Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport

Nicolas Bonneel
Univ. Lyon, CNRS and LIRIS

Gabriel Peyré
CNRS and Univ. Paris-Dauphine

Marco Cuturi
Kyoto University

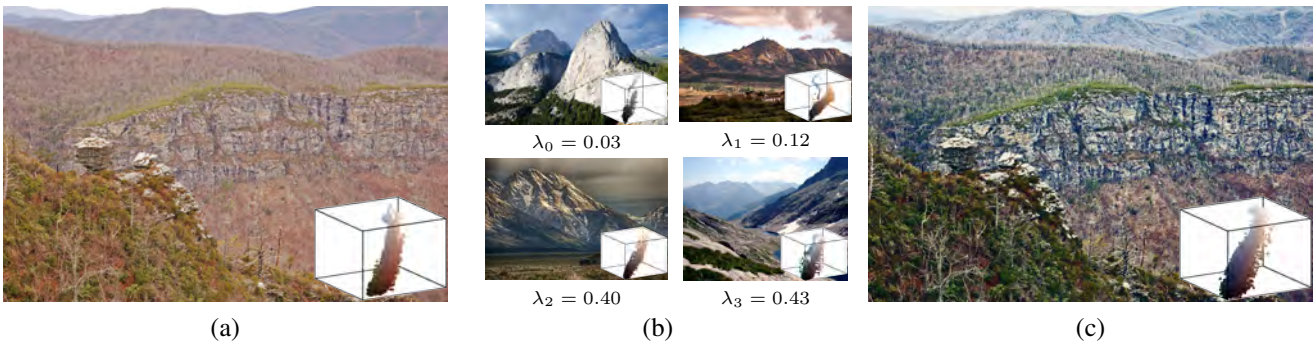


Figure 1: *Our Wasserstein projection framework can be used to automatically color grade an input photo (a) using a database of stylized color histograms, with samples shown in (b). We propose to compute the optimal transport barycenter of these stylized palettes that can approximate best the original palette, and use that barycenter to carry out color transfer without large color distortions as shown in (c), where the modified image and the barycentric palette are represented. That barycentric palette is parameterized using only the weights appearing in the captions of figure (b). Other applications include inferring reflectance functions or missing geometry (see Sec. 5).*

Abstract

This article defines a new way to perform intuitive and geometrically faithful regressions on histogram-valued data. It leverages the theory of optimal transport, and in particular the definition of Wasserstein barycenters, to introduce for the first time the notion of barycentric coordinates for histograms. These coordinates take into account the underlying geometry of the ground space on which the histograms are defined, and are thus particularly meaningful for applications in graphics to shapes, color or material modification. Beside this abstract construction, we propose a fast numerical optimization scheme to solve this backward problem (finding the barycentric coordinates of a given histogram) with a low computational overhead with respect to the forward problem (computing the barycenter). This scheme relies on a backward algorithmic differentiation of the Sinkhorn algorithm which is used to optimize the entropic regularization of Wasserstein barycenters. We showcase an illustrative set of applications of these Wasserstein coordinates to various problems in computer graphics: shape approximation, BRDF acquisition and color editing.

Keywords: optimal transport, fitting

Concepts: •**Computing methodologies** → **Computer graphics;** •**Mathematics of computing** → *Automatic differentiation;* •**Applied computing** → *Transportation;*

SIGGRAPH '16 Technical Paper,, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4279-7/16/07
DOI: http://dx.doi.org/10.1145/2897824.2925918

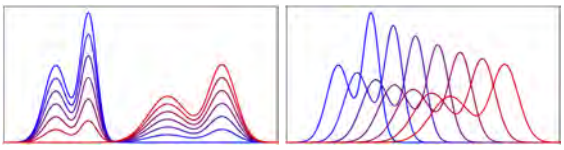


Figure 2: *Comparison between Euclidean (left) and Optimal Transport (right) barycenters between two densities, one being a translated and scaled version of the other. Colors encode the progression of the interpolation. The Euclidean interpolation results in mixtures of the two initial densities, while Optimal Transport results in a progressive translation and scaling.*

1 Introduction

Probability histograms play an important role in graphics. From color histograms to reflectance distribution functions, or even digital 3D shapes, probability histograms are routinely used to encode complex physical properties into vectors. The space of features these distributions are supported on—here, color space, sphere and spatial grid—can be often endowed with a distance, which encodes important invariances among features.

Optimal transport theory [Villani 2008; Rubner et al. 2000] proposes a natural way to lift a distance between features to define a metric between probability histograms on features. Optimal transport theory sees probability histograms as heaps of sand, and quantifies the distance between two of them by considering the least costly way to move all sand particles from one histogram to reshape it into the other. In this way, two histograms only differing by a small displacement are very near in the optimal transport sense, while they could be, in sharp contrast, regarded as very dissimilar under the ℓ_2 and ℓ_1 metrics or the KL divergence, particularly so if these two histograms have little overlap.

Because the optimal transport metric emphasizes mass displace-

ments, it also defines radically different ways to interpolate between two or more histograms. From a theoretical perspective, the mathematics of optimal transport ensure that histograms are combined through *advections*, and not linear combinations, as illustrated in Fig. 2 and Fig. 3. Because of their physical interpretation, optimal transport interpolations, *a.k.a* Wasserstein barycenters, are particularly suited to applications in graphics, where such concepts of motions are often meaningful: the rotation of the color wheel, the motion of highlights in reflectance models, Hausdorff distances between shapes etc. From a practical perspective, these interpolations can only find applications in graphics if they are computationally cheap. Although direct approaches to compute such interpolations were first believed to be intractable and degenerated [Agueh and Carlier 2011, §4], recent work has shown that regularized formulations [Cuturi 2013; Benamou et al. 2015] can provide those cheap algorithms for graphics [Solomon et al. 2015].

We consider in this paper the inverse problem associated with histogram interpolation, which is that of forming, for a given histogram, a barycentric interpolation of reference histograms that approximates it best. The barycentric interpolation itself can be interpreted as a denoised version of the original input, with respect to the prior contained in those reference histograms. The usually much shorter vector of barycentric coordinates can serve as a handy representation to compress, visualize or carry out inference on the original histogram. The crucial novelty in this paper lies in the fact that the interpolation we consider here is in the *optimal transport metric sense*, which gives our barycentric coordinate system an intuitive and geometrically faithful flavor. We call this new notion of coordinates for histograms *Wasserstein barycentric coordinates*, and provide algorithms to compute them efficiently. We apply our algorithms on histograms frequently encountered in computer graphics, ranging from color histograms to reflectance distributions.

Contributions. We propose a method to project an input histogram q onto the set of all Wasserstein barycenters formed by S histograms $(p_1, \cdots, p_S)$ (see Fig. 3). This corresponds to approximating the input histogram q by its closest (with respect to some loss) Wasserstein barycenter $P(\lambda)$ of $(p_1, \cdots, p_S)$, where $\lambda = (\lambda_1, \ldots, \lambda_S)$ is the optimal weight vector sought for. We call this weight vector λ the Wasserstein barycentric coordinates of q.

We propose the first numerical scheme to compute Wasserstein barycentric coordinates. This scheme builds upon gradient descent, and requires thus the computation of the (usually high-dimensional) Jacobian of the barycenter operator $\lambda \mapsto P(\lambda)$. To be tractable, our solution relies on an approximation of $P(\lambda)$ that uses a fixed number of steps of a fixed-point iteration computation proposed by Benamou et al. [2015]. We can therefore use a recursive differentiation method to compute that Jacobian efficiently. This leads to an algorithm which is both fast and stable, allowing for the computation of optimal barycentric weights on large scale dense 3-D grids and other domains. We showcase a set of typical applications of our methods to color analysis (Fig. 7, fitting sparse reflectance measurements (Fig. 6) and reconstructing 3D shapes (Fig. 9).

2 Previous works

Optimal transport (OT) is a powerful way to define distances (also known as Wasserstein or earth mover's distances) between probability distributions on general metric spaces, which takes into account the geometry of the underlying space. Initially formulated by Monge as an intractable non-convex optimization [Monge 1781], its modern linear programming formulation is due to Kantorovitch [1942], and is presented in much details in Villani's monograph [2003]. Its practical applications are more recent, starting with computer vision [Haker et al. ; Rubner et al. 2000], image processing [Rabin and Papadakis 2015], machine learning [Cuturi 2013; Solomon et al. 2014b] and computer graphics [Bonneel et al. 2011].

Classical linear programming and combinatorial optimization approaches to OT [Burkard et al. 2009] scale roughly with cubic complexity, and are very costly. Alternative methods somehow alleviate this issue – e.g., the special case of W_1 transport [Solomon et al. 2014a], semi-discrete approach with Laguerre's cells [Mérigot 2011] and a dynamical formulation [Benamou and Brenier 2000]. These approaches are however very restrictive in the sense that they only work for particular cost structures or discretizations (typically in low dimension) and are thus not usable for the applications this paper targets. Of particular relevance to this paper is the recent interest for entropy regularized approaches to solve OT [Cuturi 2013]. Instead of a linear program, entropic smoothing allows the use of Bregman optimization tools [Bregman 1967], in particular Sinkhorn's algorithm [Sinkhorn 1964; Deming and Stephan 1940]. This approach provides two benefits: since the computation of regularized OT only involves matrix-vector products, that problem can be computed efficiently on parallel architectures; since the problem is regularized with a strongly convex term, the regularized distance becomes a smooth function of all its parameters.

Because OT was considered expensive, computing OT distances faster has been for many years a goal in and by itself. Only recently was it realized that—now that OT distances can be efficiently approximated—far more interesting problems involving OT distances can be considered, starting with the introduction of OT barycenters [Agueh and Carlier 2011]. Among all approaches proposed to compute them in practice [Rabin et al. 2012; Cuturi and Doucet 2014; Bonneel et al. 2015], that of Benamou et al. [2015] stands out for its simplicity. These barycenters have been independently considered in statistics [Bigot and Klein 2012; Srivastava et al. 2015], image processing [Bonneel et al. 2015] and computer graphics [Solomon et al. 2015]. These developments have paved the way for even more complicated problems on the space of probability measures that incorporate OT in their formulation, such as Principal Component Analysis (PCA) [Seguy and Cuturi 2015; Bigot et al. 2015] or non-negative matrix factorization (NMF) [Sandler and Lindenbaum 2009; Rolet et al. 2016] which have been recently rephrased using OT loss functions. More precisely, the NMF problem (even when used in conjunction with a Wasserstein loss) corresponds to looking for a *linear* combination of base distributions that approximates a given input histogram. In contrast, our Wasserstein barycentric coordinate regression uses a *non-linear* combination of the base distributions, which uses the OT geometry by enabling interpolation through mass transportation (see Figures 2 and 3 for simple illustrations of this important distinction).

While we propose a generic approach, application specific solutions have been proposed. For instance, Matusik et al. [2003] has shown that measured reflectance data form a manifold of histograms which can be learned via charting [Brand and Brand 2003]. Their approach allows to locate a new measured reflectance within this manifold. Wills et al. [2009] locates reflectance functions in a manifold obtained via multi-dimensional scaling. This technique has similarly been used to build space of color histograms via optimal transportation distances [Rubner et al. 1998]. Let us also note that computing barycentric coordinates on non-Euclidean domains is an important theoretical and numerical problem, see for instance [Rustamov 2010] for the case of surfaces.

3 Wasserstein Barycenters: Background

This section first presents the—now well-understood—forward problem of numerically computing Wasserstein barycenters of his-

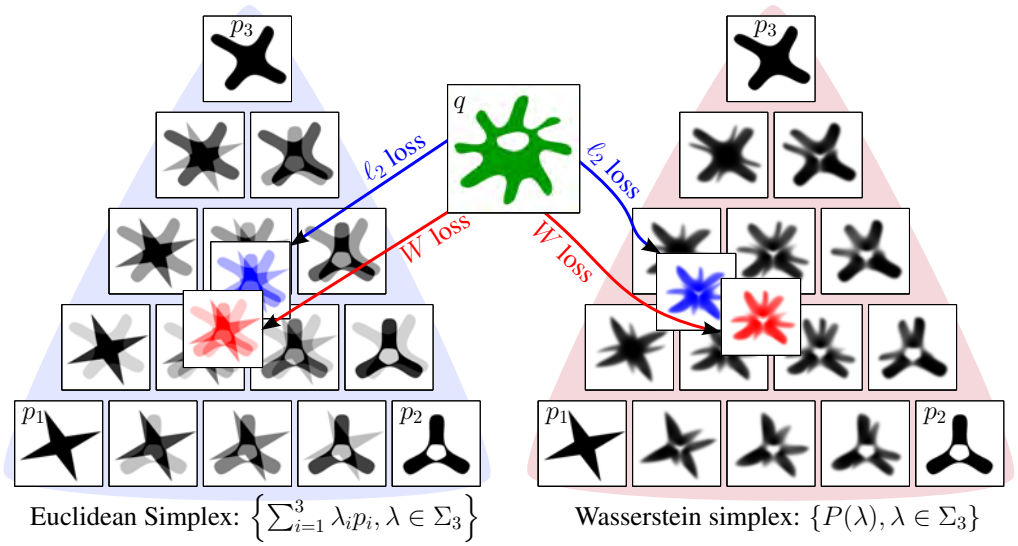


Figure 3: *We consider four monochrome* 500×500 *images,* (q, p_1, p_2, p_3) *whose total intensity is normalized to sum to* 1*. The three images* $(p_i)_i$ *generate their Euclidean simplex (left, blue background), which consists in all of their convex combinations. The* $(p_i)_i$ *also define their Wasserstein simplex (right, red background), which consists in all of their Wasserstein barycenters under varying weights* λ*, see §3.3 for a formal definition. (left arrows) Finding the best approximation of* q *on the Euclidean simplex with a* ℓ_2 *loss is a simple constrained linear regression problem. Finding such an approximation with a Wasserstein loss was recently studied in [Rolet et al. 2016]. (right arrows) projections of* q *onto the Wasserstein simplex of the histograms* $(p_i)_i$ *using either the* ℓ_2 *or the Wasserstein loss. This work proposes the first known algorithms to carry out such projections, which can be entirely parameterized by weight vectors* λ*. These coordinates (3 numbers here) are reflected in the projections' locations in their respective simplexes.*

tograms. It then exposes our main contribution: the presentation and resolution of the barycentric coordinates inverse problem.

3.1 Notations

We consider the simplex $\Sigma_N \stackrel{\text{def.}}{=} \left\{p \in \mathbb{R}_+^N \,;\, \sum_i p_i = 1\right\}$ of N-dimensional normalized histograms, and consider a family of S reference histograms $(p_1, \cdots, p_S)$ in Σ_N. To interpolate between these S histograms, we consider barycentric weights $\lambda \in \Sigma_S$. For a matrix $T \in \mathbb{R}_+^{N \times N}$, we write $H(T) = \sum_{i,j} T_{i,j} \log(T_{i,j})$ its negative entropy, with the convention $0 \log 0 = 0$. For two matrices A, B of the same size, we write $\langle A,\, B\rangle = \mathrm{tr}(A^\top B)$ for their usual inner-product, where $A^\top$ is the transpose of A. We write $\mathbb{1}$ for the vector with unit coordinates and whose size depends on the context. The ℓ_α norm for $\alpha \geq 1$ is $\|p\|_\alpha^\alpha \stackrel{\text{def.}}{=} \sum_i p_i^\alpha$. The Kullback-Leibler divergence between histograms is $\mathrm{KL}(p|q) \stackrel{\text{def.}}{=} \sum_i p_i \log(p_i/q_i)$. In this paper, multiplication ($\prod$ for products of many terms and $\odot$ for two terms) and division $/$ operators between vectors are applied entry-wise, as well as exponential $\exp$ and logarithmic $\log$ maps.

3.2 Regularized Wasserstein Distance

Following [Cuturi 2013] (see also §2), we define the entropy regularized OT distance between two histograms $(p, q) \in \Sigma_N^2$ as

$$W(p,q) \stackrel{\text{def.}}{=} \min_{T \in \mathbb{R}_+^{N \times N}} \left\{ \langle T,\, C\rangle + \gamma H(T) \,;\, T\mathbb{1} = p, T^\top \mathbb{1} = q \right\}, \tag{1}$$

where the matrix $C = (C_{i,j})_{i,j}$ quantifies the cost of transporting mass between histogram bins. For instance, if bins are sampled at some locations $(x_i)_i$ in a Euclidean space, a common choice for C would be $C_{i,j} = \|x_i - x_j\|^\alpha$ for some $\alpha > 0$. We assume that the regularization parameter γ is positive, which ensures that the optimal solution of this program is unique and that the program itself is easier (faster, parallel computations) to solve.

3.3 Regularized Wasserstein Barycenters

Agueh and Carlier defined Wasserstein barycenters [2011] as Fréchet means in the space of probability measures endowed with the Wasserstein metric. They studied several of their properties, such as uniqueness; established links with the multi-marginal OT problem; described barycenters of Gaussian distributions. We consider in this work a simplified setting for this problem, in which measures are discrete and supported on the same set of N points. Following Cuturi and Doucet [2014] (see also [Benamou et al. 2015; Solomon et al. 2015]), we propose to compute the Wasserstein barycenters of S *histograms* $(p_s)_s$ in Σ_N using the regularized Wasserstein distance defined in Eq. (1).

Definition 1. *Given a family of* S *input histograms* $(p_s)_s$*, the barycentric map* $P : \Sigma_S \to \Sigma_N$ *associates to a vector* $\lambda \in \Sigma_S$ *the barycenter of* $(p_s)_s$ *with weights* λ*, uniquely defined as*

$$P : \lambda \mapsto P(\lambda) \stackrel{\text{def.}}{=} \operatorname*{argmin}_{p \in \Sigma_N} \sum_s \lambda_s W(p, p_s). \tag{2}$$

The uniqueness of $P(\lambda)$ *comes from the strong convexity (as a function of* p*) of the energy defined on the right-hand side of Eq.* (2)*, itself inherited from the regularization term in Eq.* (1)*.*

Wasserstein barycenters are very different from the usual linear averaging $\sum_s \lambda_s p_s$, which corresponds to barycenters in the ℓ_2 sense. Indeed, the usual ℓ_2 averaging is completely blind to the geometry of the domain, and is in some sense "non-physical" (induces transport of mass at infinite speed), as highlighted by Figure 2. Note that the same remark applies to barycenters induced by other separable divergences on Σ_N such as Hellinger, Kulback-Leibler or ℓ_1.

Although there is no closed-form expression for $P(\lambda)$, Benamou et al. [2015] have shown that the celebrated Sinkhorn fixed-point algorithm—which is used to solve problem (1)—can be extended to compute Wasserstein barycenters. This extension consists in an augmented fixed-point algorithm:

Proposition 1. *[Benamou et al. 2015, Prop. 2] Define for all $s \leq S$, $a_s^{(0)} = \mathbb{1}$, and then recursively for $l \geq 0, s \leq S$:*

$$P^{(\ell)}(\lambda) \stackrel{\text{def.}}{=} \prod_s \left(K^\top a_s^{(\ell)}\right)^{\lambda_s} \text{ and } \begin{cases} b_s^{(\ell+1)} \stackrel{\text{def.}}{=} \frac{P^{(\ell)}(\lambda)}{K^\top a_s^{(\ell)}}, \\ a_s^{(\ell+1)} \stackrel{\text{def.}}{=} \frac{p_s}{K b_s^{(\ell+1)}}. \end{cases} \quad (3)$$

where $K \stackrel{\text{def.}}{=} e^{-C/\gamma}$ is the $N \times N$ kernel matrix corresponding to the cost C and regularization γ. Then $P^{(\ell)}(\lambda) \xrightarrow[\ell \to \infty]{} P(\lambda)$.

As detailed in [Benamou et al. 2015], iterations (3) correspond to iterative projections for the KL divergence on a set of affine constraints. The main computational burden of these iterations is that of applying the kernel K or $K^\top$ to S vectors. In our settings, where the cost C is translation invariant, these operations are cheap because they amount to carrying out S convolutions in parallel.

4 Barycentric Coordinate Regression

We now come to the core of our contributions. Given a histogram $q \in \Sigma_N$, our goal is to define and compute the barycentric coordinates of q within a family of S reference histograms $(p_s)_s$, namely to find the vector of probability weights $\lambda \in \Sigma_S$ such that $q \approx P(\lambda)$ with respect to a loss function $\mathcal{L} : \Sigma_N \times \Sigma_N \to \mathbb{R}_+$:

Definition 2. *Let $q, p_1, \ldots, p_S \in \Sigma_N$. The barycentric coordinates of q with respect to $(p_s)_s$ are any optimal solution to problem*

$$\operatorname*{argmin}_{\lambda \in \Sigma_S} \mathcal{E}(\lambda), \quad \text{where } \mathcal{E}(\lambda) \stackrel{\text{def.}}{=} \mathcal{L}(P(\lambda), q). \quad (4)$$

In contrast to the convexity of problem (2), the energy of problem (4) is in general not convex. Our goal is thus to recover a stationary point of that energy through gradient descent. The gradient of $\mathcal{E}$ with respect to λ can be computed using the chain rule:

$$\nabla \mathcal{E}(\lambda) = [\partial P(\lambda)]^\top \nabla \mathcal{L}(P(\lambda), q), \quad (5)$$

where $\partial P(\lambda)$ is the Jacobian of $\lambda \mapsto P(\lambda)$, $\nabla \mathcal{L}(p, q)$ is the gradient of the loss $p \mapsto \mathcal{L}(p, q)$, and, with these notations, $\nabla \mathcal{L}(P(\lambda), q)$ is that gradient evaluated at $P(\lambda)$.

Among the two quantities in Eq. (5), the gradient of the loss $\nabla \mathcal{L}(P(\lambda), q)$ is the least problematic since it can be easily derived for several common losses as shown below, and evaluated at $P(\lambda)$. Applying the transpose of the Jacobian $[\partial P(\lambda)]^\top$ to that gradient is more challenging, both in theory and practice: We show first in §4.2 that, although an exact expression for that Jacobian can be obtained, computing it is impractical for large dimensions N. We present in §4.2 an efficient alternative, by replacing the true barycenter $P(\lambda)$ in the definition of the energy $\mathcal{E}$ by the running estimate $P^{(L)}(\lambda)$ obtained after L iterations of the map described in Eq. (3), where L is a number of iterations fixed beforehand.

Gradient of the loss. The gradient with respect to p of commonly used separable losses $\mathcal{L}(p, q)$ is

$$\nabla \frac{1}{2}\|p - q\|_2^2 = p - q, \quad \nabla \|p - q\|_1 = \operatorname{sign}(p - q), \quad (6)$$

$$\nabla \operatorname{KL}(p|q) = \log(\tfrac{p}{q}), \quad \nabla W(p, q) = \gamma \log(a), \quad (7)$$

where, for the gradient of $W(p, q)$, $a \in \mathbb{R}^N$ is the left scaling produced by Sinkhorn's fixed-point algorithm, namely the unique vector with geometric mean 1 such that the matrix $\operatorname{diag}(a) K \operatorname{diag}(q / K^\top a)$ has row-sum p and column-sum q [Cuturi and Doucet 2014, §5]. Note that the notation $\nabla \|p - q\|_1$ is not rigorous, since the ℓ_1-norm is not differentiable everywhere and the sign vector is only a subgradient of that quantity. We side-step this issue in this paper, which is only problematic for the 1-norm, by using quasi-Newton solvers such as L-BFGS that work well even with non-smooth objectives [Lewis and Overton 2013].

4.1 Exact Computation of the Jacobian

We show in Proposition 2 below that $[\partial P(\lambda)]^\top$ can be computed, assuming that the barycenter $P(\lambda)$ can been computed exactly. To simplify this exposition, we introduce two bi-variate functions (Φ, Ψ) to rewrite the iterations of Proposition 1 as operating only on the scalings $b^{(\ell)}(\lambda) \stackrel{\text{def.}}{=} (b_s^{(\ell)}(\lambda))_{s=1}^S$:

$$P^{(\ell)}(\lambda) = \Psi(b^{(\ell)}(\lambda), \lambda) \text{ where } \Psi(b, \lambda) \stackrel{\text{def.}}{=} \prod_s \varphi_s(b_s)^{\lambda_s} \quad (8)$$

$$b^{(\ell+1)}(\lambda) = \Phi(b^{(\ell)}(\lambda), \lambda) \text{ where } \Phi(b, \lambda) \stackrel{\text{def.}}{=} \left(\frac{\Psi(b, \lambda)}{\varphi_s(b_s)}\right)_s, \quad (9)$$

and $\varphi_s(b_s) \stackrel{\text{def.}}{=} K^\top \frac{p_s}{K b_s}$, using this time the initialization $b_s^{(0)} = \mathbb{1}$.

Proposition 2. *One has*

$$[\partial P(\lambda)]^\top = [\partial b(\lambda)]^\top [\partial_b \Psi(b(\lambda), \lambda)]^\top + [\partial_\lambda \Psi(b(\lambda), \lambda)]^\top \quad (10)$$

$$[\partial b(\lambda)]^\top = [\partial_\lambda \Psi(b(\lambda), \lambda)]^\top \left(\operatorname{Id} - [\partial_b \Phi(b(\lambda), \lambda)]^\top\right)^{-1}, \quad (11)$$

where $\partial_b \Phi$ and $\partial_\lambda \Phi$ (resp. $\partial_b \Psi$ and $\partial_\lambda \Psi$) correspond to the partial derivatives of $\Phi(b, \lambda)$ (resp. $\Psi(b, \lambda)$) with respect to its first and second variables. The matrices corresponding to these differentials are spelled out in Appendix A.

Proof. When iterations (8) have converged, the barycenter satisfies $P(\lambda) = \Psi(b(\lambda), \lambda)$ where $b(\lambda)$ satisfies the fixed-point equation $b(\lambda) = \Phi(b(\lambda), \lambda)$. Differentiating these two relations gives a linear equation that can be inverted to give the desired expression. □

While mathematically correct, formulas (10) and (11) are difficult to implement in practice: *(i)* They make the hypothesis that one is able to compute vectors $P(\lambda)$ and $b(\lambda)$ that solve the fixed-point equations exactly, which is not the case in practice, since these vectors are only approximated by iterating a sufficient number of times the map in Eq. (3) to converge to a sufficient accuracy; *(ii)* Eq. (11) requires solving a $N \times N$ linear system, which is prohibitive for all of the settings considered in this paper, where N is usually of the order of 10^6. Even for much smaller N, toy experiments have shown that this approach was not only extremely costly, but also unstable, unless one uses an extremely small convergence criterion to control the number of iterations of the fixed point map.

4.2 Algorithmic Differentiation of the Jacobian

Because the exact computation of $P(\lambda)$ outlined above in §4.1 is impractical, we propose in this section to minimize a loss on the *approximate* barycenter $P^{(L)}(\lambda)$ computed after a finite number of iterations $L \geq 1$, to solve instead:

$$\operatorname*{argmin}_{\lambda \in \Sigma_S} \mathcal{E}_L(\lambda) \stackrel{\text{def.}}{=} \mathcal{L}(P^{(L)}(\lambda), q). \quad (12)$$

The gradient formula (5) thus needs to be replaced by

$$\nabla \mathcal{E}_L(\lambda) = [\partial P^{(L)}(\lambda)]^\top (u^{(L)}), \; u^{(L)} \stackrel{\text{def.}}{=} \nabla \mathcal{L}(P^{(L)}(\lambda), q). \quad (13)$$

Because $P^{(L)}(\lambda)$ is obtained by recursively applying the same map L times, the application of the transposed Jacobian $[\partial P^{(L)}(\lambda)]^\top$ to

function SINKHORN-DIFFERENTIATE($(p_s)_{s=1}^S, q, \lambda$)

$\forall\, s, b_s^{(0)} \leftarrow \mathbb{1}$

$(w, r) \leftarrow (0^S, 0^{S\times N})$

for $\ell = 1, 2, \ldots, L$ *// Sinkhorn loop*

$\forall\, s, \varphi_s^{(\ell)} \leftarrow K^\top \frac{p_s}{K b_s^{(\ell-1)}}$

$p \leftarrow \prod_s \left(\varphi_s^{(\ell)}\right)^{\lambda_s}$

$\forall\, s, b_s^{(\ell)} \leftarrow \frac{p}{\varphi_s^{(\ell)}}$

$g \leftarrow \nabla\mathcal{L}(p, q) \odot p$

for $\ell = L, L-1, \ldots, 1$ *// Reverse loop*

$\forall\, s, w_s \leftarrow w_s + \langle \log \varphi_s^{(\ell)}, g\rangle$

$\forall\, s, r_s \leftarrow -K^\top\left(K\left(\frac{\lambda_s g - r_s}{\varphi_s^{(\ell)}}\right) \odot \frac{p_s}{(K b_s^{(\ell-1)})^2}\right) \odot b_s^{(\ell-1)}$

$g \leftarrow \sum_s r_s$

return $P^{(L)}(\lambda) \leftarrow p, \nabla\mathcal{E}_L(\lambda) \leftarrow w$

Algorithm 1: *Given a database of histograms $(p_s)_{s=1}^S$, the input distribution q, weights λ, this function computes $\nabla\mathcal{E}_L(\lambda) \in \mathbb{R}^S$. The barycenter $P^{(L)}(\lambda)$ is obtained as a by-product.*

the vector $u^{(L)}$ can be computed using *backward recursive differentiation* [Neidinger 2010]. This turns out to be particularly efficient, since the overall complexity of computing $[\partial P^{(L)}(\lambda)]^\top(u^{(L)})$ is the same as that of computing the approximate barycenter $P^{(L)}(\lambda)$. Proposition 3 shows that one can compute $[\partial P^{(L)}(\lambda)]^\top(u^{(L)})$ and thus $\nabla\mathcal{E}_L(\lambda)$ using a simple *backward* recursion. Its proof is given in Appendix B.

Proposition 3. *Let us denote, for $\ell \geq 0$,*

$$\Phi_\lambda^{(\ell)} \stackrel{\text{def.}}{=} [\partial_\lambda \Phi(b^{(\ell)}(\lambda), \lambda)]^\top \quad \text{and} \quad \Phi_b^{(\ell)} \stackrel{\text{def.}}{=} [\partial_b \Phi(b^{(\ell)}, \lambda)]^\top,$$
$$\Psi_\lambda^{(\ell)} \stackrel{\text{def.}}{=} [\partial_\lambda \Psi(b^{(\ell)}(\lambda), \lambda)]^\top \quad \text{and} \quad \Psi_b^{(\ell)} \stackrel{\text{def.}}{=} [\partial_b \Psi(b^{(\ell)}, \lambda)]^\top.$$

One has

$$\nabla\mathcal{E}_L(\lambda) = \Psi_\lambda^{(L)}(u^{(L)}) + \sum_{\ell=0}^{L-1} \Phi_\lambda^{(\ell)}(v^{(\ell)}) \tag{14}$$

where $u^{(L)}$ is defined in (13) *and the vectors $(v^{(\ell)})_{\ell=0}^{L-1}$ are computed using the following backward recursion*

$$\forall\, \ell = L-1, L-2, \ldots, 0, \quad v^{(\ell-1)} \stackrel{\text{def.}}{=} \Phi_b^{(\ell-1)}(v^{(\ell)}) \tag{15}$$

initialized with $v^{(L)} \stackrel{\text{def.}}{=} \Psi_b^{(L)}(u^{(L)})$.

The overall numerical scheme to compute $\nabla\mathcal{E}_L(\lambda)$ is detailed in Algorithm 1, which can be obtained by plugging the expression for the differential of (Φ, Ψ)—used to prove Proposition 2 and detailed in Appendix A—into the formulas of Proposition 3. The algorithm first performs a forward loop to compute the barycenter $P^{(L)}(\lambda)$ and then an inverse loop to implement (15) and accumulate the sum appearing in (14). This efficient implementation computes both the gradient and barycenter in twice as many Gibbs kernel K and $K^\top$ applications as required to compute the barycenter alone, making it a competitive approach, even against naive approximate numerical finite differentiation, which would run $(S+1)/2$ times slower. To summarize, this algorithm only requires $4SL$ convolutions and additional storage for $3NLS$ scalar values at each gradient step carried out to minimize the energy $\mathcal{E}_L$.

4.3 Barycentric Coordinates using Quasi-Newton

With the function SINKHORN-DIFFERENTIATE at hand, which is able to compute both the current barycenter estimate $P^{(L)}(\lambda)$ *and* the gradient $\nabla\mathcal{E}_L(\lambda)$, one can now efficiently compute barycentric coordinates λ as a local minimizer of (12) through a descent method. Quasi-Newton methods proved very efficient in our experiments. We tested two methods, which turned out to be equally effective: The PQN constrained quasi-Newton of [Schmidt et al. 2009], which can optimize smooth functions such as $\mathcal{E}_L$ over the simplex Σ_S; a standard quasi-Newton (L-BFGS) over a logarithmic domain using the change of variables $\lambda = \frac{e^\alpha}{\sum_s e^{\alpha_s}} \in \Sigma_S$ and carrying out the optimization over $\alpha \in \mathbb{R}^S$.

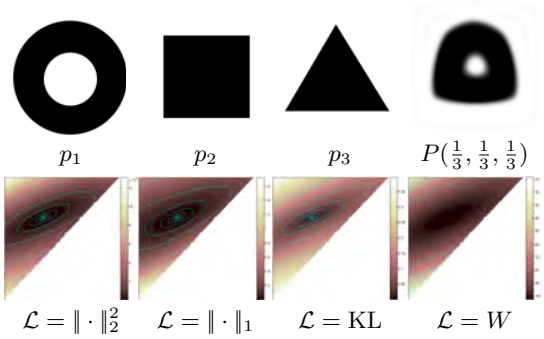


Figure 4: *To assess the convexity of our energy function (Eq. 4), we used three simple 2D shapes and their iso-barycenter. We computed the energy of fitting the isobarycenter for various barycentric coordinates. The bottom row shows these energy landscapes for various loss functions $\mathcal{L}$. In all cases, the energies appear convex.*

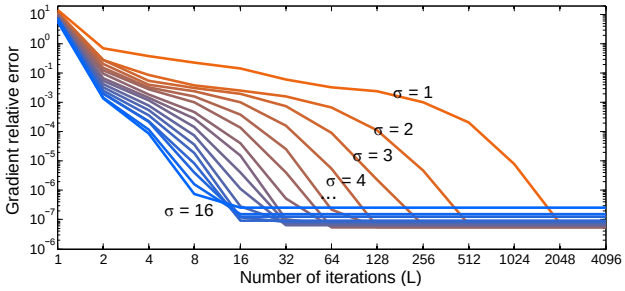
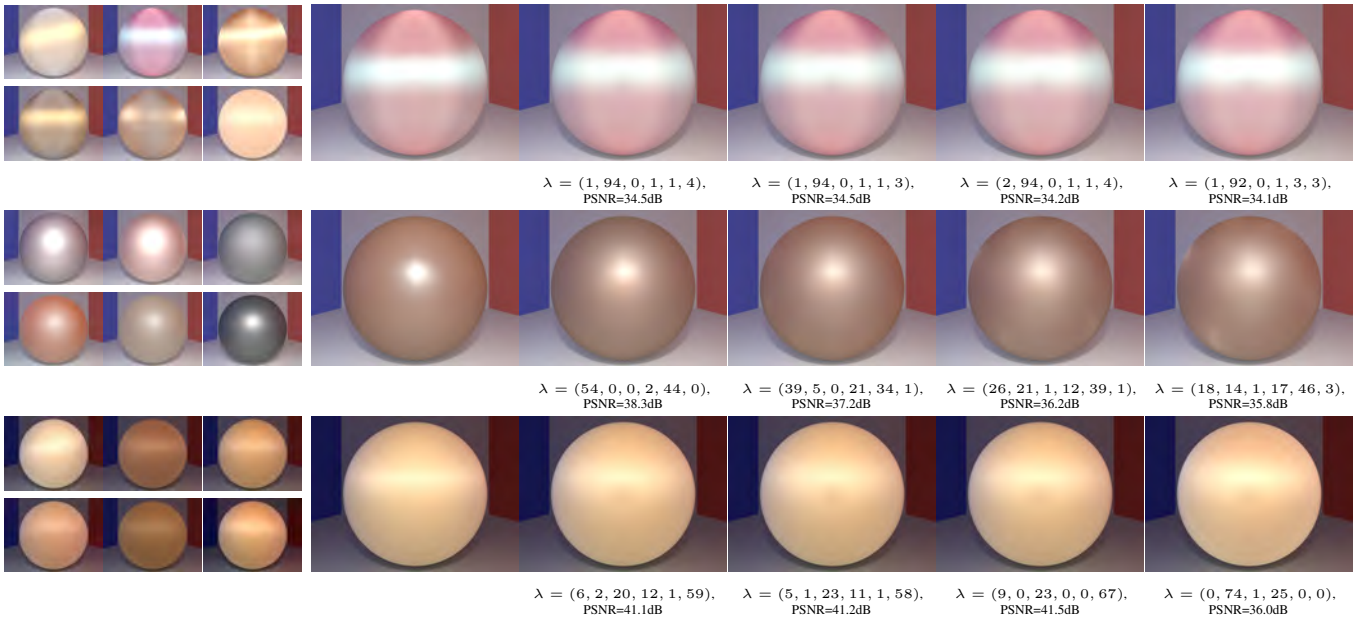


Figure 5: *Gradient accuracy compared to finite differentiation. We use the $S = 3$ 2D histograms of Fig. 4 (downsampled to $N = 64 \times 64$), and vary the number of iterations L and $\gamma = 2\sigma^2$.*

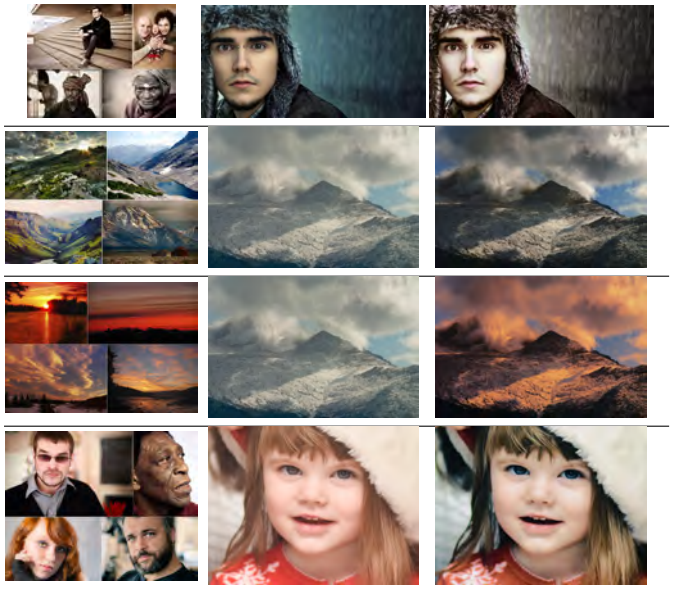
4.4 Evaluation

Problem (12) is non-convex, and optimization techniques may converge to local minima. In Figure 4, we illustrate the energy landscape of $\mathcal{E}_L(\lambda)$ for various loss functions on simple 2D shapes. The resulting landscapes show that the energy seems nearly convex. In practice, we observed repeatedly that the algorithm converged to the same minimum for all initializations λ_0 within Σ_S. In our experiments, we hence set $\lambda_0 = \mathbb{1}/S$. With this setting, quasi-newton approaches typically converge within 10 iterations. Figure 5 evaluates the gradient relative accuracy $\|\nabla\mathcal{E}_L(\lambda) - \nabla\mathcal{E}(\lambda)\|/\|\nabla\mathcal{E}(\lambda)\|$ when varying the number of iterations L for various regularizations $\gamma = 2\sigma^2$. As the exact Jacobian (Section 4.2) cannot be reliably evaluated, we approximate $\nabla\mathcal{E}(\lambda)$ using numerical finite differentiation with 5000 iterations. This shows a small number of iterations often suffices.



(a) BRDF database $(\bar{p}_s)_s$ (b) Reference $\bar{q}_0$ (c) $\bar{P}(\lambda)$, all data (d) $\bar{P}(\lambda)$, 25% data (e) $\bar{P}(\lambda)$, 10% data (f) $\bar{P}(\lambda)$, 5% data

Figure 6: *We fit a measured anisotropic BRDF $\bar{q}$ (b) using a basis of $S = 6$ measured BRDFs $(\bar{p}_s)_{s=1}^S$ (a). We obtain an approximation $\bar{P}(\lambda)$ (c) that remains robust when decimating measurements prior to the fitting (d,e,f). Reported λ are normalized so that $\sum_s \lambda_s = 100$, and PSNR values computed on BRDFs.*



(a) database $(f_s)_s$ (b) input f (c) color corrected $\tilde{f}$

Figure 7: *From a set $(f_s)_{s=1}^S$ of professional photographs (consisting of, from top to bottom: $S = 7$, $S = 12$, $S = 6$, and $S = 15$ photos), our algorithm projects a photograph f (b) to improved color-corrected or stylized photographs (c). Note that only the four most contributing professional photos (i.e. highest weights λ_s) are shown in (a). Their corresponding optimal weights $(\lambda_s)_s$, from top to bottom, are: $\lambda = (5.10^{-6}, 2.10^{-5}, 0.40, 0.60)$, $\lambda = (10^{-6}, 7.10^{-4}, 0.34, 0.66)$, $\lambda = (3.10^{-6}, 6.10^{-6}, 0.23, 0.77)$ and $\lambda = (0.05, 0.09, 0.29, 0.54)$.*

5 Applications

This section illustrates our histogram barycentric coordinates for various computer graphics applications.

5.1 Optimal Image Color Palettes

Color grading refers to the task of changing the color palette of a given image to improve its visual aspect. This task is often carried out manually by professional colorists [Pitié et al. 2007; Bonneel et al. 2013]. In this section, we build upon the automatic color grading approach pioneered by Reinhard et al. [2001] (see for instance [Bonneel et al. 2015] and references therein). This approach has shown that the visual style of images is contained to a large extent in the distribution of their colors, and that this visual aspect can be changed by adjusting the color palette of a given image to make it match that of another target image. In most automatic color grading approaches, that target palette is defined beforehand by choosing a single image's palette. We propose in this section a new approach to define such target palettes adaptively using multiple images.

Given an input photograph f and a small database of relevant color palettes, we compute first, among all barycenters of these palettes, the one that is the closest to the color palette of f. We then modify the distribution of colors in f to make it match that of this closest palette. Color transfer towards a single predefined palette often results in artefacts, especially when the target palette is very far from the original one [Rabin et al. 2010]. Our approach sidesteps this problem, and considers automatically an infinite family of target palettes that retain the characteristics of the database.

We discretize RGB histograms with $N = 128^3$ values on a uniform grid $(x_i)_{i=1}^N$ of the RGB cube. This defines the histograms $(p_s)_{s=1}^S$ of the input database $(f_s)_{s=1}^S$, and the histogram q of the image f to process. Our algorithm computes the optimal barycenter $P(\lambda)$ using the ground cost $C_{i,j} = \|x_i - x_j\|^2$ and the quadratic loss function

$\mathcal{L}(p,p') = \|p-p'\|^2$. The image f is modified into an image $\tilde{f}$, so that the histogram of $\tilde{f}$ is equal, up to a small approximation error, to $P(\lambda)$. This is achieved using the barycentric projection method detailed in [Solomon et al. 2015]. More precisely, if we denote p the histogram of the input image f, we use the Sinkhorn algorithm [Cuturi 2013] to determine a transport plan T between p and $P(\lambda)$, so that T solves (1) when setting $q = P(\lambda)$. The transport map in the RGB color space is then approximated by the barycentric projection map $x_i \in \mathbb{R}^3 \mapsto \frac{1}{p_i}\sum_j T_{i,j}x_j \in \mathbb{R}^3$. This map is applied to each pixel of f using linear interpolation (since these pixel values do not necessary fall on the discretization grid $(x_i)_i$) to obtain the modified image $\tilde{f}$. Note that this barycentric projection step is not related to the barycentric coordinate and the associated Wasserstein projection $P(\lambda)$ defined in Section 4. Figure 7 illustrates our method using a database of professional photographs. Figure 8 shows an application in the context of text-based user interfaces. We use the top 10 results of the Flickr image search engine (www.flickr.com) for the query *autumn* to stylize an input summer photograph with a more autumnal aspect. Among various loss functions, the quadratic loss offered the best quality/speed tradeoff for this application.

5.2 Sparse Reflectance Inference

Acquiring reflectance data can be cumbersome. Our method makes it possible to infer reflectance values from sparse data given a database of densely measured reflectances. A Bidirectional Reflectance Distribution Function (BRDF) $\bar{p}$ is a function $\bar{p}(\omega,\xi)$ describing the probability for a photon hitting a surface with a direction ω to be reflected off that surface with a direction ξ, or to be absorbed by that surface. These functions are sampled on discretized hemispheres $(\omega,\xi) \in \Omega^2$ of $N = 288$ points. Since $\bar{p}$ describes distributions of energy, we consider $(\bar{p}(\omega,\cdot))_{\omega\in\Omega}$ as a set of un-normalized histograms on the hemisphere. We thus first normalize the BRDF and define $p(\omega,\xi) \stackrel{\text{def.}}{=} \bar{p}(\omega,\xi)/\sum_{\xi'}\bar{p}(\omega,\xi')$, so that $(p(\omega,\cdot))_{\omega\in\Omega}$ is a collection of normalized histograms.

Given a database $(p_s)_{s=1}^S$ of such normalized BRDF $p_s(\omega,\xi)$, we extend (2) to define barycenters by jointly optimizing over all incoming direction ω

$$P(\lambda) \stackrel{\text{def.}}{=} \underset{(p(\omega,\xi))_{\xi,\lambda}}{\operatorname{argmin}} \sum_{\omega\in\Omega}\sum_s \lambda_s W(p(\omega,\cdot),p_s(\omega,\cdot)). \qquad (16)$$

We use the cost matrix $C_{i,j} = d(x_i,x_j)^2$ where d is the geodesic distance on the hemisphere. Using this extended notion of barycenters, we use (12) to define barycentric coordinates of a normalized BRDF q computed from some BRDF $\bar{q}$, where the loss $\mathcal{L}$ is the Wasserstein loss , extended to collections of histograms. The gradient of $\mathcal{E}_L$ is now obtained by summing the gradient contribution of all incident directions ω, so that we can use Algorithm 1 for the computation of the optimal λ. One finally recovers the interpolated BRDF $\bar{P}(\lambda)$ from $P(\lambda)$ by re-introducing the initial scaling factor of $\bar{q}$, i.e. $\bar{P}(\lambda)(\omega,\xi) \stackrel{\text{def.}}{=} P(\lambda)(\omega,\xi)\sum_{\xi'}\bar{q}(\omega,\xi')$

Fig. 6 shows typical results for isotropic and anisotropic BRDFs from the UTIA database [Filip and Vávra 2014]. To simulate a sparse acquisition setup, we progressively decimate (which corresponds to replacing some histogram values by 0) an input BRDF $\bar{q}_0$ by up to 95% to obtain the input BRDF q, prior to computing barycentric coordinates λ. We observe relatively good reconstruction quality even for highly degraded BRDFs. This suggests our approach could alleviate the BRDF capture process when reasonably similar materials have already been captured at higher resolution.

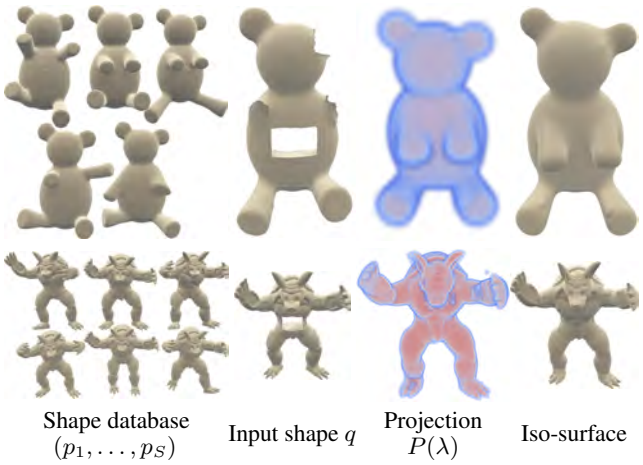


Figure 9: *We fit a 192^3 voxelized digital shape q, on a database of similar shapes $(p_1,\ldots,p_S)$. We obtain a projection $P(\lambda)$, with computed weights $\lambda = (7.10^{-4}, 0.928, 0.070, 6.10^{-4}, 4.10^{-4})$ (top row) and $\lambda = (0.295, 0.121, 0.067, 0.084, 0.163, 0.269)$ (bottom row), from which we extract a smooth iso-surface. Shapes in the first row are from the Princeton database [Chen et al. 2009].*

Figure 10: *Comparison of the Euclidean projection (left, $\lambda = (0.251, 0.287, 0.013, 0.076, 0.112, 0.187)$) and our Wasserstein projection (right, $\lambda = (0.295, 0.121, 0.067, 0.084, 0.163, 0.269)$. The Euclidean projection results in linearly blended shapes.*

5.3 Inferring missing geometry

Capturing geometries can be difficult due to partial occlusions, measurement noise, or unreachable camera angles. Given a database of input 3-D models, our tool can be used to infer missing geometry in an input mesh. We voxelize all shapes on a $N = 192^3$ uniform 3-D grid $(x_i)_{i=1}^N$ and we use a ground cost $C_{i,j} = \|x_i - x_j\|^2$, and $\mathcal{L} = \mathrm{KL}$ as loss function. Each shape is represented as a normalized histogram representing the uniform distribution inside this shape, and a uniform mass of $\varepsilon = 10^{-4}$ outside for compatibility with KL loss. We account for the mass missing in the input geometry by roughly estimating the amount of missing mass, and normalizing the input histogram accordingly. Specifically, if α percent of the input shape is missing, we use a loss of the form $\mathrm{KL}(P(\lambda),(1-\alpha)\frac{q}{\sum q})$. Figure 9 illustrates our results, and Figure 10 compares the Wasserstein and Euclidean projections.

5.4 MRI Data

We consider processed data from the Human Connectome Project Q1 data set. The processing includes skull stripping, brain segmentation and cortical reconstruction of the cerebrum and the cerebellum at the native resolution of 0.7mm, as provided by the NITRC website. We selected $S = 14$ original MRIs represented as volumetric histograms of dimension $208 \times 276 \times 225$, and considered an additional test one that we project on both the Euclidean simplex and

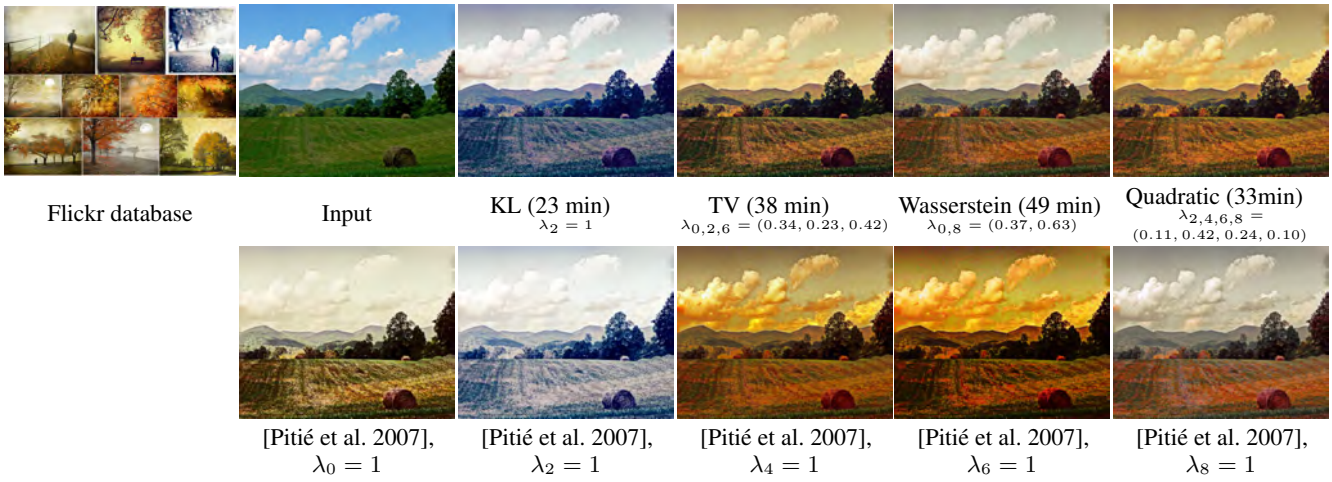


Figure 8: *Using the image search engine Flickr, we use the top 10 results for the query autumn (here, with Commercial use allowed and sorted by Interesting) and use them to color grade a summer image. (First row) For different loss functions, we show the non-zero barycentric coordinates and total computation time using 128^3 voxel RGB color histograms, $L = 60$ and our CPU implementation. (Second row) We use the color matching of Pitie et al. [2007] to transfer colors from the most contributing photographs (numbered 0, 2, 4, 6 and 8). As existing techniques use a single target histogram, this can lead to large color distortion.*

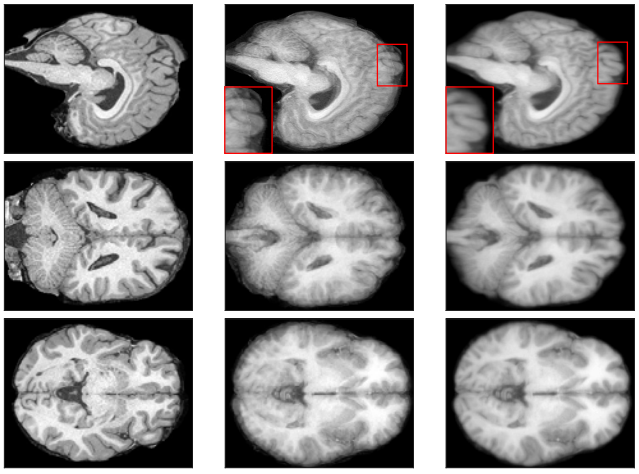


Figure 11: *(left) Original MRI, followed by two $208 \times 276 \times 225$ histogram projections, using the Euclidean simplex (middle) and the Wasserstein simplex (right), both computed using an ℓ_2 loss. The Euclidean barycentric coordinates consist in 8 non-zero values, while the Wasserstein barycentric coordinates have 9.*

the Wasserstein simplex of the 14 original MRIs, as illustrated in Figure 11. The test MRI is projected on both simplexes using a ℓ_2 loss. The coefficients selected by these two procedures have a sparse support, with 9 and 8 non-zero weights respectively. These two projections share 7 of these coefficients, with comparable weights. Although the Euclidean barycenter looks sharper, close inspection reveals overlapping boundaries and edges (see insets) while our Wasserstein projection results in well-defined contours.

6 Discussion

This paper introduces the concept of Wasserstein barycentric coordinates. We illustrate this tool with applications to color manipulation, reflectance approximation, and shape inference.

Figure 12: *When the database is too far from the input shape, our method produces poor reconstructions. The computed weight is $\lambda = (0.62, 4.10^{-4}, 0, 0, 0.38)$.*

Performance. While our method scales to large densely sampled histograms (we experimented with grids of size up to 256^3 and histograms supported on the sphere such as BRDFs), our method is limited by its memory requirements, and remains slow for databases exceeding more than 10-20 dense histograms. Memory requirements increase linearly with the number of iterations L, the number of input histograms S, and the number of bins N. In practice, we used between $L = 50$ and 100 iterations. A memory-free implementation would make the time complexity of the algorithm quadratic in the number of iterations instead of linear. Regarding speed, for a regression on a 10-histogram database typically converging within 10 L-BFGS iterations, each consisting of 100 fixed-point iterations, both our multicore CPU C++ implementation and multi-GPU matlab implementations perform about 40k convolutions. This ranges from seconds for 1D and 2D histograms to minutes for small 3D histograms ($\sim 64^3$) or hours for denser 3D histograms with the C++ implementation. The latter computations only require a few minutes on four K-80 GPUs. We found that initial L-BFGS iterations can be carried out using coarser gradient approximations, without impacting convergence.

Quality. When the histogram database is far from the input histogram, the input histogram will unlikely be faithfully approximated by Wasserstein barycenters. For applications such as shape inference, this can lead to erroneous reconstructions (Fig. 12).

Future work. We observed that our Wasserstein barycentric coordinates are often very sparse. This sparsity might be attributed

to the minimization being carried out over the set, in Σ_N, of all Wasserstein barycentric combinations of the histograms $(p_i)_i$. If the reference histograms are close to each other (meaning that the volume of that set is small) and the input histogram is far apart, the projection onto that set is likely to lie in one of the faces of that set, i.e., on barycentric combinations that have sparse weights. This poses an interesting theoretical problem, which would likely benefit from a better understanding of the energy landscape of $\mathcal{E}_L(\lambda)$. We believe that our method can find other practical applications in graphics, and find applications in vision and machine learning—notably if we consider extensions in which the basis $(p_i)_i$ is learned from data.

Acknowledgements

The work of G. Peyré has been supported by the European Research Council (ERC project SIGMA-Vision). N. Bonneel thanks Adobe for software donations. M. Cuturi gratefully acknowledges the support of JSPS young researcher A grant 26700002. Images from Flickr users: Joe Giordano, taquiman, Tom Babich (Fig. 1 and 7), Michael Villavicencio, Rod Waddington (Fig. 1), Paul Stevenson, d_pham, Rolands Lakis, NeilsPhotography, Shruti Biyani (Fig. 7, row 1), Ree Dexter, Richard P J Lambert (Fig. 7, row 2), Chris Sorge, Susanne Nilsson, Peggy2012CREATIVELENZ, Axel (Fig. 7, row 3), Yuri Samoilov, Neil Piddock, William Matthews, Luca Sartoni, Erik Drost (Fig. 7, row 4), Nick Kenrick, Gary Millar (Fig. 8)

A Differentials of Φ and Ψ

Proposition 4. *One has*

$$[\partial\varphi_s(b_s)]^\top = -K^\top \operatorname{diag}\left(\tfrac{p_s}{(Kb_s)^2}\right) K \tag{17}$$

$$[\partial\varphi(b)]^\top = \operatorname{diag}_s\left([\partial\varphi_s(b_s)]^\top\right)_s \tag{18}$$

$$[\partial_b\Phi(b,\lambda)]^\top = [\partial\varphi(b)]^\top \operatorname{diag}(\Phi(b,\lambda))P_\lambda \operatorname{diag}\left(\tfrac{1}{\varphi(b)}\right) \tag{19}$$

$$[\partial_b\Psi(b,\lambda)]^\top = [\partial\varphi(b)]^\top \operatorname{diag}(\Phi(b,\lambda))J_\lambda \tag{20}$$

$$[\partial_\lambda\Phi(b,\lambda)]^\top = \log(\varphi(b))^\top I_{N,S} \operatorname{diag}(\Phi(b,\lambda)) \tag{21}$$

$$[\partial_\lambda\Psi(b,\lambda)]^\top = \log(\varphi(b))^\top \operatorname{diag}(\Psi(b,\lambda)) \tag{22}$$

where we denoted

$$I_{N,S} \overset{\text{def.}}{=} (\mathrm{Id}_N,\ldots,I_N) \in \mathbb{R}^{N\times NS},$$
$$\forall a\in\mathbb{R}^N, J_\lambda(a) \overset{\text{def.}}{=} (\lambda_1 a,\ldots,\lambda_S a)\in\mathbb{R}^{N\times NS},$$
$$\forall b\in\mathbb{R}^{NS}, P_\lambda(b) \overset{\text{def.}}{=} (\lambda_s\sum_t b_t - b_s)_s \in \mathbb{R}^{NS},$$
$$\log(\varphi(b)) \overset{\text{def.}}{=} [\log(\varphi_1(b_1))|\ldots|\log(\varphi_S(b_s))] \in \mathbb{R}^{N\times S}.$$

Proof. For the sake of simplicity, we omit the dependency with respect to (b,λ) and write e.g. Ψ in place of $\Psi(b,\lambda)$. We also write $\Delta \overset{\text{def.}}{=} \operatorname{diag}$. Formula (17) and (18) are obtained by differentiating the definitions of φ_s and φ. Differentiating

$$\log(\Psi(b,\lambda)) = \sum_s \lambda_s\varphi_s(b_s) \tag{23}$$

with respect to b leads to

$$\Delta(\tfrac{1}{\Psi})\partial_b\Psi = (\Delta(\tfrac{\lambda_1}{\varphi_1})\partial_{b_1}\varphi_1|\ldots|\Delta(\tfrac{\lambda_S}{\varphi_S})\partial_{b_S}\varphi_S) = J_\lambda^\top\Delta(\tfrac{1}{\varphi})\partial_b\varphi$$

and hence $[\partial_b\Psi]^\top = [\partial_b\varphi]^\top\Delta(\frac{1}{\varphi})J_\lambda\Delta(\Psi)$. We then use the fact that $\Delta(\frac{1}{\varphi})J_\lambda\Delta(\Psi) = \Delta(\Phi)J_\lambda$ to obtain formula (20). Differentiating $\Phi = (\frac{\Psi(b)}{\varphi_s(b_s)})_s$ with respect to b leads to

$$\begin{aligned}\partial_b\Phi &= \begin{pmatrix}\Delta(\frac{1}{\varphi_1})\partial_b\Psi\\ \ldots\\ \Delta(\frac{1}{\varphi_S})\partial_b\Psi\end{pmatrix} - \operatorname{diag}\left(\Delta(\tfrac{\Psi}{\varphi_1^2})\partial\varphi_1,\ldots,\Delta(\tfrac{\Psi}{\varphi_S^2})\partial\varphi_S\right)\\ &= \Delta(\tfrac{1}{\varphi})I_{N,S}^\top[\partial_b\Psi] - \Delta(\tfrac{1}{\varphi})\Delta(\Phi)\partial\varphi\end{aligned}$$

and hence, transposing this relation,

$$[\partial_b\Phi]^\top = [\partial\varphi]^\top\Delta(\Phi)\left(J_\lambda I_{N,S} - \mathrm{Id}_{NS}\right)\Delta(\tfrac{1}{\varphi})$$

which is the desired formula (19) since $J_\lambda I_{N,S} - \mathrm{Id}_{NS} = P_\lambda$. Differentiating (23) with respect to λ gives

$$\Delta(\tfrac{1}{\Psi})\partial_\lambda\Psi = (\log(\varphi_1(b_1))|\ldots|\log(\varphi_s(b_S)))$$

and hence formula (22). Differentiating $\Phi = (\frac{\Phi(b)}{\varphi_s(b_s)})_s$ with respect to λ leads to

$$\partial_\lambda\Phi = \begin{pmatrix}\Delta(\frac{1}{\varphi_1})\partial_\lambda\Psi\\ \ldots\\ \Delta(\frac{1}{\varphi_S})\partial_\lambda\Psi\end{pmatrix} = \Delta(\tfrac{1}{\varphi})I_{N,S}^\top\partial_\lambda\Psi$$

and hence, transposing this relation,

$$[\partial_\lambda\Phi]^\top = \log(\varphi)^\top\Delta(\Psi)I_{N,S}^\top\Delta(\tfrac{1}{\varphi}) = \log(\varphi)^\top I_{N,S}^\top\Delta(\Phi)$$

which shows (21). □

B Proof of Proposition 3

Differentiating (8) and (9) for $\ell\geq 0$ leads to

$$[\partial P^{(\ell)}(\lambda)]^\top = B^{(\ell)}\Psi_b^{(\ell)} + \Psi_\lambda^{(\ell)}, \tag{8'}$$

$$B^{(\ell+1)} = B^{(\ell)}\Phi_b^{(\ell)} + \Phi_\lambda^{(\ell)}, \tag{9'}$$

where, $B^{(\ell)} \overset{\text{def.}}{=} [\partial b^{(\ell)}(\lambda)]^\top$ and $B^{(0)}=0$ to agree with the constant initialization $b_s^{(0)} = \mathbb{1}$. Applying (8') to $u^{(L)}$ shows that

$$\nabla\mathcal{E}_L(\lambda) = [\partial P^{(L)}(\lambda)]^\top(u^{(L)}) = B^{(L)}(v^{(L)}) + \Psi_\lambda^{(L)}(u^{(L)}) \tag{24}$$

where we denoted $v^{(L)} \overset{\text{def.}}{=} \Psi_b^{(L)}(u^{(L)})$. So all we need to compute now is $B^{(L)}(v^{(L)})$. Using (9') shows that

$$B^{(L)}(v^{(L)}) = \sum_{\ell=0}^{L-1}\Phi_\lambda^{(\ell)}(v^{(\ell)}), \tag{25}$$

where the $(v^{(\ell)})_{\ell=0}^{L-1}$ are computed using the backward recursion (15). Note that the sum appearing in (25) starts at $\ell=0$ because $B^{(0)}=0$. Plugging (25) into expression (24) gives the desired formula (14).

References

AGUEH, M., AND CARLIER, G. 2011. Barycenters in the Wasserstein space. *SIAM J. on Mathematical Analysis 43*, 2.

BENAMOU, J.-D., AND BRENIER, Y. 2000. A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem. *Numerische Mathematik 84*, 3, 375–393.

BENAMOU, J.-D., CARLIER, G., CUTURI, M., NENNA, M., AND PEYRÉ, G. 2015. Iterative Bregman projections for regularized transportation problems. *SIAM J. on Sci. Computing 2*, 37.

BIGOT, J., AND KLEIN, T. 2012. Consistent estimation of a population barycenter in the Wasserstein space. *Preprint arXiv:1212.2562*.

BIGOT, J., GOUET, R., KLEIN, T., AND LÓPEZ, A. 2015. Geodesic PCA in the Wasserstein space by Convex PCA. *Annales de l'Institut Henri Poincaré B: Probability and Statistics*.

BONNEEL, N., VAN DE PANNE, M., PARIS, S., AND HEIDRICH, W. 2011. Displacement interpolation using lagrangian mass transport. *ACM Trans. Graph. (SIGGRAPH Asia) 30*, 6.

BONNEEL, N., SUNKAVALLI, K., PARIS, S., AND PFISTER, H. 2013. Example-Based Video Color Grading. *ACM Trans. Graph. (SIGGRAPH) 32*, 4.

BONNEEL, N., RABIN, J., PEYRÉ, G., AND PFISTER, H. 2015. Sliced and Radon Wasserstein barycenters of measures. *J. of Mathematical Imaging and Vision 51*, 1.

BRAND, M., AND BRAND, M. 2003. Charting a manifold. In *Adv. in Neural Information Proc. Sys.*

BREGMAN, L. M. 1967. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR computational mathematics and mathematical physics 7*, 3, 200–217.

BURKARD, R., DELL'AMICO, M., AND MARTELLO, S. 2009. *Assignment Problems*. Society for Industrial and App. Math.

CHEN, X., GOLOVINSKIY, A., AND FUNKHOUSER, T. 2009. A benchmark for 3D mesh segmentation. *ACM Trans. Graph. (SIGGRAPH) 28*, 3.

CUTURI, M., AND DOUCET, A. 2014. Fast computation of Wasserstein barycenters. In *Int. Conf. on Machine Learning (ICML)*.

CUTURI, M. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *Adv. in Neural Information Proc. Sys.*

DEMING, W. E., AND STEPHAN, F. F. 1940. On a least squares adjustment of a sampled frequency table when the expected marginal totals are known. *Annals Mathematical Statistics 11*, 4.

FILIP, J., AND VÁVRA, R. 2014. Template-based sampling of anisotropic BRDFs. *Computer Graphics Forum (PG) 33*, 7.

HAKER, S., ZHU, L., TANNENBAUM, A., AND ANGENENT, S. Optimal mass transport for registration and warping. *International Journal of Computer Vision 60*, 3.

KANTOROVICH, L. 1942. On the transfer of masses (in russian). *Doklady Akademii Nauk 37*, 2, 227–229.

LEWIS, A. S., AND OVERTON, M. L. 2013. Nonsmooth optimization via quasi-newton methods. *Math. Programming 141*.

MATUSIK, W., PFISTER, H., BRAND, M., AND MCMILLAN, L. 2003. A data-driven reflectance model. *ACM Trans. Graph. (SIGGRAPH) 22*, 3.

MÉRIGOT, Q. 2011. A multiscale approach to optimal transport. *Computer Graphics Forum (SGP) 30*, 5.

MONGE, G. 1781. Mémoire sur la théorie des déblais et des remblais. *Histoire de l'Académie Royale des Sciences*, 666–704.

NEIDINGER, R. D. 2010. Introduction to automatic differentiation and MATLAB object-oriented programming. *SIAM Rev. 52*, 3.

PITIÉ, F., KOKARAM, A., AND DAHYOT, R. 2007. Automated colour grading using colour distribution transfer. *Computer Vision and Image Understanding*.

RABIN, J., AND PAPADAKIS, N. 2015. Convex color image segmentation with optimal transport distances. In *Proc. SSVM'15*.

RABIN, J., DELON, J., AND GOUSSEAU, Y. 2010. Regularization of transportation maps for color and contrast transfer. In *IEEE International Conference on Image Processing (ICIP)*.

RABIN, J., PEYRÉ, G., DELON, J., AND BERNOT, M. 2012. Wasserstein barycenter and its application to texture mixing. In *Scale Space and Variational Methods in Computer Vision*.

REINHARD, E., ASHIKHMIN, M., GOOCH, B., AND SHIRLEY, P. 2001. Color transfer between images. *IEEE Comput. Graph. Appl. 21*, 5.

ROLET, A., CUTURI, M., AND PEYRÉ, G. 2016. Fast dictionary learning with a smoothed Wasserstein loss. In *Proc. AISTATS'16*.

RUBNER, Y., TOMASI, C., AND GUIBAS, L. 1998. A metric for distributions with applications to image databases. In *Computer Vision, 1998. Sixth International Conference on*, 59–66.

RUBNER, Y., TOMASI, C., AND GUIBAS, M. J. 2000. The earth mover's distance as a metric for image retrieval. *International J. of Computer Vision 40*, 2000.

RUSTAMOV, R. M. 2010. Barycentric coordinates on surfaces. *Computer Graphics Forum 5*, 29.

SANDLER, R., AND LINDENBAUM, M. 2009. Nonnegative matrix factorization with earth mover's distance metric. In *IEEE Computer Vision and Pattern Recognition (CVPR)*.

SCHMIDT, M. W., VAN DEN BERG, E., FRIEDLANDER, M. P., AND MURPHY, K. P. 2009. Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 5.

SEGUY, V., AND CUTURI, M. 2015. Principal geodesic analysis for probability measures under the optimal transport metric. In *Adv. in Neural Information Proc. Sys.*

SINKHORN, R. 1964. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Statist. 35*.

SOLOMON, J., RUSTAMOV, R., GUIBAS, L., AND BUTSCHER, A. 2014. Earth mover's distances on discrete surfaces. *ACM Trans. Graph. (SIGGRAPH) 33*, 4.

SOLOMON, J., RUSTAMOV, R., GUIBAS, L., AND BUTSCHER, A. 2014. Wasserstein propagation for semi-supervised learning. In *Int. Conf. on Machine Learning (ICML)*.

SOLOMON, J., DE GOES, F., PEYRÉ, G., CUTURI, M., BUTSCHER, A., NGUYEN, A., DU, T., AND GUIBAS, L. 2015. Convolutional Wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Trans. Graph. (SIGGRAPH) 34*, 4.

SRIVASTAVA, S., CEVHER, V., TRAN-DINH, Q., AND DUNSON, D. B. 2015. Wasp: Scalable bayes via barycenters of subset posteriors. In *International Conference on Artificial Intelligence and Statistics*.

VILLANI, C. 2003. *Topics in optimal transportation*. American Mathematical Soc.

VILLANI, C. 2008. *Optimal transport: old and new*, vol. 338.

WILLS, J., AGARWAL, S., KRIEGMAN, D., AND BELONGIE, S. 2009. Toward a perceptual space for gloss. *ACM Trans. Graph. 28*, 4.