Supplement to "Improving Relational Regularized Autoencoders with Spherical Sliced Fused Gromov Wasserstein"

In this supplementary material, we collect several proofs and remaining materials that were deferred from the main paper. In Appendix A, we provide the proofs of the main results in the paper. In Appendix B, further computational details of spherical sliced fused Gromov Wasserstein and its corresponding relational regularized autoencoder are given. We discuss two key extensions of spherical sliced fused Gromov Wasserstein (SSFG) in Appendices C and D. Additional experiments for the generative models are presented in Appendix E. Finally, detailed experimental settings are in Appendix F.

A PROOFS

In this appendix, we give the detailed proof for all the results in the main text.

A.1 PROOF OF THEOREM 1

In order to facilitate the ensuing presentation, for any probability measures μ and ν on $\mathbb{P}(\mathbb{R}^d)$ and joint probability measure γ of μ and ν, we denote $\text{proj}^1(\gamma) = \mu$ and $\text{proj}^2(\gamma) = \nu$. From the definition of spherical sliced fused Gromov Wasserstein, it is clear that SSFG is symmetric and non-negative. Therefore, we will only need to prove that it satisfies the weak triangle inequality. Recall that, since d_1 is pseudo-metric, there exists a constant $C > 0$ such that for any $x, y, z \in \mathbb{R}$, we have

$$d_1(x, z) \leq C\big[d_1(x, y) + d_1(y, z)\big].$$

Now, assume that μ, ν, ξ are probability measures on $\mathbb{R}^d$. For any $\theta \in \mathbb{S}^{d-1}$, we respectively denote π_1 and π_2 as the optimal couplings that minimize the fused Gromov Wasserstein between $\theta\sharp\mu$ and $\theta\sharp\nu$ and π_2 and the fused Gromov Wasserstein between $\theta\sharp\nu$ and $\theta\sharp\xi$. Then, by the gluing lemma (Villani, 2003), there exists a probability measure $\gamma \in \mathbb{P}(\mathbb{R} \times \mathbb{R} \times \mathbb{R})$ such that $\text{proj}^{1,2}(\gamma) = \pi_1$ and $\text{proj}^{2,3}(\gamma) = \pi_2$. Since $\text{proj}^1(\gamma) = \theta\sharp\mu$ and $\text{proj}^3(\gamma) = \theta\sharp\xi$, we obtain that $\text{proj}^{1,3}(\gamma) \in \Pi(\theta\sharp\mu, \theta\sharp\xi)$. From the definition of fused Gromov Wasserstein, we find that

$$\begin{aligned}
D_{fgw}(\theta\sharp\mu, \theta\sharp\xi; \beta, d_1) = &\min_{\pi \in \Pi(\theta\sharp\mu, \theta\sharp\xi; \beta)} \bigg\{(1-\beta)\int_{\mathbb{R}^d\times\mathbb{R}^d} d_1(\theta^\top x, \theta^\top z) d\pi(x, z) \\
&+ \beta \int_{(\mathbb{R}^d)^4} \big[d_1(\theta^\top x, \theta^\top x') - d_1(\theta^\top z, \theta^\top z')\big]^2 d\pi(x, z) d\pi(x', z') \bigg\} \\
&\leq (1-\beta)\int_{\mathbb{R}^d\times\mathbb{R}^d} d_1(\theta^\top x, \theta^\top z) d\text{proj}^{1,3}(\gamma)(x, z) \\
&+ \beta \int_{(\mathbb{R}^d)^4} \big[d_1(\theta^\top x, \theta^\top x') - d_1(\theta^\top z, \theta^\top z')\big]^2 d\text{proj}^{1,3}(\gamma)(x, z) d\text{proj}^{1,3}(\gamma)(x', z') \\
&= (1-\beta)\int_{\mathbb{R}^d\times\mathbb{R}^d\times\mathbb{R}^d} d_1(\theta^\top x, \theta^\top z) d\gamma(x, y, z) \\
&+ \beta \int_{(\mathbb{R}^d)^6} \big[d_1(\theta^\top x, \theta^\top x') - d_1(\theta^\top z, \theta^\top z')\big]^2 d\gamma(x, y, z) d\gamma(x', y', z').
\end{aligned} \tag{8}$$

Since $d_1(\theta^\top x, \theta^\top z) \leq C\big[d_1(\theta^\top x, \theta^\top y) + d_1(\theta^\top y, \theta^\top z)\big]$, we obtain that

$$\begin{aligned}
\int_{\mathbb{R}^d\times\mathbb{R}^d} d_1(\theta^\top x, \theta^\top z) d\gamma(x, y, z) &\leq C \int_{\mathbb{R}^d\times\mathbb{R}^d\times\mathbb{R}^d} \big[d_1(\theta^\top x, \theta^\top y) + d_1(\theta^\top y, \theta^\top z)\big] d\gamma(x, y, z) \\
&= C\bigg\{\int_{\mathbb{R}^d\times\mathbb{R}^d} d_1(\theta^\top x, \theta^\top y) d\pi_1(x, y) \\
&\qquad + \int_{\mathbb{R}^d\times\mathbb{R}^d} d_1(\theta^\top y, \theta^\top z) d\pi_1(y, z)\bigg\}.
\end{aligned} \tag{9}$$

Furthermore, an application of Cauchy-Schwarz inequality leads to

$$\begin{aligned}
&\int_{(\mathbb{R}^d)^6} \big[d_1(\theta^\top x, \theta^\top x') - d_1(\theta^\top z, \theta^\top z')\big]^2 d\gamma(x,y,z)d\gamma(x',y',z') \\
&\quad\leq 2\Bigg\{\int_{(\mathbb{R}^d)^6} \big[d_1(\theta^\top x, \theta^\top x') - d_1(\theta^\top y, \theta^\top y')\big]^2 d\gamma(x,y,z)d\gamma(x',y',z') \\
&\qquad + \int_{(\mathbb{R}^d)^6} \big[d_1(\theta^\top y, \theta^\top y') - d_1(\theta^\top z, \theta^\top z')\big]^2 d\gamma(x,y,z)d\gamma(x',y',z')\Bigg\} \\
&\quad= 2\Bigg\{\int_{(\mathbb{R}^d)^4} \big[d_1(\theta^\top x, \theta^\top x') - d_1(\theta^\top y, \theta^\top y')\big]^2 d\pi_1(x,y)d\pi_1(x',y') \\
&\qquad + \int_{(\mathbb{R}^d)^4} \big[d_1(\theta^\top y, \theta^\top y') - d_1(\theta^\top z, \theta^\top z')\big]^2 d\pi_2(y,z)d\pi_2(y',z')\Bigg\}. \qquad (10)
\end{aligned}$$

Plugging the results in equations (9) and (10) into the upper bound of $D_{fgw}(\theta\sharp\mu, \theta\sharp\xi; \beta, d_1)$ in equation (8), for any $\theta \in \mathbb{S}^{d-1}$ we have

$$D_{fgw}(\theta\sharp\mu, \theta\sharp\xi; \beta, d_1) \leq \max\{C, 2\}\Bigg\{D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1) + D_{fgw}(\theta\sharp\nu, \theta\sharp\xi; \beta, d_1)\Bigg\}.$$

Based on that inequality, we obtain that

$$\text{SSFG}(\mu, \xi; \beta, \kappa) \leq \max\{C, 2\}\big[\text{SSFG}(\mu, \nu; \beta, \kappa) + \text{SSFG}(\nu, \xi; \beta, \kappa)\big].$$

As a consequence, the spherical sliced fused Gromov Wasserstein satisfies the weak triangle inequality, which concludes the theorem.

A.2 Proof of Theorem 2

(a) We first prove that $\text{SSFG}(\mu, \nu; \beta, \kappa)$ is continuous in terms of κ. Indeed, the function $\mathbb{E}_{\theta\sim\text{vMF}(.|\epsilon,\kappa)}\big[D^2_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)\big]$ is continuous in terms of both ϵ and κ. Therefore, based on maximum theorem, $\text{SSFG}(\mu, \nu; \beta, \kappa)$ is continuous in terms of κ.

When κ goes to 0, the density of $\text{vMF}(.|\epsilon, \kappa)$ converges to that of $\mathcal{U}(\mathbb{S}^{d-1})$ (Sra, 2016). Therefore, given the continuity of SSFG in terms of κ we have

$$\begin{aligned}
\lim_{\kappa\to 0} \text{SSFG}(\mu, \nu; \beta, \kappa) &= \max_{\epsilon\in\mathbb{S}^{d-1}}\Bigg(\lim_{\kappa\to 0} \mathbb{E}_{\theta\sim\text{vMF}(.|\epsilon,\kappa)}\big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)\big]\Bigg) \\
&= \max_{\epsilon\in\mathbb{S}^{d-1}}\Bigg(\mathbb{E}_{\theta\sim\mathcal{U}(\mathbb{S}^{d-1})}\big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)\big]\Bigg) = \text{SFG}(\mu, \nu; \beta),
\end{aligned}$$

which confirms our conclusion that the spherical sliced fused Gromov Wasserstein becomes the sliced fused Gromov Wasserstein when $\kappa \to 0$.

When $\kappa \to \infty$, the density of $\text{vMF}(.|\epsilon, \kappa)$ converges to the density of δ_ϵ (Sra, 2016). Therefore, based on Scheffe's lemma (Feller, 1966), the density of $\text{vMF}(.|\epsilon, \kappa)$ converges in L1-norm to that of δ_ϵ. Given that result, as $D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)$ is uniformly bounded for all $\theta \in \mathbb{S}^{d-1}$, we arrive at

$$\begin{aligned}
\lim_{\kappa\to\infty} \text{SSFG}(\mu, \nu; \beta, \kappa) &= \max_{\epsilon\in\mathbb{S}^{d-1}}\Bigg(\lim_{\kappa\to\infty} \mathbb{E}_{\theta\sim\text{vMF}(.|\epsilon,\kappa)}\big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)\big]\Bigg) \\
&= \max_{\epsilon\in\mathbb{S}^{d-1}}\Bigg(\mathbb{E}_{\theta\sim\delta_\epsilon}\big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)\big]\Bigg) = \text{max-SFG}(\mu, \nu; \beta).
\end{aligned}$$

As a consequence, we obtain the conclusion of part (a) of the theorem.

(b) For any $\epsilon, \theta \in \mathbb{S}^{d-1}$, using Cauchy-Schwarz inequality, we get $-1 \leq \epsilon^\top x \leq 1$. Hence, for $\kappa > 0$, it is clear that $\exp(-\kappa) \leq \exp(\kappa\epsilon^\top x) \leq \exp(\kappa)$. Consequently, we obtain that

$$\begin{aligned}
\text{SSFG}(\mu, \nu; \beta, \kappa) &= \max_{\epsilon\in\mathbb{S}^{d-1}}\Bigg(\mathbb{E}_{\theta\sim\text{vMF}(\cdot|\epsilon,\kappa)}\big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)\big]\Bigg) \\
&\leq \max_{\epsilon\in\mathbb{S}^{d-1}}\Bigg(\exp(\kappa)\mathbb{E}_{\theta\sim\text{vMF}(\cdot|\epsilon,\kappa)}\big[D^2_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta)\big]\Bigg) \\
&= \exp(\kappa)C_d(\kappa)\text{SFG}(\mu, \nu; \beta).
\end{aligned}$$

Similarly, by using the bound $\exp(-\kappa) \leq \exp(\kappa\epsilon^\top x)$, we also arrive at the bound $\exp(-\kappa)C_d(\kappa)\text{SFG}(\mu,\nu;\beta) \leq \text{SSFG}(\mu,\nu;\beta,\kappa)$. Therefore, we obtain the bounds of SSFG based on SFG.

Regarding the upper bound of SSFG based on max-SFG, it is straight-forward from the inequality $D_{fgw}(\theta\sharp\mu,\theta\sharp\nu;\beta) \leq \max_{\theta'\in\mathbb{S}^{d-1}} D_{fgw}(\theta'\sharp\mu,\theta'\sharp\nu;\beta)$. As a consequence, we obtain the conclusion of part (b) of the theorem.

A.3 Proof of Theorem 3

For any positive integer r, we first prove the following simple inequality

$$\Big\{|\theta^\top x - \theta^\top x'|^r - |\theta^\top y - \theta^\top y'|^r\Big\}^2 \leq C\Big\{|\theta^\top x - \theta^\top y|^2 + |\theta^\top x' - \theta^\top y'|^2\Big\}, \tag{11}$$

for any $\theta \in \mathbb{S}^{d-1}$ and $x, y, x', y' \in \Theta$. Here, C is some universal constant depending only on the diameter of Θ. For simplicity, denote

$$\theta^\top x = a; \quad \theta^\top x' = b; \quad \theta^\top y = c; \quad \theta^\top y' = d;$$
$$C(\Omega) = \text{diameter of } \Omega.$$

By triangle's inequality, we find that

$$\Big||a-b| - |c-d|\Big| \leq |a-c| + |b-d|.$$

Moreover, we have the identity

$$|a-b|^r - |c-d|^r = \big[|a-b| - |c-d|\big]\sum_{i=0}^{r-1}|a-b|^i|c-d|^{r-i-1}.$$

Note that, the absolute values of a, b, c, d are not greater than $C(\Omega)$. It follows that

$$\Big||a-b|^r - |c-d|^r\Big| \leq \big[|a-c| + |b-d|\big]r2^{r-1}C(\Omega)^{r-1}.$$

By using Cauchy-Schwarz inequality, we obtain

$$\begin{aligned}\Big\{|a-b|^r - |c-d|^r\Big\}^2 &\leq 2^{2r-2}r^2C(\Omega)^{2r-2}\Big[|a-c| + |b-d|\Big]^2 \\ &\leq 2^{2r-1}r^2C(\Omega)^{2r-2}\Big[|a-c|^2 + b-d|^2\Big].\end{aligned}$$

Given the claim (11), we obtain that

$$\text{SFG}(\mu_n,\mu;\beta) \leq C_1\mathbb{E}_{\theta\sim\mathcal{U}(\mathbb{S}^{d-1})}\Big[\min_{\pi\in\Pi(\theta\sharp\mu_n,\theta\sharp\mu)}(\theta^\top x - \theta^\top y)^2 d\pi(x,y)\Big] := SW(\mu_n,\mu),$$

where C_1 is some universal constant. The RHS of the above inequality is known as second order sliced Wasserstein distance between μ_n and μ. Since Θ is compact, the result of Bobkov & Ledoux (2019) leads to

$$\mathbb{E}\big[SW(\mu_n,\mu)\big] \leq \frac{C_2}{n}.$$

Hence, it demonstrates that

$$\mathbb{E}\big[\text{SFG}(\mu_n,\mu;\beta)\big] \leq \frac{C_1C_2}{n},$$

for all $\beta \in [0,1]$. Combining the above result with the result of part (b) in Theorem A.2, we obtain that

$$\mathbb{E}\Big[\text{SSFG}(\mu_n,\mu;\beta,\kappa)\Big] \leq \mathbb{E}\big[\text{SFG}(\mu_n,\mu;\beta)\big] \leq \frac{c}{n},$$

where $c = C_1C_2$. As a consequence, we obtain the conclusion of the theorem.

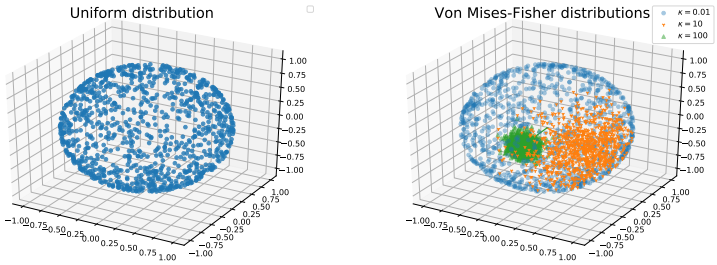


Figure 3: Illustrations of uniform distribution and von Mises-Fisher distribution.

B COMPUTATIONAL DETAILS OF SPHERICAL SLICED FUSED GROMOV-WASSERSTEIN AND ITS RELATIONAL REGULARIZED AUTOENCODER

In this section, we present the sampling algorithm (Algorithm 1) of von Mises-Fisher distribution, which follows the scheme in (Ulrich, 1984; Davidson et al., 2018), and how to derive the gradient estimator of the SSFG. Finally, we give the detail of the training procedure of s-DRAE (Algorithm 3).

B.1 ILLUSTRATION OF VON MISES-FISHER DISTRIBUTION

In Figure 3, we show the uniform distribution on the unit sphere and the von Mises-Fisher distribution with 3 different concentration values $\kappa = \{0.01, 10, 100\}$. When $\kappa = 0.01$, the distribution is very close to the uniform distribution; however, when κ is bigger (e.g. 10, 100), the masses of vMF distribution are concentrated near the location vectors.

B.2 SAMPLING FROM VON MISES-FISHER DISTRIBUTION

As discussed in (Davidson et al., 2018) to sample $\theta \sim \text{vMF}(.|\epsilon, \kappa)$, we first need to sample $h_1 \sim \text{vMF}(\cdot|e_1, \kappa)$ where $e_1 = (1, 0, .., 0)$. To do this step, we need to sample ω from the univariate density $g(\omega|\kappa, d) \propto \exp(\kappa\omega)(1-\omega^2)^{\frac{d-3}{2}}$ ($\omega \in [-1, 1]$) using an acceptance-rejection scheme, then compute $h_1 = (\omega, \sqrt{1-\omega^2}v^\top)^\top$ where $v \sim \mathcal{U}(\mathbb{S}^{d-2})$. Next, we find a orthogonal transformation U such that $U(\epsilon)e_1 = \epsilon$ by using Householder reflection. Finally, we compute $\theta = Uh_1$, then θ is a sample from vMF$(.|\epsilon, \kappa)$ as proved in (Ulrich, 1984). Detailed process is provided in pseudo-code in Algorithm 1.

Algorithm 1 Sampling from vMF distribution

Input: location ϵ, concentration κ, dimension d, unit vector $e_1 = (1, 0, .., 0)$
Sample $v \sim \mathcal{U}(\mathbb{S}^{d-2})$
$b \leftarrow \frac{-2\kappa+\sqrt{4\kappa^2+(d-1)^2}}{d-1}$, $a \leftarrow \frac{(d-1)+2\kappa+\sqrt{4\kappa^2+(d-1)^2}}{4}$, $m \leftarrow \frac{4ab}{(1+b)} - (d-1)\ln(d-1)$
repeat
  Sample $\psi \sim \text{Beta}\left(\frac{1}{2}(d-1), \frac{1}{2}(d-1)\right)$
  $\omega \leftarrow h(\psi, \kappa) = \frac{1-(1+b)\psi}{1-(1-b)\psi}$
  $t \leftarrow \frac{2ab}{1-(1-b)\psi}$
  Sample $u \sim \mathcal{U}(0, 1)$
until $(d-1)\ln(t) - t + m \geq \ln(u)$
$h_1 \leftarrow (\omega, \sqrt{1-\omega^2}v^\top)^\top$
$\epsilon' \leftarrow e_1 - \epsilon$
$u = \frac{\epsilon'}{\|\epsilon'\|}$
$U = \mathbb{I} - 2uu^\top$
Output: Uh_1

B.3 GRADIENT ESTIMATOR

Recall that d is the dimension of latent space, (ϵ, κ) be the parameters of vMF distribution, $b = \frac{-2\kappa+\sqrt{4\kappa^2+(d-1)^2}}{d-1}$, two distributions:

$$g(\omega \mid \kappa) = \frac{2\left(\pi^{d/2}\right)}{\Gamma(d/2)}\mathcal{C}_d(\kappa)\frac{\exp(\omega\kappa)\left(1-\omega^2\right)^{\frac{1}{2}(d-3)}}{\text{Beta}\left(\frac{1}{2},\frac{1}{2}(d-1)\right)},$$
$$r(\omega|\kappa) = \frac{2b^{1/2(d-1)}}{\text{Beta}\left(\frac{1}{2}(d-1),\frac{1}{2}(d-1)\right)}\frac{\left(1-\omega^2\right)^{1/2(d-3)}}{[(1+b)-(1-b)\omega]^{d-1}},$$

distribution $s(\psi) := \text{Beta}\left(\frac{1}{2}(d-1),\frac{1}{2}(d-1)\right)$, function $h(\psi,\kappa) = \frac{1-(1+b)\psi}{1-(1-b)\psi}$, distributions $\pi_1(\psi|\kappa) = s(\psi)\frac{g(h(\psi,\kappa)|\kappa)}{r(h(\psi,\kappa)|\kappa)}$, $\pi_2(v) := \mathcal{U}(\mathbb{S}^{d-2})$, and function

$$T(\omega, v, \epsilon) = \Big(\mathbb{I} - 2\frac{e_1-\epsilon}{\|e_1-\epsilon\|}\frac{e_1-\epsilon}{\|e_1-\epsilon\|}^\top\Big)\big(\omega, \sqrt{1-\omega^2}v^\top\big)^\top := \theta.$$

From Lemma 2 in (Davidson et al., 2018), we have:

$$\mathbb{E}_{\text{vMF}(\theta|\epsilon,\kappa)}\big[f(\theta)\big] = \mathbb{E}_{(\psi,v)\sim\pi_1(\psi|\kappa)\pi_2(v)}\Big[f\big(T(h(\psi,\kappa),v,\epsilon)\big)\Big], \tag{12}$$

Note that, in SSFG we only need $\nabla_\epsilon\mathbb{E}_{\text{vMF}(\theta|\epsilon,\kappa)}[f(\theta)]$ which can be obtained directly from the previous lemma:

$$\nabla_\epsilon\mathbb{E}_{\text{vMF}(\theta|\epsilon,\kappa)}\big[f(\theta)\big] = \mathbb{E}_{(\psi,v)\sim\pi_1(\psi|\kappa)\pi_2(v)}\Big[\nabla_\epsilon f\big(T(h(\psi,\kappa),v,\epsilon)\big)\Big], \tag{13}$$

Then we can get a gradient estimator by using Monte-Carlo estimation scheme:

$$\nabla_\epsilon\mathbb{E}_{\text{vMF}(\theta|\epsilon,\kappa)}\big[f(\theta)\big] \approx \frac{1}{L}\sum_{i=1}^{L}\Big[\nabla_\epsilon f\big(T(h(\psi_i,\kappa),v_i,\epsilon)\big)\Big], \tag{14}$$

where $\{\psi_i\}_{i=1}^L \sim \pi_1(\psi|\kappa)$ and $\{v_i\}_{i=1}^L \sim \pi_2(v)$ and L is the number of projections. Sampling from $\pi_1(\psi|\kappa)$ is equivalent to the acceptance-rejection scheme in vMF sampling procedure, sampling $\pi_2(v)$ is directly from $\mathcal{U}(\mathbb{S}^{d-2})$. Moreover, we also can derive a gradient estimator for $\nabla_\kappa\mathbb{E}_{\text{vMF}(\theta|\epsilon,\kappa)}\big[f(\theta)\big]$ by using the log-derivative trick, however, this is unnecessary in our SSFG.

From this estimator we can estimate the gradient $\nabla_\epsilon\mathbb{E}_{\text{vMF}(\theta|\epsilon,\kappa)}\big[D_{fgw}(\theta\sharp\mu,\theta\sharp\nu;\beta,d_1)\big]$, then use stochastic gradient ascent to find the optimal location ϵ^*, which induces the optimal vMF distribution.

B.4 COMPUTATIONAL DETAIL OF SPHERICAL SLICED FUSED GROMOV WASSERSTEIN

By using the estimator in section B.3, we present the algorithm to compute SSFG (Algorithm 2).

B.5 TRAINING PROCEDURE OF S-DRAE

In this section, we give a detailed procedure to train the spherical deterministic relational autoencoder in Algorithm 3. For each minibatch, the training procedure of s-DRAE includes computation of the reconstruction loss and the SSFG between the empirical distribution of the prior and the encoded latent distribution. The reconstruction loss is easy to compute while the computation of SSFG is harder. To obtain the best vMF distribution from the SSFG, we use the stochastic gradient ascent algorithm where the update depends on the gradient estimator of the location parameter of vMF, which is derived in Lemma 2 in (Davidson et al., 2018). Its details are in Appendix B.3. Then we sample L unit vectors from the obtained vMF and apply 1-d projections to obtain two sliced distributions. The SSFG distance can be computed easily after sorting the supports of the two distributions.

Algorithm 2 Computation of spherical sliced fused Gromov Wasserstein

Input: Empirical distribution $\{x_i\}_{i=1}^N$,$\{y_i\}_{i=1}^N$, concentration κ, fused parameter β, the number of projections L, max_iter
repeat
  Initialize SSFG $\leftarrow 0$
  for $l = 1$ **to** L **do**
    Sample $\theta_l \sim \text{vMF}(\theta|\epsilon, \kappa)$ based on Algorithm 1
    Sort $\{\theta_l^\top x_i\}_{i=1}^N$ and $\{\theta_l^\top y_i\}_{i=1}^N$ respectively
    Caculate $D_{fgw}(\{\theta_l^\top x_i\}_{i=1}^N, \{\theta_l^\top y_i\}_{i=1}^N; \beta)$ based on its closed-form and sorted samples
    SSFG $\leftarrow$ SSFG + $D_{fgw}(\{\theta_l^\top z_i\}_{i=1}^N, \{\theta_l^\top z_i'\}_{i=1}^N; \beta)$
  end for
  SSFG $\leftarrow \frac{\text{SSFG}}{L}$
  Estimate ∇_ϵSSFG
  Update $\epsilon \leftarrow \text{Proj}_{\mathbb{S}^{d-1}}(\text{Adam}(\nabla_\epsilon \text{SSFG}))$
until ϵ converges **or** reach max_iter
Output: SSFG

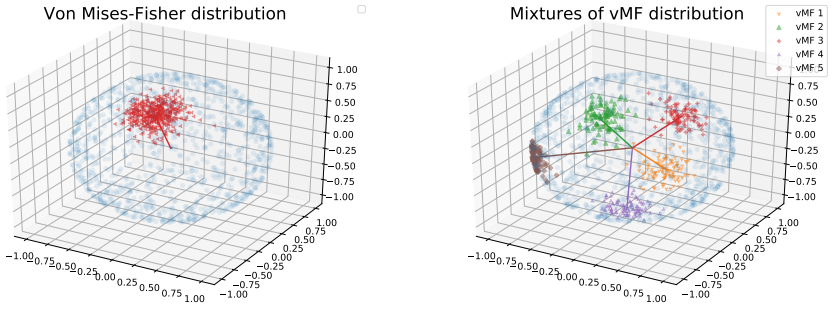


Figure 4: Illustrations of von Mises-Fisher distribution and mixture of von Mises-Fisher distributions.

C MIXTURE SPHERICAL SLICED FUSED GROMOV WASSERSTEIN

In this appendix, we consider an extension of the spherical sliced fused Gromov Wasserstein to the mixture spherical sliced fused Gromov Wasserstein (MSSFG). Then, we discuss an application of MSSFG to the deterministic relational regularized autoencoder framework.

In order to facilitate the ensuing discussion, we recall the definitions of mixture of vMF distributions and mixture spherical sliced fused Gromov Wasserstein (MSSFG). We first define mixture of von Mises-Fisher distributions (Banerjee et al., 2005), which plays an important role in the definition of mixture spherical sliced fused Gromov Wasserstein.

Definition 5. *Given $k \geq 1$ distinct pairs $(\epsilon_1, \kappa_1), \ldots, (\epsilon_k, \kappa_k)$ and the mixture weights $\{\alpha_i\}_{i=1}^k$, i.e., $\alpha_i \geq 0$ and $\sum_{i=1}^k \alpha_i = 1$, the **mixture of vMF distributions** is defined as:*

$$\text{MovMF}(\cdot|\epsilon_{1:k}, \kappa_{1:k}, \alpha_{1:k}) := \sum_{i=1}^k \alpha_i \text{vMF}(\cdot|\epsilon_i, \kappa_i).$$

When $k = 1$, the mixture of von Mises-Fisher distributions becomes the standard von Mises-Fisher distribution. When $k \geq 2$, we provide an illustration of mixture of vMF distributions in Figure 4. In order to sample from mixture of vMF distribution, we first sample the mixture index by categorical distribution parametrized by $\{\alpha_i\}_{i=1}^k$. Then, we sample the corresponding vMF component. Details of the sampling procedure with mixture of vMF distributions in Algorithm 4.

With the definition of the mixture of vMF distributions in hand, we now define the mixture spherical sliced fused Gromov Wasserstein between the probability distributions.

Definition 6. *(MSSFG) Let $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ be two probability distributions, $\beta \in [0, 1]$ be a constant, $\{\alpha_i\}_{i=1}^k$ be given mixture weights, and $\{\kappa_i\}_{i=1}^k$ be given mixture concentration parameters. Fur-*

Algorithm 3 Training DRAE with spherical sliced fused Gromov Wasserstein

Input: Concentration κ, fused parameter β, coefficient λ, the number of mixtures K, max_iter minibatch's size N, the number of projections L, empirical data distribution $\hat{p}(x)$
Initialize $G_\theta, E_\phi, \mu_{1:K}, \Sigma_{1:K}, \epsilon$
for each epoch **do**
  for each minibatch $\{x_i\}_{i=1}^N \sim \hat{p}(x)$ **do**
    $\{z_i\}_{i=1}^N \leftarrow E_\phi(\{x_i\}_{i=1}^N)$
    Sample $\{z_i'\}_{i=1}^N \sim p(z) := \frac{1}{K}\sum_{i=1}^K \mathcal{N}(\mu_k, \Sigma_k)$
    Compute SSFG between $\{z_i\}_{i=1}^N$ and $\{z_i'\}_{i=1}^N$ with $\beta, \kappa, L, max\_iter$ via the Algorithm 2.
    Reconstruction $\leftarrow \frac{1}{N}\sum_{i=1}^N d(x_i, G_\theta(z_i))$
    Update $\theta, \phi, \mu_{1:K}, \Sigma_{1:K} \leftarrow$ Adam$(\nabla_{\theta,\phi,\mu_{1:K},\Sigma_{1:K}}$(Reconstruction + λSSFG))
  end for
end for
Output: $G_{\theta^*}, E_{\phi^*}, \mu_{1:K}^*, \Sigma_{1:K}^*$

Algorithm 4 Sampling from mixture of vMF distributions

Input: The number of vMF components k, location $\{\epsilon_i\}_{i=1}^k$, concentration $\{\kappa_i\}_{i=1}^k$, mixture weights $\{\alpha_i\}_{i=1}^k$, dimension d, unit vector $e_1 = (1, 0, .., 0)$.
Sample index $i \sim$ Categorical$(\alpha_1, ..., \alpha_k)$
Sample $v \sim \mathcal{U}(\mathbb{S}^{d-2})$
$b \leftarrow \frac{-2\kappa_i + \sqrt{4\kappa_i^2 + (d-1)^2}}{d-1}$, $a \leftarrow \frac{(d-1) + 2\kappa_i + \sqrt{4\kappa_i^2 + (d-1)^2}}{4}$, $m \leftarrow \frac{4ab}{(1+b)} - (d-1)\ln(d-1)$
repeat
  Sample $\psi \sim$ Beta $\left(\frac{1}{2}(d-1), \frac{1}{2}(d-1)\right)$
  $\omega \leftarrow h(\psi, \kappa_i) = \frac{1-(1+b)\psi}{1-(1-b)\psi}$
  $t \leftarrow \frac{2ab}{1-(1-b)\psi}$
  Sample $u \sim \mathcal{U}(0, 1)$
until $(d-1)\ln(t) - t + m \geq \ln(u)$
$h_1 \leftarrow (\omega, \sqrt{1-\omega^2} v^\top)^\top$
$\epsilon' \leftarrow e_1 - \epsilon_i$
$u = \frac{\epsilon'}{\|\epsilon'\|}$
$U = \mathbb{I} - 2uu^\top$
Output: Uh_1

*thermore, let $d_1 : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ be a pseudo-metric on $\mathbb{R}$. Then, the **mixture spherical sliced fused Gromov Wasserstein** (MSSFG) between μ and ν is defined as follows:*

$$MSSFG(\mu, \nu; \beta, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k) := \max_{\epsilon_{1:k} \in \mathbb{S}^{d-1}} \mathbb{E}_{\theta \sim \text{MovMF}(\cdot|\epsilon_{1:k}, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k)} \left[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1)\right], \quad (15)$$

where the fused Gromov Wasserstein D_{fgw} is defined in equation (3).

Computational complexity of MSSFG: Let μ and ν be two discrete distributions that have n supports with uniform weights. For the general case of d_1, computing MSSFG is costly as SFG and SSFG due to the quadratic programming problem. However, MSSFG also has the complexity of order $\mathcal{O}(n \log n)$ with $d_1(x, y) = (x - y)^2$. To solve the optimization problem of MSSFG, we can reuse the vMF's gradient estimators in Section B.3 and find the locations of components of the mixture vMF by stochastic gradient ascent. In detail, the gradient estimator of each vMF component's location is derived as follow:

$$\begin{aligned}
&\nabla_{\epsilon_i} \mathbb{E}_{\text{MovMF}(\theta|\epsilon_{1:k}, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k)} \left[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1)\right] \\
&= \nabla_{\epsilon_i} \mathbb{E}_{\frac{1}{k}\sum_{i=1}^k \alpha_i \text{vMF}(\theta|\epsilon_i, \kappa_i)} \left[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1)\right] \\
&= \frac{\alpha_i}{k} \nabla_{\epsilon_i} \mathbb{E}_{\text{vMF}(\theta|\epsilon_i, \kappa_i)} \left[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1)\right] \quad (16)
\end{aligned}$$

Here, we can reuse the result from Section B.3 to get an estimator.

Key properties of MSSFG: Similar to SSFG, MSSFG is also a pseudo-metric on the probability space.

Theorem 4. *For any $\beta \in [0, 1]$, mixture concentration parameters $\{\kappa_i\}_{i=1}^k$, and mixture weights $\{\alpha_i\}_{i=1}^k$, MSSFG$(., .; \beta, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k)$ is a well-defined pseudo-metric in the space of probability measures, namely, it is non-negative, symmetric, and satisfies the weak triangle inequality.*

The proof of Theorem 4 is similar to the proof of Theorem 1; therefore, it is omitted. Our next result establishes some relations between MSSFG, SSFG, SFG, and max-SFG (see part (a) in Theorem 2 for a definition of max-SFG).

Theorem 5. *For any probability measures $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$, the following holds:*

(a) For any mixture weights $\{\alpha_i\}_{i=1}^k$, we obtain that

$$\lim_{\kappa_1,\ldots,\kappa_k \to 0} \textit{MSSFG}(\mu, \nu; \beta, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k) = \textit{SFG}(\mu, \nu; \beta),$$

$$\lim_{\kappa_1,\ldots,\kappa_k \to \infty} \textit{MSSFG}(\mu, \nu; \beta, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k) = \textit{max-SFG}(\mu, \nu; \beta).$$

(b) For any mixture concentration parameters $\{\kappa_i\}_{i=1}^k$ and mixture weights $\{\alpha_i\}_{i=1}^k$, we find that

$$\alpha_{\bar{i}} \max_{1 \le i \le k} \{\textit{SSFG}(\mu, \nu; \beta, \kappa_i)\} \le \textit{MSSFG}(\mu, \nu; \beta, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k) \le \max_{1 \le i \le k} \{\textit{SSFG}(\mu, \nu; \beta, \kappa_i)\},$$

where $\bar{i} \in \arg\max_{1 \le i \le k} \{\textit{SSFG}(\mu, \nu; \beta, \kappa_i)\}$.

The proof of part (a) in Theorem 5 is a simple extension of the proof of part (a) in Theorem 2 while the proof of part (b) in Theorem 5 is straight-forward from the definition of MSSFG. Therefore, the proof of Theorem 5 is omitted. The result of part (a) in Theorem 5 demonstrates that MSSFG is an interpolation between SFG and max-SFG. Furthermore, the result of part (b) in Theorem 5 shows that MSSFG is equivalent to $\max_{1 \le i \le k}\{\text{SSFG}(\mu, \nu; \beta, \kappa_i)\}$. Based on the result of part (b) in Theorem 2, MSSFG is also equivalent to SFG.

Our next result shows that MSSFG also does not suffer from the curse of dimensionality as SSFG.

Theorem 6. *Assume that μ is a probability measure supported on a compact subset $\Theta \subset \mathbb{R}^d$. Let $X_1, \ldots, X_n$ be i.i.d. data from P and $d_1(x, y) = |x - y|^r$ for a positive integer r. We denote $\mu_n = \frac{1}{n}\sum_{i=1}^n \delta_{X_i}$ the empirical measure of the data points $X_1, \ldots, X_n$. Then, for any $\beta \in [0, 1]$, mixture concentration parameters $\{\kappa_i\}_{i=1}^k$, and mixture weights $\{\alpha_i\}_{i=1}^k$, there exists a constant c depending only on r and the diameter of Ω such that*

$$\mathbb{E}\Big[\textit{MSSFG}(\mu_n, \mu; \beta, \{\kappa_i\}_{i=1}^k, \{\alpha_i\}_{i=1}^k)\Big] \le \frac{c}{n}.$$

The proof of Theorem 6 is direct from the upper bound of MSSFG based on SSFG in part (b) of Theorem 5 and the convergence rate of SSFG in expectation in Theorem 3; therefore, it is omitted.

Application of MSSFG to relational regularized autoencoder framework: Similar to the SSFG, we can apply MSSFG to relational regularized autoencoder by using it as the discrepancy between prior distribution and latent code distribution. We name this autoencoder – *mixture spherical deterministic autoencoder* (ms-DRAE). The training procedure of ms-DRAE is similar to s-DRAE, it includes computation of the reconstruction loss and the MSSFG between the empirical distribution of the prior and the encoded latent distribution. To compute MSSFG, it also requires the stochastic gradient ascent scheme to find the best MovMF distribution. Note that, in this autoencoder, we set a uniform mixture weights $\alpha_i = \frac{1}{k}$ and each vMF component uses the same value of concentration parameter κ.

D POWER SPHERICAL SLICED FUSED GROMOV WASSERSTEIN

In this appendix, we consider another variant of spherical sliced fused Gromov Wasserstein, which is power spherical sliced fused Gromov Wasserstein (PSSFG). Then, we discuss an application of PSSFG to the deterministic relational regularized autoencoder framework.

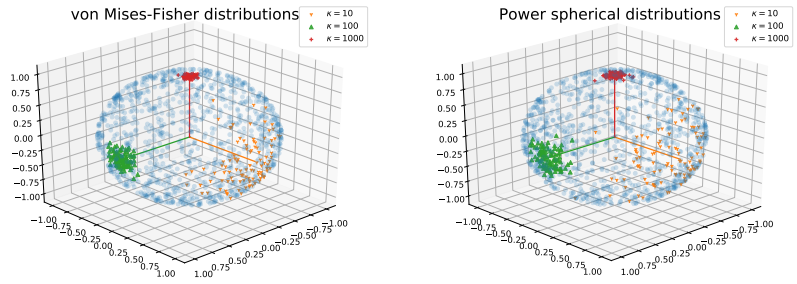


Figure 5: Illustrations of von Mises-Fisher and Power spherical distribution.

We first define power spherical distribution (De Cao & Aziz, 2020):

Definition 7. *The power spherical distribution (PS) is a probability distribution on the unit sphere* $\mathbb{S}^{d-1}$ *where its density function is given by:*

$$f(x|\epsilon,\kappa) = C(\kappa,d)(1+\epsilon^\top x)^\kappa, \tag{17}$$

where $\kappa \geq 0$ *is the concentration parameter,* $\epsilon \in \mathbb{S}^{d-1}$ *is the location vector, and* $C_d(\kappa) := \left\{2^{d-1+\kappa}\pi^{\frac{d-1}{2}}\frac{\Gamma(\frac{d-1}{2}+\kappa)}{\Gamma(d-1+\kappa)}\right\}^{-1}$.

We now provide the following result showing that similar to vMF, PS is also an interpolation between the uniform distribution and Dirac delta measure on the sphere.

Lemma 1. *(a) When* $\kappa \to 0$*, the density of PS*$(.|\epsilon,\kappa)$ *converges to that of* $\mathcal{U}(\mathbb{S}^{d-1})$.

(b) When $\kappa \to \infty$*, the density of PS*$(.|\epsilon,\kappa)$ *converges to the density of* δ_ϵ.

Proof. (a) When $\kappa \to 0$, the power spherical distribution approaches to the uniform distribution, because for $\kappa \in (0,1]$, the density function $f(x|\epsilon,\kappa)$ is uniformly bounded. Moreover, for all x is different from $-\epsilon$

$$\lim_{\kappa\to 0} f(x|\epsilon,\kappa) = C(0,d),$$

which corresponds to the density of the uniform distribution on $\mathbb{S}^{d-1}$. Hence, by Lebesgue dominated convergence theorem, for any continuous and uniformly bounded function g on $\mathbb{S}^{d-1}$, we have

$$\lim_{\kappa\to 0}\int_{\mathbb{S}^{d-1}} g(x)f(x|\epsilon,\kappa)dx = \int_{\mathbb{S}^{d-1}} g(x)\mathcal{U}(\mathbb{S}^{d-1})dx,$$

where $\mathcal{U}(\mathbb{S}^{d-1})$ stands for the uniform distribution on $\mathbb{S}^{d-1}$. It means the convergence of the density f of the power spherical distribution to the uniform distribution when $\kappa \to 0$.

(b) When $\kappa \to \infty$, as long as the function f at the mode ϵ goes to infinity, then the following holds for any continuous and uniformly bounded function g on $\mathbb{S}^{d-1}$

$$\lim_{\kappa\to\infty}\int_{\mathbb{S}^{d-1}} g(x)f(x|\epsilon,\kappa)dx = \int_{\mathbb{S}^{d-1}} g(x)\delta_\epsilon(x)dx.$$

It is true since for $x = \epsilon$, and $\kappa \to \infty$ we have

$$f(\epsilon|\epsilon,\kappa) = 2^\kappa\frac{\Gamma(d-1+\kappa)}{\Gamma(\frac{d-1}{2}+\kappa)}\pi^{\frac{1-d}{2}}2^{1-d-\kappa} = \pi^{\frac{1-d}{2}}2^{1-d}\frac{\Gamma(d-1+\kappa)}{\Gamma(\frac{d-1}{2}+\kappa)} \to \infty.$$

As a consequence, the density of the power spherical distribution with location vector ϵ converges to the Dirac delta measure at ϵ when $\kappa \to \infty$. □

Sampling procedure of the power spherical distribution: We review the sampling algorithm of PS in Algorithm 5 in (De Cao & Aziz, 2020). The important difference between the sampling of the PS to that of the vMF is that it does not require rejection sampling as in vMF's sampling algorithm (Algorithm 1). As a result, sampling from PS is faster than sampling from vMF. Furthermore, we can get an estimation of gradient of parameters of the density of the PS easier than that of vMF since all the operations in the sampling algorithm of the PS are differentiable (note that, it also includes sampling from Beta distribution which can use the implicit reparametrization trick (Figurnov et al., 2018)).

Algorithm 5 Sampling from power spherical distribution

Input: location parameter ϵ, concentration κ, dimension d, unit vector $e_1 = (1, 0, .., 0)$.
Sample $z \sim \text{Beta}(\frac{(d-1)}{2} + \kappa, \frac{(d-1)}{2})$
Sample $v \sim \mathcal{U}(\mathbb{S}^{d-2})$
$w \leftarrow 2z - 1$
$h_1 \leftarrow (\omega, \sqrt{1-\omega^2}v^\top)^\top$
$\epsilon' \leftarrow e_1 - \epsilon$
$u = \frac{\epsilon'}{\|\epsilon'\|}$
$U = \mathbb{I} - 2uu^\top$
Output: Uh_1

Given the definition of the power spherical distribution, we are ready to define the power spherical sliced fused Gromov Wasserstein between the probability distributions.

Definition 8. *(PSSFG) Let $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$ be two probability distributions, $\kappa > 0$, $\beta \in [0, 1]$, $d_1 : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ be a pseudo-metric on $\mathbb{R}$. The **power spherical sliced fused Gromov Wasserstein (PSSFG)** between μ and ν is defined as follows:*

$$PSSFG(\mu, \nu; \beta, \kappa) := \max_{\epsilon \in \mathbb{S}^{d-1}} \mathbb{E}_{\theta \sim \text{PS}(\cdot|\epsilon,\kappa)} \big[D_{fgw}(\theta\sharp\mu, \theta\sharp\nu; \beta, d_1) \big], \tag{18}$$

where the fused Gromov Wasserstein D_{fgw} is defined in equation (3). Here, PS$(.|\epsilon, \kappa)$ denotes the power spherical distribution with location vector ϵ and concentration parameter κ.

Comparison between PSSFG and SSFG: The PS distribution has similar behavior as the vMF distribution on the unit sphere, suggesting that PSSFG and SSFG are similar discrepancies. However, PS shows better sampling speed and stability than vMF, which leads to the computational benefits of PSSFG over SSFG. In particular, to compute PSSFG and SSFG, we need to approximate the gradient of the location parameter ϵ with L samples from the PS and vMF distributions respectively. With the faster sampling algorithm of PS, we can find the best location parameter in PSSFG faster than in SSFG. Therefore, PSSFG can be computed in lower time than SSFG. Moreover, as reported in (De Cao & Aziz, 2020), the vMF distribution suffers from numerical issues on high dimension settings or on high concentration parameter settings, namely, sampling vectors from the vMF distribution can be returned as "not a number" (NaN), which greatly affects the quality of SSFG. In contrast, PSSFG does not have the similar numerical issues because the PS distribution provides more stable samples in high-dimension settings with any values of the concentration parameter.

Computational complexity of PSSFG: Similar to SSFG, PSSFG between μ and ν, two discrete distributions that have n supports and uniform weights, has the complexity of order $\mathcal{O}(n \log n)$ when $d_1(x, y) = (x - y)^2$ for all $x, y \in \mathbb{R}$. With a faster sampling process, the gradient estimation step in PSSFG consumes a smaller amount of time than SSFG. Therefore, it leads to faster optimization to find the optimal location parameter ϵ in PSSFG than in SSFG.

Key properties of PSSFG: Similar to SSFG and MSSFG, PSSFG is also a pseudo-metric on the probability space.

Theorem 7. *For any $\beta \in [0, 1]$ and $\kappa > 0$, PSSFG$(., .; \beta, \kappa)$ is a pseudo-metric in the space of probability measures, namely, it is non-negative, symmetric, and satisfies the weak triangle inequality.*

The proof of Theorem 7 is similar to the proof of Theorem 1; therefore, it is omitted.

Our next result establishes some connections between PSSFG, SSFG, SFG, and max-SFG (see part (a) in Theorem 2 for a definition of max-SFG).

Theorem 8. *For any probability measures $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$, the following holds:*

(a)
$$\lim_{\kappa \to 0} \textit{PSSFG}(\mu, \nu; \beta, \kappa) = \textit{SFG}(\mu, \nu; \beta),$$
$$\lim_{\kappa \to \infty} \textit{PSSFG}(\mu, \nu; \beta, \kappa) = \textit{max-SFG}(\mu, \nu; \beta).$$

(b) For any $\kappa > 0$, we find that

$$\textit{PSSFG}(\mu, \nu; \beta, \kappa) \leq 2^{\kappa} C(\kappa, d) \textit{SFG}(\mu, \nu; \beta),$$
$$\textit{PSSFG}(\mu, \nu; \beta, \kappa) \leq \textit{max-SFG}(\mu, \nu; \beta).$$

The proof of part (a) in Theorem 8 is a straightforward application of the proof of Theorem 2 and the asymptotic properties of the power spherical distribution. The proof of part (b) in Theorem 8 is also similar to the proof of part (b) of Theorem 2. Note that, we do not have the lower bound of PSSFG$(\mu, \nu; \beta, \kappa)$ in terms of SFG$(\mu, \nu; \beta)$. It is because the value of $(1 + \epsilon^\top x)^\kappa$ can get arbitrarily close to 0 as $\epsilon^\top x$ gets close to -1. Combining with the result of Theorem 8, we reach to a conclusion that the SSFG, SFG, max-SFG are stronger discrepancies than the PSSFG.

Our next result shows that PSSFG does not suffer from the curse of dimensionality as SSFG and MSSFG. Therefore, it is also a good discrepancy to use in the deterministic relational regularized autoencoder framework.

Theorem 9. *Assume that μ is a probability measure supported on a compact subset $\Theta \subset \mathbb{R}^d$. Let $X_1, \ldots, X_n$ be i.i.d. data from P and $d_1(x, y) = |x - y|^r$ for a positive integer r. We denote $\mu_n = \frac{1}{n} \sum_{i=1}^n \delta_{X_i}$ the empirical measure of the data points $X_1, \ldots, X_n$. Then, for any $\beta \in [0, 1]$ and $\kappa > 0$, there exists a constant c depending only on r and diameter of Θ such that*

$$\mathbb{E}\Big[\textit{PSSFG}(\mu_n, \mu; \beta, \kappa)\Big] \leq \frac{c}{n}.$$

The proof of Theorem 9 is straightforward from the upper bound of PSSFG in terms of SSFG and the convergence rate of SSFG in expectation in Theorem 3; therefore, it is omitted.

Application of PSSFG to relational regularized autoencoder framework: Similar to the SSFG, we can apply PSSFG to relational regularized autoencoder by using it as the discrepancy between prior distribution and latent code distribution. We name this autoencoder *power spherical deterministic autoencoder* (ps-DRAE). The training procedure of ps-DRAE is similar to s-DRAE, namely, it includes computation of the reconstruction loss and the PSSFG between the empirical distribution of the prior and the encoded latent distribution.

E Additional experiments

In this appendix, we provide qualitative results on MNIST and CelebA datasets, including randomly generated images, reconstruction images, and latent space visualizations for the autoencoders in the main paper. In addition, we provide more applications where the proposed discrepancies can be applied. The first application in Section E.4 trains autoencoder via ex-post density estimation, which is a new procedure to train an autoencoder and is proposed currently in (Ghosh et al., 2019). In this application, we compare the SSFG with SFG and max-SFG to show the benefits of the SSFG. The second application is GAN (Goodfellow et al., 2014) which is used to learn a generator only. In this second application, we compare qualitatively our spherical sliced discrepancies (SSFG and MSSFG) with SFG and max-SFG.

E.1 Results on spherical deterministic relational autoencoder

Visualization of the latent space: First, we plot the t-SNE visualization of the latent distribution of various types of autoencoders. For autoencoders that have a mixture of Gaussian distributions prior, the Gaussian means are also visualized. In Figure 6 with MNIST dataset, we find that Vampprior only learns collapsed Gaussian distributions that cannot cover the encoded distribution. Regarding

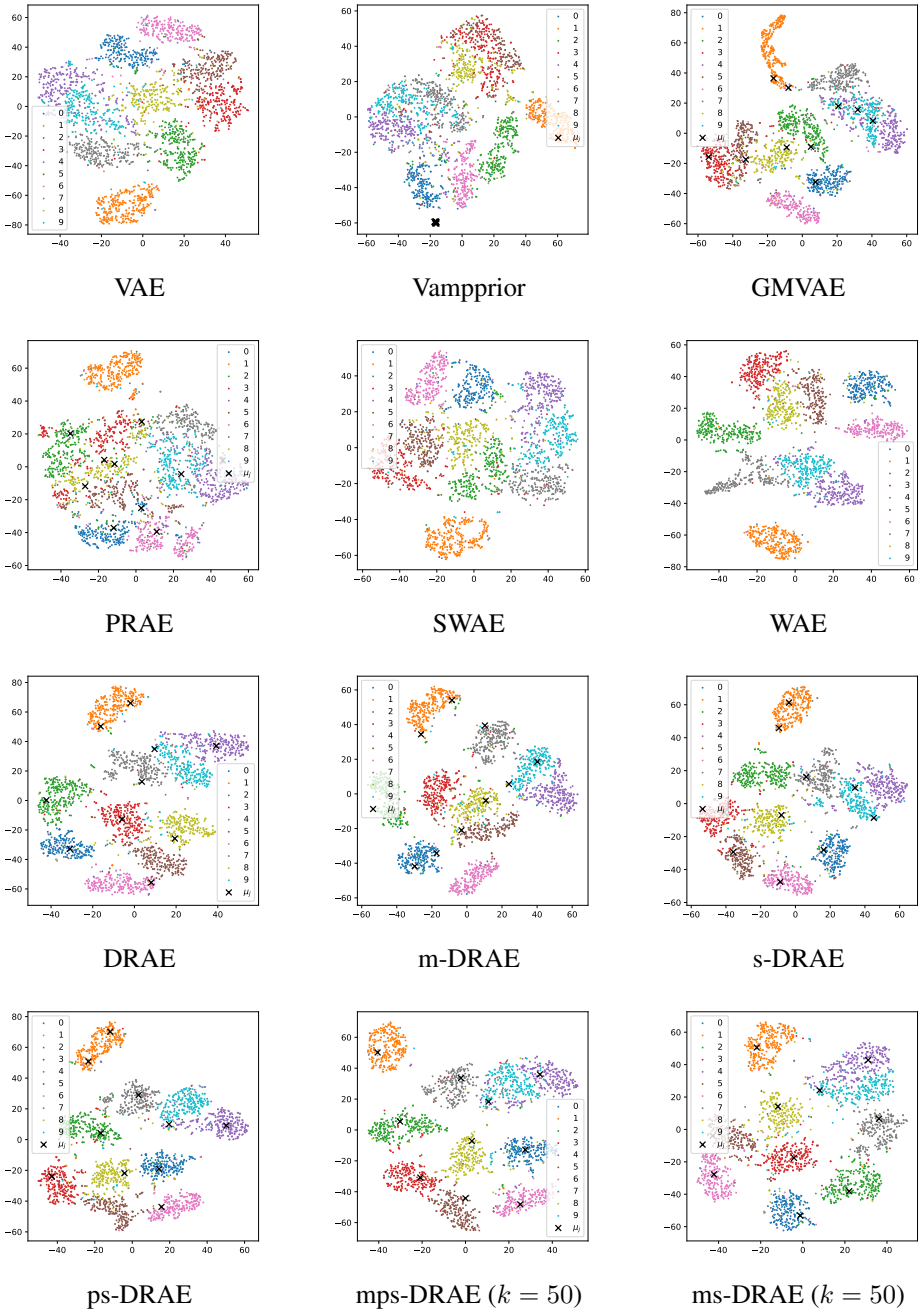


Figure 6: t-SNE on MNIST latent code, the μ_j are the means of components in the Gaussian mixture prior.

GMVAE, its latent modes are overlapped, which might affect the generative diversity. For PRAE, a probabilistic relational regularized autoencoder, its Gaussian means seem to cover all the space but the clustering effect is not obvious. Therefore, the generated images from PRAE might be blurred. Three deterministic relational regularized autoencoders (DRAE, m-DRAE, s-DRAE) have diverse Gaussian components, which cover the latent space well, however, they still miss some small parts. With the CelebA dataset in Figure 7 whose latent space dimension is larger than that of MNIST (64 dimensions compared to 8 dimensions), the differences between DRAE, m-DRAE, and s-DRAE can be seen clearly. In particular, s-DRAE gets a better visualized latent space and its prior can cover well that space while m-DRAE and DRAE seem to miss some parts of the space.

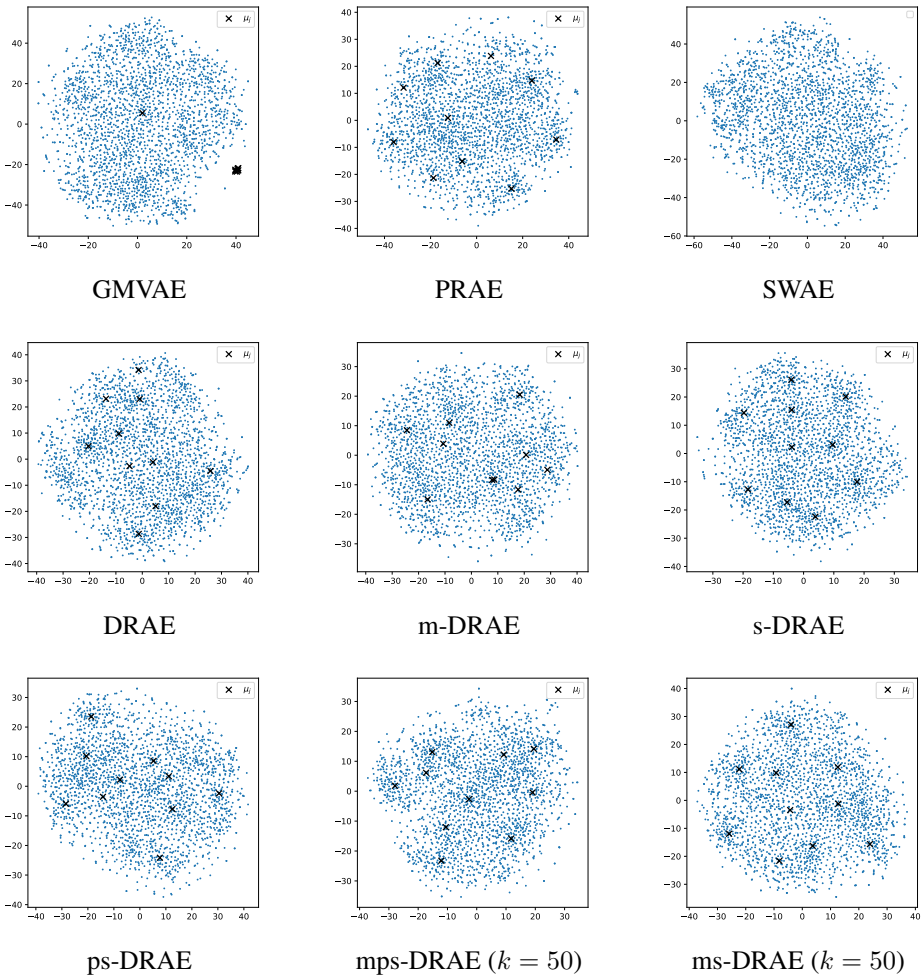


Figure 7: t-SNE on CelebA latent code, the μ_j are the means of components in the Gaussian mixture prior.

Synthesis images: We first present randomly generated samples on MNIST dataset in Figure 8. Based on these images, we can see that DRAE, m-DRAE and s-DRAE produce significantly more realistic images than other autoencoders. Because of the collapsing prior (see Figure 6), Vampprior cannot generate diverse digits. Furthermore, by looking closely, images from s-DRAE are clear and can be recognized as digits while some images from DRAE and m-DRAE are blurred.

Now, we move to the generated images with CelebA datasets in Figure 9. We see that GMVAE and PRAE are not able to generate acceptable images. These two probabilistic autoencoders seem unstable to train on CelebA dataset. On the other hand, images obtained from DRAEs, i.e., DRAE, m-DRAE, and s-DRAE, are significantly better than other autoencoders. Nevertheless, just by only looking at the generated images, it is hard to know which autoencoder among these three autoencoders can produce better images. In order to evaluate their performances, we use FID scores and present them in Table 1. According to those tables, s-DRAE is better than DRAE, m-DRAE and previous autoencoders on CelebA dataset.

Reconstruction images: Regarding the reconstruction ability, we show the reconstructed test-set images on MNIST dataset in Figure 10, and the reconstructed images test-set images on CelebA dataset in Figure 11. These results show that relational regularizations in DRAE, m-DRAE, s-DRAE do not harm the reconstruction ability of the autoencoders and the reconstruction images from these models have at least the same quality as other considered autoencoders.

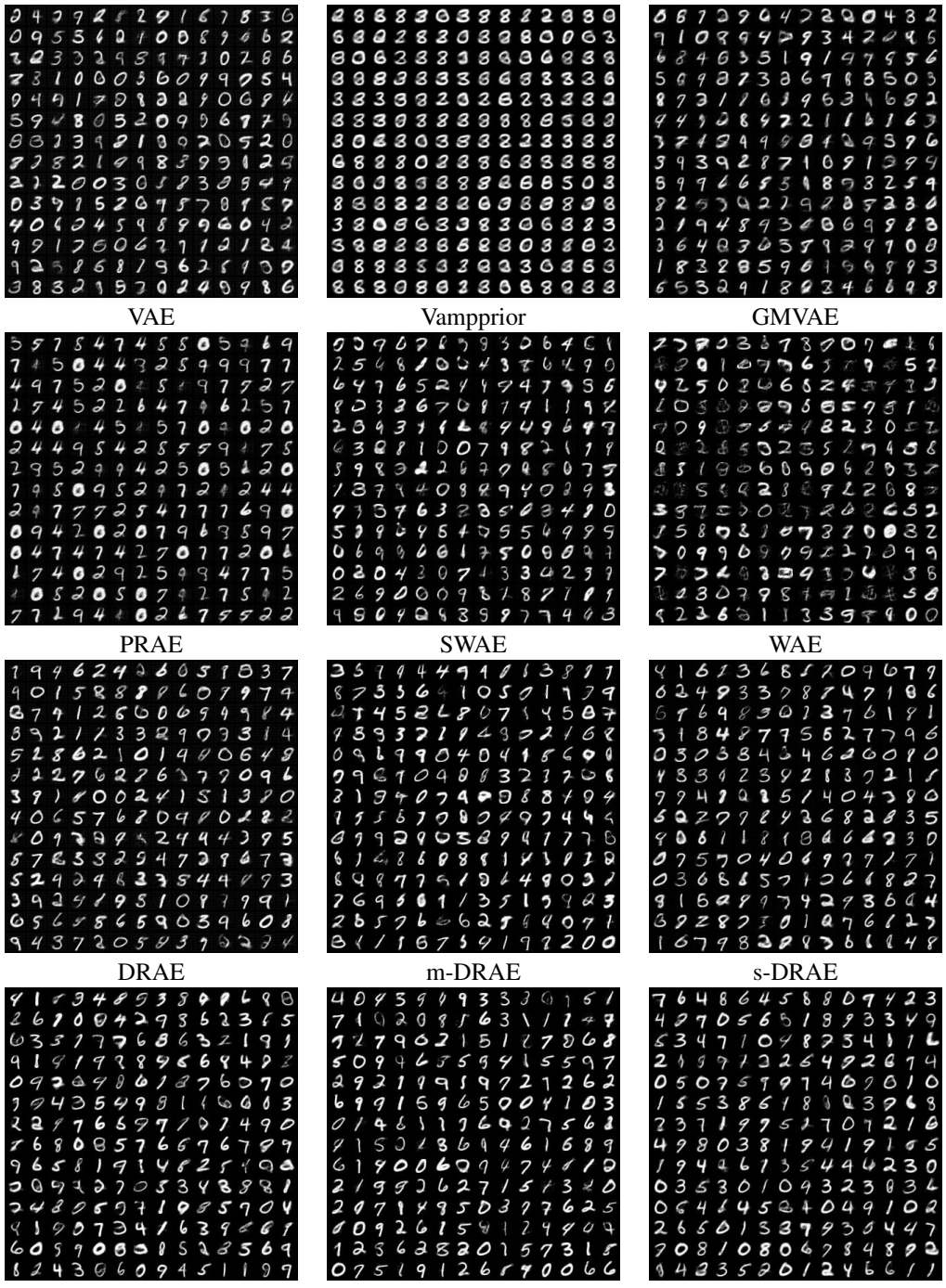


Figure 8: Generated images on MNIST dataset. Here, k denotes the number of components of mixture of vMF distributions in MSSFG.

Sensitivity to the concentration parameter: Here, we investigate the effect of κ in training s-DRAE. We set the number of slices equal to 50, and $\kappa \in \{0.001, 0.01, 0.1, 1, 5, 10, 50, 100\}$ and report reconstruction loss and FID score in Figure 12 and Figure 13 . We find that the performance of s-DRAE is close to that of DRAE when $\kappa = \{0.001, 0.01, 0.1, 1\}$, namely, the reconstruction loss and FID score is nearly equal to the scores of DRAE . On the other extreme, when $\kappa = 100$, s-DRAE behaves like m-DRAE in both evaluation metrics. It confirms our claim that SSFG is a

Figure 9: CelebA generated images.

generalization of m-DRAE and DRAE. About the best choice of κ, on MINIST, the best value of κ is 10 for both evaluation metrics. In detail, FID score and reconstruction loss decrease considerably when setting κ from 1 to 10, then increase when $\kappa > 10$. Meanwhile, on CelebA, the best value of κ for reconstruction loss and FID score are different. With $\kappa = 10$, s-DRAE reaches the best FID score of about 46.6, while $\kappa = 1$ produces the lowest reconstruction loss. This partly explains why our s-DRAE gets the best FID score, not the best reconstruction loss among all tested autoencoders.

Sensitivity to the size of minibatch: We conduct an experiment to see how the minibatch's size affects the performance of s-DRAE. In particular, we set the minibatch's size in $\{10, 50, 100, 150\}$ and adjust the number of epochs to get the same number of iterations for each setting. The experiment results are given in Table 2. These results show that the size of minibatch has a crucial role in the quality of both DRAE and s-DRAE. The bigger minibatch size, the lower FID score and reconstruction score.

E.2 Results on mixture spherical deterministic relational autoencoder

As mentioned in Section 3.2 and Appendix C, there is an extension of SSFG that uses mixtures of vMF as the slicing distribution over directions and we denote the RAE using this new discrepancy as

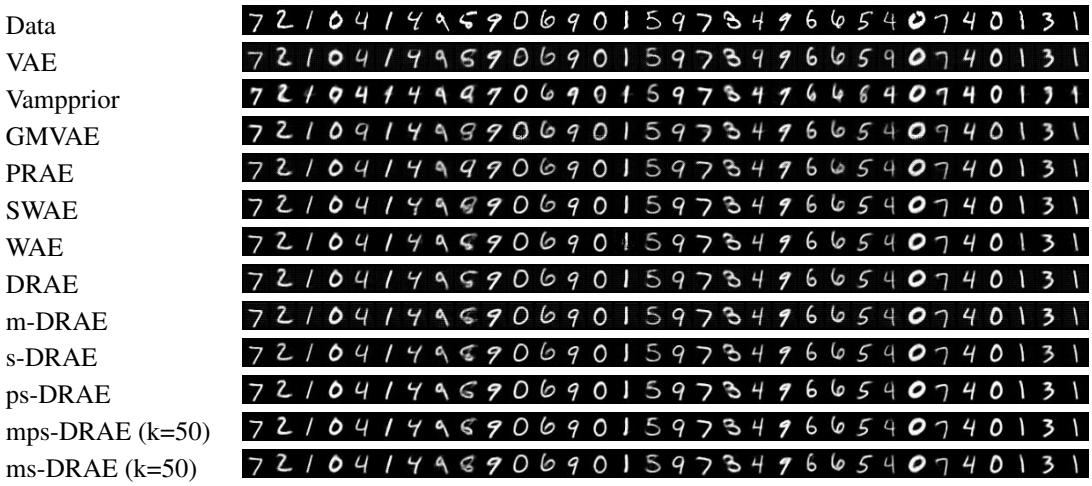


Figure 10: Reconstruction images on MNIST test set.

Figure 11: Reconstruction images on CelebA test set.

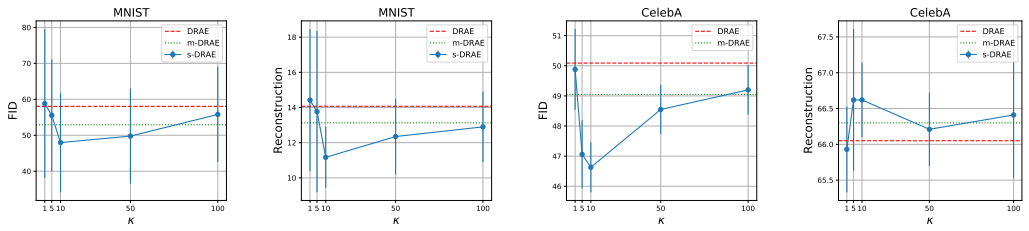


Figure 12: The performance of s-DRAE ($L = 50$) on MNIST and CelebA datasets when changing the value of $\kappa \in \{1, 5, 10, 50, 100\}$.

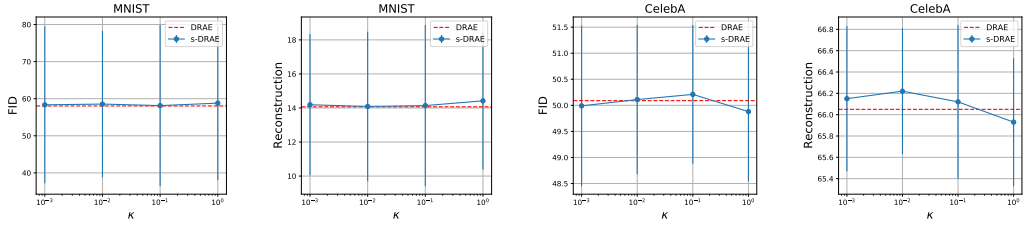


Figure 13: The performance of s-DRAE ($L = 50$) on MNIST and CelebA datasets when changing the value of $\kappa \in \{0.001, 0.01, 0.1, 1\}$.

Table 2: FID scores and reconstruction losses of DRAE, s-DRAE, and ps-DRAE with different sizes of minibatches.

Method	Minibatch's size	MNIST		CelebA	
		FID	Reconstruction	FID	Reconstruction
DRAE	10	100.31	20.51	67.41	72.34
DRAE	50	73.41	16.99	58.55	67.28
DRAE	100	58.04	14.07	50.09	66.05
DRAE	150	55.64	13.21	47.67	65.98
s-DRAE	10	88.45	22.72	67.02	67.73
s-DRAE	50	60.11	14.57	51.02	66.98
s-DRAE	100	47.97	11.17	46.63	66.62
s-DRAE	150	47.01	11.06	46.04	66.53
ps-DRAE	10	90.15	21.04	68.38	67.54
ps-DRAE	50	62.33	14.54	55.68	66.87
ps-DRAE	100	49.15	11.71	48.21	66.31
ps-DRAE	150	48.89	11.82	48.19	66.33

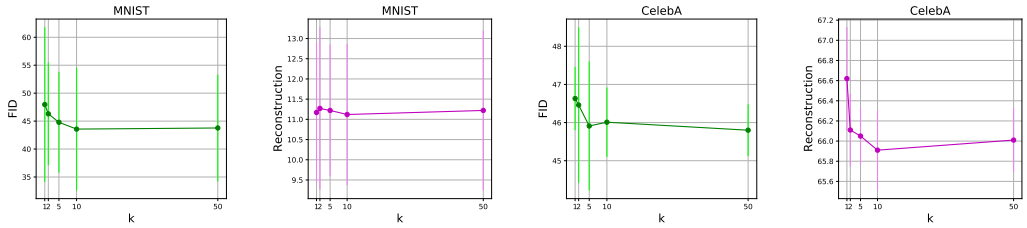


Figure 14: The performance of ms-DRAE when changing the number of vMF components (k).

ms-DRAE. In practice, we use the uniform weight $\alpha_i = \frac{1}{k}$ for the vMF mixture and use the same value of κ for every component. For the number of projections, we set $L = 50$.

Visualization of the latent space: In Figure 6, with MNIST dataset, the ms-DRAE performs very well, with $k = 10$ and $k = 50$, ms-DRAE does not miss any data's mode, the clustering effect is also very clear. With CelebA dataset in Figure 7, like s-DRAE, the latent space visualization of ms-DRAE with $k \in \{10, 50\}$ is well covered by the mixtures of Gaussian prior.

Synthesis images: As shown in Figure 8, generated MNIST images from ms-DRAE are very realistic and easy to classify. With CelebA dataset, in Figure 9, ms-DRAE can also produce the highest quality images among considered autoencoder. This quality is quantified in Table 1, ms-DRAE gets the best FID score among all autoencoders.

Reconstruction images: According to the reconstructed test-set images on MNIST dataset in Figure 10, and the reconstructed images test-set images on CelebA dataset in Figure 11, ms-DRAE provides reconstructed images that are at least comparable to other autoencoders.

Increasing the number of vMF components: We conduct experiment to see the effect of increasing the number of components, says k, and report results in Figure 14. For each value of $k \in \{1, 2, 5, 10, 50\}$ we search for its best value of the concentration parameter $\kappa \in \{1, 5, 10, 50, 100\}$. The figure shows that more vMF components enhance the quality of the generator on MNIST until $k = 10$, then flats after that. The reconstruction loss on MNIST changes slightly when k increases, but it always between 11.0 and 11.5. Whereas on CelebA dataset, until $k = 5$, both FID and reconstruction loss go down sharply. When $k = 10$, the FID score rises a little bit to about 46, while reconstruction loss continues to fall. In contrast, when $k = 50$, FID score is reduced to about 45.8 while reconstruction loss climbs to about 66. Overall, increasing the number of vMF components can affect positively the performance of the learned autoencoder.

E.3 RESULTS ON POWER SPHERICAL DETERMINISTIC RELATIONAL AUTOENCODER

In this appendix, we provide additional experiments with power spherical deterministic relational autoencoder (ps-DRAE).

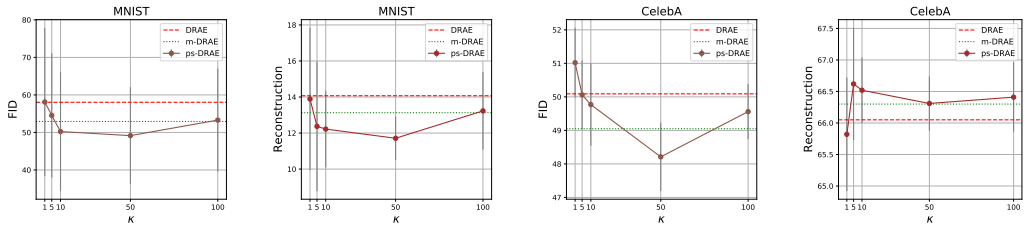


Figure 15: The performance of ps-DRAE ($L = 50$) on MNIST and CelebA datasets when changing the value of $\kappa \in \{1, 5, 10, 50, 100\}$.

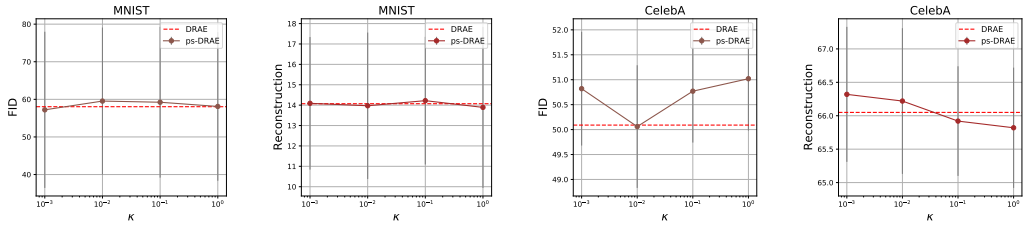


Figure 16: The performance of ps-DRAE ($L = 50$) on MNIST and CelebA datasets when changing the value of $\kappa \in \{0.001, 0.01, 0.1, 1\}$.

Visualization of the latent space: In Figure 6, with the MNIST dataset, the ps-DRAE produces a comparable latent structure to s-DRAE, namely, its mixture Gaussian prior can capture quite well the latent code. With the CelebA dataset in Figure 7, the latent space visualization of ps-DRAE is also well covered by the prior.

Synthesis images: As shown in Figure 8, generated MNIST images from ps-DRAE are comparable to s-DRAE, namely, the images are easy to classify into classes. With the CelebA dataset, in Figure 9, ps-DRAE can also produce the good images of human faces.

Reconstruction images: We find that ps-DRAE is also good at reconstructing images as s-DRAE. In Figures 10, and 11, reconstructed images of ps-DRAE are also similar to the ground truth images.

Sensitivity to the concentration parameter κ: We find that the ps-DRAE has a similar effect as s-DRAE when we change the value of the concentration parameter κ. In particular, we plot the FID score and the reconstruction loss of ps-DRAE in Figure 15 and Figure 15 for each value of $\kappa \in \{0.001, 0.01, 0.1, 1, 5, 10, 50, 100\}$. On the MNIST dataset, when $\kappa = 1$, both metrics of ps-DRAE are close to the values of DRAE. When $\kappa = 100$, ps-DRAE behaves like m-DRAE in both FID score and reconstruction loss. When $\kappa \in \{5, 10, 50\}$, ps-DRAE has better FID score and reconstruction loss than both DRAE and m-DRAE. On the CelebA dataset, we also observe quite similar phenomenon with the FID score as that of the MNIST dataset. The reconstruction losses of ps-DRAE and those of m-DRAE and DRAE are very close regardless of the choice of κ.

Sensitivity to the size of minibatch: Like DRAE and s-DRAE, minibatch's size affects the performance of ps-DRAE greatly (cf. Table 2), namely, a larger minibatch helps to learn better an autoencoder in both generation and reconstruction.

E.4 EX-POST DENSITY ESTIMATION AUTOENCODERS

In order to demonstrate the favorable performance of the SSFG, PSSFG and MSSFG over SFG, we adapt the DRAE framework to the ex-post density estimation procedure and test its generative quality.

Ex-post density estimation is a new procedure for training a generative autoencoder. It was proposed by Ghosh et al. (2019) and consists of two main steps. The first step is to train a regularized autoencoder by using the following objective:

$$\min_{\theta,\phi} \mathbb{E}_{p_d(x)}[\|x - G_\theta(E_\phi(x))\|_2^2 + \lambda_1 \|E_\phi(x)\|_2^2] + \lambda_2 \|\theta\|_2^2, \tag{19}$$

where λ_1, λ_2 are regularized positive parameters that will be chosen. After we find the optimal parameters θ^* and ϕ^* of the above objective function, the second step is to fit a density estimator $p_\psi(z)$ to the latent distribution $q_E(z) := \frac{1}{N}\sum_{i=1}^N \delta_{E_{\phi^*}(x_i)}$. To generate samples from this model, we will first sample $z \sim p_\psi(z)$ and then get a new $x = G_{\theta^*}(z)$.

In the experiments with ex-post density estimation, we compare SSFG and other relational discrepancies in the training procedure. More specifically, we learn an autoencoder with $\lambda_2 = 1$ and $\lambda_1 = 0.1$ on the MNIST dataset (and $\lambda_2 = 1$ and $\lambda_1 = 1$ on the CelebA dataset) in the first step with 50 epochs. After this step, the trained autoencoder is shared among all methods. In the next step, we again choose $p_\psi(z)$ as a mixture of Gaussian distributions and fit it to the latent distribution using relational discrepancies, e.g., SFG, max-SFG, SSFG, PSSFG, and MSSFG with again 50 epochs.

Generative quality: We compute the FID score on MNIST dataset and CelebA dataset, then present them in Table 3. According to this table, SSFG achieves a better FID score than SFG and max-SFG on this application. Compare to the traditional training of DRAE, the FID score on MNIST is significantly improved, it is better than the score in Table 1. For example, s-DRAE gets 47.97 while s-DRAE with ex-post density estimation training gets 37.42. On CelebA dataset, the new procedure performs worse than the traditional procedure in Table 1, however, SSFG still gives the lowest FID score among all distances. Moreover, PSSFG also performs well in this task, it gives a comparable result to SSFG. About MSSFG, with more vMF components, the FID scores of MSSFG in both datasets decrease. So, MSSFG becomes the best choice of discrepancy for this task.

Table 3: FID table of ex-post density estimation autoencoders

Method	MNIST	CelebA
SFG	41.85 $\pm$ 12.29	60.28 $\pm$ 2.56
max-SFG	40.69 $\pm$ 5.96	60.06 $\pm$ 2.45
SSFG	37.42 $\pm$ 6.06	58.8 $\pm$ 1.97
PSSFG	38.05 $\pm$ 5.17	58.76 $\pm$ 1.88
MSSFG (k=5)	33.54 $\pm$ 7.12	58.21 $\pm$ 1.75
MSSFG (k=10)	33.16 $\pm$ 6.96	57.48 $\pm$ 1.72
MSSFG (k=50)	**33.11** $\pm$ **6.99**	**57.22** $\pm$ **1.7**

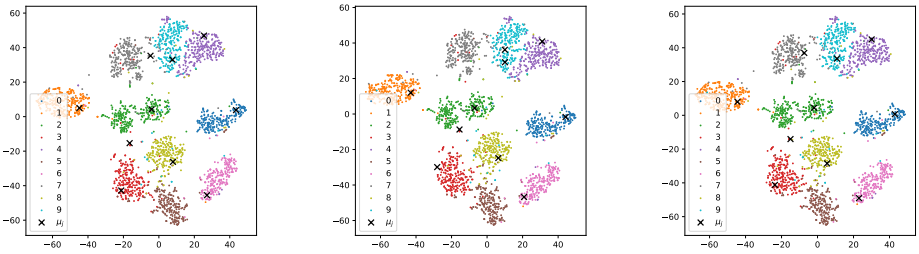
Visualization of the latent space: Next, we show the t-SNE visualization of the autoencoder. On MNIST, the mixture of Gaussian prior learned with MSSFG (k=50) can cover all the modes of the latent code. SSFG and PSSFG performs quite well, they only miss one mode. About SFG and Max-SFG, they both miss two modes of the latent code distribution. On CelebA, there is not too much difference, however, MSSFG seems to learn Gaussian means the best, those can cover almost all the latent space.

We show the generated images from trained models in Figure 19 and Figure 20 (the reconstruction images are not shown because they are the same among all methods due to the shared first step in ex-post density estimation procedure). It is easy to see that the quality of SSFG's images is slightly better than SFG and Max-SFG.

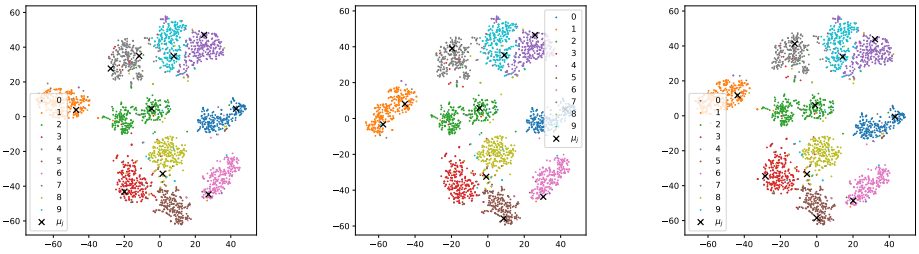
E.5 GAN MODELS

Authors in (Bunne et al., 2019) have introduced a generative adversarial net (Goodfellow et al., 2014) variant that is based on Gromov Wasserstein, and it is adapted to used sliced Gromov Wasserstein in (Vayer et al., 2019). In this work, we replace SG by our spherical discrepancies and compare their performance in this application.

$$G^* = \arg\min_G D(p_{data}, G(Z)), \tag{20}$$

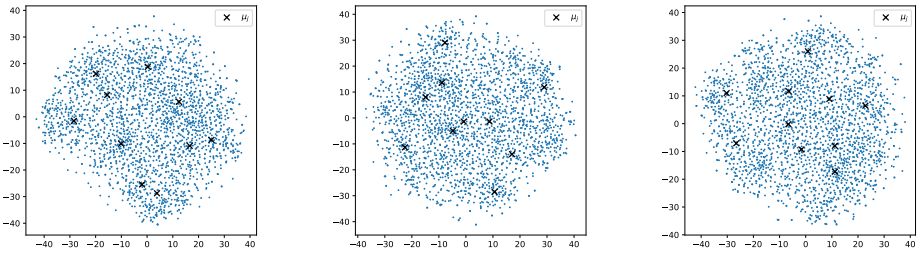


SFG Max-SFG SSFG

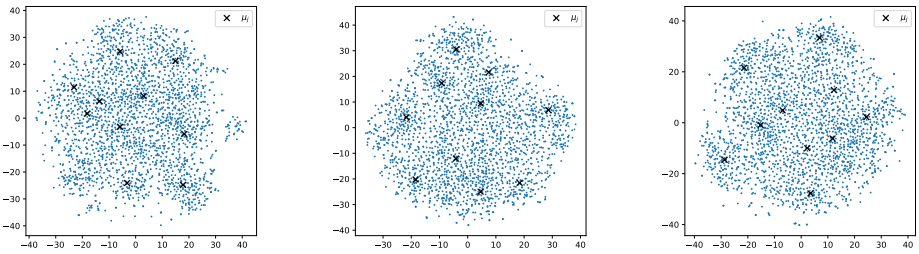


PSSFG MSSFG ($k = 10$) MSSFG ($k = 50$)

Figure 17: t-SNE on MNIST latent code with ex-post density estimation procedure, the μ_j are the means of components in the Gaussian mixture prior.



SFG Max-SFG SSFG



PSSFG MSSFG ($k = 10$) MSSFG ($k = 50$)

Figure 18: t-SNE on CelebA latent code with ex-post density estimation procedure, the μ_j are the means of components in the Gaussian mixture prior.

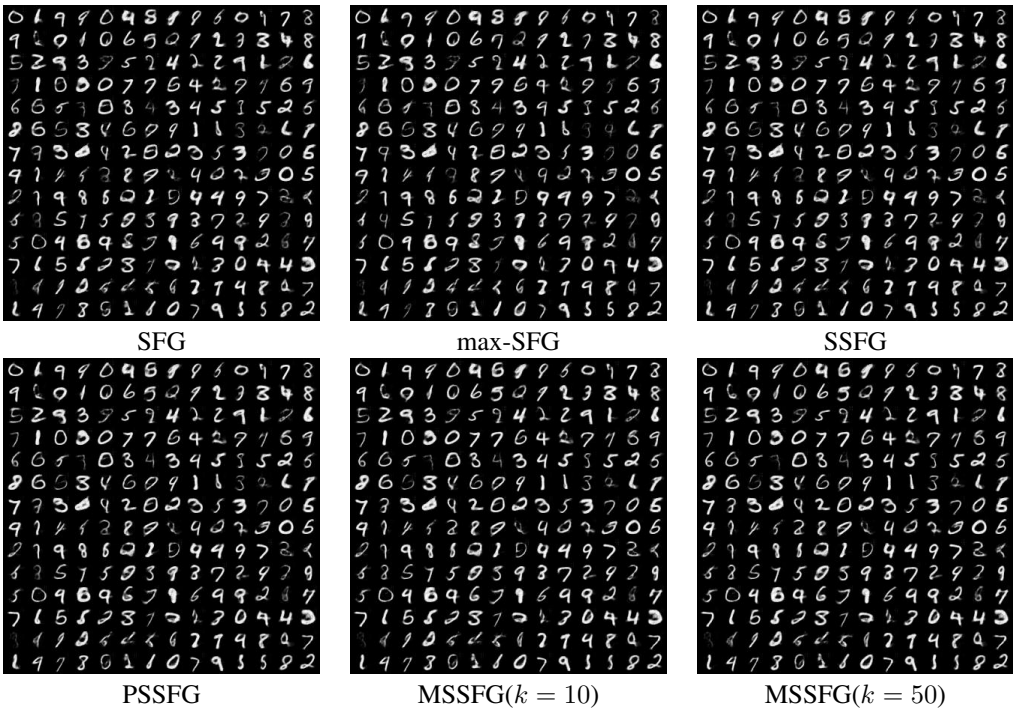


Figure 19: MNIST ex-post density estimation generated images

Figure 20: CelebA ex-post density estimation generated images

where Z is a low-dimensional random noise distribution (typically Gaussian), $G(Z)$ transforms Z to a desired dimensional space.

In this toy experiment, we learn a G function that is parametrized by a neural network to generate 4 Gaussian modes in 2-dimension. All settings are adapted from `https://github.com/bunnech/gw_gan`, and we choose $\kappa = 1000$ for SSFG.

In Figure 22 , we show the learned distributions through iterations. The qualitative results show that max-SFG, SSFG, PSSFG and MSSFG helps the distribution converge faster to the target than SFG, the evidence is that the model distributions in the iteration 1000 of max-SFG and SSFG look more similar to 4 Gaussian modes. The difference between SSFG, max-SFG, PSSFG and MSSFG is not too clear, however, in the iteration 1000 we can see that MSSFG's distributions do not put too many samples on the intersection area between modes while distributions of other discrepancies do.

E.6 Image color adaptation

In this section, we show how our proposed sliced-fused discrepancies can be applied to image color adaptation (color transfer) task, which is the most well-known application of optimal transport (Rabin et al., 2014; Bonneel et al., 2015; Perrot et al., 2016). By using the vMF distribution, the Power Spherical distribution, and their mixtures to capture area of "informative" directions, we can enhance the performance of the sliced based color transfer algorithms (Muzellec & Cuturi, 2019; Bonneel et al., 2015; Rabin et al., 2010). First, we get the quantizations of two input images by K-means algorithm (with 3000 clusters) to reduce the size of the images. With two compressed images, we find the best von Mises-Fisher distribution (or power spherical distribution, or mixture of vMF distributions, or mixture of power spherical distributions) by maximizing the expectation of the 1D Wasserstein distance. After that, we draw samples (directions) from these distributions and then find a optimal 1D Monge map between projected distributions of each direction. Finally, we move the color plate of the source image to the color plate of the target image by the found Monge maps, then average them to get the final result.

We present the qualitative results in Figure 21. We illustrate the adapted images between the source images and the target images. Note that in all these experiments, we set $\kappa = 10$ in (M)vMF and (M)PS. Furthermore, the subscript number is the number of directions that is used, and the superscript number is the number of components of mixture of vMF (respectively PS). According to the experiment results, images from the uniform distribution (i.e., normal slicing methods) are quite blurred and not catchy. "Max" direction often creates noises in the images (see the first two examples in Figure 21). On the other hand, the proposed discrepancies vMF, PS, MvMF, and MPS perform very well, namely, their adapted images have similar color style to the corresponding target images. It demonstrates that in the color transfer task, the proposed sliced-fused discrepancies also outperform both the uniform and max-sliced approaches. Finally, in the final experiment in Figure 21, we would like to remark that the orange color from the suns in the target image is still in the transferred images though it is quite blurred and may look a bit like yellow color. There are two reasons for this phenomenon. First, the K-means clustering step uses the yellow color to represent the cluster that contains the orange pixels, which reduces the brightness of the orange pixels. This phenomenon happens because there are more yellow pixels than orange ones. We think that increasing the number of clusters of K-means may help but with the cost of slower computation. Second, the average of transferred images in the final step might reduce the brightness of the color. It is due to the difference between the 1D transport maps which are found by different projecting directions in the sliced distances. For example, in one map, a pixel is transferred to the orange color; however, in another map it is transferred to the blue color. Therefore, the average will be no longer orange. Developing an efficient way to deal with these two possible problems in the color transfer tasks is an important direction and we leave it for the future work.

F Experimental settings

In this section, we provide detailed settings of experiments that we conduct in this paper.

F.1 Neural Network architecture

For MNIST dataset:

Encoder E: $x \in \mathbb{R}^{28\times 28} \to \text{Conv}_{128} \to \text{BatchNorm} \to \text{ReLU} \to \text{Conv}_{256} \to \text{BatchNorm} \to \text{ReLU} \to \text{Conv}_{512} \to \text{BatchNorm} \to \text{ReLU} \to \text{Conv}_{1024} \to \text{BatchNorm} \to \text{ReLU} \to \text{FC}_8 \to z \in \mathbb{R}^8$

Decoder G: $z \in \mathbb{R}^8 \to \text{FC}_{7x7x1024} \to \text{BatchNorm} \to \text{ReLU} \to \text{FSConv}_{512} \to \text{BatchNorm} \to \text{ReLU} \to \text{FSConv}_{256} \to \text{BatchNorm} \to \text{ReLU} \to \text{FSConv}_1 \to x \in \mathbb{R}^{28\times 28}$

where Conv_k denotes for a convolution with k 4×4 filters, FSConv_k denotes the fractional strided convolution with k 4×4 filters and FC_k is the fully connected layer mapping to $\mathbb{R}^k$. For VAE, PRAE, GMVAE and Vampprior, the encoder contains two FC layers for the mean and the logarithmic variance in the last layer.

For CelebA dataset:

Encoder E: $x \in \mathbb{R}^{64\times 64\times 3} \to \text{Conv}_{128} \to \text{BatchNorm} \to \text{ReLU} \to \text{Conv}_{256} \to \text{BatchNorm} \to \text{ReLU} \to \text{Conv}_{512} \to \text{BatchNorm} \to \text{ReLU} \to \text{Conv}_{1024} \to \text{BatchNorm} \to \text{ReLU} \to \text{FC}_6 4 \to z \in \mathbb{R}^{64}$

Decoder G: $z \in \mathbb{R}^{64} \to \text{FC}_{8x8x1024} \to \text{BatchNorm} \to \text{ReLU} \to \text{FSConv}_{512} \to \text{BatchNorm} \to \text{ReLU} \to \text{FSConv}_{256} \to \text{BatchNorm} \to \text{ReLU} \to \text{FSConv}_3 \to x \in \mathbb{R}^{64\times 64\times 3}$

The Conv in CelebA also uses 4×4 filters.

F.2 Hyperparameter settings

To train the autoencoder, we use Adam optimizer Kingma & Ba (2014) with learning rate equals 0.001, $\beta_1 = 0.5$, $\beta_2 = 0.999$. The number of epochs is 50 on MNIST and 40 on CelebA, batch size is 100 for both datasets. The coefficient λ (autoencoder regularization) is 1. The number of components of Gaussian mixture prior is set to 10. The fused parameter of fused Gromov Wasserstein $\beta = 0.1$. The number of projections of sliced-discrepancies is 50 for fairness.

For max-SFG, SSFG, PSSFG and MSSFG: The maximum iteration of the optimization for max-direction (distribution over directions) is 10.

F.3 Reconstruction and FID computation

The reconstruction score is computed by the Mean square error between reconstructed images and original images in corresponding test set.

The FID score is computed between 10000 randomly generated images and all images from test set (for evaluation), and all images from validation set (for model selection). We use the implementation at `https://github.com/bioinf-jku/TTUR`.

F.4 Tuning parameters

For SSFG, PSSFG and MSSFG: We choose the κ that has the lowest FID score on corresponding validation set.

F.5 Code and computing system

We use the code for SFG and autoencoder-baselines from `https://github.com/HongtengXu/Relational-AutoEncoders`, the code for GAN from `https://github.com/bunnech/gw_gan`, We run code on a single NVIDIA RTX 2080 Ti.

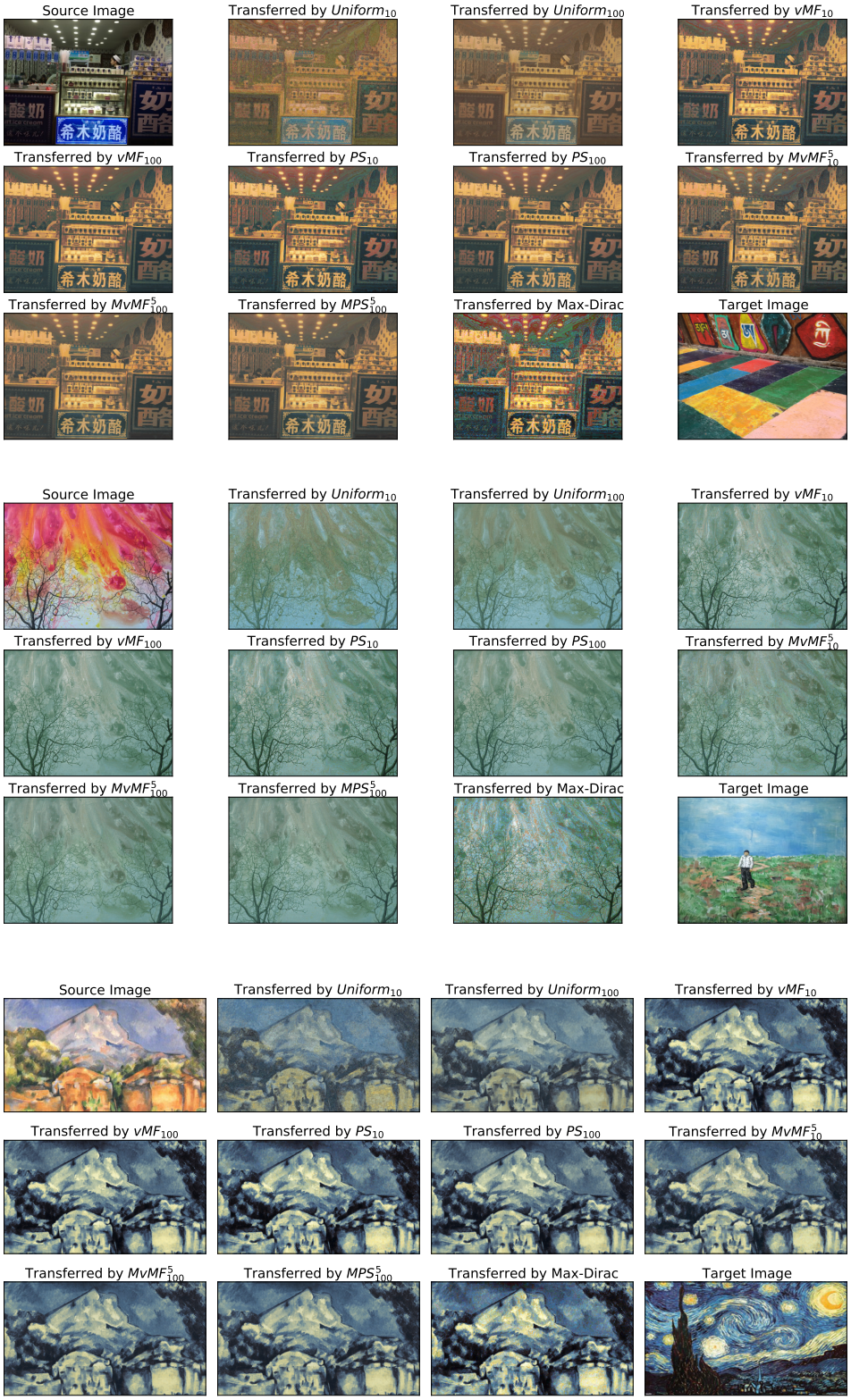


Figure 21: Sliced-based image color adaptation using various distributions over projecting directions. Images are taken from (Muzellec & Cuturi, 2019), `https://github.com/HYPJUDY/color-transfer-between-images/tree/master/images`.

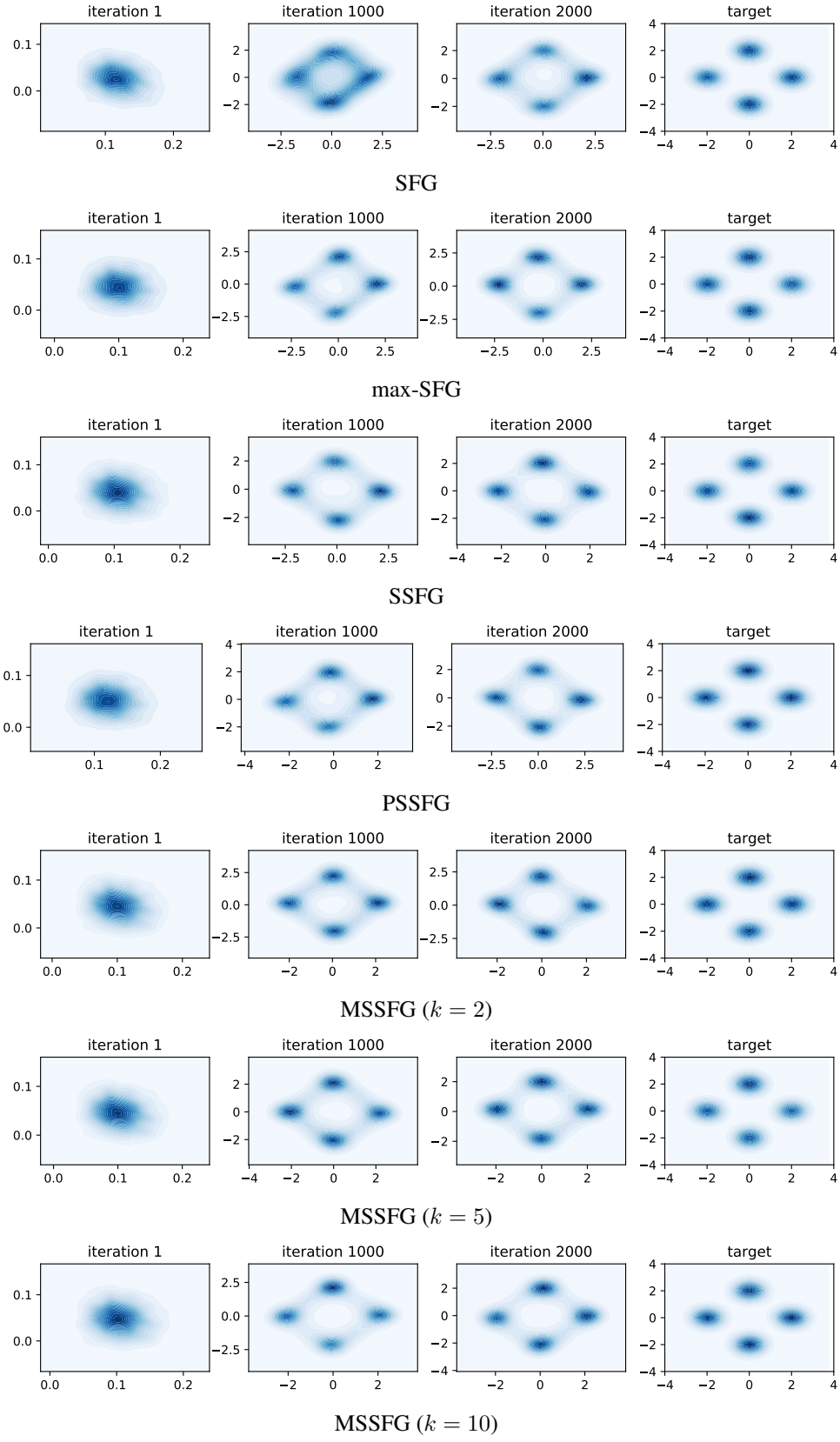


Figure 22: 4 Gaussian modes generation by variants of fused Gromov Wasserstein