Scalable Computations of Wasserstein Barycenter via Input Convex Neural Networks

Jiaojiao Fan\textsuperscript{1} Amarhossein Taghvaei\textsuperscript{2} Yongxin Chen\textsuperscript{1}

Abstract

Wasserstein Barycenter is a principled approach to represent the weighted mean of a given set of probability distributions, utilizing the geometry induced by optimal transport. In this work, we present a novel scalable algorithm to approximate the Wasserstein Barycenters aiming at high-dimensional applications in machine learning. Our proposed algorithm is based on the Kantorovich dual formulation of the Wasserstein-2 distance as well as a recent neural network architecture, input convex neural network, that is known to parametrize convex functions. The distinguishing features of our method are: i) it only requires samples from the marginal distributions; ii) unlike the existing approaches, it represents the Barycenter with a generative model and can thus generate infinite samples from the barycenter without querying the marginal distributions; iii) it works similar to Generative Adversarial Model in one marginal case. We demonstrate the efficacy of our algorithm by comparing it with the state-of-art methods in multiple experiments.

1. Introduction

The Wasserstein barycenter is concerned with the (weighted) average of multiple given probability distributions. It is based on the natural geometry over the space of distributions induced by optimal transport (Villani, 2003) theory and serves as a counterpart of arithmetic mean/average for data of distribution-type. Compared to other methods, Wasserstein barycenter provides a principled approach to average probability distributions, fully utilizing the underlying geometric structure of the data (Agueh & Carlier, 2011). During the past few years, it has found applications in several machine learning problems. For instance, in sensor fusion, Wasserstein barycenter is used to merge/average datasets collected from multiple sensors to generate a single collective result (Elvander et al., 2018). The advantage of Wasserstein barycenter is its ability to preserves the modality of the different datasets, a highly desirable property in practice (Jiang et al., 2012). Wasserstein Barycenter has also been observed to be effective in removing batch effects of the sensor measurements (Yang & Tabak, 2019). It has also found application in large scale Bayesian inference for averaging the results from Markov chain Monte Carlo (MCMC) Bayesian inference carried out over subsets of the observations (Srivastava et al., 2015; Staib et al., 2017; Srivastava et al., 2018). It has also been useful in image processing for texture mixing (Rabin et al., 2011) and shape interpolation (Solomon et al., 2015).

The bottleneck of utilizing Wasserstein barycenter in machine learning applications remains to be computational. Indeed, when the data is discrete, namely, the given probability distributions are over discrete space (e.g., grid), the Wasserstein barycenter problem can be solved using linear programming (Anderes et al., 2016). This has been greatly accelerated by introducing an entropy term (Cuturi & Doucet, 2014, Algorithm 1) as in Sinkhorn algorithm (Cuturi, 2013). However, these methods are not suitable for many machine learning applications involving distributions over continuous space. First of all, it requires discretization of the distribution to implement these methods and thus doesn’t scale to high dimensional settings. Secondly, in some applications such as MCMC Bayesian inference (Andrieu et al., 2003; Srivastava et al., 2018) the explicit formulas of the distributions are not accessible, which precludes these discretization-based algorithms. When the support atoms are free to move, there are algorithms that interchangeably optimize the support weights and locations (Cuturi & Doucet, 2014, Algorithm 2) (Luise et al., 2019; Altschuler & Boix-Adsera, 2021) which can also be formulated in stochastic optimization framework (Claici et al., 2018). But these free-support methods become computationally highly expensive when the number of support points is large.
**Contribution:** We propose a computationally efficient and scalable algorithm for estimating the Wasserstein barycenter of probability distributions over continuous spaces. Our method is based on a Kantorovich-type dual characterization of the Wasserstein barycenter, which involves optimization over convex functions, and the recently introduced input convex neural networks (ICNN) (Amos et al., 2017; Chen et al., 2018b), that provides powerful representation of convex functions. Remarkably, in our framework, the (weighted) barycenter is represented by a generator network (Goodfellow et al., 2014; Arjovsky et al., 2017), that allows characterization of continuous distributions and fast and unlimited sampling from barycenter. We prove the consistency of our formulation in Proposition 1, and demonstrate its performance and its scaling properties in truly high-dimensional setting through extensive evaluations over various benchmark experiments including synthetic and real data-set and provide comparisons with several state-of-art algorithms: Korotin et al. (2021b), Li et al. (2020), Cuturi & Doucet (2014). Our experiments reveal significant improvement in estimating the barycenter in high-dimensional setting compared to most existing algorithms. We also showcase the the ability of our method to perform as a generative adversarial network (GAN) in the one marginal case and propose a heuristic extension to learn barycenter of arbitrary weights through a single training process.

**Related work:** Our proposed algorithm is closely related to continuous Wasserstein barycenter proposed by Li et al. (2020) and Korotin et al. (2021b). Similar to our approach, both of them are based on dual formulation of the Wasserstein barycenter problem and representing the potential functions with neural networks. However, the approach in Li et al. (2020) does not restrict the potential functions to be convex. Instead, an entropic or $l_2$ regularization term is added to ensure that the optimal potential functions are approximately convex. The addition of the regularization term introduces undesirable bias error which becomes severe in high-dimensional problems as shown in Figure 5-6 and also reported in (Korotin et al., 2021b, Table 1-4) (Li et al., 2020, Section 5). The approach in Korotin et al. (2021b) restricts the potentials to be convex using ICNN, however, it includes a cycle regularizer term to ensure the potential functions are dual conjugate. Their formulation also involves a congruence regularizer to guarantee that the optimal potential functions are consistent with the true barycenter. The congruence regularizer requires selection of a priori probability distribution that is bounded below by the true barycenter, which is a non-trivial task. Moreover, the addition of the regularization terms distorts the nice optimization landscape of the original problem. The problem may become non-convex even for the simple setting of restriction to quadratic functions (see Sec. G in supplementary material). In contrast, our formulation does not involve additional regularization terms and retains the optimization landscape of the original problem. Moreover, a distinct feature of our algorithm is representing the barycenter using a generative model which allows a low-dimensional representation of the barycenter and access to infinitely many samples, while both of these methods represent the barycenter using the Monge maps from the marginals, and limits the number of samples to the number available from marginal distributions.

Earlier stochastic Wasserstein barycenter method (Claici et al., 2018) also aims at calculating barycenters for continuous distributions using samples. However, they adopt a semi-discrete approach that models the barycenter with a finite set of points. That is, even though the marginals are continuous, the barycenter is discrete. Several other sample-based algorithms (Staib et al., 2017; Kuang & Tabak, 2019; Mi et al., 2020) are also of semi-discrete-type. Most other Wasserstein barycenter algorithms are for discrete distributions and require discretization if applied to continuous distributions. An incomplete list includes (Cuturi & Doucet, 2014; Benamou et al., 2015; Solomon et al., 2015).

The subject of this work is also related to the vast amount of literature on estimating the optimal transport map and Wasserstein distance (see Peyré et al., 2019) for a complete list). Closely related to this paper are the recent works that aim to extend the optimal transport map estimation to large-scale machine learning settings (Genevay et al., 2016; Seguy et al., 2017; Liu et al., 2018; Chen et al., 2018a; Leygonie et al., 2019; Xie et al., 2019). In particular, our algorithm is inspired by the recent advances in estimation of optimal transport map and Wasserstein-2 distance using ICNNs (Taghvaei & Jalali, 2019; Makkuvu et al., 2020; Korotin et al., 2021a).

## 2. Background

### 2.1. Optimal transport and Wasserstein distance

Given two probability distributions $\nu, \mu$ over Euclidean space $\mathbb{R}^n$ with finite second moments, the optimal transport (Villani, 2003) (OT) problem with quadratic unit cost seeks an optimal joint distribution of $\nu, \mu$ that minimizes the total transport cost. More specifically, it is formulated as $W_2^2(\nu, \mu) := \min_{\pi \in \Pi(\nu, \mu)} \int_{\mathbb{R}^n \times \mathbb{R}^n} ||x - y||^2 d\pi(x, y)$, where $\Pi(\nu, \mu)$ denotes the set of all joint distributions of $\nu$ and $\mu$. The square-root of the minimum transport cost defines the celebrated Wasserstein-2 distance $W_2$, which is known to enjoy many nice geometrical properties compared to other distances for distributions (Ambrosio et al., 2008).

The Kantorovich dual of the OT problem reads

$$\frac{1}{2} W_2^2(\nu, \mu) = \sup_{(\phi, \psi) \in \Phi} \mathbb{E}_\nu[\phi(X)] + \mathbb{E}_\mu[\psi(Y)],$$

where $\Phi := \{ (\phi, \psi) : L^1(\nu) \times L^1(\mu) : \phi(x) + \psi(y) \leq 0 \}$.
\[ \frac{1}{2} \| x - y \|^2, \quad \forall x, y. \]

Let \( f(x) = \| x \|^2 / 2 - \phi(x) \), then (1) can be rewritten as
\[
\frac{1}{2} W^2_2 (\nu, \mu) = \mathbb{E}_{\nu} [f(X)] + \mathbb{E}_{\mu} [f^*(Y)]
\]
where \( \text{CVX} \) stands for the set of convex functions. \( C_{\nu, \mu} := (1/2) \{ \mathbb{E}_{\nu} [\| X \|^2] + \mathbb{E}_{\mu} [\| Y \|^2] \} \), and the \( f^* \) is the convex conjugate (Rockafellar, 1970) function of \( f \). The formulation (2) is known as the semi-dual formulation of OT. The CVX condition restricts the search space for \( f \) which becomes handy for design of optimization algorithms.

**Remark 1** When both of the marginal distributions have densities, Brenier’s Theorem gives that \( \nabla f^* \) is the optimal transport map from \( \mu \) to \( \nu \) (Villani, 2003) and \( \nabla f \) is the optimal map from \( \nu \) to \( \mu \).

### 2.2. Wasserstein Barycenter

Wasserstein barycenter is an OT-based average of probability distributions. Given a set of probability distributions \( \mu_i, \ i = 1, 2, \ldots, N \) and a weight vector \( a \in \mathbb{R}^N \) \( (a_i \geq 0, \ i = 1, 2, \ldots, N \) and \( \sum_{i=1}^N a_i = 1 \), the associated Wasserstein barycenter is defined as the minimizer of
\[
\min_{\nu} \sum_{i=1}^N a_i W^2_2 (\nu, \mu_i).
\]
The barycenter problem (3) can be reformulated as a linear programming (Agueh & Carlier, 2011). However, the linear programming-base algorithms don’t scale well for high dimensional problems. A special case that can be solved efficiently is when the marginal distributions \( \{ \mu_i \} \) are Gaussian. Denote the mean and covariance of \( \mu_i \) as \( m_i \) and \( \Sigma_i \) respectively, then their Wasserstein barycenter is a Gaussian distribution with mean being \( m = \sum_{i=1}^N a_i m_i \) and covariance \( \Sigma \) being the unique solution to the fixed-point equation \( \Sigma = \sum_{i=1}^N a_i (\Sigma_1^1 / 2 \Sigma_i^1 / 2) \). In Álvarez-Esteban et al. (2016), a simple however efficient algorithm was proposed to solve for \( \Sigma \).

### 2.3. Input Convex Neural Network

Input Convex Neural Network (ICNN) is a type of deep neural networks architecture that characterize convex functions (Amos et al., 2017). A fully ICNN (FICNN) leads to a function that is convex with respect to all inputs.

Figure 1: Fully input convex neural network (FICNN)

The FICNN architecture is shown in Fig. 1. It is a \( L \)-layer feedforward neural network propagating following:

\[ z_{l+1} = \sigma_l (W_l z_l + A_l x + b_l), \]

where \( \{ W_l \}, \{ A_l \} \) are weight matrices (with the convention that \( W_0 = 0 \), \( \{ b_l \} \) are the bias terms, and \( \sigma_l \) denotes the entry-wise activation function at the layer \( l \). Denote the total set of parameters by \( \theta = \{ \{ W_l \}, \{ A_l \}, \{ b_l \} \} \), then this network defines a map from input \( x \) to \( f(x; \theta) = z_L \). This map \( f(x; \theta) \) is convex in \( x \) provided 1) \( W_{1:L} \) are non-negative; 2) \( \sigma_{0:L-1} \) are convex; 3) \( \sigma_{1:L-1} \) are non-decreasing (Makkuva et al., 2020). We remark that FICNN has the ability to approximate any convex function over a compact domain with a desired accuracy (Chen et al., 2018b), which makes FICNN an ideal candidate for modeling convex functions.

### 3. Methods and algorithms

We study the Wasserstein barycenter problem (3) for a given set of marginal distributions \( \{ \mu_i; \ i = 1, \ldots, N \} \). We consider the setting where the analytic forms of the marginals are not available. Instead, we only have access to independent samples from them. It can be either the cases where a fix set of samples is provided as in supervised learning, or the cases where one can keep sampling from the marginals like in the MCMC Bayesian (Srivastava et al., 2018). Our goal is to recover the true continuous Barycenter \( \nu \).

#### 3.1. Deriving the dual problem over convex functions

For a fixed \( \nu \), the objective function of (3) is simply a (scaled) summation of the Wasserstein cost between \( \mu_i \) and \( \mu \). Thus, we utilize the semi-dual formulation (2) of OT to evaluate the objective function of (3). However, convex conjugate function \( f^* \) is not available explicitly in most of applications, thus we characterize it as
\[
f^*(y) = \sup_{g \in \text{CVX}} \langle g, \nabla g(y) \rangle - f(\nabla g(y)).
\]

with the maximum being achieved at \( g = f^* \), the semi-dual formulation (2) can be rewritten as
\[
\frac{1}{2} W^2_2 (\nu, \mu) = \mathbb{E}_{\nu} [f(X)] + \mathbb{E}_{\mu} [f^*(Y)]
\]
where \( \mathbb{E}_{\nu} [f(X)] \) is a functional of \( f \) and \( g \) defined as
\[
\mathbb{E}_{\nu} [f(X)] = \mathbb{E}_{\nu} [f(X) - \mathbb{E}_{\mu} [f^*(Y)]].
\]
This formulation (6) has been utilized in conjunction with FICNN to solve OT problem in (Makkuva et al., 2020) and proved to be advantageous.

Plugging (6) into the Wasserstein barycenter problem (3), we obtain the following reformulation
\[
\min_{\nu} \sum_{i=1}^N a_i \left\{ \sup_{f_i \in \text{CVX}} \inf_{g_i \in \text{CVX}} \mathbb{E}_{\nu} [f_i, g_i] + C_{\nu, \mu} \right\}.
\]
Note that we have used different functions \((f_i, g_i)\) to estimate \(W_2^2(\nu, \mu_i)\). The first minimization is over all the possible probability distributions to search for the Wasserstein barycenter. This min-max-min formulation enjoys the following property, whose proof is in the supplementary material.

**Proposition 1** When all the marginal distributions \(\mu_i\) are absolutely continuous with respect to the Lebesgue measure, the unique Wasserstein barycenter \(\nu^*\) of them solves (8). Moreover, the corresponding optimal \(f_i^*\) is the optimal potential in (2) associated with marginals \(\nu^*\) and \(\mu_i\).

**Remark 2** Obtaining convergence rate for first-order optimization algorithms solving (8) is challenging even in the ideal setting that the optimization is carried out in the space of probability distributions. The difficulty arises because of the optimization over \(\nu\). While the inner optimization over \(f_i\) and \(g_i\) are concave and convex respectively, the optimization over \(\nu\) is not convex. Precisely, it is not geodesically convex on the space of probability distributions equipped with Wasserstein-2 metric (Ambrosio et al., 2008). However, it is possible to obtain guarantees in a restricted setting by establishing a Polyak-Lojasiewicz type inequality. In particular, assuming all \(\mu_i\) are Gaussian with positive-definite covariance matrices, the gradient-descent algorithm admits a linear convergence rate (Chewi et al., 2020).

### 3.2. Solving the barycenter problem

Consider the Wasserstein barycenter problem for a fixed weight vector \(a\). Following (Makkuva et al., 2020) we use FICNN architecture to represent convex functions \(f_i\) and \(g_i\). We now use a generator \(h\) to model the distribution \(\nu\), by transforming samples from a simple distribution \(\eta\) (e.g., Gaussian, uniform) to a complicated distribution, thereby we recover a continuous Barycenter distribution. Thus, using this network parametrization and discarding constant terms, we arrive at the following optimization problem

\[
\min_h \sup_{f_i \in \text{FICNN}, g_i \in \text{FICNN}} \inf \frac{1}{N} \mathbb{E}_\eta \|h(Z)\|^2 + \sum_{i=1}^{N} a_i \mathbb{V}_{\eta, \mu_i} (f_i, g_i),
\]

where \(\mathbb{V}_{\eta, \mu_i} (f, g)\) is defined as

\[-\mathbb{E}_\eta [f_i(h(Z))] - \mathbb{E}_{\mu_i}[g_i(Y^{i \perp}, \nabla g_i(Y^{i \perp})) - f_i(\nabla g_i(Y^{i \perp})).\]

We propose Neural Wasserstein Barycenter (NWB) algorithm (Algorithm 1) to solve this three-loop min-max-min problem by alternatively updating \(h, f_i, \) and \(g_i\) using stochastic optimization algorithms. This pipeline is illustrated by the block diagram (Figure 2). We remark that the objective function in (9) can be estimated using samples from \(\mu_i, \eta\).

Thus, we just need access to the samples generated by the marginal distributions \(\mu_i\) instead of their analytic form to compute their Wasserstein barycenter. In practice, we found it more effective to replace the convexity constraints for \(g_i\) with a convexity penalty, that is, the negative values of the weight matrices \(W_i\) in FICNN (4).

Denoting the parameters of \(h, f_i, g_i\) by \(\theta_h, \theta_{f_i}, \theta_{g_i}\) respectively and the batch size by \(M\), we arrive at the batch estimation of the objective

\[
\sum_{i=1}^{N} a_i \left[ J(\theta_{f_i}, \theta_{g_i}, \theta_h) + R(\theta_{g_i}) \right] + \frac{1}{2M} \sum_{j=1}^{M} \|h(Z_j)\|^2,
\]

where \(J(\theta_{f_i}, \theta_{g_i}, \theta_h)\) represents

\[
\frac{1}{M} \sum_{j=1}^{M} f_i(\nabla g_i(Y_j^i)) - \langle \nabla g_i(Y_j^i), f_i(h(Z_j)) \rangle - f_i(h(Z_j)),
\]

\(R(\theta_{g_i}) = \lambda \sum_{W_i \in \theta_{g_i}} \|\max(-W_i, 0)\|^2\). \(Y_j^i\) represents the \(j^{th}\) sample generated by \(\mu_i\), \(\{Z_j\}\) are samples from \(\eta\), and \(\lambda > 0\) is a hyper-parameter weighing the intensity of regularization.

Algorithm 1 can be extended to obtain the barycenters for all weights in one shot. This extension is included in Sec. B of the supplementary material.

**Remark 3** It is tempting to combine the two minimization steps over \(h\) and \(g_i\) into one and reduce (9) into a min-max saddle point problem. The resulting algorithm only alternates between \(f_i\) updates and \(h, g_i\) updates instead of the three-way alternating in Algorithm 1. However, in our implementations, we observed that this strategy is unstable.

**Computation complexity** For our algorithm, as well as the algorithms recently proposed in (Korotin et al., 2021b; Li et al., 2020), the computational complexity per iteration scales with \(O(NM^2p)\) where \(N\) is the number of marginals, \(M\) is the batch-size, and \(p\) is the size of the network (the size of network scales almost linearly with dimension \(d\)). This should be compared with \(O(NMK)\) for Claici et al. (2018) where \(K\) is the size of the support for barycenter, and \(O(NnK)\) for Cuturi & Doucet (2014) where \(n\) is the number of samples of the marginals. Although the size of the network is large, our approach is favored for large scale
Algorithm 1 Neural Wasserstein Barycenter (NWB)

**Input**: Marginal dist. $\mu_1:N$, Generator dist. $\eta$, Batch size $M$

for $k_3 = 1, \ldots, K_3$ do
    Sample batch $\{Z_j\}_{j=1}^M \sim \eta$
    Sample batch $\{Y_j^i\}_{j=1}^M \sim \mu_i$ for all $i = 1, \ldots, N$
    for $k_2 = 1, \ldots, K_2$ do
        for $k_1 = 1, \ldots, K_1$ do
            Update all $\theta_{gi}$ to decrease (10)
        end for
        Update all $\theta_{fi}$, to increase (10)
        Clip: $W_i = \max(W_i, 0)$ for all $\theta_{fi}$
        Update $\theta_h$, to decrease (10)
    end for
end for

problems where the number of samples $n$ and the dimension $d$ are large (the number of atoms required to approximate a density scales exponentially with dimension).

### 3.3. Recovering the barycenter

Once Algorithm 1 converges, there are two distinct approaches to recover the Wasserstein barycenter: one through $h$ and one through $g_i$.

**Generative model** $h_2 \eta$: In our problem formulation (9), the barycenter center is modeled by $h(Z)$ where $Z$ is sampled from a simple distribution $\eta$. That is, the barycenter $\nu$ is the pushforward of $\eta$ through the map $h$, denoted by $h_2 \eta$. Once the optimal $h$ is obtained, we can easily sample from the barycenter by sampling $Z_j$ from $\eta$ and apply the map $h(Z_j)$.

**Pushforward map** $\nabla g_i \mu_i$: An alternative method to recover the barycenter is based on the fact, once Algorithm 1 converges, the pair $(f_i, g_i)$ solves the OT problem (6) between the barycenter $\nu$ and the marginal $\mu_i$. As mentioned in Remark 1, $\nabla f^*$ is the optimal map from marginal distribution to the barycenter. Moreover, the optimal $g_i$ is achieved at $g = f^*$. Hence, the pushforward of $\mu_i$ through map $\nabla g_i$ is the barycenter. Thus, to sample from the barycenter, we can sample $Y_j^i$ from a marginal and then apply map $\nabla g_i(Y_j^i)$. Note that this approach cannot generate more samples than those are already available in the marginals.

### 3.4. Barycenter serving as GAN

In case where there is only one marginal distribution, that is, $N = 1$ in (9), Algorithm 1 can be viewed as a type of generative adversarial network (GAN). More specifically, when $N = 1$, the barycenter $\nu$ coincides with the marginal distribution $\mu_1$. Given samples $\{Y_j^1\}$ from the marginal $\mu_1$, Algorithm 1 produces a generative model $h(Z)$ whose distribution matches the marginal $\mu_1$. Note that one can easily sample using $h(Z)$ and get samples that do not exist in the training data $\{Y_j^1\}$.

In fact, when $N = 1$, Algorithm 1 works very much like a Wasserstein Generative Adversarial Network (WGAN) which leverages the Wasserstein distance to distinguish fake and real samples in GAN. The original WGAN (Arjovsky et al., 2017) is based on the dual formulation for the Wasserstein-1 distance $W_1$. A heuristic weight clipping (Arjovsky et al., 2017) technique is used to enforce the Lipschitz condition on the optimal function in the dual formulation. The WGAN was later improved in WGAN-GP (Gulrajani et al., 2017) via adding a gradient penalty term to promote the Lipschitz condition. From this point of view, Algorithm 1 provides an alternative way to train the generative model with Wasserstein-2 metric (c.f. Leygonie et al. (2019); Korotin et al. (2021a); Salimans et al. (2018); Genevay et al. (2018)).

### 4. Experiments

In Section 4.1, we present numerical experiments on 2D/3D datasets which serve as proof of concept and qualitatively illustrate the performance of our approach. In Section 4.2, we numerically study the effects of dimensionality and demonstrate the scalability of our algorithms to high-dimensional problems. In Section 4.3 and 4.4, we apply our algorithm in tasks such as Bayesian inference with large scale dataset and color transfer. In Section 4.5, we illustrate the ability of our algorithm to serve as a generative model. The implementation details and further experiments are included in the supplementary materials.

For comparison, we choose the following state of the art algorithms: (i) fast free-support Wasserstein barycenter (CDWB) (Cuturi & Doucet, 2014, Section 4.4); (ii) continuous Wasserstein barycenter without minimax optimization (CWB) (Korotin et al., 2021b); (iii) continuous regularized Wasserstein barycenter (CRWB) (Li et al., 2020). CWB and CRWB involve optimization over $N$ pairs of potentials $\{f_i, g_i\}$ as in NWB, and recover barycenter through $\nabla g_i \mu_i$. The implementations of these algorithms are based on published code associated with the papers.

To evaluate the performance of these algorithms, we use the Bures-Wasserstein UVP (Korotin et al., 2021b, Section 5)

$$BW_2^2(UVP(\nu, \tilde{\nu})) = \frac{100}{\text{Var}(\tilde{\nu})} \text{BW}_2^2(\nu, \tilde{\nu}) \%,$$

where $\text{BW}_2^2(\nu, \tilde{\nu})$ equals

$$\frac{1}{2} \left\| m_\nu - m_{\tilde{\nu}} \right\|^2 + \left[ \frac{1}{2} \text{Tr} \Sigma_\nu + \frac{1}{2} \text{Tr} \Sigma_{\tilde{\nu}} - \text{Tr} \left( \Sigma_\nu \Sigma_{\tilde{\nu}} \Sigma_{\nu} \right)^{\frac{1}{2}} \right].$$

Here $\nu$ is the estimated barycenter, $\tilde{\nu}$ is the exact barycenter,
and $m_{\nu_i}, \Sigma_{\nu_i}$ are the mean and the covariance of the distribution $\nu$. For barycenter given by pushforward $\nabla g_i z^{\#}\mu_i$, we report the weighted average of $BW_{2}^{2}$-UVP scores from $N$ marginal distributions: $\sum_{i=1}^{N} a_i BW_{2}^{2}$-UVP $(\nabla g_i z^{\#}\mu_i, \nu)$.

**Hyper-parameter choice** We choose a neural network architecture of $3 \sim 4$ hidden layers of size $1 \sim 2$ times of input dimension in high dim cases and size $16 \sim 32$ in 2D/3D cases, with preLU activation function. We observed that the performance is not sensitive to number of hidden layers, but sensitive to the choice of activation function (ReLU and leaky ReLU do not perform as well as preLU).

**Training time:** The training time for our method is almost once to twice longer than of CRWB and CWB due to the inner optimization cycles. The time consumption for CDWB is shorter for small number of samples, as it does not involve training neural networks.

**4.1. Learning the Wasserstein Barycenter in 2D and 3D**

The qualitative performance of our algorithm in three benchmark examples is depicted in Figure 4. Each example is represented in a row. The first column contains the marginal distributions, and the second and third column contains the learned barycenter through $h^{\#}\eta$ and $\nabla g_i z^{\#}\mu_i$ respectively. It is observed that both representations learn the barycenter qualitatively well, however, representation through $\nabla g_i z^{\#}\mu_i$ inherits the geometrical properties of the marginal distributions, highlighted with sharp boundaries in the first and second row and pixelated image in the third row. For comparison with CRWB, see Li et al. (2020, Figure 1).

**4.2. Scalability with the dimension**

**Gaussian:** We study the performance of our proposed algorithm in learning the barycenter of three Gaussian marginal distributions as dimension grows. The Gaussian marginal distributions have zero mean and a random non-diagonal covariance matrix whose conditional number is less than 10. The exact barycenter of Gaussian distributions is available to serve as the baseline in evaluating the Bures-Wasserstein UVP error criteria (11). The results are displayed in Figure 5. It is observed that the estimation error of NWB and CWB exhibit a slow rate of growth with respect to the dimension compared with CDWB and CRWB. The NWB and CWB algorithms are quite comparable in performance, but still different with respect to the optimization landscape. The optimization landscape of CWB is sensitive to the choice of regularization parameters (see Appendix G). We conjecture that the observed error curve of CWB in Figure 5, is due to the effect of regularization terms in distorting the optimization landscape which becomes more severe as dimension grows. Note that the error curve for CRWB contains irregularities and high variance that are probably due to the regularization term.

**MNIST:** To further investigate the performance of our algorithm in high dimension setting with real dataset, we use the MNIST data set. We consider the task of learning the barycenter of two marginal distributions. The first marginal $\mu_1$ is an empirical distribution of digit 0 samples and the second marginal $\mu_2$ is of digit 1. Each image has $28 \times 28$ pixels, yielding a 784-dimensional problem. The result of learning the barycenter is depicted in Figure 6. Both our algorithm NWB and CWB give reasonable results, with
Figure 5: Numerical result for scalability of the error with dimension for estimating barycenter of Gaussian distributions. The error criteria is \( (11) \). We generate \( 10^4 \) samples from barycenter for NWB, CWB, CRWB algorithms and 1500 samples for CDWB (the maximum number it can generate). The plot on the left includes the results for all four methods and the plot on the right highlights the detailed difference between CWB and NWB.

slightly sharper boundaries in NWB, whereas CRWB does not perform well in this high dimension setting. Note that the images in panel (a) are genuinely new samples generated using the trained generator, while the images in other panels are pushforward of marginal samples.

In order to demonstrate the inner-workings of our algorithm and its ability to learn the structure of barycenter, we implement the following experiment. We generate fresh samples from the barycenter using the generator \( h(Z) \), where \( Z \sim \mathcal{N}(0, I) \), and push-forward it through the maps \( \nabla f_1(h(Z)) \) and \( \nabla f_2(h(Z)) \). It is expected that through this procedure, we would recover the digits 0 and 1, because \( \nabla f_i \) represent the optimal transport map from Barycenter to the marginal \( \mu_i \) (see Remark 1). The experimental result confirms our expectation as shown in Figure 7. This implies that our proposed framework can serve as a generative model, not only for barycenter distribution, but also for the marginal distributions, by taking a random Gaussian random variable as input and output samples from marginal distributions.

4.3. Subset posterior aggregation

MCMC Bayesian inference is often carried out on splitted datasets in big data setting. However, the subset posterior distributions need to be merged into a single posterior to reflect the entire dataset property. This subset posterior aggregation scheme has been shown as an advantageous substitute to the full posterior (Srivastava et al., 2015) (Srivastava et al., 2018). The barycenter of subset posteriors is proved to converge to the full data posterior (Srivastava et al., 2018). Similar to (Li et al., 2020), we consider the Poisson regression for predicting the hourly number of bike hires with predictors such as the season and the weather conditions\(^2\). We consider the posterior on the 8-dimensional coefficients for the Poisson regression model. We randomly split the data into 5 equal-size subsets and simultaneously use the stochastic approximation trick (Minsker et al., 2014) to promise the subset posterior samples do not vary consistently from the full posterior in covariance. We obtain \( 10^5 \) samples from each subset posterior using the PyMC3 library (Salvatier et al., 2016).

We use the full posterior samples as the ground truth and report the Bures-Wasserstein UVP to compare the estimated barycenter and the ground truth. The results are shown in Table 1. All methods approach the true barycenter well (UVP \( < 1\% \)) and the performance of NWB is better than or on par with existing algorithms.

\[ \text{http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset} \]
Table 1: Comparison of UVP for recovered barycenters in our subset posterior aggregation task

<table>
<thead>
<tr>
<th>Metric</th>
<th>NWB $h_{#\mu}$</th>
<th>CDWB</th>
<th>CWB</th>
<th>CRWB</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{BW}_2^2$-UVP, %</td>
<td>0.06</td>
<td>0.26</td>
<td>0.07</td>
<td>1.67</td>
</tr>
</tbody>
</table>

4.4. Color palette averaging

Color transfer is a method to change the appearance of a source image according to the color pattern of a target image (Reinhard et al., 2001). Given several images, we can solve for Wasserstein barycenter to aggregate color palettes of images to achieve color transfer among them. Given an RGB image $I$, its color palette is the empirical distribution $\mu(I) = \sum^K_{k=1} \frac{1}{K} \delta_{p_k}$, where $\{p_k\}$ represents the pixels $\in [0, 1]^3$ and $\delta_{p_k}$ is the Dirac distribution concentrated on $p_k$. In our example, each image contains $1980 \times 1024$ pixels, so the number of samples for each marginal distribution is more than 2 million.

In Figure 8, the upper panel shows the original images $\{I_1, I_2, I_3\}$, and the bottom panel shows pixel-wise “pushforward” images. Figure 9 shows the RGB cloud to visualize the color palettes of images. In Figure 8, the appearance of the pushforward images are different from the source images thanks to the color averaging: the leaves in the first picture become greener and darker, the sunbeams in the second picture become more red, and the sky in the last picture receives an orange color toning.

Figure 8: Qualitative results by pixel-wise pushforward of the source images

4.5. Serving as a Generative Adversarial Model in the one marginal setting

We study the performance of our proposed algorithm in the case of one marginal distribution, where it behaves as a generative adversarial network using the $W_2$ metric. For comparison, we use WGAN (Arjovsky et al., 2017) and WGAN-GP (Gulrajani et al., 2017), which are based on $W_1$ metric, and W2GN (Korotin et al., 2021a), which is based on $W_2$ metric. Note that our goal is not to provide a competitive GAN algorithm, but to demonstrate the ability of our algorithm in performing as GAN.

We first consider an example of learning a Gaussian mixture model with 10 components shown in Figure 3. It can be seen that NWB avoids mode collapse. We then investigate the performance of our algorithm NWB in learning MNIST digits dataset (784 dim, 60000 sample size). From Figure 10, it is observed that our algorithm could output all the digits from 0 to 9 without mode collapse and the quality is on par with WGAN and WGAN-GP. W2GN seeks an optimal transport map from 784 dim standard Gaussian to MNIST; the generated digits are of poor quality. Note that in Korotin et al. (2021a) W2GN was tested on MNIST dataset but the optimal transport is addressed in the feature/latent space (see Section 5.2, Section C.7 in Korotin et al. (2021a)), which is of much lower dimension than the pixel space (784 dim).

5. Conclusion

During the last decade, many algorithms have been proposed for Wasserstein Barycenter estimation. A majority of these algorithms are designed for discrete setting (either discretization of space or discretization of distribution from samples). There are several algorithms that are designed for semi-discrete setting, in the sense that even though the marginal distributions are continuous, the barycenter computed from the algorithms is supported on finite points. More recently, two algorithms (Korotin et al., 2021b; Li et al., 2020) have been proposed to approximate the barycenter by learning the optimal transport maps from the marginal distributions to the barycenter using samples from the marginal distributions. Compared to all these existing Wasserstein Barycenter estimation algorithm, the NWB algorithm we develop is the only algorithm that gives a continuous representation of the barycenter through a generative model and is capable of generating infinitely many samples from the barycenter.
Scalable Computations of Wasserstein Barycenter via Input Convex Neural Networks

Figure 10: Performance of our algorithm as GAN in single marginal case: learning the MNIST digit dataset

Acknowledgments

The authors would like to thank the anonymous reviewers for useful comments. JF and YC are supported in part by grants NSF CAREER ECCS-1942523, NSF CCF-2008513, and NSF CCF-1740776. The authors also thank Qinsheng Zhang for fruitful discussions.

References


Scalable Computations of Wasserstein Barycenter via Input Convex Neural Networks


