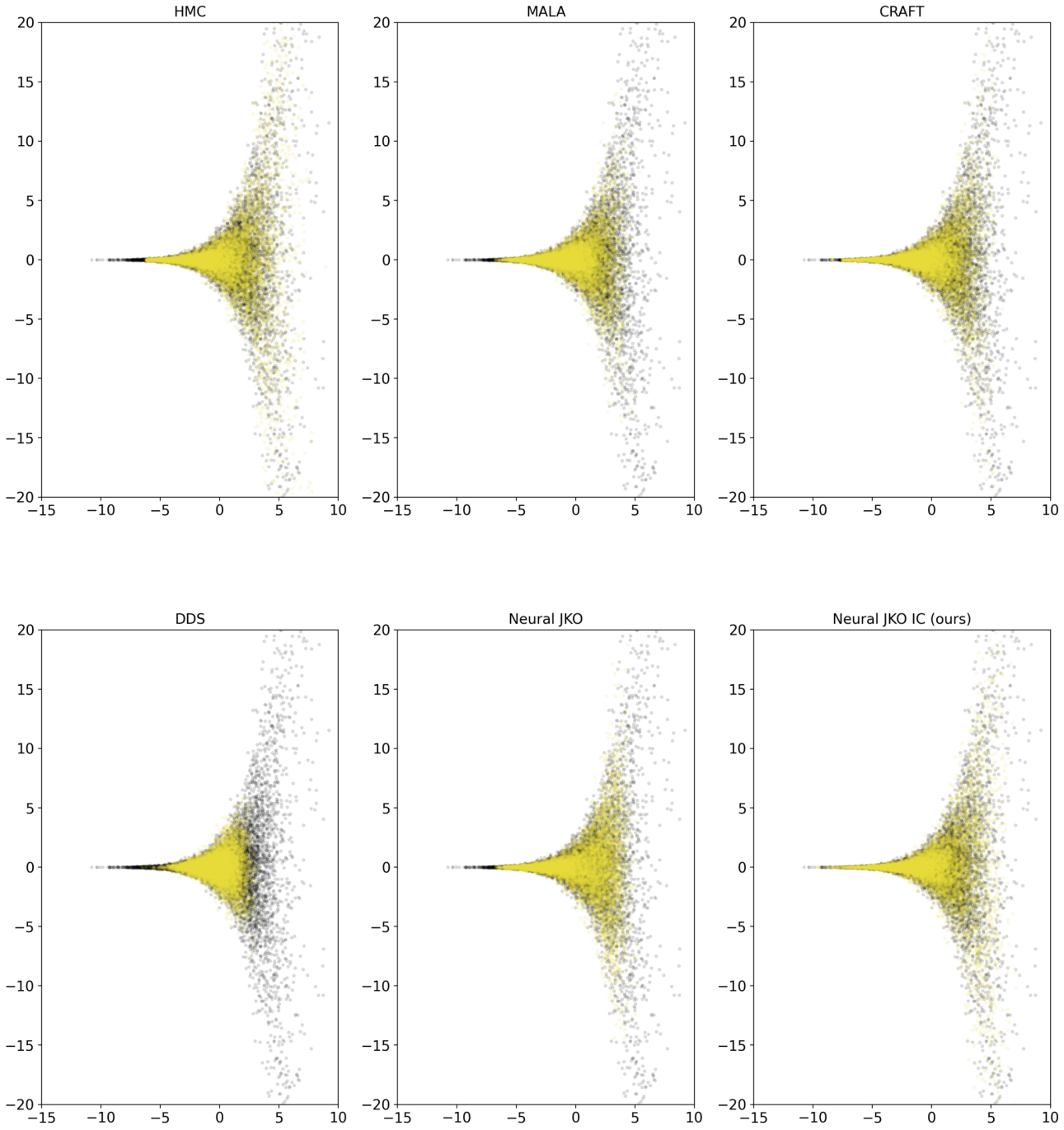

Figure 7: Marginalized sample generation with various methods for the $d = 10$ funnel distribution with *ground truth* samples and *generated* samples for each associated method. We observe that only the importance corrected neural JKO covers the thin part of the funnel well.

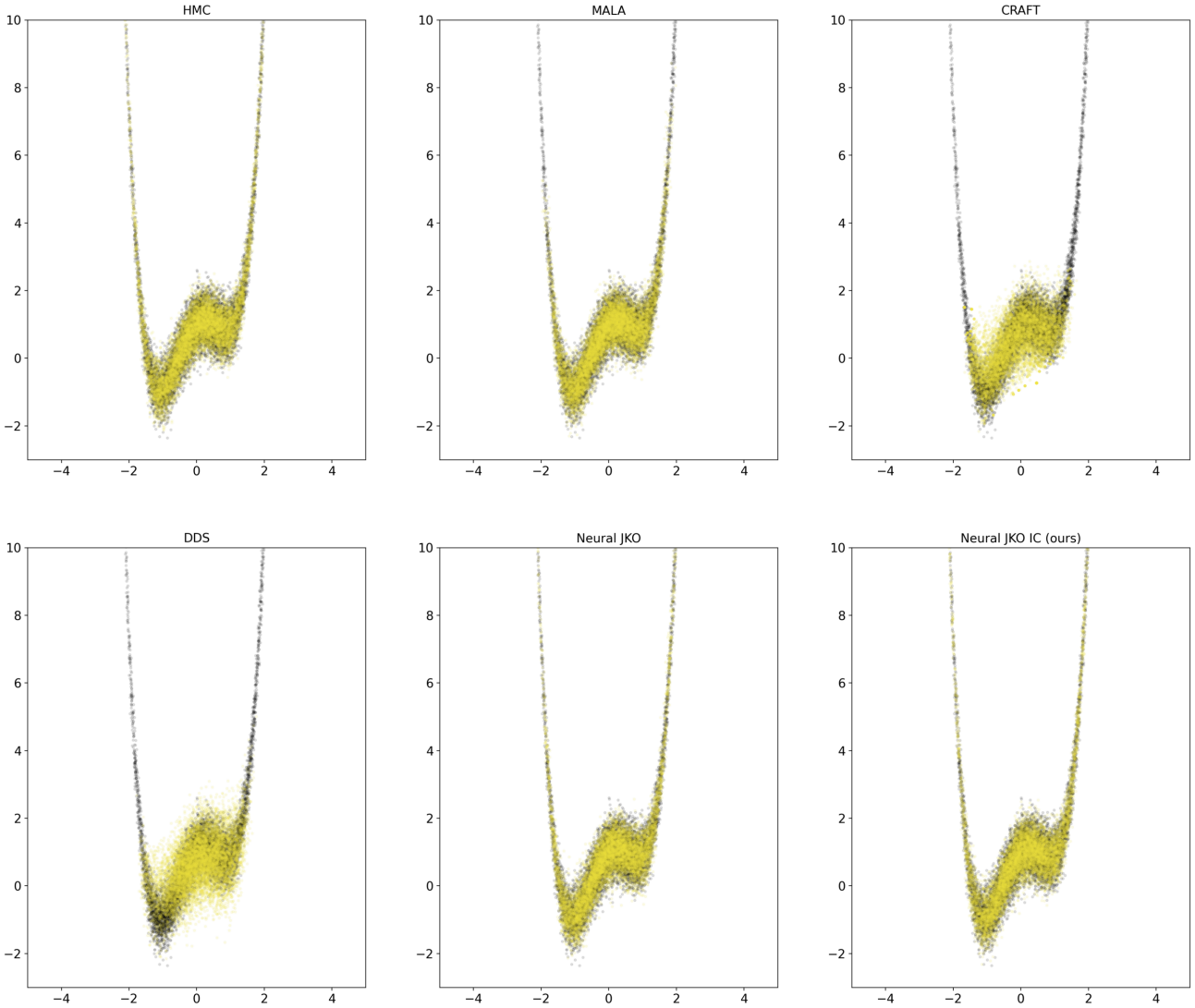

Figure 8: Sample generation with various methods for the $d = 2$ mustache distribution with *ground truth* samples and *generated* samples for each associated method. We can see that MALA, CRAFT and DDS have difficulties to model the long tails of the distribution properly.

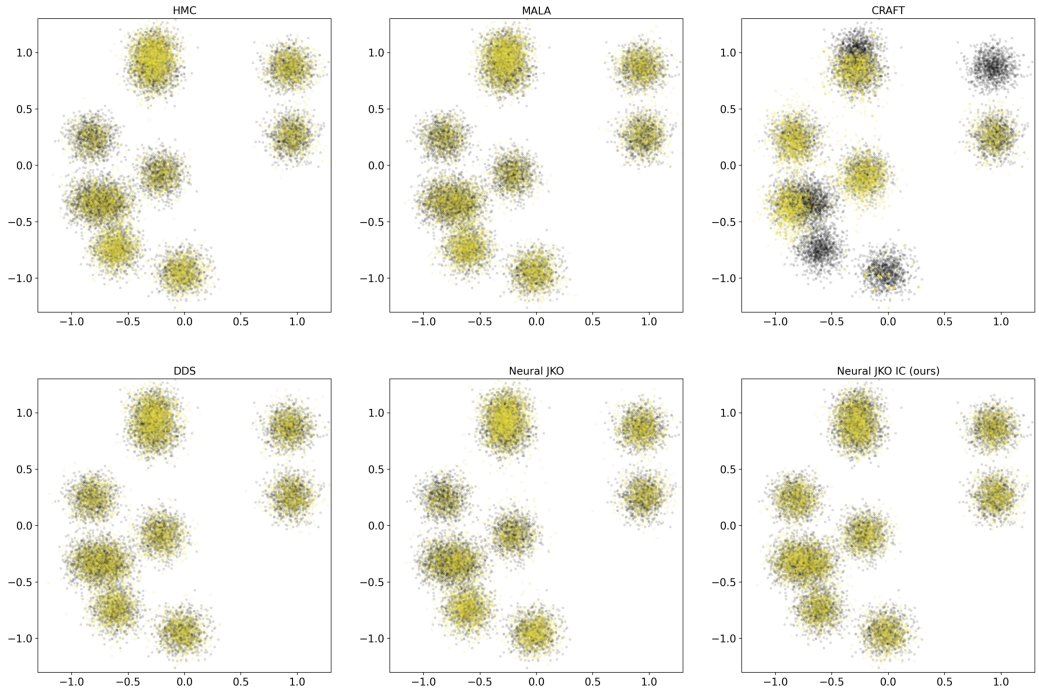

Figure 9: Marginalized sample generation with various methods for the GMM-10 distribution with *ground truth* samples and *generated* samples for each associated method. We observe that CRAFT mode collapses and that only the importance corrected neural JKO model distributes the mass correctly onto the modes.

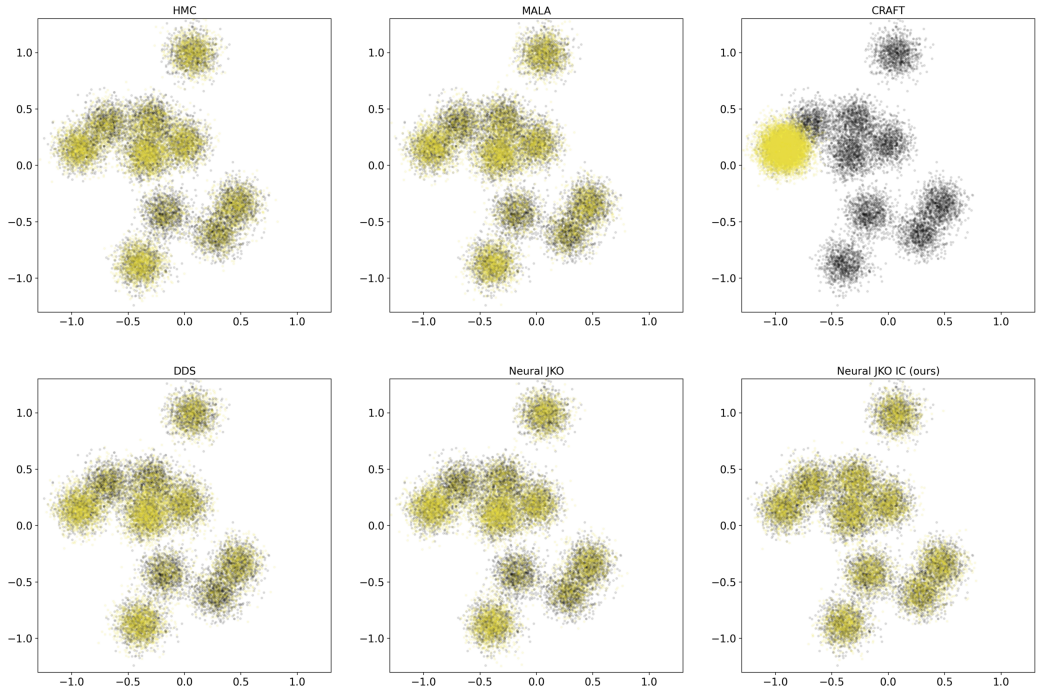

Figure 10: Marginalized sample generation with various methods for the GMM-200 distribution with *ground truth* samples and *generated* samples for each associated method. We observe that CRAFT mode collapses and that only the importance corrected neural JKO model distributes the mass correctly onto the modes.

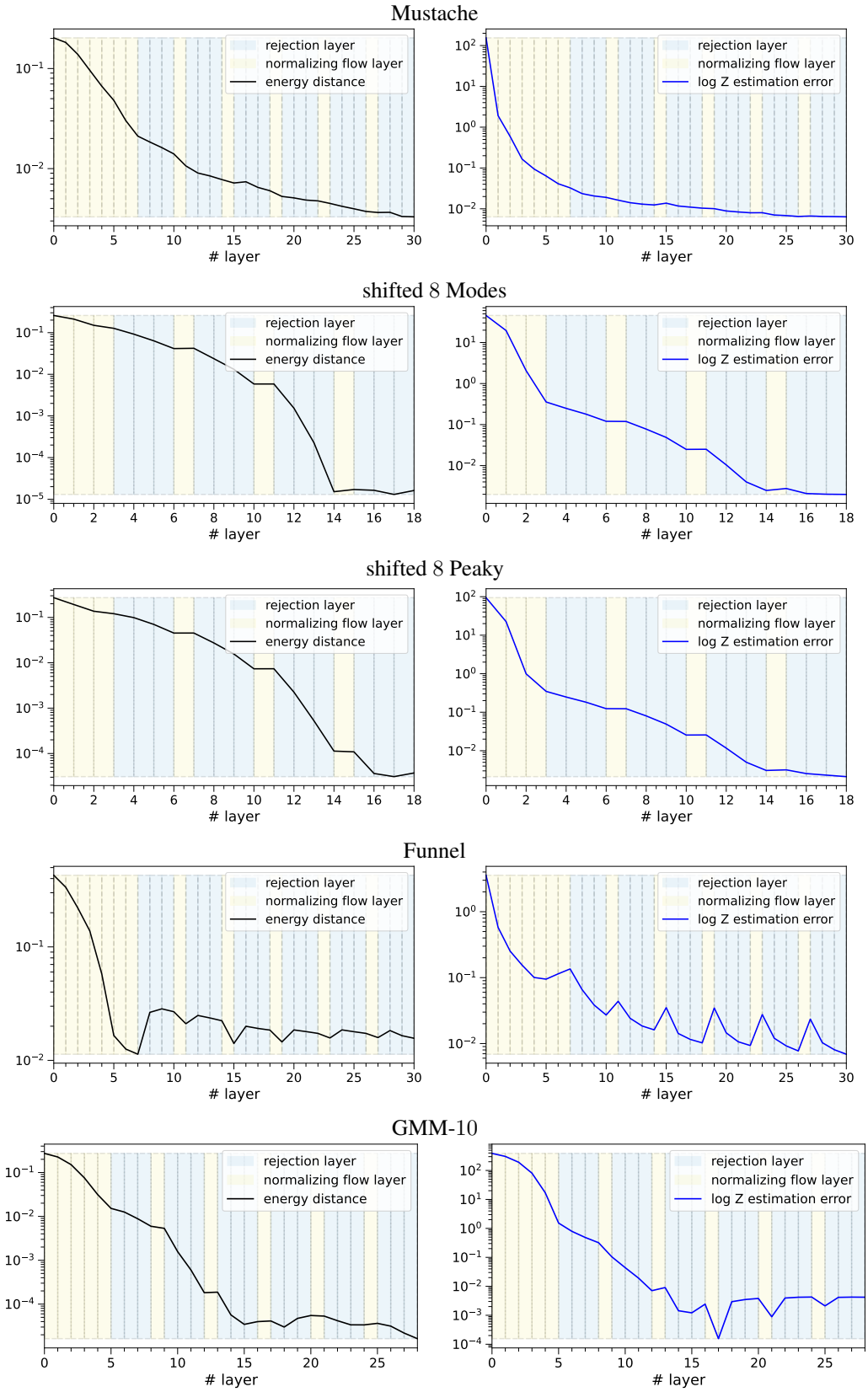

Figure 11: We plot the energy distance (**left**) and $\log(Z)$ estimate (**right**) over the steps of our importance corrected neural JKO method for different examples. The error measures decrease in the beginning and then saturate at some values.

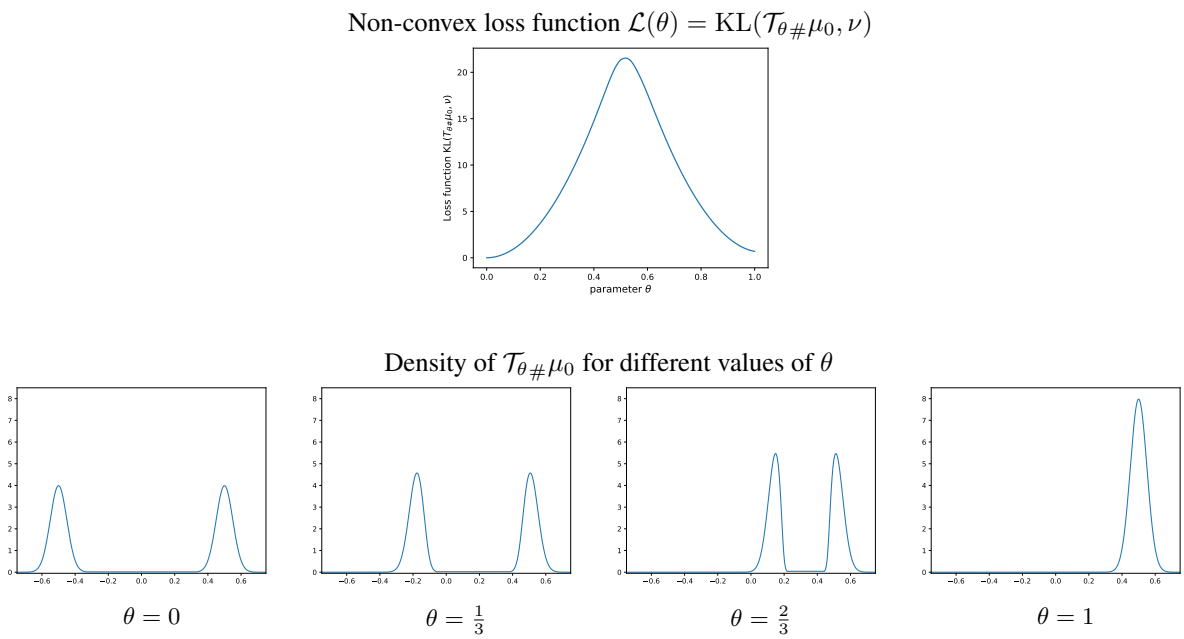

Figure 12: Illustration of the loss function $[0,1] \ni \theta \mapsto \mathcal{L}(\theta) = \mathrm{KL}(\mathcal{T}_{\theta\#}\mu_0, \nu)$ and the densities of the generated distributions $\mathcal{T}_{\theta\#}\mu_0$ from Example F.1. Both values $\theta = 0$ and $\theta = 1$ correspond to local minima (note that $\mathcal{L}(1) > 0 = \mathcal{L}(0)$). In particular $\theta = 1$ corresponds to the case of mode collapse.

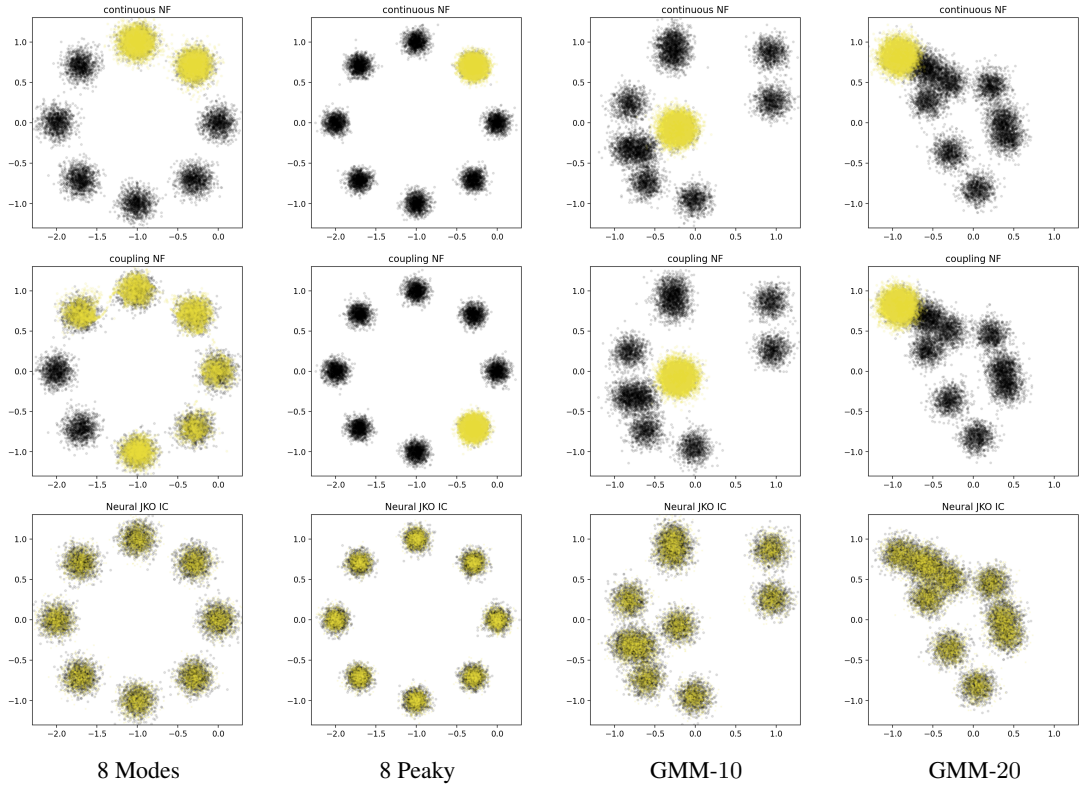

Figure 13: Marginalized sample generation for a single normalizing flow compared with neural JKO IC for different example distributions with *ground truth* samples and *generated* samples for each associated method. We observe that the standard normalizing flow architectures always collapse to one or few modes while neural JKO IC recovers all modes correctly.