Master 2 Mathématiques, Vision, Apprentissage

ENS Paris-Saclay

Wasserstein Barycenter and Its Application to Texture Mixing

Course — Geometric Data Analysis (Jean Feydy)

Report by

Ewerthon MELZANI
Amine MAAZIZI
Sammy OUKACI

Academic year 2025–2026

Efficient Wasserstein Barycenter Computation with Application to Texture Mixing

Ewerthon Melzani
ewerthon.melzani@ensta.fr
ewerthon.melzani@ip-paris.fr
ENSTA Paris
ENS Paris-Saclay
France

Amine Maazizi
amine.maazizi@ensta.fr
amine.maazizi@ip-paris.fr
ENSTA Paris
ENS Paris-Saclay
France

Sammy Oukaci
sammy.oukaci@ensta.fr
sammy.oukaci@ip-paris.fr
ENSTA Paris
ENS Paris-Saclay
France

Abstract

This report presents a detailed study of a paper by Rabin Julien, Gabriel Peyré, Julie Delon, and Bernot Marc, published in 2011. This article is about the notion of Wasserstein Barycenter and its application to texture mixing.
At that time, Optimal Transport emerged as a powerful tool in numerous domains such as computer vision and machine learning, however, for large datasets, it suffers from high computational costs. To remedy this issue, this paper introduces the notion of Wasserstein barycenter and the use of sliced optimal transport to approximate it. This study illustrates the potential of optimal transport for large-scale data analysis and texture synthesis.
In addition, The studied paper has limitations in efficiency and reproducibility. To address this, we present an optimized implementation that overcomes these issues and ensures consistent results. Furthermore, we make our code publicly available to facilitate reproducibility and support further research. https://github.com/EwerthonMelzani/MVA_Wasserstein_barycenter_et_Application_to_texture_mixing.

Keywords

Wassterstein distance, Sliced Wasserstein distance, Wasserstein barycenter, stochastic Newton/gradient descent, texture mixing.

1 Introduction

It is common to represent images and textures as point clouds, which can be seen as discrete densities within the space in which they are defined. Optimal transport is a tool to deal with such spaces of densities, by making sense of the notion of distance between two of them, through a distance called the Wasserstein distance. Hence, research in this field seems natural in order to find new ways to exploit such data in useful ways.

1.1 Context of the paper.

This paper [7] was written in 2010, and presented in the 2011 `Scale Space and Variational Methods` conference (SSVM), before Cuturi's 2013 breakthrough [4], introducing entropic regularization. The main problem, at that time, for exploiting optimal transport is its intractability, computing directly the Wasserstein distance requires to solve a linear program which can be done with an $O(N^{2.5}log(N))$ complexity, with N the number of considered points. That is why the authors consider what is called the sliced Wasserstein distance, a more tractable version, based on the special structure of 1D optimal transport. This remarkable paper has given birth to a subfield of optimal transport called `Sliced Optimal Transport` (SOT), still studied and developed today with publications in top-tier conferences, for example (Kolouri et al [10]), studying generalization for other types of sliced distances, has been presented during the 2019 NeurIPS conference.
Indeed SOT has important applications because in the case where one considers continuous densities, optimal transport suffers severely from the curse of dimensionality for approximating these densities with samples, sliced Wasserstein distance helps to remedy this issue.

1.2 Target applications

This article aims to use the framework of sliced Wasserstein barycenters to address texture mixing problems, where the goal is to synthesize a new texture from a collection of exemplar textures $(f_j)_j$ such that the output meaningfully integrates the colors and structural attributes of the exemplars. By meaningful, we mean that the resulting texture preserves important geometric features and statistical properties of the input textures.
Prior to 2010, various approaches existed, such as texture metamorphosis (2002) or Heeger & Bergen (1995) [11], primarily based on patch copying or statistical matching of filter responses. In contrast, this work adopts a geometric perspective, leveraging the notion of the Wasserstein barycenter for distributions of wavelet or pixel coefficients, as introduced by Agueh & Carlier (2011, SIAM [6]). Using sliced optimal transport (SOT) allows for a tractable approximation of these barycenters and enables meaningful texture synthesis from multiple exemplars.

1.3 Plan of the analysis

We aim to provide a complete presentation of this document, for both the mathematical part and the practical part. We have implemented and optimized as much as possible the presented algorithms. Our presentation is linear, following the same order as the paper. The article has a strong theoretical background, that we will present in detail in what follows, in particular, we will provide proofs for the important results and sketches of proofs for other results. Then we will detail the gradient descent algorithm proposed in the article and its implementation. Following that, we present the results we obtain for the texture mixing problem. Then, we explore the limitations and the potential improvements. Finally, we conclude by summarizing all has been done in a short paragraph.

2 Theoretical background

Optimal transport is a complex mathematical theory connecting optimization and probability. It aims to define a framework in which

we can build a topology for measures.
The article only focuses on discrete measures of the form $\sum_{i=1}^{N} \delta_{X_i}$ in $\mathbb{R}^d$ for some $d \in \mathbb{N}$ with $(X_i)_{i\in[\![1,N]\!]}$ being a point cloud with N points with equal weight on each point, but most of the results presented remain true when considering measures with density.

2.1 Wasserstein distance

We recall what is the Wasserstein distance in the discrete case: when one considers two point clouds, $(X_i)_{i\in[\![1,N]\!]}$ and $(Y_j)_{j\in[\![1,N]\!]}$, with the same number of points N, let Σ_N be the set of all permutations of $[\![1, N]\!]$ and we define the quadratic Wasserstein distance between X and Y as:

$$W(X,Y)^2 := \min_{\sigma\in\Sigma_N} \sum_{i=1}^{N} \left\|X_i - Y_{\sigma(i)}\right\|^2$$

If we consider $X = (X_i)_{i\in[\![1,N]\!]}$, then for any permutation $\sigma \in \Sigma_N$ X represents the same point cloud as $(X_{\sigma(i)})_{i\in[\![1,N]\!]}$. So in this case we say that two point clouds X and Y are equal if and only if there exists $\sigma \in \Sigma_N$ such that $\forall i \in [\![1, N]\!]$, $X_i = Y_{\sigma(i)}$, and we can define the quotient space $(\mathbf{R}^d)^N \text{modulo} \Sigma_N$.
The Wasserstein distance is indeed a distance between two point clouds with N points. We can prove this result using the gluing lemma and the properties of the Euclidean distance. Note that not only is it important to compute the value of $W(X, Y)$, but also, the optimal permutation, that can bring meaningful information, as we will see in the application to texture mixing.

The main problem of the Wasserstein distance is its intractability. To see that, we can view this problem as being equivalent to a linear programming problem, solvable with $O(N^{2.5}log(N))$ complexity. We provide the proof in which we assume two theorems. We denote by $\mathcal{B}_N$ the set of bistochastic matrices, i.e., matrices with non-negative coefficients that sum to 1 along the rows and the columns, and we denote by $\mathcal{P}_N$ the set of permutation matrices, i.e., the matrices with only one coefficient equal to 1 on each row and each column. We see that $\mathcal{P}_N = \mathcal{B}_N \cap \{0, 1\}^{N\times N} \subset \mathcal{B}_N$. We can write the optimization problem defining W with this notation:

$$W(X,Y)^2 := \min_{P\in\mathcal{P}_N} \sum_{i=1}^{N}\sum_{j=1}^{N} P_{i,j} \left\|X_i - Y_j\right\|^2$$

In [7], the authors claim (section 2.1 page 3) that this problem is strictly equivalent to:

$$\min_{P\in\mathcal{B}_N} \sum_{i=1}^{N}\sum_{j=1}^{N} P_{i,j} \left\|X_i - Y_j\right\|^2$$

This quantity is necessarily less than $W(X, Y)^2$, because $\mathcal{P}_N \subset \mathcal{B}_N$, we want to prove the equality.
It is a linear programming problem, because the cost function and the constraints defining $\mathcal{B}_N$ are linear in P. To prove the equivalence, we use the fundamental theorem of linear programming, stating that among all the solutions of a linear program problem, there is at least one of them that is an extreme point of the constraint set. The set of extreme points of a set C is defined as: $\text{Extr}(C) = \{p \in C, \text{ such that }, \forall (p_1, p_2) \in C^2, p = \frac{1}{2}(p_1 + p_2) \implies p = p_1 \text{ and } p = p_2\}$.
Moreover, we can now use the Birkhoff–von Neumann theorem, stating that $\text{Extr}(\mathcal{B}_N) = \mathcal{P}_N$, using the previous theorem, we can conclude that the values of the two problems are indeed equal.
More precisely, $\forall c \in \mathbb{R}$ we define the hyperplanes (using the variable $M = (m_{i,j})_{i,j} \in \mathbb{R}^{N\times N}$).

$$H(c) = \{M \in \mathbb{R}^{N\times N} \text{ such that }, \sum_{i=1}^{N}\sum_{j=1}^{N} m_{i,j} \left\|X_i - Y_j\right\|^2 = c\}$$

When c describes $\mathbb{R}$, the hyperplanes describe the set of all the $N\times N$ real matrices. In particular, since the bounded set $\mathcal{B}_N$, is connex, there exists an interval $I := [c_1, c_2]$ such that $\forall c \in I, H(c) \cap \mathcal{B}_N \neq \emptyset$. Thus c_1 is the minimum of the problem, and $H(c_1)$ the set of optimal solutions. If this hyperplane $H(c_1)$ intersects a face of the polytope $\mathcal{B}_N$, then there is an infinite number of solutions, if not, then $H(c_1)$ is just a single extremal point. since the first case is very unlikely, we then understand that an algorithm solving solving the linear programming problem almost surely returns a permutation matrix and then the permutation we are looking for, not only the value of the Wasserstein distance.
It is then explained that the best algorithm available to the authors to solve this problem has an $O(N^{2.5}log(N))$ complexity. This is intractable for the applications.

In the special case of dimension 1, we know a direct method to find the optimal permutation: We want to show that the optimal permutation is $\sigma := \sigma_Y \circ \sigma_X^{-1}$ with σ_X and σ_Y the permutations that order X and Y increasingly. We show this result in a more general case, with a cost of the form $h(x - y)$ with h convex. In our case $t \mapsto t^2$ is convex.
To prove this result, we need the following lemma: If $a < c$ and $b < d$, then for a convex function h:

$$h(a - d) + h(c - b) \geq h(a - b) + h(c - d).$$

Proof. We have (note that a convex function is almost everywhere differentiable):

$$h(a - d) - h(a - b) = \int_b^d h'(a - t)dt$$

and

$$h(c - d) - h(c - b) = \int_b^d h'(c - t)dt$$

but since h is convex and $a < c$ we know that: $h'(a - t) \leq h'(c - t)$ for almost all t . Therefore, we can conclude for this lemma since $\int_b^d h'(a - t)dt \leq \int_b^d h'(c - t)dt$.
If we have σ such that, for some i, j $X_i < X_j$ and $Y_{\sigma(j)} < Y_{\sigma(i)}$, using the lemma, we obtain:

$$h(X_i - Y_{\sigma(j)}) + h(X_j - Y_{\sigma(i)}) \leq h(X_i - Y_{\sigma(i)}) + h(X_j - Y_{\sigma(j)})$$

Thus, defining $\tilde{\sigma}$ such that $\tilde{\sigma}(t) = \sigma(t)$ if $t \neq i, j, \tilde{\sigma}(i) = \sigma(j)$ and $\tilde{\sigma}(j) = \sigma(i)$ gives a better permutation. This shows, iterating this process, we progressively order the mapping until eventually reaching $\sigma := \sigma_Y \circ \sigma_X^{-1}$, which shows that this permutation is optimal (but may not be unique if h is not strictly convex).
In order to find the best permutation, and the value of the Wasserstein distance, one simply has to sort X and Y, which can be done

with an $O(Nlog(N))$ complexity, using, for example, a merge sort algorithm.

2.2 Sliced Wasserstein distance

The brilliant idea of the paper is to use this special structure of the 1D Wasserstein distance to define a new, more tractable distance. We still consider that X and Y are N-points clouds in $\mathbb{R}^d$, and we denote $\mathcal{S}^{d-1}$ the unit sphere of $\mathbb{R}^d$. For $\theta \in \mathcal{S}^{d-1}$, we define $X_\theta = (\langle X_i\, \theta\rangle)_{i\in[\![1,N]\!]}$ (we define Y_θ similarly).
The quadratic sliced Wasserstein distance is defined as:

$$\begin{aligned}\tilde{W}(X,Y)^2 &= \int_{\theta\in\mathcal{S}^{d-1}} W(X_\theta, Y_\theta)^2\, d\theta \\ &= \int_{\theta\in\mathcal{S}^{d-1}} \min_{\sigma_\theta\in\Sigma_N} \sum_{i=1}^{N} \left|\langle X_i - Y_{\sigma_\theta(i)},\, \theta\rangle\right|^2 d\theta\end{aligned}$$

Note that in the literature, the distance is usually normalized by the measure $|\mathcal{S}^{d-1}|$ of the unit sphere.
We show that the sliced Wasserstein distance is indeed a distance between point clouds. We use the fact that the Wasserstein distance is itself a distance.
We can show that $\tilde{W}$ satisfies the conditions to be a distance:

Non negativity: follows directly from the non-negativity of W.

Symmetry: By the symmetry of W, we have:

$$\begin{aligned}\tilde{W}^2(X,Y) &= \int_{\theta\in\mathcal{S}^{d-1}} W^2(X_\theta, Y_\theta)d\theta = \\ &= \int_{\theta\in\mathcal{S}^{d-1}} W^2(Y_\theta, X_\theta)d\theta = \tilde{W}^2(Y,X)\end{aligned}$$

Triangle inequality: Let X,Y,Z be three point clouds: We have, since W is a distance:

$$\begin{aligned}\tilde{W}(X,Y) &\le \left[\int_{\theta\in\mathcal{S}^{d-1}} \left(W(X_\theta, Z_\theta) + W(Z_\theta, Y_\theta)\right)^2 d\theta\right]^{\frac{1}{2}} \\ &\le \tilde{W}(X,Z) + \tilde{W}(Z,Y)\end{aligned}$$

by the Minkowski inequality.

4/Identity of indiscernibles:
If $X = Y$, it implies $\tilde{W}^2(X,Y) = \int_{\theta\in\mathcal{S}^{d-1}} 0^2 d\theta = 0$.
And if $\tilde{W}(X,Y) = 0$. Then, by definition, one has:

$$\int_{\theta\in\mathcal{S}^{d-1}} \min_{\sigma_\theta\in\Sigma_N} \sum_{i=1}^{N} \left|\langle X_i - Y_{\sigma_\theta(i)},\, \theta\rangle\right|^2 d\theta = 0$$

This implies that for almost every $\theta \in \mathcal{S}^{d-1}$,

$$\min_{\sigma_\theta\in\Sigma_N} \sum_{i=1}^{N} \left|\langle X_i - Y_{\sigma_\theta(i)},\, \theta\rangle\right|^2 = 0$$

Since the set of $\theta \in \mathcal{S}^{d-1}$ orthogonal to one of the $X_i - Y_j$ for $(i,j) \in [\![1,N]\!]^2$ has a measure 0, because it is the union of a finite number of hyperplanes, there exists a permutation σ such that $\forall i \in [\![1,N]\!]$, $X_i = Y_{\sigma_{(i)}}$. We can then conclude that X and Y are the same point cloud in the quotient space.

Thus, the sliced Wasserstein distance is a distance. We could show that it remains a distance in the general case when one considers continuous distributions, using the notion of push-forward to generalize the 1D projection and the use of a Fourier transform in point 4.

The sliced Wasserstein distance is presented as an approximation of the true Wasserstein distance. However, it is not the case, is the sense that the values of the two distances are not close.
We show the following result explaining that for all N-point clouds X and Y in $\mathbb{R}^d$:

$$\tilde{W}(X,Y)^2 \le \frac{|\mathcal{S}^{d-1}|}{d} W(X,Y)^2.$$

To prove this result, we first provide a proof for the following lemma:
For a fixed dimension d, there exists c_d only depending on d such that, for all $v \in \mathbb{R}^d$ we have:

$$\|v\|^2 = c_d \int_{\theta\in\mathcal{S}^{d-1}} \langle v,\theta\rangle^2 d\theta.$$

To show that, let $v \in \mathbb{R}^d$, $e_1 = \frac{v}{\|v\|}$, and complete an orthonormal basis $(e_1, e_2, ...e_d)$ we define the scalar product with respect to this basis.
Then, still using this base, $\forall\theta = (\theta_1, ...\theta_d) \in \mathcal{S}^{d-1}$ one has:

$$\int_{\theta\in\mathcal{S}^{d-1}} \langle v,\theta\rangle^2 d\theta = \|v\|^2 \int_{\theta\in\mathcal{S}^{d-1}} \theta_1^2 d\theta$$

Now, we can use the symmetry of the sphere to say that

$$\forall i \in [\![1,N]\!], \int_{\theta\in\mathcal{S}^{d-1}} \theta_1^2 d\theta = \int_{\theta\in\mathcal{S}^{d-1}} \theta_i^2 d\theta.$$

Finally:

$$\int_{\theta\in\mathcal{S}^{d-1}} \langle v,\theta\rangle^2 d\theta = \|v\|^2 \frac{1}{d} \int_{\theta\in\mathcal{S}^{d-1}} \sum_{i=1}^{d} \theta_i^2 d\theta = \|v\|^2 \frac{|\mathcal{S}^{d-1}|}{d}$$

So $c_d = \frac{d}{|\mathcal{S}^{d-1}|}$
We can use this fact with $v = X_i - Y_{\sigma_\theta(i)}$ and get, for any permutations σ:

$$\begin{aligned}\tilde{W}^2(X,Y) &= \int_{\theta\in\mathcal{S}^{d-1}} \min_{\sigma_\theta\in\Sigma_N} \sum_{i=1}^{N} \left|\langle X_i - Y_{\sigma_\theta(i)},\, \theta\rangle\right|^2 d\theta \\ &\le \int_{\theta\in\mathcal{S}^{d-1}} \sum_{i=1}^{N} \left|\langle X_i - Y_{\sigma(i)},\, \theta\rangle\right|^2 d\theta \\ &= c_d^{-1} \sum_{i=1}^{N} \|X_i - Y_{\sigma(i)}\|^2\end{aligned}$$

Since it holds for any permutations σ, it holds for the minimum, and we obtain the result we wanted, by the definition of W. In his phd thesis from 2013[2], Nicolas Bonnotte (theorem 5.1.5) , proves this result in the general case and provides the following result (that we adapt in the case of discrete measures and quadratic cost, so with p=2 with the notations of Bonnotte):

If X and Y are both contained in a ball of radius R, in $\mathbb{R}^d$, then, there exists C_d only depending on the dimension d, such that:

$$W(X,Y)^2 \leq C_d R^{\frac{1}{d+1}} \tilde{W}(X,Y)^{\frac{1}{d+1}}$$

We now understand that the sliced Wasserstein distance is linked to standard Wasserstein distance in the sense that for X an N-point cloud and $(X_n)_{n\in\mathbb{N}}$ a sequence of N-point clouds $W(X_n, X) \to 0$ is equivalent $\tilde{W}(X_n, X) \to 0$. We note that the convergence in both cases implies boundedness of $(X_n)_{n\in\mathbb{N}}$, so we can use Bonnotte's result.

2.3 Wasserstein barycenters

The link between the theory and practical application is made through the introduction of the Wasserstein barycenter, which is a crucial tool to understand the geometry of point clouds and understand how to make sense of weighted averages for those objects.

The general framework described the paper by Agueh and Carlier [6] involves deep topological concepts, they explain that the study of such barycenters had been studied extensively by Sturm in the context of nonpositively curved spaces, which is not the case of the Wasserstein space.

We consider $(Y^j)_{j\in[\![1,m]\!]}$ m N-point clouds in $\mathbb{R}^d$, and $(\rho_j)_{j\in[\![1,m]\!]}$ the weights such that, $\forall j \in [\![1, N]\!]$ $\rho_j \geq 0$ and $\sum_{j=1}^m \rho_j = 1$.

The Wasserstein barycenter is defined as:

$$\mathrm{Bar}(\rho_j, Y^j)_{j\in[\![1,m]\!]} \in \arg\min_X E(X), \qquad E(X) = \sum_{j=1}^m \rho_j W(X, Y^j)^2$$

In a similar way as for the Wasserstein distance, this optimization problem can be written as a linear programming problem involving a very large set of variables, and then is computationally intractable. Once again, we study the special properties of 1D optimal transport, so we suppose the Y^j are N-points clouds in dimension 1.

We assume (like in the article) without loss of generality, since we work in the quotient space, the X_i in the cost function $E(X)$ are in increasing order. We want to show that, in dimension one, the barycenter is the N-point cloud defined by:

$$\forall\, 1 \leq i \leq N, \quad \left(\mathrm{Bar}(\rho_j, Y^j)_{j\in[\![1,m]\!]}\right)_i = \sum_{j=1}^m \rho_j\, Y^j_{\sigma_{Yj}(i)}$$

With σ_{Yj} the permutation ordering the elements of Y^j in increasing order.

To do that, we write the definition of the barycenter:

$$\arg\min_X E(X) = \arg\min_X \sum_{j=1}^m \rho_j \min_{\sigma\in\Sigma_N} \sum_{i=1}^N \|X_i - Y^j_{\sigma(i)}\|^2$$

Since, in dimension one, as we saw, the optimal permutation is the one ordering increasingly the elements of Y^j, for all $j \in [\![1, m]\!]$, so σ_{Yj} we end up with:

$$\arg\min_X E(X) = \arg\min_X \sum_{i=1}^N \sum_{j=1}^m \rho_j \|X_i - Y^j_{\sigma_{Yj}(i)}\|^2$$

We observe that the minimization problem is separable across the X_i, it then remains to solve the N minimization problems, indexed by $i = 1, ..., N$:

$$\arg\min_{X_i} \sum_{j=1}^m \rho_j \|X_i - Y^j_{\sigma_{Yj}(i)}\|^2$$

Which can be seen as a classical barycenter in $\mathbb{R}$, and the solution of this minimization problem is exactly the formula we wanted to prove.

Remark: we make the same assumption as in the article: the X_i are in increasing order. So the separation is not immediate, since there is this constraint, but it is not important since in the end the optimal values are indeed increasing by definition of the σ_{Yj}.

The computation can be done in $O(Nlog(N))$ (if we consider that m if fixed and a parameter of our problem). Then combined with the sliced Wasserstein distance, we can define a sliced version of the Wasserstein barycenter, much more tractable:

$$\tilde{\mathrm{Bar}}\left((\rho_j, Y^j)_{j\in[\![1,m]\!]}\right) \in \arg\min_X \tilde{E}(X), \qquad \tilde{E}(X) = \sum_{j=1}^m \rho_j\, \tilde{W}(X, Y^j)^2$$

3 Optimization Methodology

3.1 Selection and Analysis of the Second-Order Newton's Method

The functional optimization under consideration is amenable to a broad class of iterative methods. Nevertheless, consistent with the reference article, the Second-Order Newton's Method was adopted.

Theorem (Hypotheses for Local Newton Convergence): Let $f : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable ($f \in C^2$), and let x^* be a point satisfying $\nabla f(x^*) = 0$ and the Hessian matrix $\nabla^2 f(x^*)$ is non-singular. Then, Newton's method, $x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$, exhibits local and quadratic convergence to x^*.

Key Limitation: The primary constraint of the standard Newton's method is its susceptibility to divergence if the local hypotheses of the theorem are not satisfied. An common adaptation, the Damped Newton Method, mitigates this but introduces an additional computational cost associated with sorting procedures(necessary to calculate the distance on our metric , during the linear search performed by the Damped method). This overhead corresponds precisely to the computational *bottleneck* identified during empirical testing, justifying the retention of the standard method due to its satisfactory practical performance.

3.2 Approximation Strategy and Computational Efficiency

To circumvent the $O(n^3)$ computational cost of calculating and inverting the Hessian at each iteration, an approximation was implemented. This adjustment places the method within the class of Quasi-Newton Methods by shaping the Hessian favorably.

Core Idea: The approximation avoids the explicit inversion of a dense Hessian matrix by replacing it with a scalar multiple of the identity. This substantially reduces the cost of the Newton update, although the overall iteration still requires projections, sorting, and aggregation over the sampled directions.

Disadvantage: The trade-off for this efficiency gain is the requirement for extensive angular sampling to ensure convergence in law

(statistical convergence) of the solution. This sampling strategy was executed in practice and yielded high-quality results.

3.3 Stochastic Newton's Method

In this work, we employ a second-order optimization technique known as Newton's Method. It is important to distinguish this from the Stochastic Gradient Descent (SGD) algorithms often cited in related literature. While both are iterative, Newton's method utilizes curvature information (the Hessian), making the classification as standard SGD mathematically imprecise, moreover descent methods have certain properties,for each iteration that does not ,necessarily, belong to the Newton's Method;

Algorithm 1 Newton's Method

Require: Function $f : \mathbb{R}^d \to \mathbb{R}$, twice differentiable.
1: **Initialize:** $\mathbf{x}_0 \in \mathbb{R}^d$
2: **Iterate for** $k = 0, 1, 2, \dots$ **:**
3: Solve the linear system for the search direction $\mathbf{d}_k$:
4: $\nabla^2 f(\mathbf{x}_k)\mathbf{d}_k = \nabla f(\mathbf{x}_k)$
5: Update parameter:
6: $\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{d}_k$
7: **Stop** if convergence criteria met.
8: **return** Final point $\mathbf{x}_{end}$.

3.4 Analysis of the Method

3.4.1 Advantages. It is well established that Newton's method exhibits quadratic convergence close to the solution, particularly for convex functions.

- **Convergence Rate:** To achieve an accuracy of $\epsilon \approx 10^{-20}$, the method typically requires only $5 - 6$ iterations, assuming the starting point is within the basin of attraction (as demonstrated in the Appendix for the CONVEX case).

3.4.2 Drawbacks. Despite its theoretical speed, the practical application involves significant challenges:

- **Computational Complexity:** The algorithm requires solving a linear system involving the Hessian matrix. The worst-case complexity for solving a system of dimension n is $O(n^3)$. In the context of high-dimensional image processing, where n is large, this operation is computationally prohibitive. Generally, for optimization problems where $n > 10^3$, exact Newton steps are often infeasible. This is typically addressed using Quasi-Newton methods (which approximate the Hessian) or, as we propose, by exploiting the specific structure of our Hessian to accelerate calculation.
- **Lack of Global Descent Guarantee:** Unlike gradient descent, the standard Newton method is not guaranteed to be a descent method (i.e., $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$), even in convex scenarios, unless the Hessian is positive definite. Furthermore, it guarantees only local convergence. While techniques such as Damped Newton's Method can mitigate divergence through line search, we propose an alternative approach derived directly from the statistical properties of our Hessian.

3.4.3 Proposed Solution: Statistical Analysis of the Hessian. To overcome the computational bottleneck, we analyze the structure of the Hessian matrix used in our problem formulation.

4 Derivation of the Hessian Expectation

Let the random vector $\theta \in \mathbb{R}^D$ be uniformly distributed on the surface of a D-dimensional hypersphere of radius R, denoted as $\theta \sim \mathcal{U}(S^{D-1}(R))$.

4.1 Probability Density Function (PDF)

The distribution is supported entirely on the manifold defined by $\|\theta\| = R$. With respect to the standard Lebesgue measure on $\mathbb{R}^D$, the Probability Density Function $f(\theta)$ is a generalized function involving the Dirac delta distribution:

$$f(\theta) = \frac{1}{A_{D-1}(R)} \delta(\|\theta\| - R) \tag{1}$$

where $\delta(\cdot)$ is the Dirac delta function and $A_{D-1}(R)$ is the surface area of the sphere:

$$A_{D-1}(R) = \frac{2\pi^{D/2}}{\Gamma(D/2)} R^{D-1} \tag{2}$$

We seek the value of the expectation matrix $\Sigma = \mathbb{E}[\theta\theta^T]$.

4.2 Step-by-Step Derivation

Step 1: Symmetry Argument (Off-Diagonal Terms). Consider an off-diagonal element $(\theta\theta^T)_{ij} = \theta_i\theta_j$ where $i \neq j$. The domain $S^{D-1}(R)$ is symmetric with respect to the reflection transformation $T_i(\theta)$ which negates the i-th component: $T_i(\theta_1, \dots, \theta_i, \dots) = (\theta_1, \dots, -\theta_i, \dots)$. Since $f(\theta)$ depends only on the norm $\|\theta\|$, it is invariant under this reflection ($f(T_i(\theta)) = f(\theta)$). However, the integrand $g(\theta) = \theta_i\theta_j$ is an odd function with respect to θ_i:

$$g(T_i(\theta)) = (-\theta_i)\theta_j = -g(\theta)$$

Theorem (Integration of Odd Functions): The integral of an odd function over a symmetric domain is zero.

$$\mathbb{E}[\theta_i\theta_j] = \int_{\mathbb{R}^D} \theta_i\theta_j f(\theta)\, d\theta = 0 \quad \forall i \neq j \tag{3}$$

Thus, Σ is a diagonal matrix.

Step 2: Isotropy Argument (Diagonal Terms). The uniform distribution on a sphere is **isotropic** (rotationally invariant). For any rotation matrix $Q \in O(D)$, the distribution of $Q\theta$ is identical to θ. This implies there is no preferred direction, so the variance along any canonical axis must be identical:

$$\mathbb{E}[\theta_1^2] = \mathbb{E}[\theta_2^2] = \cdots = \mathbb{E}[\theta_D^2] = \sigma^2 \tag{4}$$

Consequently, the covariance matrix is a scalar multiple of the identity matrix:

$$\mathbb{E}[\theta\theta^T] = \sigma^2 I_D \tag{5}$$

Step 3: Trace Method. To determine the scalar constant σ^2, we employ the trace operator. Using the linearity of the trace and expectation:

$$\mathrm{Tr}(\mathbb{E}[\theta\theta^T]) = \mathbb{E}[\mathrm{Tr}(\theta\theta^T)] \tag{6}$$

Using the cyclic property of the trace ($\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$):

$$\mathbb{E}[\mathrm{Tr}(\theta\theta^T)] = \mathbb{E}[\mathrm{Tr}(\theta^T\theta)] = \mathbb{E}[\|\theta\|^2] \tag{7}$$

Since θ is constrained to the surface of the sphere, $\|\theta\|^2 = R^2$ is deterministic. Thus:

$$\mathrm{Tr}(\mathbb{E}[\theta\theta^T]) = R^2 \tag{8}$$

Alternatively, calculating the trace from the result in Step 2:

$$\mathrm{Tr}(\sigma^2 I_D) = \sum_{i=1}^{D} \sigma^2 = D\sigma^2 \tag{9}$$

Step 4: Solving for σ^2. Equating the two expressions for the trace:

$$D\sigma^2 = R^2 \implies \sigma^2 = \frac{R^2}{D} \tag{10}$$

Substituting σ^2 back into the matrix form:

$$\mathbb{E}[\theta\theta^T] = \frac{1}{D} I_D \text{ , in our case R=1;} \tag{11}$$

4.3 Showing it's not a Descent method:

In order to prove that it's a descent method we need to be able to answer one question , along all the iterations is it true that:

$$f(X_{k+1}) < f(X_k)$$

But in our case we have this Taylor Series:

$$f(x_k + \eta d_k) = f(x_k) + \eta f(x_k)^T d_k + O(\eta^2)$$

$$f(x_{k+1}) = f(x_k) - \eta f(x_k)^T H_k f(x_k) + O(\eta^2)$$

What is not necessarily true with our step size that is always equal to 1; But at the same type at least we have the first order term is always negative, since the Hessian has positive eigenvalues(asymptotically) , since it should be a factor of a Identity matrix, so the only way it would belong to a class of descent methods is to use a small step η ;

4.4 Explicite Newton's Method:

4.4.1 The Monte Carlo Approximation: The expected cost function $\tilde{E}(X)$ is approximated by a discrete sum over sampled directions $\theta_k \in \Omega$. Let Σ_N be the set of all permutations of N elements.

The initial cost function formulation is:

$$\tilde{E}(X) = \sum_{j\in J} \rho_j \tilde{W}(X, Y^j) = \sum_{j\in J} \rho_j \int_{\theta\in S^{D-1}(1)} W(X_\theta, Y^j_\theta)^2 d\theta$$

Expanding the projected cost $W(\cdot,\cdot)$:

$$\tilde{E}(X) = \sum_{j\in J} \rho_j \int_{\theta\in S^{D-1}(1)} \min_{\sigma_\theta \in \Sigma_N} \sum_{i\in I} |\langle X_i - Y^j_{\sigma_\theta(i)}, \theta\rangle|^2 d\theta$$

The Monte Carlo approximation uses samples $\theta_k \sim \boldsymbol{\mathcal{U}}(S^{D-1}(\mathbb{R}))$:

$$\tilde{E}(X) \approx \sum_{j\in J} \rho_j \sum_{\theta_k\in\Omega} \min_{\sigma_{\theta_k} \in \Sigma_N} \sum_{i\in I} |\langle X_i - Y^j_{\sigma_{\theta_k}(i)}, \theta_k\rangle|^2$$

4.4.2 The Gradient: Let $\sigma^*_{\theta_k}$ be the permutation that achieves the minimum at θ_k.

By linearity and applying the Envelope Theorem, we can differentiate the minimum-value function $\tilde{W}(X, Y^j, \theta_k)$ by evaluating the gradient with respect to X at the fixed optimal permutation $\sigma^*_{\theta_k}$.

The partial gradient with respect to one component X_i is:

$$\nabla\tilde{E}(X)_i = \sum_{j\in J} \rho_j \sum_{\theta_k\in\Omega} \nabla_{X_i} |\langle X_i - Y^j_{\sigma^*_{\theta_k}(i)}, \theta_k\rangle|^2$$

Applying the chain rule $\nabla_x \langle x, v\rangle^2 = 2\langle x, v\rangle v$:

$$\nabla\tilde{E}(X)_i = 2\sum_{j\in J} \rho_j \sum_{\theta_k\in\Omega} \left(\langle X_i, \theta_k\rangle - \langle Y^j_{\sigma^*_{\theta_k}(i)}, \theta_k\rangle\right)\theta_k$$

Rearranging terms to isolate X_i and utilizing the identity $(\langle X_i, \theta_k\rangle)\theta_k = (\theta_k\theta_k^T)X_i$:

$$\nabla\tilde{E}(X)_i = 2\left[\left(\sum_{\theta_k\in\Omega} \theta_k\theta_k^T\right)X_i - \sum_{\theta_k\in\Omega}\sum_{j\in J} \rho_j \langle Y^j_{\sigma^*_{\theta_k}(i)}, \theta_k\rangle\theta_k\right]$$

4.4.3 The Hessian: The Hessian matrix $H \in \mathbb{R}^{d\times d}$ corresponds to the second derivative of the energy functional with respect to X. Differentiating the gradient expression derived above:

$$H = \frac{\partial}{\partial X_i}(\nabla\tilde{E}(X)_i) = \frac{\partial}{\partial X_i}\left(2\sum_{\theta_k\in\Omega} (\theta_k\theta_k^T)X_i\right)$$

The second term of the gradient vanishes as it depends only on Y. Thus, the Hessian is the scaled covariance matrix of the directions Ω:

$$H = 2\sum_{\theta_k\in\Omega} \theta_k\theta_k^T$$

4.4.4 The Final Formula: The Newton's method update rule is given by solving $\nabla^2 f(X^k)\mathbf{d}_k = \nabla f(X^k)$, where $x_{k+1} = x_k - \eta_k\mathbf{d}_k$ (In the classical newton's method $\eta_k = 1$),writing in naive form we have:

$$x_{k+1} = x_k - \eta_k[\nabla^2 f(X^k)]^{-1}\nabla f(X^k)$$

Making $f(x) = \tilde{E}(X)$ and writing $H = 2\sum_{\theta_k\in\Omega} \theta_k\theta_k^T$ we have the following :

$$x_{k+1} = x_k - \eta_k[H]^{-1}\nabla\tilde{E}(X)$$

Now, assuming that $H \approx \mathbb{E}[H]$, as developed previously, the empirical Hessian can be approximated by its expectation when a sufficiently large number of directions is sampled. This approximation motivates the following simplified update, although it does not by itself guarantee convergence of the iterative scheme:

$$x_{k+1} = x_k - \eta_k[E[H]]^{-1}\nabla\tilde{E}(X)$$

$$x_{k+1} = x_k - \eta_k D I_d \frac{\nabla\tilde{E}(X)}{2|\Omega|}$$

$$x_{k+1} = x_k - \eta_k \frac{D}{|\Omega|} \left[\left(\sum_{\theta_k \in \Omega} \theta_k \theta_k^T \right) X_i - \sum_{\theta_k \in \Omega} \sum_{j \in J} \rho_j \langle Y^j_{\sigma_{\theta_k}(i)}, \theta_k \rangle \theta_k \right]$$

We could go even further and propose the following, by the same approximation of the Hessian:

$$x_{k+1} = x_k - \eta_k \frac{D}{|\Omega|} \left[\left(\frac{1}{D} I_D X_i - \sum_{\theta_k \in \Omega} \sum_{j \in J} \rho_j \langle Y^j_{\sigma_{\theta_k}(i)}, \theta_k \rangle \theta_k \right]$$

Having the final step as:

$$x_{k+1} = x_k - \frac{\eta_k}{|\Omega|} \left[\left(X_i - D \sum_{\theta_k \in \Omega} \sum_{j \in J} \rho_j \langle Y^j_{\sigma_{\theta_k}(i)}, \theta_k \rangle \theta_k \right]$$

4.5 Algorithm Complexity

The analysis of the theoretical complexity is centered on the cost per iteration k, primarily determined by the total dimension $d = ND$, where N is the number of points and D is the space dimensionality.

4.5.1 Comparison of Computational Complexity. The table below summarizes the complexity per iteration for the three methods considered, highlighting the dependence on the total dimension $d = ND$.

Table 1: Comparative Analysis of Complexity per Iteration

Meth.	Dominant Operation	Comp.
1. Stand Newt	Dense Matrix Inversion / System Solve	$O((\mathbf{ND})^3)$
2. Moore-Penrose ($H^\dagger$)	Full SVD for Pseudoinverse	$O((\mathbf{ND})^3)$
3. Prop Quasi-Newt.	Sorting projected samples	$O(\mathbf{\lvert\Omega\rvert \cdot \lvert J\rvert \cdot N \log N})$

4.5.2 Conclusion on Speedup. The Standard Newton and the numerically stable Moore-Penrose Pseudoinverse methods require solving or inverting a large Hessian system, which can become computationally prohibitive when the number of points N and the ambient dimension D are large.

The Proposed Quasi-Newton method reduces this cost by replacing the empirical Hessian H with its expectation $\mathbb{E}[H] \propto I_D$. This removes the need for an explicit dense Hessian inversion and considerably simplifies the Newton update.

- The expensive Hessian inversion is replaced by a simple rescaling involving the identity matrix.
- The main computational bottleneck therefore shifts toward the sliced optimal transport operations, which require projecting the samples and sorting the resulting one-dimensional values.

For $|\Omega|$ sampled directions and $|J|$ target distributions, the dominant sorting operations have an approximate complexity of

$$O\left(|\Omega| \cdot |J| \cdot N \log N\right)$$

per iteration.

Thus, the proposed approximation substantially reduces the computational cost associated with the Newton step, particularly in high-dimensional settings, while the remaining computational effort is mainly determined by the number of sampled directions and the sorting operations required by sliced optimal transport.

5 Texture Mixing

An illustrative application of *sliced Wasserstein barycenters* and *sliced Wasserstein projections*, as introduced in [7], is *texture mixing*, The goal is to synthesize a new texture from a collection of exemplars $(f_j)_j$ such that the output meaningfully integrates the colors and texture attributes of the exemplars In this section, we briefly review this problem setting and present results obtained from our implementation of their method.

5.1 Steerable Wavelets

We first review relevant concepts related to wavelets and steerable wavelets that were not explicitly detailed in [7].

A general framework for texture modeling relies on projecting an image onto a set of atoms $\{\psi_{\ell,n}\}_{\ell \in L, n}$ [7]. These atoms, also referred to as *basis functions* [9], are directional derivative operators of a desired order, localized both in space and in spatial-frequency.

Formally, let $f(k)$ denote a texture patch or signal defined over the index set K. The decomposition of $f(k)$ onto the basis $\{\psi_{\ell,n}\}$ is then written as

$$f(k) = \sum_{\ell \in L} \sum_{n} c_{\ell,n}(k)\, \psi_{\ell,n}, \tag{12}$$

where $c_{\ell,n}(k)$ are the coefficients representing the contribution of each atom $\psi_{\ell,n}$ at location k.

All atoms corresponding to a given ℓ share the same scale s and orientation θ. The *steerable pyramid* is specifically designed to decompose an image into subbands localized in both scale and orientation.

Scale (s): Scale captures the multi-scale decomposition of the image. The decomposition is recursive, breaking down the signal into components at different scales, which significantly improves computational efficiency. The radial portion of the filter, $B(\omega)$, determines the frequency tuning.

The radial decomposition uses a recursive pyramid algorithm involving downsampling and upsampling by a factor of 2 (denoted $2D$ and $2U$). This dyadic progression halves the resolution or frequency band at each level of recursion. The recursive, subsampled approach enables multi-scale analysis while maintaining computational efficiency [9].

Orientation (θ): Orientation refers to directional analysis. The angular portion of the filter, $A(\theta)$, determines orientation tuning, constrained by the *steerability property*. Formally, this property allows a rotated filter to be expressed as a linear combination of a

fixed set of basis filters:

$$\psi_\theta(x) = \sum_k c_k(\theta)\psi_{\theta_k}(x), \tag{13}$$

where $c_k(\theta)$ are interpolation coefficients and ψ_{θ_k} are basis filters at predefined orientations. The full decomposition is then achieved using basis functions related by dilations (scale changes) and rotations (orientation changes) [9].

Coarse-scale frame (Low-pass residual): The low-pass residual corresponds to the coarsest scale of the decomposition. It is repeatedly subsampled and processed recursively, capturing the low-frequency components that remain after all higher-frequency oriented details have been extracted [9].

High-frequency frame (Details): The decomposition also produces high-pass subbands. Oriented bandpass filters $B_i(\tilde{\omega})$ capture specific frequency bands and orientations. The recursive filtering subsystem separates the signal into low-pass (coarse scale) and high-pass (detail) components, enabling a rich multi-scale, multi-orientation representation [9].

5.2 Texture Modeling

Texture modeling aims to capture the statistical structure of a textured image. Following the framework in [7], the image is represented using a set of wavelet atoms $\{\psi_{\ell,n}\}_{\ell\in L,n}$, where each atom is indexed by $\ell = (s, \theta)$, specifying its dyadic scale 2^s and orientation θ, and by a spatial location n.

Atoms sharing the same ℓ correspond to the same geometric configuration but are translated to different spatial positions. This structure will later be used to perform operations such as sliced Wasserstein projections and barycenters.

This multiscale, multi-orientation representation is preferred over treating each pixel as a 3D RGB vector because texture is inherently geometric: it is characterized by repeated patterns, directional structure, and interactions across scales. Projections onto $\psi_{\ell,n}$ capture these geometric features, yielding a representation that is far more informative and invariant than raw color values. Moreover, separating information by scale and orientation reduces redundancy and mitigates the curse of dimensionality: instead of operating in a high-dimensional RGB neighborhood space, the synthesis algorithm works with compact, structured coefficients that encode the statistics most relevant for texture.

For our experiments, we follow [7], using 4 dyadic scales and 4 orientations, along with coarse (low-pass) and fine (high-frequency) frames, resulting in a total of 18 atom frames.

5.3 First-Order Statistical Mixing

For each exemplar texture f_j and each steerable-pyramid band ℓ, we collect the coefficient cloud

$$Y_{\ell,j} = \{\, \langle f_j, \psi_{\ell,n}\rangle \,\}_n \in \mathbb{R}^3,$$

and we similarly gather the RGB pixel cloud

$$Y_j = \{\, f_j(x) \,\}_x \in \mathbb{R}^3.$$

Given mixing weights (ρ_j), first-order texture mixing computes the sliced Wasserstein barycenter of these distributions:

$$Y_\ell = \widetilde{\mathrm{Bar}}(\rho_j, Y_{\ell,j}), \qquad Y = \widetilde{\mathrm{Bar}}(\rho_j, Y_j).$$

Starting from a white-noise initialization $f^{(0)}$, synthesis iteratively enforces these barycentric distributions. At iteration k, the steerable-pyramid coefficients are projected onto the target band-wise distributions:

$$c_{\ell,n}^{(k)} = \widetilde{\mathrm{Proj}}\big(\langle f^{(k)}, \psi_{\ell,n}\rangle; Y_\ell\big),$$

after which an intermediate image is reconstructed via the tight-frame inverse:

$$\tilde{f}^{(k)} = \sum_{\ell,n} c_{\ell,n}^{(k)}\, \psi_{\ell,n}.$$

A final pixelwise projection enforces the mixed RGB distribution:

$$f^{(k+1)}(x) = \widetilde{\mathrm{Proj}}\big(\tilde{f}^{(k)}(x); Y\big).$$

Algorithm 2 First-Order Texture Mixing

Require: Exemplars $\{f_j\}_{j\in J}$, weights $\{\rho_j\}$, pyramid parameters
For each exemplar f_j**:**
  Build steerable pyramid; collect coefficient clouds $Y_{\ell,j}$
  Collect pixel RGB cloud Y_j
For each band ℓ**:**
  Compute barycenter $Y_\ell = \widetilde{\mathrm{Bar}}(\{\rho_j, Y_{\ell,j}\})$
Compute pixel-value barycenter $Y = \widetilde{\mathrm{Bar}}(\{\rho_j, Y_j\})$
Initialize $f^{(0)}$ as white noise
for $k = 0, 1, 2, \ldots$ **do**
  Decompose $f^{(k)}$ to get coefficients $\{c_{\ell,n}\}$
  for each band ℓ **do**
    $\{c_{\ell,n}\} \leftarrow \widetilde{\mathrm{Proj}}_{Y_\ell}(\{c_{\ell,n}\})$
  end for
  Reconstruct $\tilde{f}^{(k)}$ from updated coefficients
  $f^{(k+1)}(x) \leftarrow \widetilde{\mathrm{Proj}}_Y(\tilde{f}^{(k)}(x))$ for all pixels x
end for
return final synthesized texture $f^{(K)}$

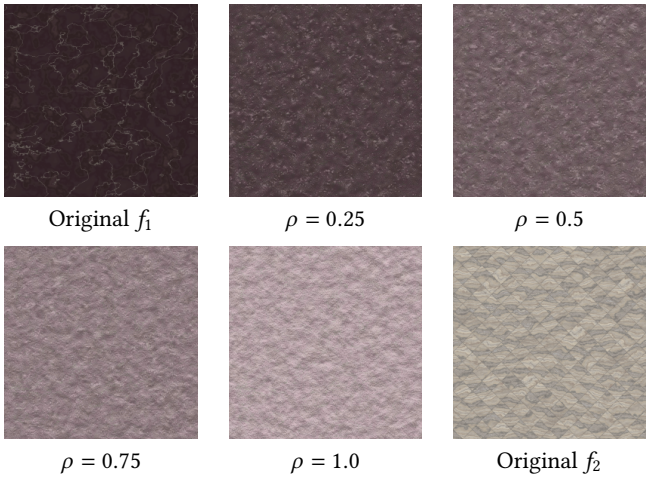


Figure 1: Texture interpolation: mixing between two textures using the first-order statistical model.

5.4 Higher-Order Statistical Mixing

First–order texture models operate independently on each wavelet coefficient. While this is effective for stochastic textures, it fails for structured materials whose appearance depends on spatially organized patterns. [7] In these cases, matching only marginal coefficient distributions is insufficient: textures such as brick, wood grain, or foliage rely on spatial correlations across nearby coefficients. To address this, they extended the sliced Wasserstein first–order model to include joint distributions of local coefficient neighborhoods. For each band ℓ and each exemplar j, they've assembled the joint neighborhood cloud

$$C_{\ell,j}^N = \left\{ Y_{\ell,j}^{\{m\}} \ : \ m \in N(n) \right\}_n \in \mathbb{R}^{3|N|},$$

where $N(n)$ denotes a fixed spatial neighborhood around each coefficient location n. In our experiments N is a 4×4 square neighborhood, giving a 3D vector for each offset and thus capturing local spatial dependencies.

Given mixing weights (ρ_j), we compute sliced Wasserstein barycenters of these joint distributions:

$$C_\ell^N = \widetilde{\mathrm{Bar}}\big(\rho_j,\ C_{\ell,j}^N\big) \subset \mathbb{R}^{3|N|}.$$

These provide the target higher–order statistics to impose during synthesis.

During iterative synthesis, steerable–pyramid coefficients of the current image are grouped into blocks matching the neighborhood pattern N and projected onto the barycentric joint distribution:

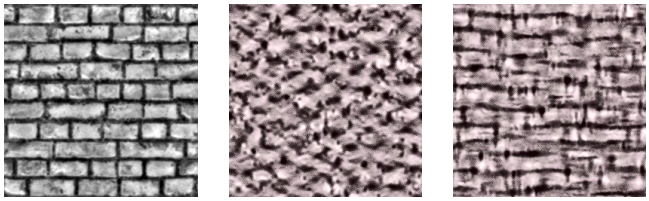
$$\hat{c}_{\ell,n}^{(k)} = \widetilde{\mathrm{Proj}}\big(\{\, c_{\ell,m}^{(k)} : m \in N(n)\,\}; C_\ell^N\big).$$

Reconstruction proceeds as in the first–order case, using these updated coefficients in the inverse tight-frame transform to produce $f^{(k+1)}$.

Algorithm 3 Higher-Order Texture Mixing

Require: Exemplars $\{f_j\}_{j\in J}$, weights $\{\rho_j\}$, neighborhood N

For each exemplar f_j:
  Build steerable pyramid; collect marginal clouds $Y_{\ell,j}$
  Form joint neighborhood clouds $C_{\ell,j}^N$
  Collect pixel RGB cloud Y_j
For each band ℓ:
  $Y_\ell \leftarrow \widetilde{\mathrm{Bar}}(\{\rho_j, Y_{\ell,j}\})$
  $C_\ell^N \leftarrow \widetilde{\mathrm{Bar}}(\{\rho_j, C_{\ell,j}^N\})$
Compute pixel barycenter $Y = \widetilde{\mathrm{Bar}}(\{\rho_j, Y_j\})$
Initialize $f^{(0)}$ as white noise
for $k = 0, 1, 2, \ldots$ **do**
  Decompose $f^{(k)}$ to obtain coefficients $\{c_{\ell,n}\}$
  for each band ℓ **do**
    Form neighborhoods $\{c_{\ell,m} : m \in N(n)\}_n$
    $\hat{c}_{\ell,n} \leftarrow \widetilde{\mathrm{Proj}}_{C_\ell^N}(\{c_{\ell,m} : m \in N(n)\})$
    $\hat{c}_{\ell,n} \leftarrow \widetilde{\mathrm{Proj}}_{Y_\ell}(\hat{c}_{\ell,n})$
  end for
  Reconstruct $\tilde{f}^{(k)}$ from $\{\hat{c}_{\ell,n}\}$
  $f^{(k+1)}(x) \leftarrow \widetilde{\mathrm{Proj}}_Y(\tilde{f}^{(k)}(x))$
end for
return final synthesized texture $f^{(K)}$



(a) Original texture (b) Heeger-Bergen (c) Their method

Figure 2: Comparison of original texture, Heeger-Bergen synthesis, and our method using joint-distributions.

6 Limitations & Extensions

The main theoretical limitation of sliced optimal transport is the loss of information, especially concerning the optimal mapping function, based on the optimal permutation. Moreover, the Wasserstein distance and its sliced variation do not define that same geometry, and as we saw, we only have a upper bound of the latter by a multiple of the former. In other words, it is not possible to control the absolute difference between the two distances. Also, we do not have results in [7] concerning the amount of sampling on the unit sphere required to obtain a satisfying approximation of the sliced Wasserstein distance, especially in high dimension.

These theoretical limitations naturally manifest in practice: while [7] provides an elegant computational framework, we observed several practical issues affecting reproducibility, stability, and efficiency.

Extension regarding a Quasi-Newton Method: We have proposed an approximate Newton method that replaces the empirical Hessian with its expectation, avoiding the explicit inversion of a large Hessian matrix and reducing the computational cost of the Newton update. To maintain the accuracy of this approximation, as explained, it is necessary to increase the number of samples, $|\Omega|$. While this necessity for increased sampling might initially be viewed as a computational burden, it strategically utilizes the inherent properties of our convergence analysis. By leveraging the disadvantage associated with Monte Carlo estimation—the variance reduction requires extensive sampling we can simultaneously improve the accuracy of the functional evaluation and potentially establish *asymptotic guarantees* related to the stability or convergence of the resulting iterative scheme. This integration allows the sampling constraint to serve as a mechanism for method refinement.

Divergence Analysis of the Optimization Algorithm. Throughout the empirical execution of this project, divergence of the iterative procedure was not observed. However, within a general theoretical framework, the standard Newton's method lacks assurances of global convergence or of possessing the properties of a descent algorithm, leading to divergent steps , even in the convex case. This potential instability can be circumvented by employing methods from the descent class of Newton algorithms, such as the Damped Newton Method.The inclusion of a robust line search in the Damped Newton Method necessarily increases the computational cost per iteration. Given that the iteration cost of the unadapted method is already high, the inclusion of damping would not change the complexity, but rather its constant multiplicative factor. It is imperative

to note that the observed stability is an empirical result of practical performance, not a conclusion derived from asymptotic theoretical guarantees.

Convergence Speed and Computational Constraint. The underlying algorithm relies fundamentally on the Sliced Wasserstein Distance. A critical step in the evaluation of this distance involves vector sorting. According to established modern methods for solving this class of problems, the complexity of the sorting operation adheres to the Comparison Sort Lower Bound Theorem, requiring $O(N \log N)$ time.

For large-scale data, such as an image of dimension $N = 1024 \times 1024 \times 3$, this $O(N \log N)$ complexity constitutes a significant computational bottleneck, potentially exceeding admissible time constraints. Furthermore, the inherent computational expense of Hessian inversion techniques (or the associated system solving) renders the standard Newton's method with complexity $O((ND)^3)$ largely inapplicable for applications involving larger images and requiring real-time performance.

Extension regarding Sorting Complexity: This section serves as a recommendation for future work to optimize the required sorting step. Given that the Sliced Wasserstein Distance relies on projections onto the unitary sphere, studying the final distribution of these projected values is crucial. If an underlying pattern, such as a near-uniform distribution, can be demonstrated for these projections, we could exploit this structure by employing algorithms like Bucket Sort to achieve a linear-time complexity of $O(N)$. Similarly, if the projected values could be treated as integers or approximated as fixed-point numbers, Radix Sort could also be utilized to attain the same complexity. However, in the general case, the existing complexity of $O(N \log N)$ obtained via comparison-based methods already represents an acceptable performance threshold for sorting.

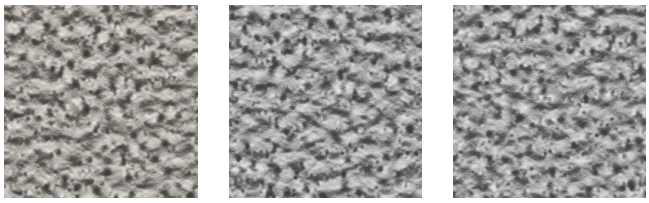
Incomplete Experimental Details: The original source specifies only a few hyperparameters, such as the number of iterations and learning rate, but omits critical details regarding barycenter and projection iterations computation iterations and which color channels are processed. This lack of specification makes exact reproduction of their results difficult. In our experiments, we had to empirically determine these parameters to achieve stable texture synthesis, which introduces potential variability when comparing outcomes.

Color Space Ambiguity: The authors describe the input textures as 3-channel vectors $f_j(x) \in \mathbb{R}^3$, but do not explicitly state the intended color space (e.g., RGB, L*a*b*, YUV).In our implementation, using RGB directly for texture mixing produced noticeable chromatic artifacts and hue shifts (Fig. 3, left). This is because all three channels are mixed jointly as 3D vectors with no separation between luminance and chrominance, leading to interpolation in RGB space that often introduces hue drift and partial desaturation due to its non-perceptual geometry.

To mitigate these issues, we experimented with perceptually uniform spaces:

- **LAB:** Mixing only the luminance channel L while preserving chrominance channels a and b reduced color aberrations (Fig. 3, middle).
- **YUV:** Similarly, mixing the luminance Y and preserving U, V channels produced visually coherent textures (Fig. 3, right).

This confirms that the choice of color space and which channels are mixed has a significant impact on synthesis quality, an aspect not addressed in the original work. All results in this report were therefore generated using the LAB configuration.



RGB LAB YUV

Figure 3: First-order statistical mixing results using different color spaces (RGB, L*A*B*, YUV).

7 Conclusion

We reviewed the 2011 sliced Wasserstein framework of Rabin et al. [8] and detailed its theory, barycenters, and use in texture synthesis. We implemented the sliced barycenter algorithm and reproduced its behavior on texture-mixing tasks. Although influential, this approach is now dated compared with modern texture-synthesis pipelines driven by diffusion models and neural systems. Recent state-of-the-art methods, such as CasTex [1], TexTailor [5], and multi-texture NCAs [3], demonstrate far greater fidelity, viewpoint consistency, and controllability. These contemporary methods rely on explicit PBR parameterizations, cascaded diffusion stages, resampling and preservation-loss strategies, or genome-conditioned NCA dynamics, enabling capabilities that classical optimal transport cannot match. Still, sliced Wasserstein distances remain relevant as a lightweight, interpretable tool and as a principled loss for feature-distribution matching. Our implementation aims to support reproducibility and provide context for how modern techniques evolved beyond this foundational work.

References

[1] Mishan Aliev, Dmitry Baranchuk, and Kirill Struminsky. 2025. CasTex: Cascaded Text-to-Texture Synthesis via Explicit Texture Maps and Physically-Based Shading. arXiv:2504.06856 [cs.CV] https://arxiv.org/abs/2504.06856

[2] Nicolas Bonnotte. 2013. *Unidimensional and Evolution Methods for Optimal Transportation.* Ph. D. Dissertation. Université Paris-Sud – Paris XI and Scuola Normale Superiore (Pisa, Italy).

[3] Mirela-Magdalena Catrina, Ioana Cristina Plajer, and Alexandra Băicoianu. 2025. Multi-texture synthesis through signal responsive neural cellular automata. *Scientific Reports* 15, 1 (Nov. 2025). doi:10.1038/s41598-025-23997-7

[4] Marco Cuturi. 2013. SINKHORN DISTANCES: LIGHTSPEED COMPUTATION OF OPTIMAL TRANSPORTATION DISTANCES. In *Advances in neural information processing systems.* 2292–2300.

[5] Suin Lee and Dae-Shik Kim. 2025. TexTailor: Customized Text-aligned Texturing via Effective Resampling. arXiv:2506.10612 [cs.CV] https://arxiv.org/abs/2506.10612

[6] Guillaume Carlier Martial, Agueh. 2013. Barycenters in the Wasserstein space. In *SIAM Journal on Mathematical Analysis.* 904–924.

[7] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. 2011. Wasserstein Barycenter and its Application to Texture Mixing. In *Scale Space and Variational Methods in Computer Vision (SSVM).* Israel, 435–446.

[8] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. 2011. A Wasserstein framework for texture synthesis. *International Journal of Computer Vision* (2011).

[9] E.P. Simoncelli and W.T. Freeman. 1995. The steerable pyramid: a flexible architecture for multi-scale derivative computation. In *Proceedings., International Conference on Image Processing*, Vol. 3. 444–447 vol.3. doi:10.1109/ICIP.1995.537667
[10] Kimia Nadjahi Umut Simsekli Roland Badeau Gustavo Rohde Soheil, Kolouri. 2019. Generalized Sliced Wasserstein Distances. In *Advances in neural information processing systems*.
[11] Jonathan Vacher Bruno Galerne Julien Rabin Thibaud, Briand. 1995. TheHeeger-BergenPyramid-BasedTextureSynthesis Algorithm. In *Siggraph '95. Annual Conference Series*. 229—-238.

A Convergence of Newton's Method on Convex Case

The Newton method, applied to unconstrained optimization problems for cost functions $f \in C^{2,2}_M(\mathbb{R}^n)$ (i.e., twice differentiable functions with the Hessian $\nabla^2 f(x)$ being Lipschitz continuous with constant M), and verifying the strong convexity property $\nabla^2 f(x) \succcurlyeq mI_n$, has the following local convergence certificate.

Theorem A.1 (Local Convergence Certificate of Newton's Method). *Let $f \in C^{2,2}_M(\mathbb{R}^n)$ such that $\nabla^2 f(x) \succcurlyeq mI_n$ for $m > 0$ (strong convexity). For any $R > 2m/M$, and starting with an initial condition x_0 such that $\|x_0 - x^*\| < \min\{1, 2m/M\}$, the distance to the optimizer x^* after t iterations is bounded by:*

$$\frac{M}{2m}\|x_t - x^*\| \leq \left(\frac{M}{2m}\|x_0 - x^*\|\right)^{2^t} \leq C^{2^t}$$

where $C < 1$.

Proof. Let $r_k := \|x_k - x^*\|$. The Newton update step is $x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1}\nabla f(x_k)$.

(1) Expression for the Error r_{k+1}:

$$r_{k+1} = \|x_{k+1} - x^*\| = \|x_k - x^* - (\nabla^2 f(x_k))^{-1}\nabla f(x_k)\|$$

(2) Gradient Substitution (Fundamental Theorem of Calculus for Vector Fields): The Fundamental Theorem of Calculus for the gradient states that:

$$\nabla f(x_k) = \nabla f(x^*) + \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*))(x_k - x^*)d\tau$$

Since x^* is the optimum, $\nabla f(x^*) = 0$. Substituting:

$$\nabla f(x_k) = \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*))(x_k - x^*)d\tau \tag{14}$$

(3) Update Manipulation (Addition and Subtraction of $\nabla^2 f(x_k)$ Term): The term $x_k - x^*$ can be written as $e_k = H_k^{-1}H_k e_k$, Where $e_k = (\nabla^2 f(x_k))^{-1}\nabla^2 f(x_k)(x_k - x^*)$, Substituting the gradient expression and factoring out H_k^{-1}:

$$r_{k+1} = \left\| H_k^{-1}\left[H_k e_k - \int_0^1 H_\tau e_k d\tau\right]\right\|$$

Combining terms inside the integral (using $H_k e_k = \int_0^1 H_k e_k d\tau$):

$$r_{k+1} = \left\| H_k^{-1}\left[\int_0^1 (H_k - H_\tau)\, d\tau\right] e_k\right\| \tag{15}$$

(4) Application of Properties (Matrix Norm and Lipschitz): We use the Cauchy–Schwarz inequality, the strong convexity property ($\nabla^2 f(x) \succcurlyeq mI_n \implies \|(\nabla^2 f(x))^{-1}\| \leq 1/m$), and the Lipschitz property of the Hessian ($f \in C^{2,2}_M(\mathbb{R}^n) \implies \|\nabla^2 f(x) - \nabla^2 f(y)\| \leq M\|x - y\|$). Let $r_k = \|x_k - x^*\|$.

Bounding the norm of the product (2) using submultiplicativity:

$$r_{k+1} \leq \|(\nabla^2 f(x_k))^{-1}\| \cdot r_k \cdot \int_0^1 \left\|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\right\| d\tau$$

Applying strong convexity and Lipschitz continuity:

$$r_{k+1} \leq \frac{r_k}{m}\int_0^1 M\|x_k - (x^* + \tau(x_k - x^*))\|d\tau$$

Simplifying the distance term $\|x_k - (x^* + \tau(x_k - x^*))\| = (1 - \tau)r_k$:

$$r_{k+1} \leq \frac{Mr_k^2}{m}\int_0^1 (1 - \tau)d\tau$$

Evaluating the integral $\int_0^1 (1 - \tau)d\tau = 1/2$:

$$r_{k+1} \leq \frac{M}{2m}r_k^2$$

Thus, the fundamental convergence inequality is:

$$r_{k+1} \leq \frac{M}{2m}r_k^2 \tag{16}$$

(5) Conclusion (Quadratic Convergence): If the initial condition $r_0 < 2m/M$ is satisfied, then $\frac{M}{2m}r_0 < 1$. Multiplying the fundamental inequality (3) by $\frac{M}{2m}$:

$$\frac{M}{2m}r_{k+1} \leq \left(\frac{M}{2m}r_k\right)^2$$

Applying recursively for t iterations:

$$\frac{M}{2m}r_t \leq \left(\frac{M}{2m}r_0\right)^{2^t}$$

Since $\frac{M}{2m}r_0 < 1$, convergence is established.

□

Analysis of Convergence

This result shows that the convergence is doubly exponential or quadratic. An accuracy of ε (i.e., $\|x_t - x^*\| \leq \varepsilon$) is achieved in a number of iterations t that grows only logarithmically with the logarithm of the inverse accuracy:

$$t \approx \log\log\frac{1}{\varepsilon}$$

In practice, high accuracy $\varepsilon \approx 10^{-20}$ is typically reached within 5 to 6 steps, and this rate is largely independent of the problem's condition number κ, unlike first-order methods.

Corollary A.2 (Case of Quadratic Functions). *For strictly quadratic functions, the Lipschitz constant is $M = 0$, so convergence is global and achieved in exactly one iteration.*