Sliced Iterative Normalizing Flows

Biwei Dai [1] **Uroš Seljak** [1][2]

Abstract

We develop an iterative (greedy) deep learning (DL) algorithm which is able to transform an arbitrary probability distribution function (PDF) into the target PDF. The model is based on iterative Optimal Transport of a series of 1D slices, matching on each slice the marginal PDF to the target. The axes of the orthogonal slices are chosen to maximize the PDF difference using Wasserstein distance at each iteration, which enables the algorithm to scale well to high dimensions. As special cases of this algorithm, we introduce two sliced iterative Normalizing Flow (SINF) models, which map from the data to the latent space (GIS) and vice versa (SIG). We show that SIG is able to generate high quality samples of image datasets, which match the GAN benchmarks, while GIS obtains competitive results on density estimation tasks compared to the density trained NFs, and is more stable, faster, and achieves higher $p(x)$ when trained on small training sets. SINF approach deviates significantly from the current DL paradigm, as it is greedy and does not use concepts such as mini-batching, stochastic gradient descent and gradient back-propagation through deep layers.

1. Introduction

Latent variable generative models such as Normalizing Flows (NFs) (Rezende & Mohamed, 2015; Dinh et al., 2014; 2017; Kingma & Dhariwal, 2018), Variational AutoEncoders (VAEs) (Kingma & Welling, 2014; Rezende et al., 2014) and Generative Adversarial Networks (GANs) (Goodfellow et al., 2014; Radford et al., 2016) aim to model the distribution $p(x)$ of high-dimensional input data x by introducing a mapping from a latent variable z to x, where z is assumed to follow a given prior distribution $\pi(z)$. These

[1]Department of Physics, University of California, Berkeley, California, USA [2]Lawrence Berkeley National Laboratory, Berkeley, California, USA. Correspondence to: Biwei Dai <biwei@berkeley.edu>.

models usually parameterize the mapping using neural networks, and the training of these models typically consists of minimizing a dissimilarity measure between the model distribution and the target distribution. For NFs and VAEs, maximizing the marginal likelihood is equivalent to minimizing the Kullback–Leibler (KL) divergence. While for GANs, the adversarial training leads to minimizations of the Jenson-Shannon (JS) divergence (Goodfellow et al., 2014). The performance of these models largely depends on the following aspects:

1) The parametrization of the mapping (the architecture of the neural network) should match the structure of the data and be expressive enough. Different architectures have been proposed (Kingma & Dhariwal, 2018; van den Oord et al., 2017; Karras et al., 2018; 2019), but to achieve the best performance on a new dataset one still needs extensive hyperparameter explorations (Lucic et al., 2018).

2) The dissimilarity measure (the loss function) should be appropriately chosen for the tasks. For example, in high dimensions the JS divergence is more correlated with the sample quality than KL divergence (Huszár, 2015; Theis et al., 2016), which is believed to be one of the reasons that GANs are able to generate higher quality samples than VAEs and NFs. However, JS divergence is hard to directly work with, and the adversarial training could bring many problems such as vanishing gradient, mode collapse and non-convergence (Arjovsky & Bottou, 2017; Wiatrak & Albrecht, 2019).

To avoid these complexities, in this work we adopt a different approach to build the map from latent variable z to data x. We approach this problem from the Optimal Transport (OT) point of view. OT studies whether the transport maps exist between two probability distributions, and if they do, how to construct the map to minimize the transport cost. Even though the existence of transport maps can be proved under mild conditions (Villani, 2008), it is in general hard to construct them in high dimensions. We propose to decompose the high dimensional problem into a succession of 1D transport problems, where the OT solution is known. The mapping is iteratively augmented, and it has a NF structure that allows explicit density estimation and efficient sampling. We name the algorithm Sliced Iterative Normalizing Flow (SINF). Our objective function is inspired by the Wasserstein distance, which is defined as the minimal transport

cost and has been widely used in the loss functions of generative models (Arjovsky & Bottou, 2017; Tolstikhin et al., 2018). We propose a new metric, max K-sliced Wasserstein distance, which enables the algorithm to scale well to high dimensions.

In particular, SINF algorithm has the following properties:
1) The performance is competitive compared to state-of-the-art (SOTA) deep learning generative models. We show that if the objective is optimized in data space, the model is able to produce high quality samples similar to those of GANs; and if it is optimized in latent space, the model achieves comparable performance on density estimation tasks compared to NFs trained with maximum likelihood, and achieves highest performance on small training sets.
2) Compared to generative models based on neural networks, this algorithm has very few hyperparameters, and the performance is insensitive to their choices.
3) The model training is very stable and insensitive to random seeds. In our experiments we do not observe any cases of training failures.

2. Background

2.1. Normalizing Flows

Flow-based models provide a powerful framework for density estimation (Dinh et al., 2017; Papamakarios et al., 2017) and sampling (Kingma & Dhariwal, 2018). These models map the d-dimensional data x to d-dimensional latent variables z through a sequence of invertible transformations $f = f_1 \circ f_2 \circ ... \circ f_L$, such that $z = f(x)$ and z is mapped to a base distribution $\pi(z)$, which is normally chosen to be a standard Normal distribution. The probability density of data x can be evaluated using the change of variables formula:

$$
\begin{aligned}
p(x) = \quad & \pi(f(x)) |\det\left(\tfrac{\partial f(x)}{\partial x}\right)| \\
= \quad & \pi(f(x)) \prod_{l=1}^{L} |\det\left(\tfrac{\partial f_l(x)}{\partial x}\right)|.
\end{aligned}
\quad (1)
$$

The Jacobian determinant $\det(\frac{\partial f_l(x)}{\partial x})$ must be easy to compute for evaluating the density, and the transformation f_l should be easy to invert for efficient sampling.

2.2. Radon Transform

Let $\mathbb{L}^1(X)$ be the space of absolute integrable functions on X. The Radon transform $\mathcal{R} : \mathbb{L}^1(\mathbb{R}^d) \to \mathbb{L}^1(\mathbb{R} \times \mathbb{S}^{d-1})$ is defined as

$$
(\mathcal{R}p)(t, \theta) = \int_{\mathbb{R}^d} p(x) \delta(t - \langle x, \theta \rangle) dx, \quad (2)
$$

where $\mathbb{S}^{d-1}$ denotes the unit sphere $\theta_1^2 + \cdots \theta_d^2 = 1$ in $\mathbb{R}^d$, $\delta(\cdot)$ is the Dirac delta function, and $\langle \cdot, \cdot \rangle$ is the standard

inner product in $\mathbb{R}^d$. For a given θ, the function $(\mathcal{R}p)(\cdot, \theta) : \mathbb{R} \to \mathbb{R}$ is essentially the slice (or projection) of $p(x)$ on axis θ.

Note that the Radon transform $\mathcal{R}$ is invertible. Its inverse, also known as the filtered back-projection formula, is given by (Helgason, 2010; Kolouri et al., 2019)

$$
\mathcal{R}^{-1}((\mathcal{R}p)(t, \theta))(x) = \int_{\mathbb{S}^{d-1}} ((\mathcal{R}p)(\cdot, \theta) * h)(\langle x, \theta \rangle) d\theta,
\quad (3)
$$

where $*$ is the convolution operator, and the convolution kernel h has the Fourier transform $\hat{h}(k) = c|k|^{d-1}$. The inverse Radon transform provides a practical way to reconstruct the original function $p(x)$ using its 1D slices $(\mathcal{R}p)(\cdot, \theta)$, and is widely used in medical imaging. This inverse formula implies that if the 1D slices of two functions are the same in all axes, these two functions are identical, also known as Cramér-Wold theorem (Cramér & Wold, 1936).

2.3. Sliced and Maximum Sliced Wasserstein Distances

The p-Wasserstein distance, $p \in [1, \infty)$, between two probability distributions p_1 and p_2 is defined as:

$$
W_p(p_1, p_2) = \inf_{\gamma \in \Pi(p_1, p_2)} \left(\mathbb{E}_{(x,y) \sim \gamma} \left[\|x - y\|^p \right] \right)^{\frac{1}{p}}, \quad (4)
$$

where $\Pi(p_1, p_2)$ is the set of all possible joint distributions $\gamma(x, y)$ with marginalized distributions p_1 and p_2. In 1D the Wasserstein distance has a closed form solution via Cumulative Distribution Functions (CDFs), but this evaluation is intractable in high dimension. An alternative metric, the Sliced p-Wasserstein Distance (SWD), is defined as:

$$
SW_p(p_1, p_2) = \left(\int_{\mathbb{S}^{d-1}} W_p^p(\mathcal{R}p_1(\cdot, \theta), \mathcal{R}p_2(\cdot, \theta)) d\theta \right)^{\frac{1}{p}},
\quad (5)
$$

where $d\theta$ is the normalized uniform measure on $\mathbb{S}^{d-1}$. The SWD can be calculated by approximating the high dimensional integral with Monte Carlo samples. However, in high dimensions a large number of projections is required to accurately estimate SWD. This motivates to use the maximum Sliced p-Wasserstein Distance (max SWD):

$$
\max\text{-}SW_p(p_1, p_2) = \max_{\theta \in \mathbb{S}^{d-1}} W_p(\mathcal{R}p_1(\cdot, \theta), \mathcal{R}p_2(\cdot, \theta)),
\quad (6)
$$

which is the maximum of the Wasserstein distance of the 1D marginalized distributions of all possible directions. SWD and max SWD are both proper distances (Kolouri et al., 2015; 2019).

3. Sliced Iterative Normalizing Flows

We consider the general problem of building a NF that maps an arbitrary PDF $p_1(x)$ to another arbitrary PDF $p_2(x)$ of

the same dimensionality. We first introduce our objective function in Section 3.1. The general SINF algorithm is presented in Section 3.2. We then consider the special cases of p_1 and p_2 being standard Normal distributions in Section 3.3 and Section 3.4, respectively. In Section 3.5 we discuss our patch-based hierarchical strategy for modeling images.

3.1. Maximum K-sliced Wasserstein Distance

We generalize the idea of maximum SWD and propose maximum K-Sliced p-Wasserstein Distance (max K-SWD):

$$\max\text{-}K\text{-}SW_p(p_1, p_2) = \max_{\{\theta_1, \cdots, \theta_K\} \text{ orthonormal}}$$

$$\left(\frac{1}{K} \sum_{k=1}^{K} W_p^p((\mathcal{R}p_1)(\cdot, \theta_k), (\mathcal{R}p_2)(\cdot, \theta_k)) \right)^{\frac{1}{p}}. \quad (7)$$

In this work we fix $p = 2$. The proof that max K-SWD is a proper distance is in the appendix. If $K = 1$, it becomes max SWD. For $K < d$, the idea of finding the subspace with maximum distance is similar to the subspace robust Wasserstein distance (Paty & Cuturi, 2019). Wu et al. (2019) and Rowland et al. (2019) proposed to approximate SWD with orthogonal projections, similar to max K-SWD with $K = d$. max K-SWD will be used as the objective in our proposed algorithm. It defines K orthogonal axes $\{\theta_1, \cdots, \theta_K\}$ for which the marginal distributions of p_1 and p_2 are the most different, providing a natural choice for performing 1D marginal matching in our algorithm (see Section 3.2).

The optimization in max K-SWD is performed under the constraints that $\{\theta_1, \cdots, \theta_K\}$ are orthonormal vectors, or equivalently, $A^T A = I_K$ where $A = [\theta_1, \cdots, \theta_K]$ is the matrix whose i-th column vector is θ_i. Mathematically, the set of all possible A matrices is called Stiefel Manifold $V_K(\mathbb{R}^d) = \{A \in \mathbb{R}^{d \times K} : A^T A = I_K\}$, and we perform the optimization on the Stiefel Manifold following Tagare (2011). The details of the optimization is provided in the appendix, and the procedure for estimating max K-SWD and A is shown in Algorithm 1.

3.2. Proposed SINF Algorithm

The proposed SINF algorithm is based on iteratively minimizing the max K-SWD between the two distributions p_1 and p_2. Specifically, SINF iteratively solves for the orthogonal axes where the marginals of p_1 and p_2 are most different (defined by max K-SWD), and then match the 1D marginalized distribution of p_1 to p_2 on those axes. This is motivated by the inverse Radon Transform (Equation 3) and Cramér-Wold theorem, which suggest that matching the high dimensional distributions is equivalent to matching the 1D slices on all possible directions, decomposing the high dimensional problem into a series of 1D problems. See Figure 1 for an illustration of the SINF algorithm. Given a set

---

Algorithm 1 max K-SWD

Input: $\{x_i \sim p_1\}_{i=1}^N$, $\{y_i \sim p_2\}_{i=1}^N$, K, order p, max iteration J_{maxiter}
Randomly initialize $A \in V_K(\mathbb{R}^d)$
for $j = 1$ **to** J_{maxiter} **do**
    Initialize $D = 0$
    for $k = 1$ **to** K **do**
        $\theta_k = A[:, k]$
        Compute $\hat{x}_i = \theta_k \cdot x_i$ and $\hat{y}_i = \theta_k \cdot y_i$ for each i
        Sort $\hat{x}_i$ and $\hat{x}_j$ in ascending order s.t. $\hat{x}_{i[n]} \leq \hat{x}_{i[n+1]}$ and $\hat{y}_{j[n]} \leq \hat{y}_{j[n+1]}$
        $D = D + \frac{1}{KN} \sum_{i=1}^N |\hat{x}_{i[n]} - \hat{y}_{j[n]}|^p$
    end for
    $G = [-\frac{\partial D}{\partial A_{i,j}}]$, $U = [G, A]$, $V = [A, -G]$
    Determine learning rate τ with backtracking line search
    $A = A - \tau U (I_{2K} + \frac{\tau}{2} V^T U)^{-1} V^T A$
    if A has converged **then**
        Early stop
    end if
end for
Output: $D^{\frac{1}{p}} \approx \max\text{-}K\text{-}SW_p$, $A \approx [\theta_1, \cdots, \theta_K]$

---

of i.i.d. samples X drawn from p_1, in each iteration, a set of 1D marginal transformations $\{\Psi_k\}_{k=1}^K$ ($K \leq d$ where d is the dimensionality of the dataset) are applied to the samples on orthogonal axes $\{\theta_k\}_{k=1}^K$ to match the 1D marginalized PDF of p_2 along those axes. Let $A = [\theta_1, \cdots, \theta_K]$ be the matrix derived from max K-SWD optimization (algorithm 1), that contains K orthogonal axes($A^T A = I_K$). Then the transformation at iteration l of samples X_l can be written as [1]

$$X_{l+1} = A_l \Psi_l(A_l^T X_l) + X_l^\perp, \quad (8)$$

where $X_l^\perp = X_l - A_l A_l^T X_l$ contains the components that are perpendicular to $\theta_1, ..., \theta_K$ and is unchanged in iteration l. $\Psi_l = [\Psi_{l1}, \cdots, \Psi_{lK}]^T$ is the marginal mapping of each dimension of $A_l^T X_l$, and its components are required to be monotonic and differentiable. The transformation of Equation 8 can be easily inverted:

$$X_l = A_l \Psi_l^{-1}(A_l^T X_{l+1}) + X_l^\perp, \quad (9)$$

where $X_l^\perp = X_l - A_l A_l^T X_l = X_{l+1} - A_l A_l^T X_{l+1}$. The Jacobian determinant of the transformation is also efficient to calculate (see appendix for the proof):

$$\det(\frac{\partial X_{l+1}}{\partial X_l}) = \prod_{k=1}^K \frac{d\Psi_{lk}(x)}{dx}. \quad (10)$$

---

[1] **Notation definition**: In this paper we use l, k, j and m to represent different iterations of the algorithm, different axes θ_k, different gradient descent iterations of max K-SWD calculation (see Algorithm 1), and different knots in the spline functions of 1D transformation, respectively.
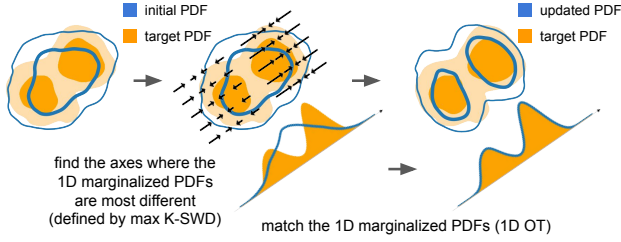
Figure 1. Illustration of 1 iteration of SINF algorithm with $K = 1$.

At iteration l, the SINF objective can be written as:

$$\mathcal{F}_l = \min_{\{\Psi_{l1}, \cdots, \Psi_{lK}\}} \max_{\{\theta_{l1}, \cdots, \theta_{lK}\} \text{ orthonormal}}$$

$$\left(\frac{1}{K} \sum_{k=1}^{K} W_p^p (\Psi_{lk}((\mathcal{R}p_{1,l})(\cdot, \theta_{lk})), (\mathcal{R}p_2)(\cdot, \theta_{lk})) \right)^{\frac{1}{p}} . \quad (11)$$

The algorithm first optimizes θ_{lk} to maximize the objective, with Ψ_{lk} fixed to identical transformations (equivalent to Equation 7). Then the axes θ_{lk} are fixed and the objective is minimized with marginal matching Ψ_l. The samples are updated, and the process is repeated until convergence.

Let $p_{1,l}$ be the transformed p_1 at iteration l. The kth component of Ψ_l, $\Psi_{l,k}$, maps the 1D marginalized PDF of $p_{1,l}$ to p_2 and has an OT solution:

$$\Psi_{l,k}(x) = F_k^{-1}(G_{l,k}(x)), \quad (12)$$

where $G_{l,k}(x) = \int_{-\infty}^{x} (\mathcal{R}p_{1,l})(t, \theta_k)dt$ and $F_k(x) = \int_{-\infty}^{x} (\mathcal{R}p_2)(t, \theta_k)dt$ are the CDFs of $p_{1,l}$ and p_2 on axis θ_k, respectively. The CDFs can be estimated using the quantiles of the samples (in SIG Section 3.3), or using Kernel Density Estimation (KDE, in GIS Section 3.4). Equation 12 is monotonic and invertible. We choose to parametrize it with monotonic rational quadratic splines (Gregory & Delbourgo, 1982; Durkan et al., 2019), which are continuously-differentiable and allows analytic inverse. More details about the spline procedure are given in the appendix. We summarize SINF in Algorithm 2.

The proposed algorithm iteratively minimizes the max K-SWD between the transformed p_1 and p_2. The orthonormal vectors $\{\theta_1, \cdots, \theta_K\}$ specify K axes along which the marginalized PDF between $p_{1,l}$ and p_2 are most different, thus maximizing the gain at each iteration and improving the efficiency of the algorithm. In the appendix we show empirically that the model is able to converge with two orders of magnitude fewer iterations than random axes, and it also leads to better sample quality. This is because as the dimensionality d grows, the number of slices $(\mathcal{R}p)(\cdot, \theta)$ required to approximate $p(x)$ using inverse Radon formula scales as L^{d-1} (Kolouri et al., 2015), where L is the number of slices needed to approximate a similar smooth 2D distribution. Therefore, if θ are randomly chosen, it takes a

---

Algorithm 2 Sliced Iterative Normalizing Flow

Input: $\{x_i \sim p_1\}_{i=1}^N$, $\{y_i \sim p_2\}_{i=1}^N$, K, number of iteration L_{iter}
for $l = 1$ **to** L_{iter} **do**
   $A_l = \max \text{ K-SWD}(x_i, y_i, K)$
   for $k = 1$ **to** K **do**
      $\theta_k = A_l[:, k]$
      Compute $\hat{x}_i = \theta_k \cdot x_i$ and $\hat{y}_i = \theta_k \cdot y_i$ for each i
      $\tilde{x}_m = \text{quantiles}(\text{PDF}(\hat{x}_i))$
      $\tilde{y}_m = \text{quantiles}(\text{PDF}(\hat{y}_i))$
      $\psi_{l,k} = \text{RationalQuadraticSpline}(\tilde{x}_m, \tilde{y}_m)$
   end for
   $\Psi_l = [\Psi_{l1}, \cdots, \Psi_{lK}]$
   Update $x_i = x_i - A_l A_l^T x_i + A_l \Psi_l(A_l^T x_i)$
end for

---

large number of iterations to converge in high dimensions due to the curse of dimensionality. Our objective function reduces the curse of dimensionality in high dimensions by identifying the most relevant directions first.

K is a free hyperparameter in our model. In the appendix we show empirically that the convergence of the algorithm is insensitive to the choice of K, and mostly depends on the total number of 1D transformations $L_{\text{iter}} \times K$.

Unlike KL-divergence, which is invariant under the flow transformations, max K-SWD is different in data space and in latent space. Therefore the direction of building the flow model is of key importance. In the next two sections we discuss two different ways of building the flow, which are good at sample generation and density estimation, respectively.

3.3. Sliced Iterative Generator (SIG)

For Sliced Iterative Generator (SIG) p_1 is a standard Normal distribution, and p_2 is the target distribution. The model iteratively maps the Normal distribution to the target distribution using 1D slice transformations. SIG directly minimizes the max K-SWD between the generated distribution and the target distribution, and is able to generate high quality samples. The properties of SIG are summarized in Table 1.

Specifically, one first draws a set of samples from the standard Normal distribution, and then iteratively updates the samples following Equation 8. Note that in the NF framework, Equation 8 is the inverse of transformation f_l in Equation 1. The Ψ transformation and the weight matrix A are learned using Equation 12 and Algorithm 1. In Equation 12 we estimate the CDFs using the quantiles of the samples.

3.4. Gaussianizing Iterative Slicing (GIS)

For Gaussianizing Iterative Slicing (GIS) p_1 is the target distribution and p_2 is a standard Normal distribution. The

Table 1. Comparison between SIG and GIS

| Model	SIG	GIS
Initial PDF p_1	Gaussian	p_{data}
Final PDF p_2	p_{data}	Gaussian
Training	Iteratively maps Gaussian to p_{data}	Iteratively maps p_{data} to Gaussian
NF structure	Yes	Yes
Advantage	Good samples	Good density estimation

Figure 2. Illustration of the patch-based approach with $S = 4$, $p = 2$ and $q = 2$. At each iteration, different patches are modeled separately. The patches are randomly shifted in each iteration assuming periodic boundaries.

model iteratively gaussianizes the target distribution, and the mapping is learned in the reverse direction of SIG. In GIS the max K-SWD between latent data and the Normal distribution is minimized, thus the model performs well in density estimation, even though its learning objective is not $\log p$. The comparison between SIG and GIS is shown in Table 1.

We add regularization to GIS for density estimation tasks to further improve the performance and reduce overfitting. The regularization is added in the following two aspects:
1) The weight matrix A_l is regularized by limiting the maximum number of iterations J_{maxiter} (see Algorithm 1). We set $J_{\text{maxiter}} = N/d$. Thus for very small datasets ($N/d \to 1$) the axes of marginal transformation are almost random. This has no effect on datasets of regular size.
2) The CDFs in Equation 12 are estimated using KDE, and the 1D marginal transformation is regularized with:

$$\tilde{\psi}_{l,k}(x) = (1-\alpha)\psi_{l,k}(x) + \alpha x, \tag{13}$$

where $\alpha \in [0, 1)$ is the regularization parameter, and $\tilde{\psi}_{l,k}$ is the regularized transformation. In the appendix we show that as α increases, the performance improves, but more iterations are needed to converge. Thus α controls the trade-off between performance and speed.

3.5. Patch-Based Hierarchical Approach

Generally speaking, the neighboring pixels in images have stronger correlations than pixels that are far apart. This fact has been taken advantage by convolutional neural networks, which outperform Fully Connected Neural Networks (FCNNs) and have become standard building blocks in computer vision tasks. Like FCNNs, vanilla SIG and GIS make no assumption about the structure of the data and cannot model high dimensional images very well. Meng et al. (2020) propose a patch-based approach, which decomposes an $S \times S$ image into $p \times p$ patches, with $q \times q$ neighboring pixels in each patch ($S = pq$). In each iteration the marginalized distribution of each patch is modeled separately without considering the correlations between different patches. This approach effectively reduces the dimensionality from S^2
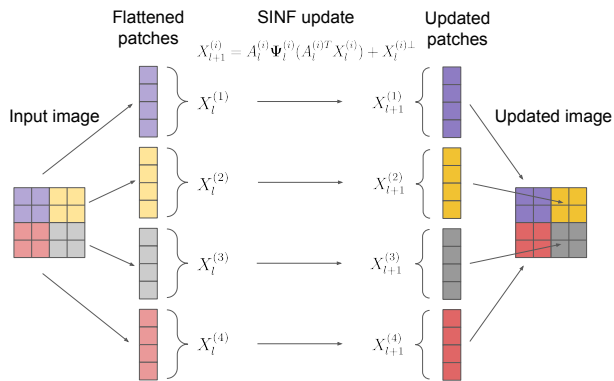
to q^2, at the cost of ignoring the long range correlations. Figure 2 shows an illustration of the patch-based approach.

To reduce the effects of ignoring the long range correlations, we propose a hierarchical model. In SIG, we start from modeling the entire images, which corresponds to $q = S$ and $p = 1$. After some iterations the samples show correct structures, indicating the long range correlations have been modeled well. We then gradually decrease the patch size q until $q = 2$, which allows us to gradually focus on the smaller scales. Assuming a periodic boundary condition, we let the patches randomly shift in each iteration. If the patch size q does not divide S, we set $p = \lfloor S/q \rfloor$ and the rest of the pixels are kept unchanged.

4. Related Work

Iterative normalizing flow models called RBIG (Chen & Gopinath, 2000; Laparra et al., 2011) are simplified versions of GIS, as they are based on a succession of rotations followed by 1D marginal Gaussianizations. Iterative Distribution Transfer (IDT) (Pitié et al., 2007) is a similar algorithm but does not require the base distribution to be a Gaussian. These models do not scale well to high dimensions because they do not have a good way of choosing the axes (slice directions), and they are not competitive against modern NFs trained on $p(x)$ (Meng et al., 2020). Meng et al. (2019) use a similar algorithm called Projection Pursuit Monge Map (PPMM) to construct OT maps. They propose to find the most informative axis using Projection Pursuit (PP) (Freedman, 1985) at each iteration, and show that PPMM works well in low-dimensional bottleneck settings ($d = 8$). PP scales as $\mathcal{O}(d^3)$, which makes PPMM scaling to high dimensions prohibitive. A DL, non-iterative version of these models is Gaussianization Flow (GF) (Meng et al., 2020), which trains on $p(x)$ and achieves good density estimation

results in low dimensions, but does not have good sampling properties in high dimensions. RBIG, GIS and GF have similar architectures but are trained differently. We compare their density estimation results in Section 5.1.

Another iterative generative model is Sliced Wasserstein Flow (SWF) (Liutkus et al., 2019). Similar to SIG, SWF tries to minimize the SWD between the distributions of samples and the data, and transforms this problem into solving a d dimensional PDE. The PDE is solved iteratively by doing a gradient flow in the Wasserstein space, and works well for low dimensional bottleneck features. However, in each iteration the algorithm requires evaluating an integral over the d dimensional unit sphere approximated with Monte Carlo integration, which does not scale well to high dimensions. Another difference with SIG is that SWF does not have a flow structure, cannot be inverted, and does not provide the likelihood. We compare the sample qualities between SWF and SIG in Section 5.2.

SWD, max SWD and other slice-based distance (e.g. Cramér-Wold distance) have been widely used in training generative models (Deshpande et al., 2018; 2019; Wu et al., 2019; Kolouri et al., 2018; Knop et al., 2018; Nguyen et al., 2020b;a; Nadjahi et al., 2020). Wu et al. (2019) propose a differentiable SWD block composed of a rotation followed by marginalized Gaussianizations, but unlike RBIG, the rotation matrix is trained in an end-to-end DL fashion. They propose Sliced Wasserstein AutoEncoder (SWAE) by adding SWD blocks to an AE to regularize the latent variables, and show that its sample quality outperforms VAE and AE + RBIG. Nguyen et al. (2020b;a) generalize the max-sliced approach using parametrized distributions over projection axes. Nguyen et al. (2020b) propose Mixture Spherical Sliced Fused Gromov Wasserstein (MSSFG), which samples the slice axes around a few informative directions following Von Mises-Fisher distribution. They apply MSSFG to training of Deterministic Relational regularized AutoEncoder (DRAE) and name it mixture spherical DRAE (ms-DRAE). Nguyen et al. (2020a) go further and propose Distributional Sliced Wasserstein distance (DSW), which tries to find the optimal axes distribution by parametrizing it with a neural network. They apply DSW to the training of GANs, and we will refer to their model as DSWGAN in this paper. We compare the sample qualities between SIG, SWAE, ms-DRAE, DSWGAN and other similar models in Section 5.2.

Grover et al. (2018) propose Flow-GAN using a NF as the generator of a GAN, so the model can perform likelihood evaluation, and allows both maximum likelihood and adversarial training. Similar to our work they find that adversarial training gives good samples but poor $p(x)$, while training by maximum likelihood results in bad samples. Similar to SIG, the adversarial version of Flow-GAN minimizes the
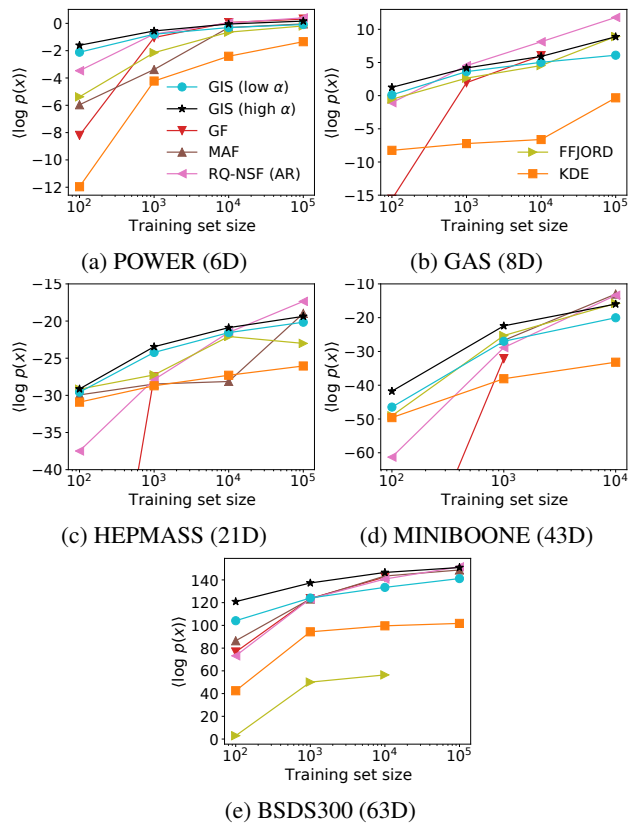

Figure 3. Density estimation on small training sets. The legends in panel (a) and (b) apply to other panels as well. At 100 training data GIS has the best performance in all cases.

Wasserstein distance between samples and data, and has a NF structure. We compare their samples in Section 5.2.

5. Experiments

5.1. Density Estimation $p(x)$ of Tabular Datasets

We perform density estimation with GIS on four UCI datasets (Lichman et al., 2013) and BSDS300 (Martin et al., 2001), as well as image datasets MNIST (LeCun et al., 1998) and Fashion-MNIST (Xiao et al., 2017). The data preprocessing of UCI datasets and BSDS300 follows Papamakarios et al. (2017). In Table 2 we compare our results with RBIG (Laparra et al., 2011) and GF (Meng et al., 2020). The former can be seen as GIS with random axes to apply 1D gaussianization, while the latter can be seen as training non-iterative GIS with MLE training on $p(x)$. We also list other NF models Real NVP (Dinh et al., 2017), Glow (Kingma & Dhariwal, 2018), FFJORD (Grathwohl et al., 2019), MAF (Papamakarios et al., 2017) and RQ-NSF (AR)(Durkan et al., 2019) for comparison.

We observe that RBIG performs significantly worse than current SOTA. GIS outperforms RBIG and is the first iterative algorithm that achieves comparable performance compared

Table 2. Negative test log-likelihood for tabular datasets measured in nats, and image datasets measured in bits/dim (lower is better).

	Method	POWER	GAS	HEPMASS	MINIBOONE	BSDS300	MNIST	Fashion
iterative	RBIG	1.02	0.05	24.59	25.41	-115.96	1.71	4.46
	GIS (this work)	-0.32	-10.30	19.00	14.26	-155.75	1.34	3.22
maximum likelihood	GF	-0.57	-10.13	17.59	10.32	-152.82	1.29	3.35
	Real NVP	-0.17	-8.33	18.71	13.55	-153.28	1.06	2.85
	Glow	-0.17	-8.15	18.92	11.35	-155.07	1.05	2.95
	FFJORD	-0.46	-8.59	14.92	10.43	-157.40	0.99	-
	MAF	-0.30	-10.08	17.39	11.68	-156.36	1.89	-
	RQ-NSF (AR)	-0.66	-13.09	14.01	9.22	-157.31	-	-

Table 3. Averaged training time of different NF models on small datasets ($N_{\text{train}} = 100$) measured in seconds. All the models are tested on both a cpu and a K80 gpu, and the faster results are reported here (the results with * are run on gpus.). P: POWER, G: GAS, H: HEPMASS, M: MINIBOONE, B: BSDS300.

Method	P	G	H	M	B
GIS (low α)	0.53	1.0	0.63	3.5	7.4
GIS (high α)	6.8	9.4	7.3	44.1	69.1
GF	113*	539*	360*	375*	122*
MAF	18.4	-[1]	10.2	-[1]	32.1
FFJORD	1051	1622	1596	499*	4548*
RQ-NSF (AR)	118	127	55.5	38.9	391

[1] Training failures.

to maximum likelihood models. This is even more impressive given that GIS is not trained on $p(x)$, yet it outperforms GF on $p(x)$ on GAS, BSDS300 and Fashion-MNIST.

The transformation at each iteration of GIS is well defined, and the algorithm is very stable even for small training sets. To test the stability and performance we compare the density estimation results with other methods varying the size of the training set N (from 10^2 to 10^5). For GIS we consider two hyperparameter settings: large regularization α (Equation 13) for better $\log p$ performance, and small regularization α for faster training. For other NFs we use settings recommended by their original paper, and set the batch size to $\min(N/10, N_{\text{batch}})$, where N_{batch} is the batch size suggested by the original paper. All the models are trained until the validation $\log p_{\text{val}}$ stops improving, and for KDE the kernel width is chosen to maximize $\log p_{\text{val}}$. Some non-GIS NF models diverged during training or used more memory than our GPU, and are not shown in the plot. The results in Figure 3 show that GIS is more stable compared to other NFs and outperforms them on small training sets. This highlights that GIS is less sensitive to hyper-parameter optimization and achieves good performance out of the box. GIS training time varies with data size, but is generally

lower than other NFs for small training sets. We report the training time for 100 training data in Table 3. GIS with small regularization α requires significantly less time than other NFs, while still outperforming them at 100 training size.

5.2. Generative Modeling of Images

We evaluate SIG as a generative model of images using the following 4 datasets: MNIST, Fashion-MNIST, CIFAR-10 (Krizhevsky et al., 2009) and Celeb-A (cropped and interpolated to 64×64 resolution) (Liu et al., 2015). In Figure 5 we show samples of these four datasets. For MNIST, Fashion-MNIST and CelebA dataset we show samples from the model with reduced temperature $T = 0.85$ (i.e., sampling from a Gaussian distribution with standard deviation $T = 0.85$ in latent space), which slightly improves the sample quality (Parmar et al., 2018; Kingma & Dhariwal, 2018). We report the final FID score (calculated using temperature T=1) in Table 4, where we compare our results with similar algorithms SWF and Flow-Gan (ADV). We also list the FID scores of some other generative models for comparison, including models using slice-based distance SWAEs (two different models with the same name) (Wu et al., 2019; Kolouri et al., 2018), Cramer-Wold AutoEncoder (CWAE) (Knop et al., 2018), ms-DRAE (Nguyen et al., 2020b) and DSWGAN (Nguyen et al., 2020a), Wasserstein GAN models (Arjovsky et al., 2017; Gulrajani et al., 2017), and other GANs and AE-based models Probablistic AutoEncoder (PAE) (Böhm & Seljak, 2020) and two-stage VAE (Dai & Wipf, 2019; Xiao et al., 2019). The scores of WGAN and WGAN-GP models are taken from Lucic et al. (2018), who performed a large-scale testing protocol over different GAN models. The "Best default GAN" is extracted from Figure 4 of Lucic et al. (2018), indicating the lowest FID scores from different GAN models with the hyperparameters suggested by original authors. Vanilla VAEs generally do not perform as well as two stage VAE, and thus are not shown in the table. NF models usually do not report FID scores. PAE combines AEs with NFs and we

Table 4. FID scores on different datasets (lower is better). The errors are generally smaller than the differences.

|	Method	MNIST	Fashion	CIFAR-10	CelebA
iterative	SWF	225.1	207.6	-	-
	SIG ($T = 1$) (this work)	**4.5**	**13.7**	66.5	37.3
adversarial training	Flow-GAN (ADV)	155.6	216.9	71.1	-
	DSWGAN	-	-	56.4	66.9
	WGAN	6.7	21.5	**55.2**	41.3
	WGAN GP	20.3	24.5	55.8	**30.0**
	Best default GAN	~ 10	~ 32	~ 70	~ 48
AE based	SWAE(Wu et al., 2019)	-	-	107.9	48.9
	SWAE(Kolouri et al., 2018)	29.8	74.3	141.9	53.9
	CWAE	23.6	57.1	120.0	49.7
	ms-DRAE	43.6	-	-	46.0
	PAE	-	28.0	-	49.2
	two-stage VAE	12.6	29.3	96.1	44.4
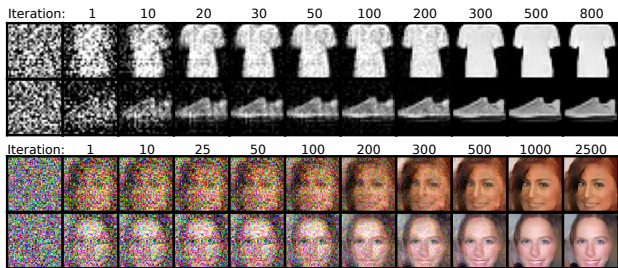
Figure 4. Gaussian noise (first column), Fashion-MNIST (top panel) and CelebA (bottom) samples at different iterations.

expect it to outperform most NF models in terms of sample quality due to the use of AEs. We notice that previous iterative algorithms are unable to produce good samples on high dimensional image datasets (see Table 4 and Figure 7 for SWF samples; see Figure 6 and 7 of Meng et al. (2020) for RBIG samples). However, SIG obtains the best FID scores on MNIST and Fashion-MNIST, while on CIFAR-10 and CelebA it also outperforms similar algorithms and AE-based models, and gets comparable results to GANs. In Figure 4 we show samples at different iterations. In Figure 6 we display interpolations between SIG samples, and the nearest training data, to verify we are not memorizing the training data.

5.3. Improving the Samples of Other Generative Models

Since SIG is able to transform any base distribution to the target distribution, it can also be used as a "Plug-and-Play" tool to improve the samples of other generative models. To demonstrate this, we train SWF, Flow-GAN(ADV) and MAF(5) on Fashion-MNIST with the default architectures in their papers, and then we apply 240 SIG iterations (30% of
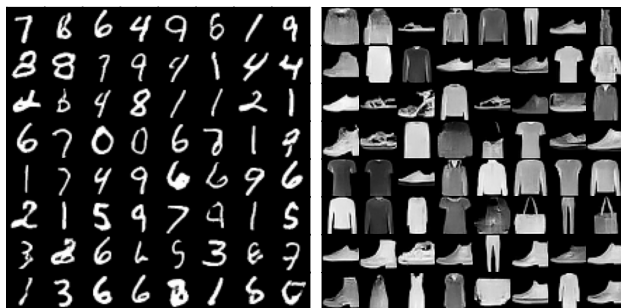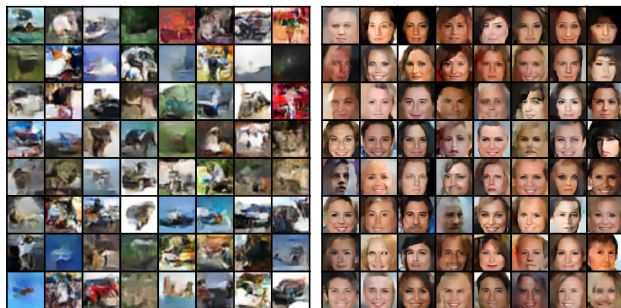

(a) MNIST (T=0.85) (b) Fashion-MNIST (T=0.85)

(c) CIFAR-10 (T=1) (d) CelebA (T=0.85)

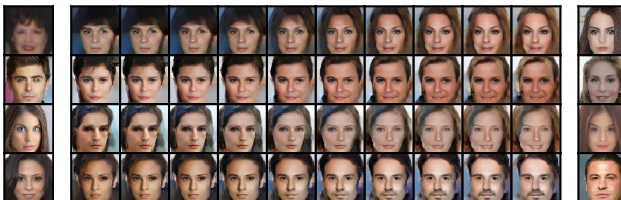*Figure 5.* Random samples from SIG.

Figure 6. Middle: interpolations between CelebA samples from SIG. Left and right: the corresponding nearest training data.
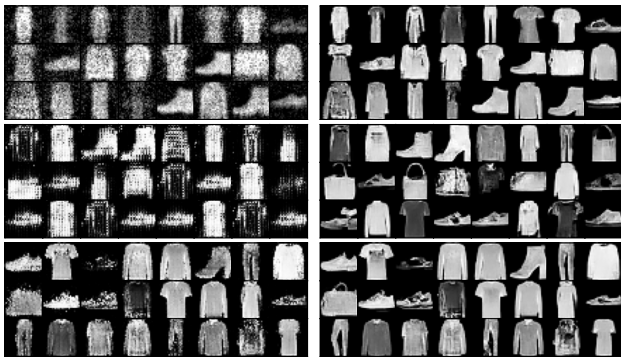
Figure 7. Fashion-MNIST samples before (left panel) and after SIG improvement (right panel). Top: SWF. Middle: Flow-GAN (ADV). Bottom: MAF.

the total number of iterations in Section 5.2) to improve the sample quality. In Figure 7 we compare the samples before and after SIG improvement. Their FID scores improve from 207.6, 216.9 and 81.2 to 23.9, 21.2 and 16.6, respectively. These results can be further improved by adding more SIG iterations.

5.4. Out of Distribution (OoD) Detection

Table 5. OoD detection accuracy quantified by the AUROC of data $p(x)$ trained on Fashion-MNIST.

Method	MNIST	OMNIGLOT
SIG (this work)	**0.980**	**0.993**
GIS (this work)	0.824	0.891
PixelCNN++	0.089	-
IWAE	0.423	0.568

OoD detection with generative models has recently attracted a lot of attention, since the $\log p$ estimates of NF and VAE have been shown to be poor OoD detectors: different generative models can assign higher probabilities to OoD data than to In Distribution (InD) training data (Nalisnick et al., 2019). One combination of datasets for which this has been observed is Fashion-MNIST and MNIST, where a model trained on the former assigns higher density to the latter.

SINF does not train on the likelihood $p(x)$, which is an advantage for OoD. Likelihood is sensitive to the smallest variance directions (Ren et al., 2019): for example, a zero variance pixel leads to an infinite $p(x)$, and noise must be added to regularize it. But zero variance directions contain little or no information on the global structure of the image. SINF objective is more sensitive to the meaningful global structures that can separate between OoD and InD. Because the patch based approach ignores the long range correlations and results in poor OoD, we use vanilla

SINF without patch based approach. We train the models on F-MNIST, and then evaluate anomaly detection on test data of MNIST and OMNIGLOT (Lake et al., 2015). In Table 5 we compare our results to maximum likelihood $p(x)$ models PixelCNN++(Salimans et al., 2017; Ren et al., 2019), and IWAE (Choi et al., 2018). Other models that perform well include VIB and WAIC (Choi et al., 2018), which achieve 0.941, 0.943 and 0.766, 0.796, for MNIST and OMNIGLOT, respectively (below our SIG results). For the MNIST case Ren et al. (2019) obtained 0.996 using the likelihood ratio between the model and its perturbed version, but they require fine-tuning on some additional OoD dataset, which may not be available in OoD applications. Lower dimensional latent space PAE (Böhm & Seljak, 2020) achieves 0.997 and 0.981 for MNIST and OMNIGLOT, respectively, while VAE based likelihood regret (Xiao et al., 2020) achieves 0.988 on MNIST, but requires additional (expensive) processing.

6. Conclusions

We introduce sliced iterative normalizig flow (SINF) that uses Sliced Optimal Transport to iteratively transform data distribution to a Gaussian (GIS) or the other way around (SIG). To the best of our knowledge, SIG is the first greedy deep learning algorithm that is competitive with the SOTA generators in high dimensions, while GIS achieves comparable results on density estimation with current NF models, but is more stable, faster to train, and achieves higher $p(x)$ when trained on small training sets, even though it does not train on $p(x)$. It also achieves better OoD performance. SINF is very stable to train, has very few hyperparameters, and is very insensitive to their choice (see appendix). SINF has deep neural network architecture, but its approach deviates significantly from the current DL paradigm, as it does not use concepts such as mini-batching, stochastic gradient descent and gradient back-propagation through deep layers. SINF is an existence proof that greedy DL without these ingredients can be state of the art for modern high dimensional ML applications. Such approaches thus deserve more detailed investigations that may have an impact on the theory and practice of DL.

Acknowledgements

References

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

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Böhm, V. and Seljak, U. Probabilistic auto-encoder. *arXiv preprint arXiv:2006.05479*, 2020.

Chen, S. S. and Gopinath, R. A. Gaussianization. In Leen, T. K., Dietterich, T. G., and Tresp, V. (eds.), *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pp. 423–429. MIT Press, 2000.

Choi, H., Jang, E., and Alemi, A. A. Waic, but why? generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018.

Cramér, H. and Wold, H. Some theorems on distribution functions. *Journal of the London Mathematical Society*, 1(4):290–294, 1936.

Dai, B. and Wipf, D. P. Diagnosing and enhancing VAE models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=B1e0X3C9tQ.

Deshpande, I., Zhang, Z., and Schwing, A. G. Generative modeling using the sliced wasserstein distance. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pp. 3483–3491. IEEE Computer Society, 2018. doi: 10.1109/CVPR.2018.00367. URL http://openaccess.thecvf.com/content_cvpr_2018/html/Deshpande_Generative_Modeling_Using_CVPR_2018_paper.html.

Deshpande, I., Hu, Y., Sun, R., Pyrros, A., Siddiqui, N., Koyejo, S., Zhao, Z., Forsyth, D. A., and Schwing, A. G. Max-sliced wasserstein distance and its use for gans. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 10648–10656. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.01090. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Deshpande_Max-Sliced_Wasserstein_Distance_and_Its_Use_for_GANs_CVPR_2019_paper.html.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

Durkan, C., Bekasov, A., Murray, I., and Papamakarios, G. Neural spline flows. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 7509–7520, 2019.

Freedman, J. Exploratory projection pursuit. *Journsl of American Statistical Association*, 82:397, 1985.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2672–2680, 2014.

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., and Duvenaud, D. FFJORD: free-form continuous dynamics for scalable reversible generative models. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJxgknCcK7.

Gregory, J. and Delbourgo, R. Piecewise rational quadratic interpolation to monotonic data. *IMA Journal of Numerical Analysis*, 2(2):123–130, 1982.

Grover, A., Dhar, M., and Ermon, S. Flow-gan: Combining maximum likelihood and adversarial learning in generative models. In McIlraith, S. A. and Weinberger, K. Q. (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 3069–3076. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17409.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5767–5777, 2017.

Helgason, S. *Integral geometry and Radon transforms*. Springer Science & Business Media, 2010.

Huszár, F. How (not) to train your generative model: Scheduled sampling, likelihood, adversary? *arXiv preprint arXiv:1511.05101*, 2015.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=Hk99zCeAb.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 4401–4410. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00453. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Karras_A_Style-Based_Generator_Architecture_for_Generative_Adversarial_Networks_CVPR_2019_paper.html.

Kingma, D. P. and Dhariwal, P. Glow: Generative flow with invertible 1x1 convolutions. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 10236–10245, 2018.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y. (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Knop, S., Tabor, J., Spurek, P., Podolak, I., Mazur, M., and Jastrzębski, S. Cramer-wold autoencoder. *arXiv preprint arXiv:1805.09235*, 2018.

Kolouri, S., Park, S. R., and Rohde, G. K. The radon cumulative distribution transform and its application to image classification. *IEEE transactions on image processing*, 25(2):920–934, 2015.

Kolouri, S., Pope, P. E., Martin, C. E., and Rohde, G. K. Sliced-wasserstein autoencoder: An embarrassingly simple generative model. *arXiv preprint arXiv:1804.01947*, 2018.

Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., and Rohde, G. K. Generalized sliced wasserstein distances. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 261–272, 2019.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Laparra, V., Camps-Valls, G., and Malo, J. Iterative gaussianization: from ica to random rotations. *IEEE transactions on neural networks*, 22(4):537–549, 2011.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lichman, M. et al. Uci machine learning repository, 2013.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pp. 3730–3738. IEEE Computer Society, 2015. doi: 10.1109/ICCV.2015.425. URL https://doi.org/10.1109/ICCV.2015.425.

Liutkus, A., Simsekli, U., Majewski, S., Durmus, A., and Stöter, F. Sliced-wasserstein flows: Nonparametric generative modeling via optimal transport and diffusions. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4104–4113. PMLR, 2019. URL http://proceedings.mlr.press/v97/liutkus19a.html.

Lucic, M., Kurach, K., Michalski, M., Gelly, S., and Bousquet, O. Are gans created equal? A large-scale study. In Bengio, S., Wallach, H. M., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in*

Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada, pp. 698–707, 2018.

Martin, D., Fowlkes, C., Tal, D., and Malik, J. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, pp. 416–423. IEEE, 2001.

Meng, C., Ke, Y., Zhang, J., Zhang, M., Zhong, W., and Ma, P. Large-scale optimal transport map estimation using projection pursuit. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 8116–8127, 2019.

Meng, C., Song, Y., Song, J., and Ermon, S. Gaussianization flows. In Chiappa, S. and Calandra, R. (eds.), *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy]*, volume 108 of *Proceedings of Machine Learning Research*, pp. 4336–4345. PMLR, 2020. URL http://proceedings.mlr.press/v108/meng20b.html.

Nadjahi, K., Durmus, A., Chizat, L., Kolouri, S., Shahrampour, S., and Simsekli, U. Statistical and topological properties of sliced probability divergences. *Advances in Neural Information Processing Systems*, 33, 2020.

Nalisnick, E. T., Matsukawa, A., Teh, Y. W., Görür, D., and Lakshminarayanan, B. Do deep generative models know what they don't know? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=H1xwNhCcYm.

Nguyen, K., Ho, N., Pham, T., and Bui, H. Distributional sliced-wasserstein and applications to generative modeling. *arXiv preprint arXiv:2002.07367*, 2020a.

Nguyen, K., Nguyen, S., Ho, N., Pham, T., and Bui, H. Improving relational regularized autoencoders with spherical sliced fused gromov wasserstein. *arXiv preprint arXiv:2010.01787*, 2020b.

Papamakarios, G., Murray, I., and Pavlakou, T. Masked autoregressive flow for density estimation. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual*

Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pp. 2338–2347, 2017.

Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., and Tran, D. Image transformer. In Dy, J. G. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4052–4061. PMLR, 2018. URL http://proceedings.mlr.press/v80/parmar18a.html.

Paty, F. and Cuturi, M. Subspace robust wasserstein distances. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5072–5081. PMLR, 2019. URL http://proceedings.mlr.press/v97/paty19a.html.

Pitié, F., Kokaram, A. C., and Dahyot, R. Automated colour grading using colour distribution transfer. *Computer Vision and Image Understanding*, 107(1-2):123–137, 2007.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. In Bengio, Y. and LeCun, Y. (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.06434.

Ren, J., Liu, P. J., Fertig, E., Snoek, J., Poplin, R., DePristo, M. A., Dillon, J. V., and Lakshminarayanan, B. Likelihood ratios for out-of-distribution detection. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 14680–14691, 2019.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. In Bach, F. R. and Blei, D. M. (eds.), *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 1530–1538. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/rezende15.html.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31th International*

Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014, volume 32 of *JMLR Workshop and Conference Proceedings*, pp. 1278–1286. JMLR.org, 2014. URL `http://proceedings.mlr.press/v32/rezende14.html`.

Rowland, M., Hron, J., Tang, Y., Choromanski, K., Sarlós, T., and Weller, A. Orthogonal estimation of wasserstein distances. In Chaudhuri, K. and Sugiyama, M. (eds.), *The 22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan*, volume 89 of *Proceedings of Machine Learning Research*, pp. 186–195. PMLR, 2019. URL `http://proceedings.mlr.press/v89/rowland19a.html`.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL `https://openreview.net/forum?id=BJrFC6ceg`.

Tagare, H. D. Notes on optimization on stiefel manifolds. In *Technical report, Technical report*. Yale University, 2011.

Theis, L., van den Oord, A., and Bethge, M. A note on the evaluation of generative models. In Bengio, Y. and LeCun, Y. (eds.), *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL `http://arxiv.org/abs/1511.01844`.

Tolstikhin, I. O., Bousquet, O., Gelly, S., and Schölkopf, B. Wasserstein auto-encoders. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=HkL7n1-0b`.

van den Oord, A., Vinyals, O., and Kavukcuoglu, K. Neural discrete representation learning. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6306–6315, 2017.

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

Wiatrak, M. and Albrecht, S. V. Stabilizing generative adversarial network training: A survey. *arXiv preprint arXiv:1910.00927*, 2019.

Wu, J., Huang, Z., Acharya, D., Li, W., Thoma, J., Paudel, D. P., and Gool, L. V. Sliced wasserstein generative models. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 3713–3722. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00383. URL `http://openaccess.thecvf.com/content_CVPR_2019/html/Wu_Sliced_Wasserstein_Generative_Models_CVPR_2019_paper.html`.

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

Xiao, Z., Yan, Q., Chen, Y., and Amit, Y. Generative latent flow: A framework for non-adversarial image generation. *arXiv preprint arXiv:1905.10485*, 2019.

Xiao, Z., Yan, Q., and Amit, Y. Likelihood regret: An out-of-distribution detection score for variational auto-encoder. *arXiv preprint arXiv:2003.02977*, 2020.