Sliced Wasserstein Generative Models

Jiqing Wu*[1] Zhiwu Huang*[1] Dinesh Acharya[1] Wen Li[1]
Janine Thoma[1] Danda Pani Paudel[1] Luc Van Gool[1,2]
[1] Computer Vision Lab, ETH Zurich, Switzerland [2] VISICS, KU Leuven, Belgium
{jwu, zhiwu.huang, liwen, jthoma, paudel, vangool}@vision.ee.ethz.ch
*Equal contribution

Abstract

In generative modeling, the Wasserstein distance (WD) has emerged as a useful metric to measure the discrepancy between generated and real data distributions. Unfortunately, it is challenging to approximate the WD of high-dimensional distributions. In contrast, the sliced Wasserstein distance (SWD) factorizes high-dimensional distributions into their multiple one-dimensional marginal distributions and is thus easier to approximate.

In this paper, we introduce novel approximations of the primal and dual SWD. Instead of using a large number of random projections, as it is done by conventional SWD approximation methods, we propose to approximate SWDs with a small number of parameterized orthogonal projections in an end-to-end deep learning fashion. As concrete applications of our SWD approximations, we design two types of differentiable SWD blocks to equip modern generative frameworks—Auto-Encoders (AE) and Generative Adversarial Networks (GAN).

In the experiments, we not only show the superiority of the proposed generative models on standard image synthesis benchmarks, but also demonstrate the state-of-the-art performance on challenging high resolution image and video generation in an unsupervised manner [1].

1. Introduction

The Wasserstein distance (WD) is an important metric, which was originally applied in the optimal transport problem[2] [33]. Recently, [3, 11, 30, 36, 31, 23, 1, 2] discovered the advantages of the WD in generative models and achieved state-of-the-art performance for image synthesis. However, the WD has some drawbacks. For instance, its primal form is generally intractable for high-dimensional probability distributions, although some works [31, 10, 30] have proposed relaxed versions of the primal form. The dual form of the WD can be more easily derived, but it remains difficult to approximate its k-Lipschitz constraint [36, 37].

Given the weaknesses of the WD, the sliced Wasserstein distance (SWD) suggests itself as a potential alternative. The SWD factorizes high-dimensional distributions into their multiple one-dimensional marginal distributions and can therefore be approximated more easily, as studied in [5, 21, 19, 9]. However, due to the inefficient approximations of the SWD in existing methods, its potential in generative modeling has not been fully explored yet.

In this paper, we address this issue. Our contributions can be summarized as follows:

- Relying on our novel primal SWD approximation, we propose sliced Wasserstein Auto-Encoder (SWAE) models. By seamlessly stacking the proposed primal SWD blocks (layers) on top of the standard encoder, we give the traditional AE generative capabilities. State-of-the-art AE-based generative models usually require an additional regularizer to achieve the same effect.

- Based on our new dual SWD approximation, we introduce a sliced version of Wasserstein Generative Adversarial Networks (SWGAN) by applying the proposed dual SWD blocks to the discriminator. To the best of our knowledge, this is the first work to study the dual SWD and its application to generative models.

- To satisfy the orthogonality constraint required by the projection matrices of the proposed SWD blocks, we apply a non-Euclidean optimization algorithm on Stiefel manifolds to update the projection matrices.

- Motivated by the improvements of visual quality and model stability demonstrated on the standard image synthesis benchmarks, we apply our proposed model to the challenging task of unsupervised high resolution image and video synthesis. These evaluations confirm the advantage of our model under non-trivial cases.

[1] Code: https://github.com/musikisomorphie/swd.git
[2] Optimal transport addresses the problem of finding an optimal plan to transfer a source distribution to a target distribution at minimal cost.

2. Background

2.1. Wasserstein Distance & Related Models

The primal Wasserstein distance (WD) is given by

$$W_p(P_X, P_Y) = \inf_{\gamma \in \Pi(P_X, P_Y)} \mathbb{E}_{(X,Y)\sim\gamma}[d^p(X,Y)]^{\frac{1}{p}}, \quad (1)$$

where X, Y are random variables, $\Pi(P_X, P_Y)$ denotes the set of all joint distributions $\gamma(X, Y)$ whose marginal distributions are P_X, P_Y respectively, d is a metric, and $p > 0$. For $p = 1$, the Kantorovich's dual of the WD is

$$W_1(P_X, P_Y) = \sup_{f \in \text{Lip}^1} \mathbb{E}_{X\sim P_X}[f(X)] - \mathbb{E}_{Y\sim P_Y}[f(Y)], \quad (2)$$

where Lip^1 is the set of all 1-Lipschitz functions. The dual WD becomes $k \cdot W_1$ if we replace Lip^1 with Lip^k for $k > 0$.

The original form of the primal WD (Eq. 1) is generally intractable. For the case of Auto-Encoders (AE), however, [31] have proven that optimizing the primal WD over tractable encoders is equivalent to optimizing it over the intractable joint distributions $\gamma(X, Y)$. This idea yields Wasserstein Auto-Encoders (WAE). Another way of avoiding the intractable primal WD is to use its dual form instead. By parameterizing the f in Eq. 2 with a neural network, [3] have found a natural way of introducing the dual WD to the GAN framework. WAE and WGAN stand for typical applications of the primal and dual WD in generative models and are closely related to our proposed methods. We therefore summarize the necessary details in the following.

2.1.1 Wasserstein AE (WAE)

The WAE proposed by [31] optimizes a relaxed version of the primal WD. In order to impose the prior distribution on the encoder, an additional divergence $\mathcal{D}$ is introduced to the objective:

$$\inf_{P_{Q(Z|X)} \in \mathcal{Q}} \mathbb{E}_{X\sim P_X} \mathbb{E}_{Q\sim P_{Q(Z|X)}}[c(X, G(Z))] + \lambda \mathcal{D}(P_Q, P_Z), \quad (3)$$

where Z is random noise, G is the decoder, $\mathcal{Q}$ is any non-parametric set of marginal distributions P_Q on encoders Q, c is the Euclidean distance, $\lambda > 0$ is a hyperparameter, and $\mathcal{D}$ is the divergence between the P_Q of the encoder and the prior distribution P_Z of Z. [31] instantiate $\mathcal{D}$ by using either maximum mean discrepancy (MMD) or GAN, both of which can be regarded as a distribution matching strategy.

2.1.2 Wasserstein GAN (WGAN)

The key challenge of WGAN is the k-Lipschitz constraint required in Eq. 2. The original WGAN [3] adopt a weight clipping strategy; however, it satisfies the k-Lipschitz constraint poorly. To alleviate this problem, the improved training of Wasserstein GAN (WGAN-GP) [11] penalizes the norm of the discriminator's gradient with respect to a few input samples. This gradient penalty is then added to the basic WGAN loss (i.e. the dual form of the WD) resulting in the following full objective:

$$\min_G \max_D \mathbb{E}_{X\sim P_X}[D(X)] - \mathbb{E}_{\tilde{X}\sim P_G}[D(G(Z))] + \lambda \mathbb{E}_{\hat{X}\sim P_{\hat{X}}}[(\|\nabla_{\hat{X}} D(\hat{X})\|_2 - 1)^2], \quad (4)$$

where G, D denotes the generator and discriminator respectively, Z is random noise, $\hat{X}$ is random samples following the distribution $P_{\hat{X}}$ which is sampled uniformly along straight lines between pairs of points sampled from P_X and P_G, and $\nabla_{\hat{X}} D(\hat{X})$ is the gradient with respect to $\hat{X}$. As studied in [36], a limited number of samples is not sufficient to impose the k-Lipschitz constraint on a high-dimensional domain. Thus, [36] further improved the Wasserstein GAN with an additional consistency term (CTGAN). Furthermore, [26] introduced a spectral normalization (SN) technique which also improves the training of GAN including the family of WGAN. The SNGAN imposes the 1-Lipschitz constraint by normalizing the weights of each layer. To strengthen GAN training stability and to achieve high resolution image generation, [16] applied WGAN-GP to a progressive growing scheme (PG-WGAN).

2.2. Sliced Wasserstein Distance & Related Models

The idea underlying the sliced Wasserstein distance (SWD) is to decompose the challenging estimation of a high-dimensional distribution into the simpler estimation of multiple one-dimensional distributions. Formally, let P_X, P_Y be probability distributions of random variables X, Y. For a unit vector $\boldsymbol{\theta} \in \mathbb{S}^{n-1}$ we define the corresponding inner product $\pi_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{x}$ and marginal distribution $\pi^*_{\boldsymbol{\theta}} P_X = P_X \circ \pi_{\boldsymbol{\theta}}^{-1}$. Then the primal SWD is given by

$$SW_p(P_X, P_Y) = \left(\int_{\mathbb{S}^{n-1}} W_p(\pi^*_{\boldsymbol{\theta}} P_X, \pi^*_{\boldsymbol{\theta}} P_Y)^p d\boldsymbol{\theta} \right)^{\frac{1}{p}}. \quad (5)$$

Several works [5, 21, 19] exploit the fact that the WD has a closed form solution for the optimal transport plan between one-dimensional probability distributions. More concretely, let F_X, F_Y be the cumulative distribution functions (CDFs) corresponding to P_X, P_Y, then for all $\boldsymbol{\theta} \in \mathbb{S}^{n-1}$ there exists a unique closed form solution

$$\tau_{\boldsymbol{\theta}} = (\pi^*_{\boldsymbol{\theta}} F_Y)^{-1} \circ \pi^*_{\boldsymbol{\theta}} F_X, \quad (6)$$

such that the integrand of Eq. 5 can be computed by

$$W_p(\pi^*_{\boldsymbol{\theta}} P_X, \pi^*_{\boldsymbol{\theta}} P_Y)^p = \int_{\mathbb{R}} d^p(x, \tau_{\boldsymbol{\theta}}(x)) d\pi^*_{\boldsymbol{\theta}} P_X. \quad (7)$$

Furthermore, as proven by [6] (Chapter 5), the SWD is not only a valid distance, but also equivalent[3] to the WD

$$SW_p(P_X, P_Y)^p \leq \alpha_1 W_p(P_X, P_Y)^p \leq \alpha_2 SW_p(P_X, P_Y)^{\frac{1}{n+1}}, \tag{8}$$

where α_1, α_2 are constants and n is the dimension of sample vectors from X, Y. Given such favorable properties, the SWD has the potential to improve modern generative modeling especially when processing samples of high-dimensional distributions such as images and videos.

The SWD is typically approximated by using a summation over the projections along random directions (random projections) [28, 21, 16, 9]. For example, [28] iteratively use a large number of random projections to estimate the SWD from samples and update the samples by gradient descent. Similarly, the sliced Wasserstein Generator (SWG) [9] optimizes its generator with an SWD loss. This SWD loss computes the difference between marginal distributions of feature maps decomposed by random projections. Unfortunately, these methods require a large amount of random projections and have not fully unlocked the potential of the SWD yet.

3. Proposed Method

Compared to a set of random vectors (projections), a set of orthogonal projections is more efficient to span an entire space. Also, neural networks have been shown to possess robust generalization abilities. We thus propose to approximate the primal and dual SWD with a small set of parameterized orthogonal matrices in a deep learning fashion. In the following, we give a detailed description of our primal and dual SWD approximations. Later, we introduce two generative modeling applications of the resulting SWD blocks—sliced Wasserstein AE (SWAE) and sliced Wasserstein GAN (SWGAN).

3.1. Primal SWD Approximation

Given $i \in \mathbb{N}$, guided by the target distribution P_Y, we define the i-th computational block which transfers the input distribution P_{X^i} to $P_{X^{i+1}}$ as follows

$$\boldsymbol{Q}^i_{\boldsymbol{\Theta}}(\boldsymbol{x}) = \boldsymbol{O}^i_{\boldsymbol{\Theta}} \boldsymbol{\Pi}^i_{\boldsymbol{\Theta}}((\boldsymbol{O}^i_{\boldsymbol{\Theta}})^T \boldsymbol{x}), \tag{9}$$

where $\boldsymbol{x}$ is a sample vector from P_{X^i}, $\boldsymbol{O}^i_{\boldsymbol{\Theta}} = [\boldsymbol{\theta}^i_1, \ldots, \boldsymbol{\theta}^i_n] \in \mathbb{R}^{n \times n}$ is a random orthogonal matrix, and $\boldsymbol{\Pi}^i_{\boldsymbol{\Theta}} = (\tau^i_{\boldsymbol{\theta}_1}, \tau^i_{\boldsymbol{\theta}_2}, \ldots, \tau^i_{\boldsymbol{\theta}_n})$ (Eq. 6) are optimal transport maps with respect to the marginal distributions of P_{X^i}, P_Y projected by $\boldsymbol{O}^i_{\boldsymbol{\Theta}}$. Stacking m computational blocks $\boldsymbol{Q}^m_{\boldsymbol{\Theta}} \circ \ldots \circ \boldsymbol{Q}^2_{\boldsymbol{\Theta}} \circ \boldsymbol{Q}^1_{\boldsymbol{\Theta}}$ results in the iterative distribution transfer (IDT) method [27]. As studied in [6], let the target distribution P_Y be Gaussian, then P_{X^m} converges to P_Y with respect to the primal SWD, and the convergence holds when $m \to \infty$. To reduce the number of computational blocks (Eq. 9) required by IDT, we propose to parameterize the orthogonal matrices and learn them in an end-to-end deep learning fashion. As a result, a small number of such parameterized computational blocks is sufficient to approximate the primal SWD.

[3] Here, we adopt the usage of 'equivalent' from [6], which is an abuse of notation.

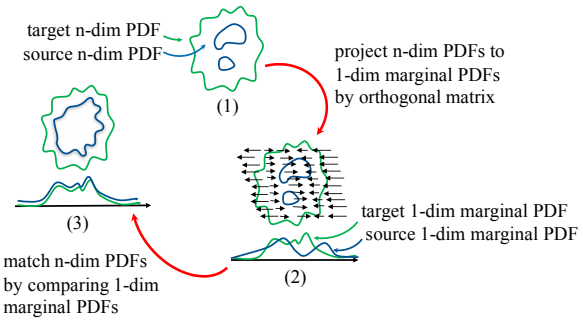


Figure 1. Illustration of our primal and dual SWD approximations. (1) - (2): By projecting samples along orthogonal unit vectors (orthogonal matrix), we decompose n-dimensional target and source probability distribution functions (PDFs) into their one-dimensional marginal PDFs. (2) - (3): We match the n-dimensional PDFs by comparing their marginal PDFs. For the primal SWD approximation, this is done implicitly through the iterative transformation of source to target distribution. For the dual approximation, the dual SWD is calculated explicitly.

3.2. Dual SWD Approximation

Since the integrand of the SWD (Eq. 5) is nothing but a one-dimensional WD, its Kantorovich's dual can be seamlessly applied and Eq. 5 can be rewritten as

$$\int_{\mathbb{S}^{n-1}} \left(\sup_{f \in \mathrm{Lip}^k} \mathbb{E}_{X_{\boldsymbol{\theta}} \sim \pi^*_{\boldsymbol{\theta}} P_X}[f(X_{\boldsymbol{\theta}})] - \mathbb{E}_{Y_{\boldsymbol{\theta}} \sim \pi^*_{\boldsymbol{\theta}} P_Y}[f(Y_{\boldsymbol{\theta}})] \right) d\boldsymbol{\theta}. \tag{10}$$

Similar to the primal SWD approximation, we propose to employ orthogonal matrices to estimate the integral over $\mathbb{S}^{n-1}$. These orthogonal matrices are also parametrized and learned in the context of deep learning. It therefore suffices to use a moderate number of orthogonal matrices to achieve a good estimation. The proposed SWD approximations are conceptually illustrated in Fig. 1.

3.3. Sliced Wasserstein AE (SWAE)

Since AE-based generative models require to impose a prior distribution on the encoder, it is natural to make Eq. 9 learnable and incorporate it into the encoder. By stacking our primal SWD blocks (layers) on top of the standard encoder, we give generative capability to the traditional Auto-encoder. In other words, we can implicitly match the encoder and prior distributions without introducing an extra regularizer such as $\mathcal{D}$ required in Eq. 3. Specifically, our encoder Q is the composition of a standard encoding network E and m primal SWD blocks $S_{p,1}, \ldots, S_{p,m}$, that is, $Q = S_{p,m} \circ \ldots \circ S_{p,1} \circ E$.

Algorithm 1 The proposed primal SWD block

Require: Orthogonal matrix $\boldsymbol{O}_{\boldsymbol{\Theta}} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_r] \in \mathbb{R}^{r\times r}$, batch of latent codes $\boldsymbol{M}_{\boldsymbol{y}} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_b] \in \mathbb{R}^{r\times b}$, batch of Gaussian noise $\boldsymbol{M}_{\boldsymbol{z}} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_b] \in \mathbb{R}^{r\times b}$, and bin number l
Output: Batch of transferred latent codes $M_{\tilde{\boldsymbol{y}}} = [\tilde{\boldsymbol{y}}_1, \ldots, \tilde{\boldsymbol{y}}_b]$
for $i \leftarrow 1, r$ **do**
  $\boldsymbol{y}'_i = \boldsymbol{\theta}_i^T \boldsymbol{M}_{\boldsymbol{y}}, \boldsymbol{z}'_i = \boldsymbol{\theta}_i^T \boldsymbol{M}_{\boldsymbol{z}}$
  $\boldsymbol{y}''_i = \frac{\boldsymbol{y}'_i - \min_j\{y'_{i,j}\}}{\max_j\{y'_{i,j}\} - \min_j\{y'_{i,j}\}}, \boldsymbol{z}''_i = \frac{\boldsymbol{z}'_i - \min_j\{z'_{i,j}\}}{\max_j\{z'_{i,j}\} - \min_j\{z'_{i,j}\}}$,
  $y'_{i,j}, z'_{i,j}$ are the j-th element of $\boldsymbol{y}'_i, \boldsymbol{z}'_i$ respectively.
  Compute soft PDF histogram $p_{y''_i}, p_{z''_i}$ of $\boldsymbol{y}''_i, \boldsymbol{z}''_i$ with l bins
  Compute CDF $F_{y''_i}, F_{z''_i}$ of $p_{y''_i}, p_{z''_i}$
  Compute $F_{y''_i}(\boldsymbol{y}''_i)$ element-wise by linear interpolation
  $\hat{\boldsymbol{y}}_i = (\max_j\{z'_{i,j}\} - \min_j\{z'_{i,j}\})(F_{z''_i})^{-1} F_{y''_i}(\boldsymbol{y}''_i) + \min_j\{z'_{i,j}\}$
end for
Compute $\boldsymbol{M}_{\tilde{\boldsymbol{y}}} = \boldsymbol{O}_{\boldsymbol{\Theta}} \boldsymbol{M}_{\hat{\boldsymbol{y}}}^T, \quad \boldsymbol{M}_{\hat{\boldsymbol{y}}} = [\hat{\boldsymbol{y}}_1^T, \ldots, \hat{\boldsymbol{y}}_r^T]$

Algorithm 2 The proposed SWAE

Require: Primal SWD block number m, batch size b, decoder G and encoder $Q = S_{p,m} \circ \ldots \circ S_{p,2} \circ S_{p,1} \circ E$, training steps h, training hyperparameters, etc.
for $t \leftarrow 1, h$ **do**
  Sample real data $\boldsymbol{M}_{\boldsymbol{x}} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_b]$ from P_X
  Sample Gaussian noise $\boldsymbol{M}_{\boldsymbol{z}} = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_b]$ from $\mathcal{N}(0, 1)$
  Update the weights $\boldsymbol{w}$ of Q and G by descending:
  $\boldsymbol{w} \leftarrow \mathrm{Adam}(\nabla_{\boldsymbol{w}}(\frac{1}{b}\|\boldsymbol{M}_{\boldsymbol{x}} - G(Q(\boldsymbol{M}_{\boldsymbol{x}}, \boldsymbol{M}_{\boldsymbol{z}}))\|_2^2), \boldsymbol{w})$
end for

By feeding the latent codes from E into the primal SWD blocks $S_{p,1}, \ldots, S_{p,m}$, the distribution of latent codes is transferred to the prior distribution. In this paper, we choose the prior distribution to be Gaussian, as it is frequently done for AE-based models. However, supported by [6], more complicated prior distributions are acceptable as well.

The implementation details of our primal SWD block are presented in Alg. 1. The idea behind the algorithm is to decompose Eq. 6 into multiple differentiable computational steps. However, the conventional histogram computation is not differentiable. We therefore propose a soft version of histogram computation to make the PDF histogram computation differentiable. More specifically, for an element y we assign the weight $e^{-\alpha\|y-c_i\|^2} / \sum_{j=1}^{l} e^{-\alpha\|y-c_j\|^2}$ to the i-th bin, where $c_1, \ldots, c_l$ are the bin centers. Eventually, we obtain the histogram by summing the weights for each bin over all elements y . Note that for $\alpha \to \infty$ this soft version returns to the original non-differentiable version. In practice, due to the minor impact of α on the generative capability, we empirically determine $\alpha = 1$. As a result, the primal SWD block (Alg. 1) is differentiable and can be trained in a deep learning manner.

The approximation error of Alg. 1 is dominated by its core steps corresponding to Eq. 6. Since Alg. 1 rescales all sample vectors to $[0, 1]$, the inverse functions of its CDFs are again CDFs. Together with the fact that a CDF can be written as an empirical distribution function (EDF) [8], we obtain the following error estimation for Alg. 1:

Theorem 1. *Given $b \in \mathbb{N}$, let $Z_1, Z_2, \ldots, Z_b$ be real-valued i.i.d. random variables with a continuous CDF F_Z^{-1} with domain $[0, 1]$. Then we define the associated EDF $F_{Z,b}^{-1}(t) = \frac{1}{b}\sum_{i=1}^{b} \mathbf{1}_{\{Z_i \le t\}}$. Assume $\tilde{F}_Y, F_Y$ are CDFs satisfying $\|\tilde{F}_Y - F_Y\|_\infty \le \gamma$, then there exists a $\delta > 0$ such that for all $\varepsilon - \delta\gamma \ge \sqrt{\frac{1}{2b}\ln 2}$ it holds that*

$$\Pr\left(\|F_{Z,b}^{-1}\tilde{F}_Y(t) - F_Z^{-1}F_Y(t)\|_\infty > \varepsilon\right) \le e^{-2b(\varepsilon-\delta\gamma)^2}. \tag{11}$$

For a proof of Theorem 1 please refer to our supplementary material. Since it is straightforward to estimate an EDF on one-dimensional data using a moderate number of samples, Theorem 1 tells us that the core steps of our primal SWD block approximate Eq. 6 well. Owing to the implicit SWD approximation by the primal SWD blocks, it is unnecessary to introduce an explicit regularization on the final objective. The objective of our proposed SWAE model is:

$$\inf_{P_{Q(Z|X)} \in \mathcal{Q}} \mathbb{E}_{X\sim P_X} \mathbb{E}_{Q\sim P_{Q(Z|X)}} [\|X - G(Q(X, Z))\|_2^2], \tag{12}$$

where Q, G are the encoder and decoder respectively, and Q is implicitly constrained by our primal SWD blocks. The corresponding algorithm is presented in Alg. 2.

3.4. Sliced Wasserstein GAN (SWGAN)

The success of WGAN indicates that the dual WD can be used as a suitable objective for the discriminator of GAN models. In order to keep the advantages of this setup, but to avoid imposing the k-Lipschitz constraint on a high dimensional distribution, we propose to use the dual SWD instead. Specifically, we introduce m dual SWD blocks $S_{d,1}, \ldots, S_{d,m}$ to the discriminator D (see Alg. 3). Image data distributions are supported by low-dimensional manifolds. For this reason, classic GAN discriminators encode their input data into lower-dimensional feature maps. We follow this setting. Our discriminator is the composition of an encoding network E and dual SWD blocks $S_{d,s}$, that is, $D = [S_{d,1} \circ E, \ldots, S_{d,m} \circ E]^T$.

Eventually, we estimate the integral over $\mathbb{S}^{n-1}$ of the dual SWD by summing over the outputs' mean value of SWD blocks $S_{d,1}, \ldots, S_{d,m}$ (see Alg. 4). In order to approximate the one-dimensional optimal $f \in \mathrm{Lip}^k$ (Eq. 10) in our SWD blocks, it suffices to use non-linear neural network layers. This is supported by the universal approximation theorem [14, 12]. For our case, we empirically set F_i in Alg. 3 to be $F_i(\boldsymbol{y}'_i) = u_i \mathrm{LeakyReLU}(w_i \boldsymbol{y}'_i + v_i)$, where u_i, v_i, w_i are scalar parameters.

The k-Lipschitz gradient penalty has its drawbacks in high dimensional space. For one-dimensional functions,

Algorithm 3 The proposed dual SWD block

Require: Orthogonal matrix $\boldsymbol{O}_{\boldsymbol{\Theta}} = [\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_r] \in \mathbb{R}^{r \times r}$ and batch of latent codes $\boldsymbol{M}_{\boldsymbol{y}} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_b] \in \mathbb{R}^{r \times b}$.
Output: Batch of $\tilde{\boldsymbol{y}}$ for dual SWD
for $i \leftarrow 1, r$ **do**
  Compute $\boldsymbol{y}_i' = \boldsymbol{\theta}_i^T \boldsymbol{M}_{\boldsymbol{y}}$
  Compute $\boldsymbol{y}_i'' = F_i(\boldsymbol{y}_i')$ element-wise, where $F = (F_1, \dots, F_r)$ are one-dimensional functions to approximate the f in Eq. 10.
end for
$\tilde{\boldsymbol{y}} = [\boldsymbol{y}_1'', \dots, \boldsymbol{y}_b'']^T$

Algorithm 4 The proposed SWGAN

Require: Number of dual SWD blocks m, batch size b, generator G and discriminator $D = [S_{d,1} \circ E, \dots, S_{d,m} \circ E]^T$, latent code dimension r, Lipschitz constant k, training steps h, training hyperparameters, etc.
for $t \leftarrow 1, h$ **do**
  Sample real data $\boldsymbol{M}_{\boldsymbol{x}} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_b]$ from P_X
  Sample Gaussian noise $\boldsymbol{M}_{\boldsymbol{z}} = [\boldsymbol{z}_1, \dots, \boldsymbol{z}_b]$ from $\mathcal{N}(0, 1)$
  Sample two vectors $\boldsymbol{\mu_1}, \boldsymbol{\mu_2}$ from uniform distribution $U[0, 1]$ and for $l = 1, \dots, b$ calculate the elements of $\boldsymbol{M}_{\hat{\boldsymbol{x}}}, \boldsymbol{M}_{\hat{\boldsymbol{y}}}$:
  $\hat{\boldsymbol{x}}_l = (1 - \mu_{1,l})\boldsymbol{x}_l + \mu_{1,l} G(\boldsymbol{z}_l)$
  $\hat{\boldsymbol{y}}_l = (1 - \mu_{2,l})E(\boldsymbol{x}_l) + \mu_{2,l} E(G(\boldsymbol{z}_l))$
  Update the weights $\boldsymbol{w}_G$ of G by descending:
  $\boldsymbol{w}_G \leftarrow \mathrm{Adam}(\nabla_{\boldsymbol{w}_G}(\frac{1}{b}\sum_{j,i=1}^{r \times m, b} D_{ji}(G(\boldsymbol{M}_{\boldsymbol{z}}))), \boldsymbol{w}_G)$
  Update the weights $\boldsymbol{w}_D$ of D by descending:
  $\boldsymbol{w}_D \leftarrow \mathrm{Adam}(\nabla_{\boldsymbol{w}_D}(\frac{1}{b}\sum_{j,i=1}^{r \times m, b}(D_{ji}(\boldsymbol{M}_{\boldsymbol{x}}) - D_{ji}(G(\boldsymbol{M}_{\boldsymbol{z}})) + \lambda_1 \|\nabla_{\boldsymbol{M}_{\hat{\boldsymbol{x}}}} D(\boldsymbol{M}_{\hat{\boldsymbol{x}}})\|_2^2 + \lambda_2 \|\nabla_{\boldsymbol{M}_{\hat{\boldsymbol{y}}}} F(\boldsymbol{M}_{\hat{\boldsymbol{y}}}) - k \cdot \mathbf{1}\|_2^2), \boldsymbol{w}_D)$, where we compute the gradients of F element-wise.
end for

however, it can easily impose the k-Lipschitz constraint. Thus, we additionally apply the gradient penalty on each dimension of the F_is' output. Since dual WD with different k-Lip constraints are equivalent to each other up to a scalar, we treat k, k' as tunable hyper-parameters for both F, D and relax the search interval to $k, k' \geq 0$, this relaxation can be justified by [37]. Consequently, the final objective is

$$\min_G \max_D \int_{\theta \in \mathbb{S}^{n-1}} \Big(\mathbb{E}_{X \sim P_X}[D(X)] - \mathbb{E}_{\tilde{X} \sim P_G}[D(G(Z))]\Big) + \lambda_1 \mathbb{E}_{\hat{X} \sim P_{\hat{X}}}[\|\nabla_{\hat{X}} D(\hat{X}) - k'\|_2^2] + \lambda_2 \mathbb{E}_{\hat{Y} \sim P_{\hat{Y}}}[\|\nabla_{\hat{Y}} F(\hat{Y}) - k \cdot \mathbf{1}\|_2^2], \tag{13}$$

where θ is embedded in D, and $\mathbf{1}$ is a vector with all entries being 1. We sample $\hat{X}, \hat{Y}$ based on [11]. λ_1, λ_2 are coefficients which balance the penalty terms (see Alg. 4). For the sake of computational efficiency our objective swaps the order of maximum and integral compared to Eq. 10. This exchange results in a lower bound estimation of Eq. 10. It implies that the objective can lead to the convergence of the dual SWD.

Discussion. In addition to the proposed SWAE and SWGAN, it is also possible to apply our proposed primal and dual SWD approximations to other generative models. For example, following [25], AE-based models can be enhanced by adversarial training using our dual SWD. Inspired by [30] it is possible to regularize GAN with our primal SWD. Moreover, by using a sorting algorithm [20] we can incorporate a simple primal SWD loss into GAN models.

3.5. Training for SWAE and SWGAN

To train the proposed SWAE and SWGAN models, we utilize the standard Adam optimization algorithm [17]. Throughout training, the projection matrices in the SWD blocks should remain orthogonal. For this purpose we first initialize the parameters of SWD blocks with random orthogonal matrices through QR decomposition, then update them on a curved manifold instead of a Euclidean space. Building upon the manifold-valued update rule [15], we optimize the orthogonal matrices on Stiefel manifolds[4].

In the t-th training step, after computing the Euclidean gradient $\nabla L_{\boldsymbol{O}_t}^{(k)}$ of an orthogonal matrix $\boldsymbol{O}_t$, we obtain its tangential component by subtracting $\boldsymbol{O}_t(\boldsymbol{O}_t^T \nabla L_{\boldsymbol{O}_t}^{(k)} + (\nabla L_{\boldsymbol{O}_t}^{(k)})^T \boldsymbol{O}_t)/2$ (see [7]), where $L^{(k)}$ is the loss for the k-th layer. For simplicity, we subsequently drop the index k. Searching along the tangential direction yields the update in the tangent space of the Stiefel manifold. In the end, the resulting update is projected back to the Stiefel manifold by a retraction operation Γ. Accordingly, the update of the current orthogonal matrix $\boldsymbol{O}_t$ on the Stiefel manifold can be written in the following form

$$\tilde{\nabla} L_{\boldsymbol{O}_t} = (\nabla L_{\boldsymbol{O}_t} - \boldsymbol{O}_t(\nabla L_{\boldsymbol{O}_t})^T \boldsymbol{O}_t)/2, \tag{14}$$

$$\boldsymbol{O}_{t+1} = \Gamma(\boldsymbol{O}_t - \Omega(\tilde{\nabla} L_{\boldsymbol{O}_t})), \tag{15}$$

where Γ denotes the retraction operation corresponding to QR decomposition and $\Omega(\cdot)$ denotes the standard Adam optimization. Note that the retraction has complexity $O(r^3)$ for r-dimensional data and is the main contributor to the time complexity of the optimization method. We therefore encode the n-dimensional input data into r-dimensional latent codes ($r < n$) before applying the SWD blocks, such that the training speed of our method remains comparable to the existing methods (see Tab. 1). The inference speed is not affected by the retraction operation.

4. Experiments under Standard Training

After discussing the theoretical merits of SWD and its applications to generative modeling, we examine the practical advantages of our proposed models under a standard training setting.

4.1. Evaluation on Toy Datasets

Following [11], we first conduct experiments on three toy datasets: Swiss Roll, 8 Gaussians and 25 Gaussians (see Fig. 2). For a fair comparison, we respect the experimental settings of [11] for the compared methods. For this experiment, the SWAE and SWGAN use only one SWD block.

[4] A compact Stiefel manifold $St(d, n)$ is defined as $St(d, n) = \{A \in \mathbb{R}^{n \times d} : A^T A = I_d\}$, where I_d is the $d \times d$ identity matrix.

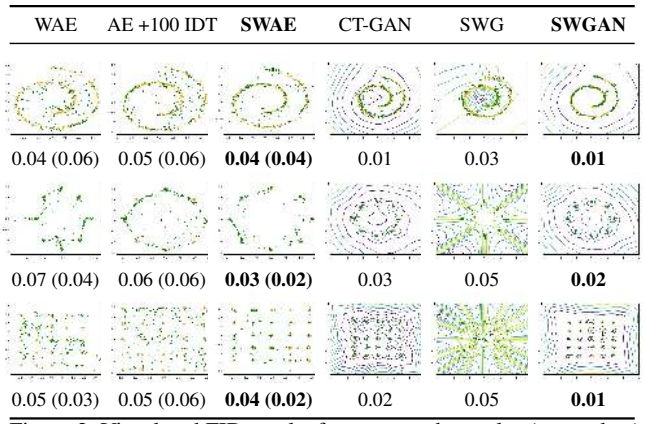


Figure 2. Visual and FID results for generated samples (green dots) compared to real samples (yellow dots) on Swiss Roll (top row), 8 Gaussians (middle row) and 25 Gaussians (bottom row). For the AE-based models, the FID scores displayed in parentheses indicate the discrepancy between generated latent codes and real noise. For the GANs, value surfaces of the discriminators are also plotted.

The superiority of our models is illustrated by visual results and the Fréchet inception distance (FID) [13].

SWD vs WD. Compared to the WD-based models–WAE and CTGAN–Fig. 2 shows that our SWAE outperforms WAE both visually and quantitatively. SWGAN also achieves better scores than CTGAN. These results support the advantages of SWD over WD for generative modeling.

SWAE vs AE + IDT. IDT [27] is the starting point of our SWAE. Thus, we also use it as a baseline. For this purpose, we stack IDT blocks on top of a regular encoder. We determine the optimal number of IDT blocks to be 100 by running multiple experiments. Then we train the IDT enhanced AE (AE + 100 IDT) under the standard Adam optimization. The FID scores in parentheses in Fig. 2 indicate that our SWAE, equipped with only one SWD block, better approximates the real noise distribution than the IDT enhanced AE with 100 IDT blocks. Moreover, the visual results confirm the improvement of SWAE over AE + 100 IDT for all three datasets. This shows that a single learnable primal SWD block is more effective than multiple original IDT blocks.

SWGAN vs SWG. We compare our SWGAN to the state-of-the-art SWD-based GAN model SWG [9]. SWG is a typical application of SWD approximation with projections along random unit vectors. Despite the use of 10000 random unit vectors in SWD, Fig. 2 shows that our SWGAN with only one dual SWD block (128 orthogonal unit vectors) is more successful at capturing the real data distribution in terms of better visual and better FID results, leveraging learnable projections along orthogonal unit vectors.

4.2. Evaluation on Standard Datasets

In addition to our toy dataset experiments, we also conduct various experiments on three widely-used benchmarks:

	CIFAR-10	CelebA	LSUN	CIFAR-10	CelebA
VAE	144.7±9.6	66.8±2.2	–	0.16s	0.64s
WAE-MMD	109.1±1.5	59.1±4.9	–	0.17s	0.63s
AAE (WAE-GAN)	**107.7±2.1**	**49.3±5.8**	–	0.25s	1.61s
SWAE	**107.9±5.2**	**48.9±4.3**	–	0.16s	0.37s
DCGAN	30.2 ± 0.9	52.5 ± 2.2	61.7 ± 2.9	0.13s	1.57s
WGAN	51.3 ± 1.5	37.1 ± 1.9	73.3 ± 2.5	0.25s	2.12s
WGAN-GP	19.0 ± 0.8	18.0 ± 0.7	26.9 ± 1.1	0.60s	2.40s
SNGAN	21.5 ± 1.3	21.7 ± 1.5	31.3 ± 2.1	0.21s	0.53s
CTGAN	17.6 ± 0.7	15.8 ± 0.6	19.5 ± 1.2	0.63s	2.61s
SWG	33.7 ± 1.5	21.9 ± 2.0	67.9 ± 2.7	0.22s	0.83s
SWGAN	**17.0 ± 1.0**	**13.2 ± 0.7**	**14.9 ± 1.0**	0.64s	2.74s

Table 1. FID (left) and runtime (right) comparison of AE-based and GAN models. The runtime is computed for one training step on a TITAN Xp GPU.

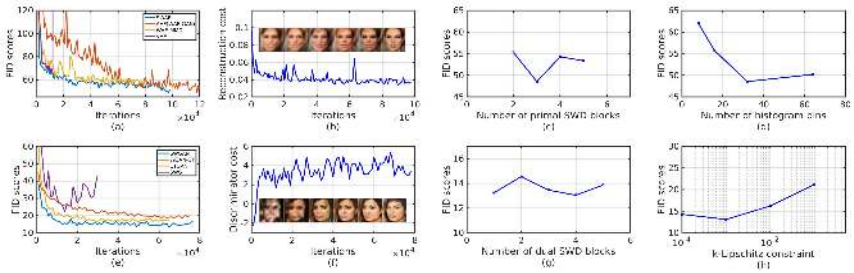
CIFAR-10 [22], CelebA [24], and LSUN [38]. We compare our SWAE to VAE [18], WAE-MMD [31], and AAE (WAE-GAN) [31, 25]. For GAN models, we compare our SWGAN against DCGAN [29], WGAN [3], WGAN-GP [11], SNGAN [26], CTGAN [36], and SWG [9].

Our proposed SWAE uses the decoder architecture suggested by [4]. For the encoder, we stack our primal SWD blocks on a shallow encoding network containing a downscaling and linear transform layer. Our SWGAN employs the ResNet structure used by [11] for the generator. For the discriminator, we apply our dual SWD blocks to an encoding network containing multiple ResNet layers. Please refer to our supplementary material for more architecture details. As to the compared methods, we use the official implementation if it is available online, and we apply the optimal settings tuned by their authors.

The evaluations of AE-based models in Tab. 1, Fig. 3 (Right: MTurk preference score) and Fig. 4 show that our proposed SWAE clearly outperforms the pure VAE model. Furthermore, our FID score is only marginally higher than the one of AAE (WAE-GAN), which additionally employs adversarial training. Due to this adversarial training, AAE (WAE-GAN) is generally less stable, while our model provides stable training (see Fig. 3 (Left a)) owing to a simple l_2 reconstruction loss without any additional regularizer.

Tab. 1 and Fig. 3 (Right) highlight the advantages of our SWGAN model in terms of FID and MTurk preference score. Relying on extra label information, SNGAN achieved a competitive FID score of 17.5 on CIFAR-10 as reported in [26]. Meanwhile, our SWGAN reaches an even lower score of 17.4 without the use of ground truth labels. The visual results reported in Fig. 4 are consistent with the FID scores in Tab. 1. We believe that the good performance of SWGAN mainly results from the efficient approximation of the proposed SWD on multiple one-dimensional marginal distributions of the training data.

Stability Study. In addition to the optimal architecture comparison, we also study the model stability under a variety of settings: ConvNet and ResNet, with normalization (w/ norm) and without normalization (w/o norm). As shown in Tab. 2, our proposed models are less sensitive in terms of FID scores, which is credited to the easier approximation of



Human Preference Score	
SWAE / VAE	**0.74** / 0.26
SWAE / WAE-MMD	**0.55** / 0.45
SWAE / AAE (WAE-GAN)	0.48 / **0.52**
SWGAN / WGAN-GP	**0.61** / 0.39
SWGAN / SNGAN	**0.66** / 0.34
SWGAN / CTGAN	**0.56** / 0.44
SWGAN / SWG	**0.69** / 0.31
PG-SWGAN / PG-WGAN	**0.55** / 0.45
PG-SWGAN-3D / VGAN	**0.91** / 0.09
PG-SWGAN-3D / MoCoGAN	**0.95** / 0.05
PG-SWGAN-3D / PG-WGAN-3D	**0.54** / 0.46

Figure 3. Left: Training and hyperparameter study for SWAE (top row) and SWGAN (bottom row). Right: Preference score from MTurk user study for generated images on CelebA and synthesized videos on TrailerFaces.

	ResNet (w/ norm)	ResNet (w/o norm)	ConvNet (w/ norm)	ConvNet (w/o norm)
WAE-MMD	64.0	61.8	55.8	67.8
AAE (WAE-GAN)	**62.3**	**56.7**	**48.3**	66.1
SWAE	63.2	59.1	65.2	**48.6**
CTGAN	16.0	16.5	19.5	19.7
SWG	24.3	29.1	22.2	28.5
SWGAN	**13.0**	**14.8**	**19.2**	**18.8**

Table 2. FID scores of various architectures on CelebA. The optimal architectures are ConvNet for SWG, WAE-MMD, AAE (WAE-GAN), ResNet for CTGAN, SWGAN, and ConvNet without normalization (w/o norm) for SWAE.

SWD (see visual results in supplementary material).

Training and Hyperparameters. Fig. 3 (Left b, f) show that the visual quality produced by our SWAE and SWGAN increases progressively with increasing training iterations. Also, using a small number of SWD blocks—3 primal and 4 dual blocks (Fig. 3 (Left c, g))—is sufficient to achieve top performances. The 4 dual blocks (4×128 unit vectors) in our SWGAN, stand in contrast to the 10000 random unit vectors required by SWG [9]. This confirms the efficiency of our learnable SWD blocks. In another experiment, we study the impact of the Lipschitz constant k, k' for SWGAN and bin number l for SWAE. Fig. 3 (Left d, h) show that SWGAN favors relatively small values of k. k' is determined to be 0 (see supplementary material). The optimal bin number for SWAE is 32. By analyzing the trade-off between computational complexity and performance, we set the r-dimension to 128 (Alg. 1, 3). We also determine λ_1, λ_2 (Eq. 13) to be $20, 10$ using grid search. Both studies are presented in the supplementary material. The learning rate of SWGAN and SWAE is determined empirically to be 0.0003. Finally, we set the discriminator iterations per training step of SWGAN to 4 for LSUN and CelebA and 5 for CIFAR-10.

5. Experiments under Progressive Training

Encouraged by the visual quality and stability improvements on standard benchmarks, we evaluate our proposed model for high resolution image and video generation under the progressive training manner suggested by [16].

Higher Resolution Image Generation. For this task, we use the CelebA-HQ [16] and LSUN [38] datasets, which contain 1024×1024 and 256×256 images respectively. To improve high resolution image generation, [16] introduces a progressive growing training scheme for GANs (PG-GAN). PG-GAN uses WGAN-GP loss (PG-WGAN) to achieve state-of-the-art high resolution image synthesis. For fair comparison, we equip the same progressive growing architecture with our proposed SWGAN objective and its dual SWD blocks (PG-SWGAN). As shown in Fig. 3 (Right) and Fig. 5, our PG-SWGAN can outperform PG-WGAN in terms of both qualitative and quantitative comparison on the CelebA-HQ and LSUN datasets.

Higher Resolution Video Generation. We introduce a new baseline unsupervised video synthesis method along with a new facial expression video dataset[5]. The dataset contains approximately 200,000 individual clips of various facial expressions, where the faces are cropped with 256×256 resolution from about 6,000 Hollywood movie trailers on YouTube. Thus, we name the dataset as TrailerFaces. For progressive video generation, we exploit a new PG-GAN network design for unsupervised video generation. We progressively scale the network in spatio-temporal dimension such that it can produce spatial appearance and temporal movement smoothly from coarse to fine. Fig. 3 (Right) shows the superiority of our model over the state-of-the-art methods [34, 32] in terms of preference score. For qualitative comparison, please refer to our supplementary videos.

Based on the proposed PG-GAN design, we evaluate the original WGAN loss (PG-WGAN-3D) and our proposed SWGAN loss (PG-SWGAN-3D). Fig. 3 (Right) and Fig. 5 present their qualitative and quantitative comparison. For the FID evaluation, we follow [35] to compute the video FID scores for PG-WGAN-3D and PG-SWGAN-3D. The higher preference score (Fig. 3 (Right)) and the lower FID score (Fig. 5) of our PG-SWGAN-3D reflect the advantage of using our proposed SWGAN.

Limitations. For pure AE-based generative models, including SWAE, the extension to high resolution image and video synthesis tasks is non-trivial. The challenges for SWAE to generate high quality images and videos on par with SWGAN remain. We plan to address this performance gap in future research.

[5]Both the baseline code and dataset will be released at https://github.com/musikisomorphie/swd.git

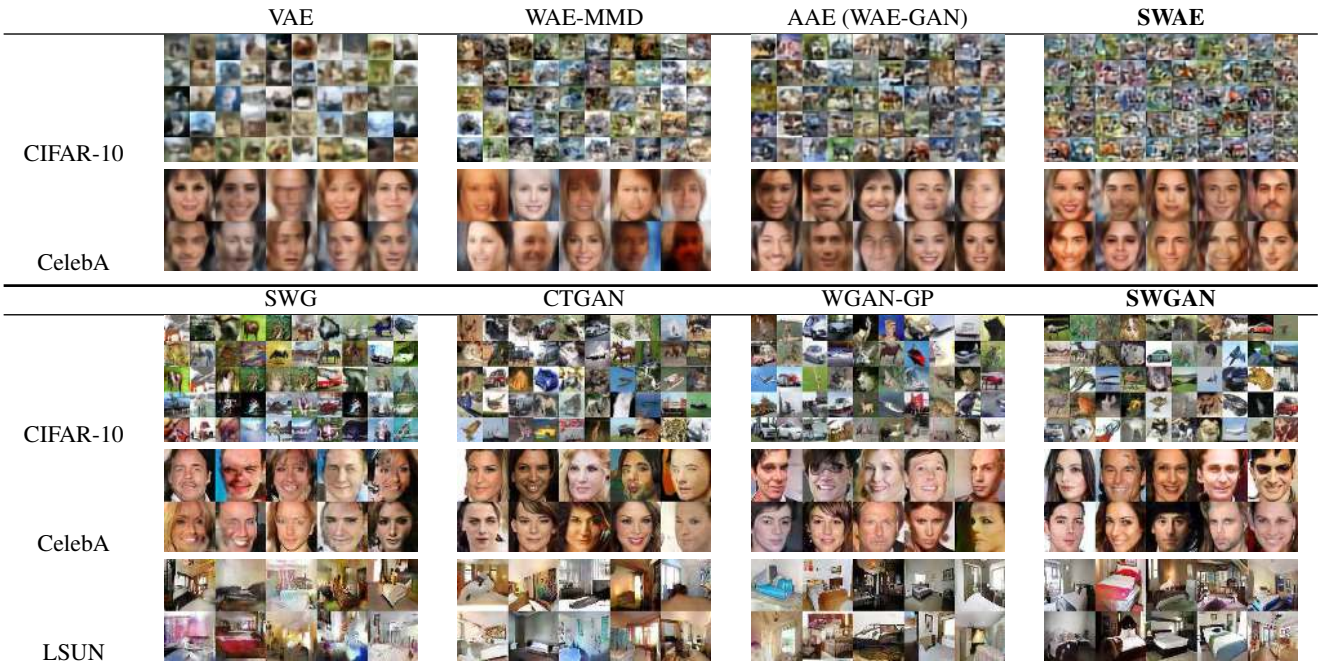


Figure 4. Visual results for SWAE, SWGAN, and compared methods. More results are available in the supplementary material.

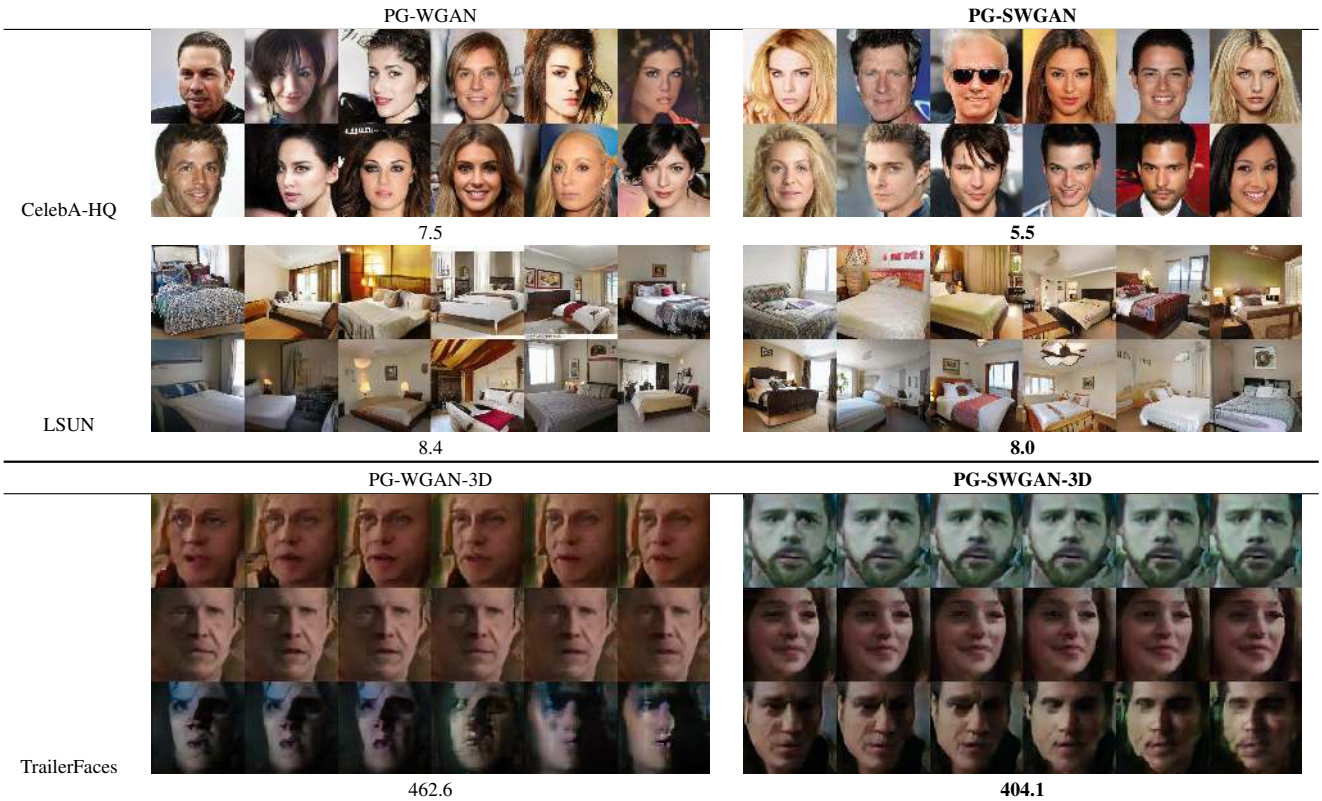


Figure 5. Visual and FID results for compared methods on higher resolution images/videos. More results are available in the supplementary.

6. Conclusion

In this paper, we introduce a novel way of efficiently approximating the primal and dual SWD. As concrete applications, we enhance modern AE-based and GAN models with the resulting primal and dual SWD blocks. For image and video synthesis, both qualitative and quantitative results show the superiority of our models over other approaches.

References

[1] Jonas Adler and Sebastian Lunz. Banach Wasserstein GAN. In *NIPS*, 2018. 1

[2] Luca Ambrogioni, Umut Güçlü, Yağmur Güçlütürk, Max Hinne, Marcel AJ van Gerven, and Eric Maris. Wasserstein variational inference. In *NIPS*, 2018. 1

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, 2017. 1, 2, 6

[4] David Berthelot, Tom Schumm, and Luke Metz. BEGAN: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017. 6

[5] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015. 1, 2

[6] Nicolas Bonnotte. *Unidimensional and evolution methods for optimal transportation*. PhD thesis, Paris 11, 2013. 3, 4

[7] Nicolas Boumal, Bamdev Mishra, P-A Absil, and Rodolphe Sepulchre. Manopt, a matlab toolbox for optimization on manifolds. *The Journal of Machine Learning Research*, 15(1):1455–1459, 2014. 5

[8] Rui Castro. The empirical distribution function and the histogram. *Lecture Notes, 2WS17-Advanced Statistics. Department of Mathematics, Eindhoven University of Technology*, 2015. 4

[9] Ishan Deshpande, Ziyu Zhang, and Alexander Schwing. Generative modeling using the sliced Wasserstein distance. In *CVPR*, 2018. 1, 3, 6, 7

[10] Aude Genevay, Gabriel Peyré, and Marco Cuturi. Learning generative models with Sinkhorn divergences. *arXiv preprint arXiv:1706.00292*, 2017. 1

[11] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of Wasserstein GANs. In *NIPS*, 2017. 1, 2, 5, 6

[12] Simon Haykin and Neural Network. A comprehensive foundation. *Neural networks*, 2(2004):41, 2004. 4

[13] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NIPS*, 2017. 6

[14] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257, 1991. 4

[15] Zhiwu Huang and Luc Van Gool. A Riemannian network for SPD matrix learning. In *AAAI*, 2017. 5

[16] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018. 2, 3, 7

[17] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[18] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 6

[19] Soheil Kolouri, Se Rim Park, Matthew Thorpe, Dejan Slepcev, and Gustavo K Rohde. Optimal mass transport: Signal processing and machine-learning applications. *IEEE Signal Processing Magazine*, 34(4):43–59, 2017. 1, 2

[20] Soheil Kolouri, Phillip E Pope, Charles E Martin, and Gustavo K Rohde. Sliced wasserstein auto-encoders. In *ICLR*, 2019. 5

[21] Soheil Kolouri, Yang Zou, and Gustavo K Rohde. Sliced Wasserstein kernels for probability distributions. In *CVPR*, 2016. 1, 2, 3

[22] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 6

[23] Huidong Liu, GU Xianfeng, and Dimitris Samaras. A two-step computation of the exact GAN Wasserstein distance. In *ICML*, 2018. 1

[24] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 6

[25] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. In *ICLR*, 2016. 5, 6

[26] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 2, 6

[27] François Pitié, Anil C Kokaram, and Rozenn Dahyot. Automated colour grading using colour distribution transfer. *CVIU*, 107(1):123–137, 2007. 3, 6

[28] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein barycenter and its application to texture mixing. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 435–446. Springer, 2011. 3

[29] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 6

[30] Tim Salimans, Han Zhang, Alec Radford, and Dimitris Metaxas. Improving GANs using optimal transport. In *ICLR*, 2018. 1, 5

[31] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. In *ICLR*, 2018. 1, 2, 6

[32] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. MoCoGAN: Decomposing motion and content for video generation. In *CVPR*, 2018. 7

[33] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008. 1

[34] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *NIPS*, 2016. 7

[35] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *NIPS*, 2018. 7

[36] Xiang Wei, Boqing Gong, Zixia Liu, Wei Lu, and Liqiang Wang. Improving the improved training of Wasserstein GANs: A consistency term and its dual effect. In *ICLR*, 2018. 1, 2, 6

[37] Jiqing Wu, Zhiwu Huang, Janine Thoma, Dinesh Acharya, and Luc Van Gool. Wasserstein divergence for GANs. In *ECCV*, 2018. 1, 5

[38] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 6, 7