A. Proof of Proposition 2.5

Proof. By (Villani, 2003, Theorem 2.44), if $T(x)$ can be uniquely determined by $T(x) = x - \nabla c^*(\nabla\varphi(x))$, where φ is defined in eq. (2.1), then $T(x)$ is the unique Monge-Kantorovich optimal transference plan. Defining $m := \arg\min_i c(x - y_i) - \hat{\psi}_i$ for convenience, we have

$$\begin{aligned}
& & T(x) &= x - \nabla c^*(\nabla\varphi(x)) \\
&\implies & x - T(x) &= \nabla c^*(\nabla\varphi(x)) \\
& & &= \nabla c^*(\nabla_x \min_i c(x - y_i) - \hat{\psi}_i) \\
& & &= \nabla c^* \nabla_x (c(x - y_m) - \hat{\psi}_m) \\
& & &= \nabla c^* \nabla c(x - y_m) \\
& & &= x - y_m \\
&\implies & T(x) &= y_m \\
& & &= y_{\arg\min_i c(x-y_i)-\hat{\psi}_i}.
\end{aligned}$$

□

B. An Example of Non-Gradual Training

We present in Figure 7 a similar synthetic example to Figure 1, except that FIT step will iterate itself until stuck at a local optimum. The local optimum after the first OTS-FIT run is a zig-zag shaped curve which does not generalize. While alternating the OTS-FIT is able to further push the generated samples to targets, the learned manifold is not as smooth as in Figure 1.

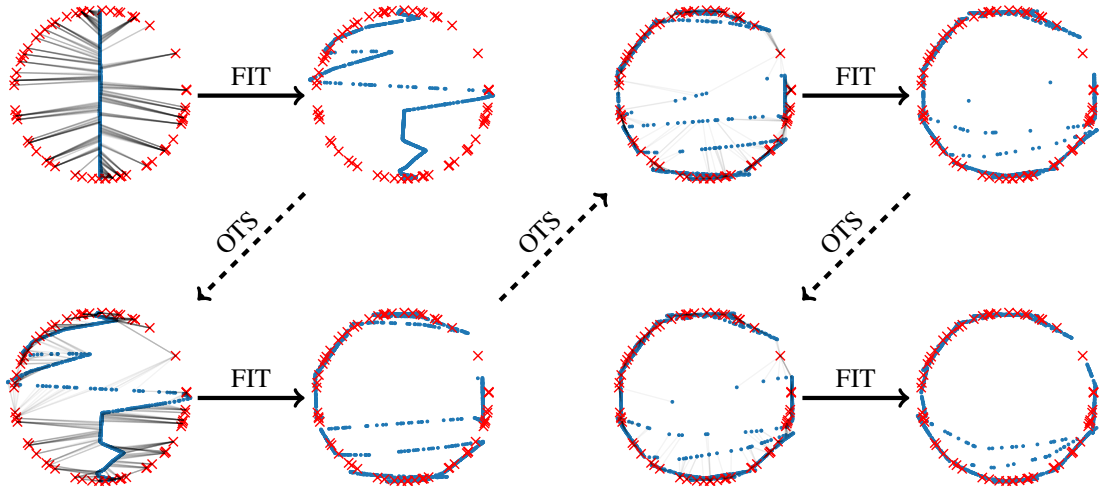


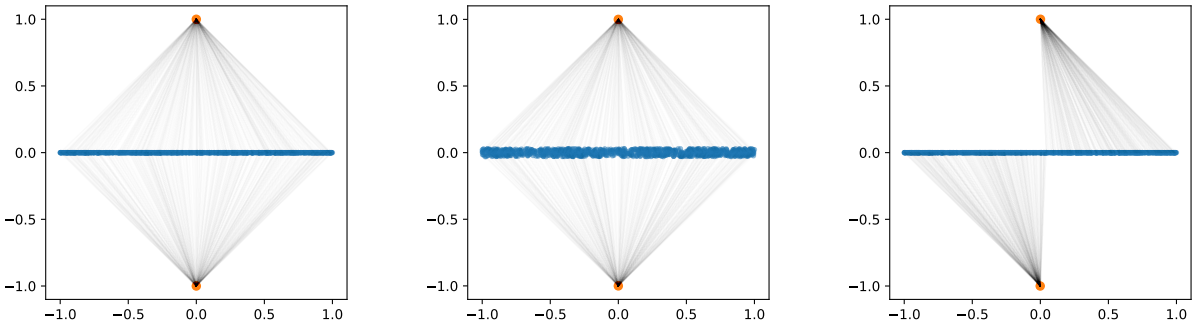
Figure 7. Example of non-gradual training.

C. Further Discussion on the Uniqueness of OT Plan

Continuity of $g\#\mu$ is required in (Villani, 2003, Theorem 2.44), which is violated if $g\#\mu$ lies in a low-dimensional manifold in $\mathbb{R}^d$, or the activation function is not strictly increasing (such as ReLU). When $g\#\mu$ is not continuous, the optimal transference plan is not unique in general, and moreover it is possible that $T(x) := y_{\arg\min_i c(x,y_i)-\hat{\psi}_i}$ is not a transference plan at all (fig. 8(a)).

However, as discussed in Proposition 2.5, the only condition in which the "argmin form" does not characterize a unique Monge OT plan is the existence of ties. To break the ties, one can add arbitrarily small d-dimensional Gaussian noise to the outputs of g (fig. 8(b)). Furthermore, this makes the distribution continuous and satisfies the conditions of (Villani, 2003,

Theorem 2.44). On the other hand, supposing that the dataset $\hat{\nu}$ is drawn from a continuous distribution in $\mathbb{R}^d$, the event that two samples achieves exactly the same minimum $\min_i c(x, y_i) - \hat{\psi}_i$ happens with probability zero (fig. 8(c)).

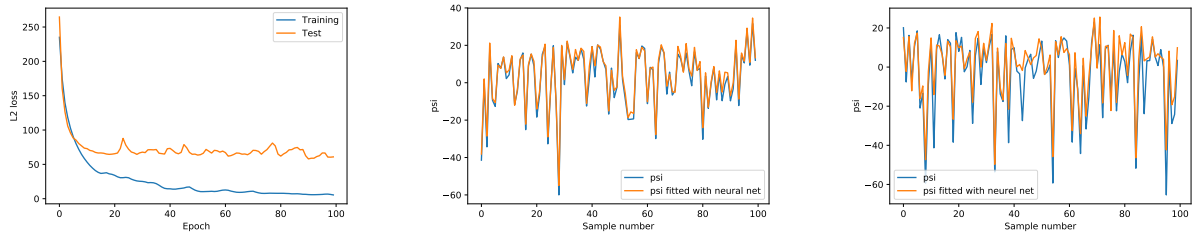


(a) Example of non-uniqueness. (b) Any perturbation breaks the ties. (c) When the samples are shifted.

Figure 8. (a) Suppose $g\#\mu$ is a uniform distribution on $[-1, 1] \times \{0\}$, and the samples are $y_1(0, 1), y_2(0, -1)$, then any equal-sized partition of $[-1, 1]$ would result to an optimal Monge plan. (b) Adding a small perturbation would make the optimal Monge plan unique. (c) If the samples become $y_1(0.001, 1), y_2(-0.001, -1)$, the ties are also broken.

D. Verifying the Assumption in Section 3.2

We train $\hat{\psi}$ between our fitted generating distribution $g\#\mu$ and MNIST full dataset $\hat{\nu}$, then fit an MLP ψ with 4 hidden layers of 512 neurons to $\hat{\psi}$ on training dataset $\hat{\nu}_{\text{train}}$, and evaluated on both $\hat{\nu}_{\text{train}}$ and test dataset $\hat{\nu}_{\text{test}}$ (ψ takes 784-dimensional images as input and outputs a scalar). Figure 9 shows that the training error goes to zero and ψ has almost the same value as $\hat{\psi}$ when evaluated on $\hat{\nu}$.



(a) L2 loss during training (b) $\hat{\psi}$ vs fitted ψ on 100 training samples (c) $\hat{\psi}$ vs fitted ψ on 100 test samples

Figure 9. $\hat{\psi}$ on MNIST fitted by neural network.

We point out some concrete cases where the exponential lower bound is defeated by introducing a dependence on the measures μ and ν, without such empirical verification. If μ and ν are both piecewise constant densities with few pieces, then the supremum in Equation (3.2) can be well-approximated with piecewise affine functions with few pieces. In the worst case, the number of pieces can be exponential in dimension, but we will depend explicitly on the measure and its properties.

Another way to achieve the generalization result without the verification process, is to parametrize ψ as a neural network in the optimal transport algorithm, an idea related to (Seguy et al., 2018). In Algorithm 1, we choose to optimize the vectorized $\hat{\psi}$ since it is a convex programming formulation guaranteed to converge to global minimum.

E. Experimental Details

E.1. Evaluation Metrics

Neural net distance (NND-WC, NND-GP). (Arora et al., 2017) define the *neural net distance* between the generated distribution $g\#\mu$ and dataset ν as:

$$\mathcal{D}_{\mathcal{F}}(g\#\mu, \nu) := \sup_{f \in \mathcal{F}} \mathbb{E}_{x\sim g\#\mu} f(x) - \mathbb{E}_{y\sim\nu} f(y),$$

where $\mathcal{F}$ is a neural network function class. We use DCGAN with weight clipping at ± 0.01, and DCGAN with gradient penalty with $\lambda = 10$ as two choices of $\mathcal{F}$. We call the corresponding neural net distances NND-WC and NND-GP respectively.

Wasserstein-1 distance (WD). This refers to the exact Wasserstein distance on ℓ_1 metric between the generated distribution μ and dataset ν, computed with our Algorithm 1.

Inception score (IS). (Salimans et al., 2016) assume there exists a pretrained external classifier outputing label distribution $p(y|x)$ given sample x. The score is defined as $\text{IS}_p(\mu) := \exp\{\mathbb{E}_{x\sim\mu}\text{KL}(p(y|x) \,||\, p(y))\}$.

Fréchet Inception distance (FID). Heusel et al. (2017) give this improvement over IS, which compares generated and real samples by the activations of a certain layer in a pretrained classifier. Assuming the activations follow Multivariate Gaussian distribution of mean μ_g, μ_r and covariance Σ_g, Σ_r, FID is defined as:

$$\text{FID}(\mu_g, \mu_r, \Sigma_g, \Sigma_r) := \|\mu_g - \mu_r\|^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}).$$

Note that the naming of IS and FID is used since the classifier is an Inception-v3 network pretrained on Imagenet. Since the pretrained Inception network is not suitable for classifying handwritten digits, we follow an idea due to Li et al. (2017) and pretrain a CNN classifier on MNIST dataset with 99.1% test accuracy.

E.2. Neural Network Architectures and Hyperparameters

E.2.1. Generator

MLP:

$$\textit{Input}(100) \xrightarrow{FC} \textit{Hidden}(512) \xrightarrow{FC} \textit{Hidden}(512) \xrightarrow{FC} \textit{Hidden}(512) \xrightarrow{FC} \textit{Output}(D).$$

DCGAN MNIST 28×28**:**

$$\textit{Input}(100, 1, 1) \xrightarrow{\textit{ConvT}(7,1,0)+\textit{BN}} \textit{Hidden}(128, 7, 7) \xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(64, 14, 14)$$
$$\xrightarrow{\textit{ConvT}(4,2,1)} \textit{Output}(1, 28, 28).$$

DCGAN Thin-8 128×128**:**

$$\textit{Input}(100, 1, 1) \xrightarrow{\textit{ConvT}(4,1,0)+\textit{BN}} \textit{Hidden}(256, 4, 4) \xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(128, 8, 8)$$
$$\xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(64, 16, 16) \xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(32, 32, 32) \xrightarrow{\textit{ConvT}(4,2,1)} \textit{Hidden}(16, 64, 64)$$
$$\xrightarrow{\textit{ConvT}} \textit{Output}(1, 128, 128).$$

E.2.2. Discriminator/Encoder

DCGAN Discriminator 28×28**:**

$$\textit{Input}(1, 28, 28) \xrightarrow{Conv(4,2,1)} \textit{Hidden}(64, 14, 14) \xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(128, 7, 7) \xrightarrow{FC} \textit{Output}(1).$$

Thin-8 Discriminator 128×128**:**

$$\textit{Input}(1, 128, 128) \xrightarrow{Conv(4,2,1)} \textit{Hidden}(16, 64, 64) \xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(32, 32, 32)$$
$$\xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(64, 16, 16) \xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(128, 8, 8) \xrightarrow{\textit{ConvT}(4,2,1)+\textit{BN}} \textit{Hidden}(256, 4, 4)$$
$$\xrightarrow{FC} \textit{Output}(1).$$

Encoder architectures are the same as discriminator architectures except for the output layers.

For WGANGP, batch normalization is not used since it affects computation of gradients. All activations used are ReLU. Learning rate is 10^{-4} for WGAN, WGANGP and our method, and 10^{-3} for VAE and WAE.

E.3. Training Details of Our Method

To get the transport mapping T in OTS, we memorize the sampled batches and their transportation targets, and reuse these batches in FIT. By this trick we avoid recomputing the maximum over the whole dataset.

Our empirical stopping criterion relies upon keeping a histogram of transportation targets in memory: if the histogram of targets is close to a uniform distribution (which is the distribution of training dataset), we stop OTS. This stopping criterion is grounded by our analysis in Section 3.1.