TP - SLICED WASSERSTEIN FLOW AND COLOR TRANSFER

ANATOLE GALLOUËT

1. Introduction

In this work, we are going to implement an estimator of the Sliced Wasserstein distance between two measures. Then compute its gradient with respect to the position of one of the two measures in order to follow a gradient descent of the distance. This gradient descent can be implemented on measures representing RGB images, allowing to apply the color palette of an image to another one, which we call color transfer.

1.1. **The Sliced Wasserstein distance.** The main idea behind Sliced Wasserstein (SW) is that there exists a closed form formula for the Wasserstein distance in dimension 1. It reads for $\mu, \nu \in \mathcal{P}(\mathbb{R})$,

$$W_p^p(\mu,\nu) = \int_0^1 |F_\mu^{-1}(t) - F_\nu^{-1}(t)|^p \mathrm{d}t$$

where F_μ and F_ν are the c.d.f of μ and ν. This formula becomes even more interesting when μ and ν are the sum of m dirac masses with uniform weights:

$$\mu = \frac{1}{m}\sum_{i=1}^m \delta_{x_i} \text{ and } \nu = \frac{1}{m}\sum_{i=1}^m \delta_{y_i}. \tag{1}$$

It can then be expressed as

$$W_p^p(\mu,\nu) = \frac{1}{m}\sum_{i=1}^m \left\| x_{\sigma(i)} - y_{\kappa(j)} \right\|^p,$$

where $\left(x_{\sigma(i)}\right)_{1\le i\le m}$ and $\left(y_{\kappa(j)}\right)_{1\le j\le m}$ are the sorted versions of $(x_i)_{1\le i\le m}$ and $(y_j)_{1\le j\le m}$. The computational benefits of this expression are consequent, as the computing cost of the 1d Wasserstein distance is the sort of both point clouds, which is $\mathcal{O}(m\log(m))$. The SW distance is a generalization of this idea to higher dimension, taking the 1D wasserstein of projected measures on a direction θ and integrating over all possible direction $\theta \in \mathcal{S}^{d-1}$. Let σ be the uniform measure on $\mathcal{S}^{d-1}$ and $\mu, \nu \in \mathcal{P}(\mathbb{R}^d)$, the Sliced Wasserstein distance is defined by

$$\mathrm{SW}_p^p(\mu,\nu) = \int_{\mathcal{S}^{d-1}} \mathrm{W}_p^p(P_{\theta\#}\mu, P_{\theta\#}\nu)\mathrm{d}\sigma(\theta)$$

Where $P_\theta : \mathbb{R}^d \to \mathbb{R}$ is the projection in the direction θ, and $P_{\theta\#}\mu$ is the pushforward of μ by P_θ defined for $A \subset \mathbb{R}$ by $P_{\theta\#}\mu(A) = \mu(P_\theta^{-1}(A))$. Since $P_{\theta\#}\mu$ (resp. ν) are 1-d measures, we can compute explicitly the distance

$W_p^p(P_{\theta\#}\mu, P_{\theta\#}\nu)$ using the above formula. When both measures are the sum of m uniform diracs as in (1), SW rewrites

$$\mathrm{SW}_2^2(\mu,\nu) = \int_{\mathcal{S}^{d-1}} \frac{1}{m}\sum_{i=1}^{m} \langle x_{\sigma_\theta(i)} - y_{\kappa_\theta(i)} | \theta\rangle^2 \mathrm{d}\theta$$

where σ_θ and κ_θ are permutations such that $(\langle x_{\sigma_\theta(i)}|\theta\rangle)_i$ and $(\langle y_{\kappa_\theta(i)}|\theta\rangle)_i$ are sorted.

1.2. **Sliced Wasserstein flows.** In this part we interest ourselves in continuously moving a points cloud toward another by doing a gradient descent of the Sliced Wasserstein distance. Consider the sliced wasserstein distance between two discrete measure $\mu = \frac{1}{m}\sum_{i=1}^m \delta_{x_i}$ and $\nu = \frac{1}{m}\sum_{i=1}^m \delta_{y_i}$ as a function of $x \in (\mathbb{R}^d)^m$, and where ν is a fixed measure, namely a function $\mathcal{E}_\nu : (\mathbb{R}^d)^m \to \mathbb{R}$ defined by

$$\mathcal{E}_\nu(x) = \mathrm{SW}_2^2(\mu,\nu) = \int_{\mathcal{S}^{d-1}} \frac{1}{m}\sum_{i=1}^{m} \langle x_{\sigma_\theta(i)} - y_{\kappa_\theta(i)} | \theta\rangle^2 \mathrm{d}\theta \tag{2}$$

Then we have the i-th coordinates of the gradient of $\mathcal{E}$, given by

$$\nabla\mathcal{E}_\nu(x)_i = 2\int_{\mathcal{S}^{d-1}} \frac{1}{m}\langle x_i - y_{\kappa_\theta\circ\sigma_\theta^{-1}(i)} | \theta\rangle\theta \mathrm{d}\theta$$

which can be approximated by a Monte Carlo methods with samples $(\theta_\ell)_{1\le\ell\le L}$ giving

$$\nabla\mathcal{E}_\nu(x)_i \approx \frac{2}{m}\sum_{\ell=1}^{L} \frac{1}{L}\langle x_i - y_{\kappa_{\theta_\ell}\circ\sigma_{\theta_\ell}^{-1}(i)} | \theta_\ell\rangle\theta_\ell$$

Thus defining a Stochastic Gradient descent of the form $x_{k+1} = x_k - \tau_k\nabla\mathcal{E}_\nu(x)$. For appropriate choice of the lerning rate sequence (τ_k), iterate x_k converges toward y.

Color transfer. An image of m pixels can be represented as a vector $x \in (\mathbb{R}^3)^m$ where x_i is the three RGB color of the i-th pixel. To this image X, can be associated a color palette, which is a discrete measure

$$\mu_x = \frac{1}{m}\sum_{i=1}^{m}\delta_{x_i}.$$

Note that in this representation, we lose the ordering of the image pixels, and just focus on the colors it contains. Take another image $y \in (\mathbb{R}^3)^m$, and apply the previous gradient descent to μ_x such that it is equal to μ_y. Then if you watch the image x with its original pixels ordering but shifted in the space of colors you get your original image with the color palette of the second one.

1.3. **Expected work.** The practical work to do is:

- Implement an estimator of the Sliced Wasserstein distance.
- Implement the computation of the gradient of the distance.
- Implement a SW gradient descent for color transfer and reproduce Image 1

FIGURE 1. An example of color transfer. Source (left), Color target (middle) and Transfered (right) images.

It is preferable to implement all computations using pytorch, and if possible with the data on GPU to improve computational speed. The source and color palette image are in the github folder, there is three different sizes (small, medium and big), in order to test validity of the code on the small images, and then improving the performance of the algorithm to make it work on bigger images. Images "source_big.jpg" and "colors_big.jpg" are of order $m = 10^6$ pixels, this means that without efficient GPU implementation, convergence of the SW flow can be quite long.

Evaluation. You will be evaluated on your code (a .py file which should be executable from command line), and a small report explaining your choices of implementation and parameters (preferably latex, html or markdown).

1.4. **Interesting extensions.** The gradient descent can be replaced by a Newton algorithm, by computing the Hessian of the function $\mathcal{E}$ defined in Equation (2). This can also give informations of the ideal learning rate for SW flows, see [2] for details. One can also define the SW barycenter between measures. For images, this implies that one can harmonize the color palette of several images. See [1] for details.

References

[1] N. Bonneel, J. Rabin, G. Peyré, and H. Pfister, "Sliced and Radon Wasserstein barycenters of measures," *Journal of Mathematical Imaging and Vision*, vol. 51, no. 1, pp. 22–45, 2015.

[2] C. Vauthier, A. Korba, and Q. Mérigot, "Towards Understanding Gradient Dynamics of the Sliced-Wasserstein Distance via Critical Point Analysis," *arXiv preprint arXiv:2502.06525*, 2025.