Differentiable Expectation-Maximisation and Applications to Gaussian Mixture Model Optimal Transport

Samuel Boïté*[1], Eloi Tanguy*[1], Julie Delon[1], Agnès Desolneux[2], and Rémi Flamary[3]

[1]Université Paris Cité, CNRS, MAP5, F-75006 Paris, France
[2]Centre Borelli, CNRS and ENS Paris-Saclay, F-91190 Gif-sur-Yvette, France
[3]CMAP, CNRS, Ecole Polytechnique, Institut Polytechnique de Paris

September 29, 2025

*: equal contribution.

Abstract

The Expectation-Maximisation (EM) algorithm is a central tool in statistics and machine learning, widely used for latent-variable models such as Gaussian Mixture Models (GMMs). Despite its ubiquity, EM is typically treated as a non-differentiable black box, preventing its integration into modern learning pipelines where end-to-end gradient propagation is essential. In this work, we present and compare several differentiation strategies for EM, from full automatic differentiation to approximate methods, assessing their accuracy and computational efficiency. As a key application, we leverage this differentiable EM in the computation of the Mixture Wasserstein distance MW_2 between GMMs, allowing MW_2 to be used as a differentiable loss in imaging and machine learning tasks. To complement our practical use of MW_2, we contribute a novel stability result which provides theoretical justification for the use of MW_2 with EM, and also introduce a novel unbalanced variant of MW_2. Numerical experiments on barycentre computation, colour and style transfer, image generation, and texture synthesis illustrate the versatility of the proposed approach in different settings.

Table of Contents

1 Introduction

The Expectation-Maximisation (EM) algorithm [DLR77] is a ubiquitous tool in statistics to fit mixture models on data [BE94; End03; VH10; Ng13]. Numerous variants of the EM algorithm have been proposed in the statistics and machine learning communities [DH97; Fri98; FH02; CJJ05; GTG07; VR08; CM09; Cap11; SCR12; GVS17; ZAC21; Kim22], and are the focus of various monographs [MK07; MP00]. From a theoretical standpoint, the EM algorithm is only known to converge under specific conditions [Wu83; Boy83; MK07; XHM16], and its behaviour is still not completely understood in full generality. In machine learning, Gaussian priors have been used for latent space representations [Ras03; KW14; RMW14; HJA20] with resounding success, while Gaussian Mixture Models (GMMs) are used more sparingly [NB06; VM19; Yua+20]. Beyond the difficulties of GMM estimation, the core challenge is that the EM algorithm is not easily integrated into end-to-end learning pipelines, as its differentiation with respect to the input data is not straightforward. Theoretical ties between the EM algorithm and an alternate minimisation of an energy involving Entropic Optimal Transport [Cut13] were recently highlighted [RW18; Men+20; Die+24; VL25]. To our knowledge, these connections do not provide insight into the differentiation of the EM algorithm. In this paper, we tackle the theory and practice of differentiating the EM algorithm, in the hope of sparking further research in this direction, beyond the various applications that we present.

One of the core contributions that sparked the Machine Learning wave is automatic differentiation [Wen64; Lin70; RHW86; GW08], which is a powerful method for complex optimisation problems. In the setting where the target objective is itself an optimisation problem, the problem is referred to as "bi-level". Numerous automatic differentiation methods for bi-level iterative minimisation were studied in [Gil92; Bec94; Sha+19; MO20; BPV22; BPV24]. Another approach to bi-level optimisation involving fixed-point problems is the *implicit method* [Lui+18; BKK19; LVD20; Bol+21; Blo+22; ER24].

In order to illustrate the potential of differentiable EM, we apply it to imaging tasks which rely on the comparison of GMMs with Optimal Transport (OT) [Mon81; Kan42]. Specifically, we leverage a variant of the Wasserstein distance between GMMs, called the Mixture-Wasserstein distance MW_2 [DD20], which compares GMMs by matching their components using a small-scale discrete OT problem that can be solved efficiently [PC19]. The Mixture-Wasserstein distance is one of many examples of recent advances in computational OT, where less costly surrogates of the Wasserstein distance such as regularised transport [Cut13] or sliced transport [Bon13] have seen a wide range of success in machine learning [Bon+11; Cou+17; Kol+19; KLA19; Fey+19]. Computing transport distances between empirical distributions remains challenging when the number of samples or the dimensionality of the space becomes too large, even though several solutions have been proposed in the literature [WB19; Gen+19; Chi+20].

The Mixture-Wasserstein distance has been used in texture synthesis [LDD23], for the evaluation

of generative neural networks [Luz+23], in quantum chemistry [Dal+23], and for domain adaptation [MMS24]. An efficient barycentre computation algorithm for this metric was also recently proposed in [TDG24]. When using MW_2 on discrete data, the space dimension d and the number of samples n only appear in two stages of the whole computation: the GMM inference on the data, and the computation of Bures distances between the covariance matrices of the GMM components. This makes the approach highly versatile and robust to dimensionality in practice. Nevertheless, a current limitation of the MW_2 distance is the inference of the GMMs, which our work renders differentiable with EM, thus allowing the use of MW_2 as a differentiable loss function between datasets in machine learning tasks.

Objectives. Our goal in this paper is to propose different ways to differentiate EM, allowing for applications in imaging or machine learning problems. As a focal application, we will pair differentiable EM with the Mixture-Wasserstein distance MW_2, which is not difficult to differentiate in practice (using classical results on the differentiation of discrete OT [PC+19], see also [TDD25, Proposition 4.3]). Differentiation of the Expectation-Maximisation algorithm is a more involved process. Surprisingly, while EM is well-known and extensively studied, the question of its differentiation seldom appears in the literature. To the best of our knowledge, the first work in this direction is [Kim22], which rewrites a Bayesian variant of EM which is related to the Optimal Transport Kernel Embedding [Mia+21]. In this paper, we propose several approaches for this differentiation, exact via automatic differentiation[1] or approximate, and compare their performance on different applications, ranging from toy examples to larger-scale machine learning tasks. Given that our contribution is methodological in nature, our goal is not to achieve state-of-the-art results on these applications, but rather to illustrate the versatility of the proposed approach.

Paper outline. In Section 2, we begin by recalling the EM algorithm and expressing it as a fixed point problem. We give precise mathematical meaning to the differentiation of a solution of the EM algorithm, and present numerous strategies to compute the differential of T steps of the method with respect to the input data. In Section 3, we provide a short reminder on the Mixture-Wasserstein distance MW_2 and show a stability result for the estimation of MW_2 between GMMs. We discuss practical difficulties in the differentiation of the MW_2 distance between GMMs estimated from data, and provide rationale for the importance of fixing EM weights. To circumvent the numerical difficulties incurred by weight optimisation and to ensure robustness to Gaussian outliers, we introduce an unbalanced variant of MW_2. In Section 4 we illustrate our methods on the flow of MW_2 composed with the EM algorithm, and perform a quantitative study on the convergence of the EM algorithm and the quality of the gradient approximations. In Section 5, we present several applications of differentiable EM: barycentre computation, colour transfer with inspiration from [Rab+12], style transfer in the spirit of [GEB15], image generation through MW_2-based Generative Adversarial Networks, and a novel texture synthesis method related to [GLR18; LDD23; Hou+23].

2 Differentiation of the Expectation-Maximisation Algorithm

2.1 Main Ideas of the EM Algorithm

The Expectation-Maximisation (EM) algorithm [DLR77] attempts to fit a GMM to a dataset $X = (x_1, \cdots, x_n) \in \mathbb{R}^{n\times d}$, with a fixed number of components K. We introduce the (hidden) quantities $Y \in [\![1, K]\!]^n$ which encode the component index of each sample x_i. The GMM parameters $\theta := (w, (m_k)_k, (\Sigma_k)_k)$ lie in the space $\Theta := \triangle_K \times (\mathbb{R}^d)^K \times (S_d^{++}(\mathbb{R}))^K$, where $\triangle_K$ is the K-simplex defined as

$$\triangle_K := \left\{ w \in (0,1)^K \mid \sum_{k=1}^{K} w_k = 1 \right\}, \tag{1}$$

[1] barring numerical approximation, which in certain cases may cause substantial errors.

and $S_d^{++}(\mathbb{R})$ is the set of symmetric positive definite matrices. We will denote by $\mu(\theta)$ the GMM probability measure of parameters θ. Given $X \in \mathbb{R}^{n\times d}$ and $Y \in [\![1, K]\!]^n$, the complete likelihood and its logarithm are respectively

$$L_\theta(X,Y) = \prod_{i=1}^{n}\prod_{k=1}^{K} \left(w_k g_{m_k,\Sigma_k}(x_i)\right)^{\mathbb{1}(Y_i=k)}, \quad \ell_\theta(X,Y) = \sum_{i=1}^{n}\sum_{k=1}^{K} \mathbb{1}(Y_i=k)\log\left(w_k g_{m_k,\Sigma_k}(x_i)\right), \tag{2}$$

where g_{m_k,Σ_k} is the Gaussian density with mean m_k and covariance Σ_k (recalled in Eq. (24)). Note that ℓ_θ cannot be optimised in θ directly since we do not know the hidden variables Y. The EM algorithm [DLR77] maximises the log-likelihood by iterating two steps over θ_t, first computing the "responsibilities" $\gamma_{ik}(\theta_t)$ which are the posterior probabilities of the hidden quantities Y_i given the data X and the current parameters $\theta_t = (w^{(t)}, (m_k^{(t)})_k, (\Sigma_k^{(t)})_k) \in \Theta$:

$$\gamma_{ik}(\theta_t) = \mathbb{P}_{(\mathbf{X},\mathbf{Y})\sim\mu(\theta_t)^{\otimes n}}\left[\mathbf{Y}_i = k | \mathbf{X} = X\right] = \left[w_k^{(t)} g_{m_k^{(t)},\Sigma_k^{(t)}}(x_i)\right] \Big/ \left[\sum_{\ell=1}^{K} w_\ell^{(t)} g_{m_\ell^{(t)},\Sigma_\ell^{(t)}}(x_i)\right]. \tag{3}$$

The next iteration θ_{t+1} corresponds to a maximisation of $\ell_\theta(X,Y)$ with the unknown variables Y replaced by the posterior probabilities $\gamma_{ik}(\theta_t)$, which leads to the following closed-form expressions for the parameters:

$$w_k^{(t+1)} = \frac{1}{n}\sum_{i=1}^{n}\gamma_{ik}(\theta_t), \quad m_k^{(t+1)} = \frac{\sum_{i=1}^{n}\gamma_{ik}(\theta_t)x_i}{\sum_{j=1}^{n}\gamma_{jk}(\theta_t)}, \quad \Sigma_k^{(t+1)} = \frac{\sum_{i=1}^{n}\gamma_{ik}(\theta_t)(x_i - m_k^{(t+1)})(x_i - m_k^{(t+1)})^\top}{\sum_{j=1}^{n}\gamma_{jk}(\theta_t)}. \tag{4}$$

It is a standard result (see [Moo96]) that the log-likelihood $\ell_\theta(X) = \sum_{i=1}^{n}\log\left(\sum_{k=1}^{K} w_k g_{m_k,\Sigma_k}(x_i)\right)$ increases with respect to $\theta = (w, (m_k), (\Sigma_k))$ with each EM iteration. In practice, we shall see that it is sometimes preferable to tweak the standard EM algorithm by not updating the weights w and keeping the weights w_0 of the initialisation θ_0 (we refer to this as fixing the weights). We summarise the two algorithms in Algorithms 1 and 2, with the difference highlighted in red.

Algorithm 1: EM Algorithm

Input: $\theta_0 \in \Theta$, $X \in \mathbb{R}^{n\times d}$, T, K.

for $t \in [\![0, T-1]\!]$ **do**

  Expectation: Compute the responsibilities $\gamma(\theta_t)$ using Eq. (3);

  Maximisation: Update $\theta_{t+1} = (w^{(t+1)}, (m_k^{(t+1)})_k, (\Sigma_k^{(t+1)})_k)$ using Eq. (4);

Algorithm 2: Fixed-Weights EM

Input: $\theta_0 \in \Theta$, $X \in \mathbb{R}^{n\times d}$, T, K.

for $t \in [\![0, T-1]\!]$ **do**

  Expectation: Compute the responsibilities $\gamma(\theta_t)$ using Eq. (3);

  Maximisation: Update $\theta_{t+1} = (w_0, (m_k^{(t+1)})_k, (\Sigma_k^{(t+1)})_k)$ using Eq. (4);

For applications minimising a loss involving the output of the EM algorithm with respect to the data X, it is important to highlight that even in the fixed-weights version (Algorithm 2), the responsibilities γ evolve with the EM steps and with the updates of X. As a result, running the EM algorithm along with X updates is paramount, and it is not sufficient to keep an initial responsibility assignment γ.

2.2 Fixed-Point Formulation and Differentiability

The goal of this section is to express an EM step as a fixed-point operation. For this, a technical condition is required to ensure that the term $\Sigma_k^{(t+1)}$ is invertible (symmetry and non-negativity of the eigenvalues is immediate), which requires a slightly stronger condition than assuming that the points (x_i) span $\mathbb{R}^d$. Since the terms $\gamma_{i,k}$ are positive, we can write the condition as:

$$X = (x_1, \cdots, x_n) \in \mathcal{X} := \left\{(x_1, \cdots, x_n) \in (\mathbb{R}^d)^n \mid \forall y \in \mathbb{R}^d,\ \mathrm{Span}((x_i - y)_{i\in[\![1,n]\!]}) = \mathbb{R}^d\right\}. \tag{5}$$

For $X \in \mathcal{X}$ and $\theta_0 \in \Theta$, the next iteration θ_1 obtained using Eq. (4) satisfies $\theta_1 \in \Theta$. Note that if ν is an absolutely continuous probability measure on $\mathbb{R}^d$, and if $n \geq d+1$, then Eq. (5) is verified almost surely for $\mathbf{X} \sim \nu^{\otimes n}$. This condition can be seen as a weaker variant of being in a "standard configuration". It can be shown that $\mathcal{X}$ is an open subset of $(\mathbb{R}^d)^n \simeq \mathbb{R}^{n \times d}$.

We study the map $F : \Theta \times \mathcal{X} \longrightarrow \Theta$ that maps a parameter θ to the value of the next M-step using Eq. (4). For convenience, we also write $F_X := F(\cdot, X)$ and F_X^t the t-th iteration $F_X \circ \cdots \circ F_X$ for $t \in \mathbb{N}$. An optimal solution to the EM algorithm can be seen as a fixed point of F, i.e. $\theta^* = F_X(\theta^*)$. Numerically, one takes a final iterate θ_T of the iteration scheme

$$\forall t \in [\![0, T-1]\!],\ \theta_{t+1} = F(\theta_t, X),$$

with an arbitrary initialisation $\theta_0 \in \Theta$. Due to the definition of Θ as the set of parameters with weights $w_k \in (0,1)$ and *positive* definite matrices Σ_k, the map F is of class $\mathcal{C}^\infty$ (jointly in (θ, X)), by virtue of the explicit expressions Eq. (4).

We now aim to give meaning to the gradient with respect to the data of a "true" solution of the EM algorithm. To this end, we need to work under the assumption of convergence of EM to a non-degenerate fixed point:

Assumption 1. There exists $(\theta_0, X_0) \in \Theta \times \mathcal{X}$ such that $\theta^*(\theta_0, X_0) := \lim\limits_{t \longrightarrow +\infty} F_{X_0}^t(\theta_0)$ exists in Θ, with $\frac{\partial F}{\partial \theta}(\theta^*(\theta_0, X_0), X_0) - I$ invertible.

While convergence is typically observed in practice numerically, from a theoretical standpoint, it is a delicate matter. See [MK07, Chapter 3] for a reference on this field of research. We are now ready to formulate Proposition 2.1, which shows that the gradient of an EM solution with respect to the data is well-defined, using the implicit function theorem.

Proposition 2.1. Under Assumption 1, there exist open neighbourhoods $\Theta^*, \mathcal{X}_0$ such that $\theta^*(\theta_0, X_0) \in \Theta^* \subset \Theta$ and $X_0 \in \mathcal{X}_0 \subset \mathcal{X}$, where there exists $\theta^*(\theta_0, \cdot) \in \mathcal{C}^\infty(\mathcal{X}_0, \Theta^*)$ with:

$$\forall (\theta, X) \in \Theta^* \times \mathcal{X}_0,\ F(\theta, X) = \theta \Longleftrightarrow \theta = \theta^*(\theta_0, X). \tag{6}$$

Proof. Let $G := \left\{ \begin{array}{ccc} \Theta \times \mathcal{X} & \longrightarrow & \Theta \\ (\theta, X) & \longmapsto & F(\theta, X) - \theta \end{array} \right.$. Thanks to the regularity of F, G is of class $\mathcal{C}^\infty$. Assumption 1 implies that $G(\theta^*(X_0), X_0) = 0$, with $\frac{\partial G}{\partial \theta}(\theta^*(X_0), X_0)$ invertible.

By the implicit function theorem, there exist open neighbourhoods $\Theta^*, \mathcal{X}_0$ such that $\theta^*(\theta_0, X_0) \in \Theta^* \subset \Theta$ and $X_0 \in \mathcal{X}_0 \subset \mathcal{X}$, and there exists $g : \mathcal{X}_0 \longrightarrow \Theta^*$ of class $\mathcal{C}^\infty$ such that $g(X_0) = \theta^*(\theta_0, X_0)$, and:

$$\forall (\theta, X) \in \Theta^* \times \mathcal{X}_0,\ F(\theta, X) = \theta \Longleftrightarrow \theta = g(X).$$

For $X \in \mathcal{X}_0$, we define $\theta^*(\theta_0, X) := g(X)$. □

To alleviate notation, we will write simply $\theta^*(X) := \theta^*(\theta_0, X)$ and continue with Assumption 1 and the map θ^* from Proposition 2.1. For the sake of completeness and to pave the way for future theoretical study, we provide the explicit expressions of the partial differentials of F in Appendix A.4. Note that from a statistical viewpoint, the parameter θ^* is not a Maximum-Likelihood Estimator (which, in fact, does not exist for GMMs), it is only the output of the EM algorithm which is often a local maximum of the likelihood.

2.3 Gradient Computation Methods

In this section, we are concerned with practical implementation of the computation of the gradient of the EM algorithm with respect to the data $\partial_X[F_X^T(\theta_0)]$, given some initialisation $\theta_0 \in \Theta$ and number

of iterations $T \geq 1$. In addition to the automatic differentiation method, we present two alternative approximate strategies which rely only on the last parameter θ_T. These methods work under the assumption that $\theta_T(X) \approx \theta^*(X)$, where θ^* is defined in Proposition 2.1 and refers to the fixed point $\lim_{t \longrightarrow +\infty} F_X^t(\theta_0)$, assuming convergence.

Full Automatic Differentiation (AD). The most naïve approach consists in computing the gradient through all iterations using the backpropagation algorithm (for instance, using PyTorch's automatic differentiation [Pas+19]). In other words, the "full automatic gradient" corresponds to letting automatic differentiation compute $\partial_X[F_X^T(\theta_0)]$ directly, using an automatically differentiable implementation of the EM algorithm. For a large number of iterations T, AD can be considered as a natural baseline for computing the exact gradient, up to numerical precision. It is still an approximation, due to propagation of numerical errors, but is to the best of our knowledge the best available approximation of the true gradient, as discussed in Appendix A.3. Nevertheless, we use AD as a natural baseline method for comparisons. The AD method may be costly (both in time and memory) if the number of iterations T is very large.

Approximate Implicit Gradient (AI). Our goal is to approximate the gradient $\frac{\partial \theta^*}{\partial X}(X)$ at a fixed point θ^* of $F(\cdot, X)$. Thanks to the differentiability property of Proposition 2.1 and under Assumption 1, we can differentiate with respect to X using the chain rule:

$$\frac{\partial \theta^*}{\partial X}(X) = \frac{\partial}{\partial X}[F(\theta^*, X)] = \frac{\partial F}{\partial \theta}(\theta^*, X)\frac{\partial \theta^*}{\partial X}(X) + \frac{\partial F}{\partial X}(\theta^*, X). \tag{7}$$

We deduce the following equation on $\frac{\partial \theta^*}{\partial X}(X)$:

$$\left(I - \frac{\partial F}{\partial \theta}(\theta^*, X)\right)\frac{\partial \theta^*}{\partial X}(X) = \frac{\partial F}{\partial X}(\theta^*, X), \tag{8}$$

using Assumption 1 again, we can invert the matrix on the left hand-side, yielding

$$\frac{\partial \theta^*}{\partial X}(X) = \left(I - \frac{\partial F}{\partial \theta}(\theta^*, X)\right)^{-1}\frac{\partial F}{\partial X}(\theta^*, X). \tag{9}$$

We define the approximate implicit gradient by approximating $\theta^* \approx \theta_T$:

$$J_{\mathrm{AI}} := \left(I - \frac{\partial F}{\partial \theta}(\theta_T, X)\right)^{-1}\frac{\partial F}{\partial X}(\theta_T, X) \tag{10}$$

The implicit approximation is theoretically exact (barring numerical imprecision in the inversion in particular) when $\theta_T = \theta^*$, however it requires additional costly computations: first evaluate the differential $\partial_\theta F(\theta_T, X)$, and then solve a large linear system to compute $(I - \partial_\theta F(\theta^*, X))^{-1}\partial_X F(\theta^*, X)$.

One-Step Gradient Approximation (OS). The One-Step method (OS) studied in [BPV24] (within a particular framework of bi-level optimisation), works under the following condition:

> **Condition 1.** $\left\|\frac{\partial F}{\partial \theta}(\theta, X)\right\|_{\mathrm{op}} \leq \rho \ll 1$ for any θ, X.

In the case of the EM algorithm, Condition 1 is not verified, since the eigenvalues of the partial differential $\frac{\partial F}{\partial \theta}(\theta, X)$ are commonly larger than 1, even in the neighbourhood of a fixed point. Under Condition 1, the OS approximation further neglects the term $(I - \partial_\theta F(\theta^*, X))^{-1}$ in Eq. (10), yielding the following expression:

$$J_{\mathrm{OS}} := \frac{\partial F}{\partial X}(\theta_{T-1}, X) \approx \frac{\partial F}{\partial X}(\theta^*, X) \approx \left(I - \frac{\partial F}{\partial \theta}(\theta^*, X)\right)^{-1}\frac{\partial F}{\partial X}(\theta^*, X) = \frac{\partial \theta^*}{\partial X}(X). \tag{11}$$

In practice, the OS method corresponds to computing the gradient of the EM output $\theta_T(X) = F_X^T(\theta_0)$ with respect to the data X only through the last iteration $\theta_T(X) = F_X(\theta_{T-1})$, neglecting the dependence of the penultimate iteration θ_{T-1} on X. Using automatic differentiation (for example with PyTorch [Pas+19]), this is done conveniently by computing $\theta_{T-1} = F_X^{T-1}(\theta_0)$ without gradient computation (e.g. `with torch.no_grad()`), and performing the last step with gradient computation. Due to this computation method, the OS gradient is numerically inexpensive compared to the others, albeit at the expense of precision. See [BPV24] for a detailed presentation.

Method	Time	Memory
Full Automatic Differentiation (AD)	$\mathcal{O}(T(nKd^2 + Kd^3))$	$\mathcal{O}(TKd^2 + nd)$
Approximate Implicit Gradient (AI)	$\mathcal{O}(TnKd^2 + K^3d^6 + nK^2d^5)$	$\mathcal{O}(nK^2d^4)$
One-Step gradient (OS)	$\mathcal{O}(T(nKd^2 + Kd^3))$	$\mathcal{O}(Kd^2 + nd)$

Table 1: Complexities of the `backward` passes (gradient computations) for T EM iterations on n points in $\mathbb{R}^d$ with K components.

Time complexity and memory footprint. The complexities of the gradient computation approaches are summarised in Table 1. As a baseline, the time complexity of the `forward` pass (EM algorithm without gradients) is $\mathcal{O}(T(nKd^2+Kd^3))$, while its memory footprint is $\mathcal{O}(Kd^2+nd)$. The complexities are deduced from the differential expressions in Appendix A.4. The $\mathcal{O}(Kd^3)$ factor corresponds to inverting the K covariance matrices of size $d\times d$ during each E-step. The $\mathcal{O}(nKd^2)$ factor comes from the M-step parameter updates (weights, means, covariances) and the differentiation of these updates with respect to X or θ.

The Warm-Start Method for Iteration of Differentiable EM. In many practical applications, we are interested in minimising a certain loss function $\mathcal{L}$ applied to the output $\theta_T(X)$ of the EM algorithm[2]. In this case, after a (small) gradient descent step X_{t+1} computed from X_t, the output of the EM algorithm with data X_t will often be a good initialisation for the EM algorithm with data X_{t+1}. As a result, we suggest operating the EM algorithm with only one iteration and using the output of the previous step as an initialisation. This leads to the following algorithm, which we refer to as the Warm-Start EM Flow of a loss $\mathcal{L} : \Theta \longrightarrow \mathbb{R}$.

Algorithm 3: Warm-Start EM Flow

Input: $\theta_0 \in \Theta$, $X_0 \in \mathcal{X}$, $T_{\text{GD}} \in \mathbb{N}$, $\mathcal{L} : \Theta \longrightarrow \mathbb{R}$.

1 **for** $t \in [\![0, T_{\text{GD}} - 1]\!]$ **do**

2 $\quad \theta_{t+1} = F(\theta_t, X_t)$;

3 $\quad X_{t+1} = X_t - \eta_t \frac{\partial \mathcal{L}}{\partial \theta}(\theta_t) \frac{\partial F}{\partial X}(\theta_t, X_t)$;

The gradient step at line 3 means that we perform automatic differentiation of the expression $\mathcal{L}(F(\theta_t, X_t))$ with respect to X at X_t, seeing θ_t as a constant and storing the value $\theta_{t+1} = F(\theta_t, X_t)$ for the next iteration. In essence, the Warm-Start method corresponds to an online OS gradient, which does not suffer from the approximation error of the OS method (since only one step is performed), yet benefits from the low memory footprint of the OS method.

[2] for example $\mathcal{L}(\theta_T(X)) = \text{MW}_2^2(\mu(F(\theta_T, X)), \nu)$, where ν is a target GMM, see Section 3.

3 Gaussian Mixture Model Optimal Transport

3.1 Reminders on GMM-OT

This section summarises the main results from [DD20]. The quadratic Wasserstein distance between two probability measures μ_0 and μ_1 on $\mathbb{R}^d$ with finite second moments is defined by

$$\mathrm{W}_2^2(\mu_0,\mu_1) := \inf_{\pi\in\Pi(\mu_0,\mu_1)} \int_{\mathbb{R}^d\times\mathbb{R}^d} \|y_0-y_1\|_2^2 \mathrm{d}\pi(y_0,y_1), \tag{12}$$

where $\Pi(\mu_0,\mu_1)$ denotes the set of probability measures with finite second moments on $\mathbb{R}^d\times\mathbb{R}^d$ whose marginals are μ_0 and μ_1. A solution π^* to Eq. (12) is called an optimal transport plan between μ_0 and μ_1. This distance has been widely used over the past fifteen years for various applications in data science. Let GMM_d denote the set of probability measures that can be written as finite Gaussian Mixture Models (GMMs) on $\mathbb{R}^d$. Transport plans and barycentres between GMMs with respect to W_2 are generally not GMMs, which is a limitation when such representations are used for data analysis or generation. For this reason, the authors of [DD20] propose to modify the W_2 formulation by restricting the couplings to be GMMs on $\mathbb{R}^d\times\mathbb{R}^d$. More precisely, given $\mu_0,\mu_1\in\mathrm{GMM}_d$, one can define

$$\mathrm{MW}_2^2(\mu_0,\mu_1) := \inf_{\pi\in\Pi^{\mathrm{GMM}}(\mu_0,\mu_1)} \int_{\mathbb{R}^{2d}} \|y_0-y_1\|_2^2 \mathrm{d}\pi(y_0,y_1), \tag{13}$$

where $\Pi^{\mathrm{GMM}}(\mu_0,\mu_1)$ denotes the set of probability measures in GMM_{2d} with marginals μ_0 and μ_1. The problem is well-defined since this set contains the product measure $\mu_0\otimes\mu_1$. The authors show that MW_2 defines a distance between elements of GMM_d. Moreover, if $\mu_0=\sum_{k=1}^{K_0} w_k^{(0)}\mu_k^{(0)}$ and $\mu_1=\sum_{\ell=1}^{K_1} w_\ell^{(1)}\mu_\ell^{(1)}$, where $(w_k^{(0)})_k\in\triangle_{K_0}$ and $(w_\ell^{(1)})_\ell\in\triangle_{K_1}$, and $\mu_k^{(0)},\mu_\ell^{(1)}$ are Gaussian measures, then it can be shown ([DD20, Proposition 4]) that

$$\mathrm{MW}_2^2(\mu_0,\mu_1) = \min_{P\in\Pi(w_0,w_1)} \sum_{k,\ell} P_{kl}\mathrm{W}_2^2(\mu_k^{(0)},\mu_\ell^{(1)}), \tag{14}$$

where $\Pi(w_0,w_1)$ is the set of $K_0\times K_1$ matrices with non-negative entries and marginals w_0 and w_1:

$$\Pi(w_0,w_1) = \left\{ P\in\mathcal{M}_{K_0,K_1}(\mathbb{R}^+);\ \forall k,\sum_j P_{kj}=w_k^{(0)} \text{ and } \forall j,\ \sum_k P_{kj}=w_j^{(1)} \right\}.$$

This discrete formulation makes MW_2 very easy to compute in practice, even in high dimensions. Indeed, the W_2 distance between two Gaussian measures $\mu=\mathcal{N}(m,\Sigma)$ and $\tilde{\mu}=\mathcal{N}(\tilde{m},\tilde{\Sigma})$ admits a closed-form expression:

$$\mathrm{W}_2^2(\mu,\tilde{\mu}) = \|m-\tilde{m}\|_2^2 + \mathrm{tr}\left(\Sigma+\tilde{\Sigma}-2\left(\Sigma^{\frac{1}{2}}\tilde{\Sigma}\Sigma^{\frac{1}{2}}\right)^{\frac{1}{2}}\right), \tag{15}$$

where $M^{\frac{1}{2}}$ denotes the unique positive semidefinite square root of the positive semidefinite matrix M. If the parameters of the GMMs μ_0 and μ_1 are known, computing Eq. (14) amounts to evaluating $K_0\times K_1$ Wasserstein distances between Gaussians and solving a discrete transport problem of size $K_0\times K_1$. It is also possible to define barycentres for MW_2 [DD20; TDG24], which leads to a similar discrete formulation. Given point clouds, [DD20] suggest using EM to fit GMMs to the data, allowing comparison of the point clouds with EM-MW_2^2.

3.2 Stability of MW_2^2 With Respect to GMM Parameters

To study the stability of the MW_2 distance with respect to the GMM parameters, we leverage a discrete OT stability result from [TFD24]. To relate this problem to discrete OT stability, we see $\mathrm{MW}_2^2(\mu_0,\mu_1)$ as a particular discrete Kantorovich problem with cost matrix $C_{k_0,k_1} := \|m_{k_0}^{(0)}-m_{k_1}^{(1)}\|_2^2 + d_{\mathrm{BW}}^2(\Sigma_{k_0}^{(0)},\Sigma_{k_1}^{(1)})$, where we recall the expression of the Bures-Wasserstein distance on $S_d^+(\mathbb{R})$:

$$\forall\Sigma,\Sigma'\in S_d^+(\mathbb{R}),\ d_{\mathrm{BW}}(\Sigma,\Sigma') := \sqrt{\mathrm{Tr}(\Sigma+\Sigma'-2(\Sigma^{1/2}\Sigma'\Sigma^{1/2})^{1/2})}. \tag{16}$$

We show that if the GMM parameters are sufficiently close (thanks to EM convergence for instance), then the MW_2^2 costs will also be close thanks to the stability result from [TFD24]. While general sample complexity results for the EM algorithm are not available, assuming a certain rate of convergence towards the true parameters, this result shows that the precision obtained using EM translates into a precision on the MW_2^2 distance. This key observation is a first step towards guaranteeing the quality of MW_2^2 as a loss function with respect to the data. We formulate two results: first, Proposition 3.1 quantifies the decrease of $\mathrm{MW}_2^2(\hat{\mu}, \mu)$ when $\hat{\mu}$ is an estimator of μ; then, Proposition 3.2 quantifies the decrease of $|\mathrm{MW}_2^2(\hat{\mu}_0, \hat{\mu}_1) - \mathrm{MW}_2^2(\mu_0, \mu_1)|$ when $\hat{\mu}_i$ are estimators of μ_i for $i \in \{0, 1\}$. We defer the proofs to Appendix A.1.

Proposition 3.1. Consider GMM parameters $(\hat{w}, \hat{m}, \hat{\Sigma}) \in \triangle_K \times \mathbb{R}^{K\times d} \times S_d^{++}(\mathbb{R})^K$ of a GMM $\hat{\mu}$ as estimators of $(w, m, \Sigma) \in \triangle_K \times \mathbb{R}^{K\times d} \times S_d^{++}(\mathbb{R})^K$ which are parameters of a target GMM μ. Assume "convergence rates" on the parameter estimations for $k \in [\![1, K]\!]$:

$$\mathbb{E}\left[\|w - \hat{w}\|_1\right] \leq \rho_w, \ \mathbb{E}\left[\mathrm{W}_2^2(\mathcal{N}(\hat{m}_k, \hat{\Sigma}_k), \mathcal{N}(m_k, \Sigma_k))\right] \leq \rho_{\mathcal{N}},$$

then the following stability bound holds:

$$\mathbb{E}\left[\mathrm{MW}_2^2(\hat{\mu}, \mu)\right] \leq \rho_{\mathcal{N}} + \frac{\rho_w}{2} \max_{k,\ell} \mathbb{E}\left[\mathrm{W}_2^2(\mathcal{N}(\hat{m}_k, \hat{\Sigma}_k), \mathcal{N}(m_\ell, \Sigma_\ell))\right]. \tag{17}$$

Proposition 3.2. For $i \in \{0, 1\}$, consider GMM parameters $(\hat{w}_i, \hat{m}_i, \hat{\Sigma}_i) \in \triangle_{K_i} \times \mathbb{R}^{K_i\times d} \times S_d^{++}(\mathbb{R})^{K_i}$ of a GMM $\hat{\mu}_i$ as estimators of $(w_i, m_i, \Sigma_i) \in \triangle_{K_i} \times \mathbb{R}^{K_i\times d} \times S_d^{++}(\mathbb{R})^{K_i}$ which are parameters of a target GMM μ_i. Assume that the means and covariances are bounded, namely that there exists $R_m > 0$, $R_\Sigma > 0$ such that:

$$\forall i \in \{0, 1\}, \ \forall k \in [\![1, K_i]\!], \ \|m_k^{(i)}\|_2 \leq R_m, \ \|\hat{m}_k^{(i)}\|_2 \leq R_m, \ \sqrt{\mathrm{Tr}\Sigma_k^{(i)}} \leq R_\Sigma, \ \sqrt{\mathrm{Tr}\hat{\Sigma}_k^{(i)}} \leq R_\Sigma.$$

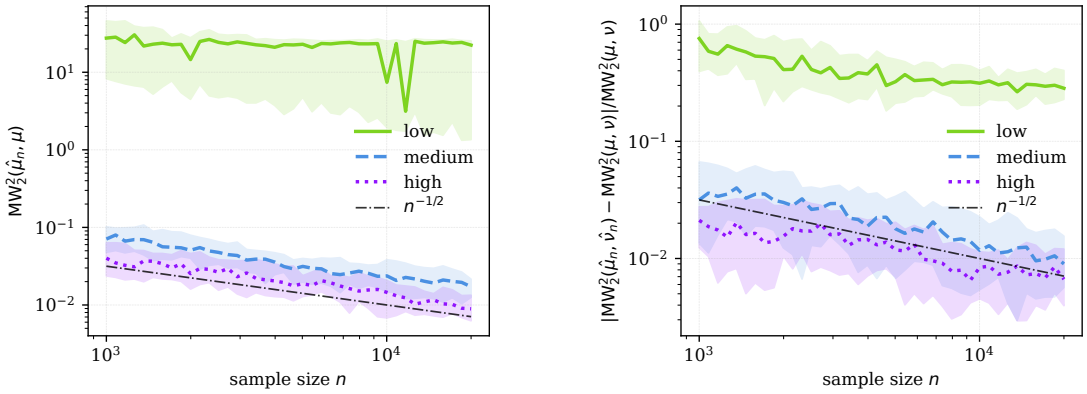
Further assume "convergence rates" on the parameter estimations $k \in [\![1, K_i]\!]$:

$$\forall i \in \{0, 1\}, \ \mathbb{E}\left[\|w_i - \hat{w}_i\|_1\right] \leq \rho_w, \ \mathbb{E}\left[\|m_k^{(i)} - \hat{m}_k^{(i)}\|_2\right] \leq \rho_m, \ \mathbb{E}\left[d_{\mathrm{BW}}\left(\Sigma_k^{(i)}, \hat{\Sigma}_k^{(i)}\right)\right] \leq \rho_\Sigma,$$

then the following stability bound holds:

$$\mathbb{E}\left[\left|\mathrm{MW}_2^2(\hat{\mu}_0, \hat{\mu}_1) - \mathrm{MW}_2^2(\mu_0, \mu_1)\right|\right] \leq 8R_m\rho_m + 8R_\Sigma\rho_\Sigma + 8(R_m^2 + R_\Sigma^2)\rho_w. \tag{18}$$

We complement the stability bounds of Proposition 3.1 and Proposition 3.2 with an empirical study. We fix $K = 3$, $d = 2$, vary the sample size n between 10^3 and $2 \cdot 10^4$, and for each n run 40 repetitions of EM with 200 iterations (with *k-means++* initialisation), noting the resulting estimates $\hat{\mu}_n$ and $\hat{\nu}_n$. Three regimes of component separation (low, medium, high) are considered, controlled by a scale parameter σ applied to the covariances. The mixtures μ, ν are fixed, and we provide a visualisation in Appendix A.2. For each n, we report the median error and interquartile range across repetitions. Fig. 1a shows $\mathrm{MW}_2^2(\hat{\mu}_n, \mu)$. With medium and high component separation, the error decreases regularly with n, while with low separation it remains flat, indicating that EM does not improve the estimation significantly in this regime. Fig. 1b shows the relative error $|\mathrm{MW}_2^2(\hat{\mu}_n, \hat{\nu}_n) - \mathrm{MW}_2^2(\mu, \nu)|/\mathrm{MW}_2^2(\mu, \nu)$. Medium and high separation again yield decay with n, while low separation plateaus. The results above translate parameter error rates of EM into rates for MW_2^2, but do not specify a universal value. In well-separated special cases, EM is known to achieve a rate of $\mathcal{O}(n^{-1/2})$ (see [KC20] for spherical covariances). When separation is weak, EM appears not to converge reliably, hence the MW_2^2 error does not decrease.



(a) One-sample: $\mathrm{MW}_2^2(\hat{\mu}_n, \mu)$ (b) Two-sample: relative error

Figure 1: Sample complexity of MW_2^2 for three specific GMMs of "low" to "high" mode separation. Curves show the median across 40 repetitions, with shaded interquartile ranges.

3.3 Minimisation of $\mathrm{EM} - \mathrm{MW}_2^2$: Local Optima and Weight Fixing

In practice, the minimisation of the energy $X \longmapsto \mathrm{MW}_2^2(F_X^T(\theta_0), \nu)$ for some initialisation $\theta_0 \in \Theta$ and a target GMM ν comes with numerous challenges. The first hurdle is the "outer" minimisation of the MW_2^2 cost. To illustrate this difficulty, we begin with a study of the simpler energy $\mu \longmapsto \mathrm{W}_2^2(\mu, \nu)$ for a fixed (discrete) measure $\nu \in \mathcal{P}_2(\mathbb{R}^d)$ with respect to the weights and support of the discrete measure $\mu = \sum_{i=1}^n a_i \delta_{x_i}$. This setting corresponds to the optimisation of MW_2^2 with known covariances, and thus highlights practical bottlenecks for the minimisation of the complete energy at stake, $\mathrm{EM}-\mathrm{MW}_2^2$. The objective of this section is to provide a theoretical rationale for fixing the mixture weights in practical applications, which is to say using Algorithm 2 instead of standard EM (Algorithm 1).

Local Minima of the Discrete 2-Wasserstein Distance. We focus on a particular instance of the minimisation of $\mu \longmapsto \mathrm{W}_2^2(\mu, \nu)$, and show the existence of a strict local minimum. We parametrise a discrete measure $\mu \in \mathcal{P}(\mathbb{R})$ with a support size of 3 as follows:

$$\forall \alpha \in [-\tfrac{1}{6}, \tfrac{1}{6}]^2,\ \forall \eta \in (-\tfrac{1}{2}, \tfrac{1}{2})^3,\ \mu_{\alpha,\eta} := (\tfrac{1}{6} + \alpha_1)\delta_{\eta_1} + (\tfrac{1}{6} + \alpha_2)\delta_{\eta_2} + (\tfrac{2}{3} - \alpha_1 - \alpha_2)\delta_{1+\eta_3},$$

and we fix a target measure $\nu := \frac{1}{3}(\delta_0 + \delta_{1-\varepsilon} + \delta_{1+\varepsilon})$ for a fixed $\varepsilon \in (0, \frac{1}{2})$. The energy to minimise is then:

$$\mathcal{E}_3 := \left\{ \begin{array}{ccc} [-\frac{1}{6}, \frac{1}{6}]^2 \times (-\frac{1}{2}, \frac{1}{2})^3 & \longrightarrow & \mathbb{R} \\ (\alpha, \eta) & \longmapsto & \mathrm{W}_2^2(\mu_{\alpha,\eta}, \nu) \end{array} \right. . \tag{19}$$

Obviously, the energy $(\alpha, \eta) \longmapsto \mathrm{W}_2^2(\mu_{\alpha,\eta}, \nu)$ has a global minimum with value 0 at all (α, η) such that $\mu_{\alpha,\eta} = \nu$. However, on the region with $(\alpha, \eta) \in [-\frac{1}{6}, \frac{1}{6}]^2 \times (-\frac{1}{2}, \frac{1}{2})^3$, we show in Appendix A.5.1 that the energy $\mathcal{E}_3$ has a minimum at $\alpha = 0_{\mathbb{R}^2}$ and $\eta = 0_{\mathbb{R}^3}$, with value $\mathcal{E}_3(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3}) > 0$. Note that for the case $n = 2$ we can show that there is a unique local minimum, see Appendix A.5.2.

Essential Stationary Points for the $\mathrm{EM}-\mathrm{MW}_2^2$ Loss. We have seen in the previous paragraph that optimising $\mu \longmapsto \mathrm{W}_2^2(\mu, \nu)$ with respect to the weights and support of μ can lead to local minima, which are not global minima. An additional difficulty arises when optimising the energy

$$\mathcal{E}_{\mathrm{EM}-\mathrm{MW}_2^2} := X \longmapsto \mathrm{MW}_2^2(\mu(F(\theta, X)), \nu), \tag{20}$$

with one iteration F of the EM algorithm, due to the update on the weights. The issue is that at some problematic points X to which the algorithm often converges in practice, the gradient $\partial_X \mathcal{E}_{\mathrm{EM}-\mathrm{MW}_2^2}(X)$ becomes extremely small, in particular when the covariances are highly localised. This leads in practice to undesirable convergence to an essential local minimum, as illustrated by an example in Section 4.3. We provide a theoretical explanation in a simple case in Appendix A.5.3. To

avoid these numerical issues, we propose fixing the weights of the GMMs in the EM steps by using Algorithm 2.

Our theoretical observations suggest that considering GMMs with uniform weights and using fixed-weights EM (Algorithm 2) is a more stable alternative to standard EM (Algorithm 1). In practice, we believe it is also preferable to keep the same number of components K between the compared GMMs for additional stability. Note that an identifiability issue remains with GMMs: if the means and covariances of two modes coincide, then the GMM can also be written by fusing both components and adding their weights. At this stage, it remains unclear whether this phenomenon has an impact on the optimisation behaviour (note that we never observed it in our experiments).

3.4 Unbalanced GMM-OT

Starting from the discrete formulation of the MW_2 distance, we relax the constraints on the transport plan π, penalising the marginal conditions instead of enforcing them in the optimisation problem. The resulting optimisation problem defines an unbalanced GMM-OT distance on the set $\mathrm{GMM}_d^+(\infty)$ of GMMs with positive weights on $\mathbb{R}^d$, as in [LMS18]. Given two Gaussian mixtures $\mu = \sum_{k_0=1}^{K_0} w_{k_0}^{(0)} g_{k_0}$, $\nu = \sum_{k_1=1}^{K_1} w_{k_1}^{(1)} g_{k_1} \in \mathrm{GMM}_d^+(\infty)$, and regularisation parameters $(\lambda_0, \lambda_1) \in (0, +\infty)^2$, the unbalanced GMM-OT cost is defined as:

$$\mathrm{UMW}_2^2(\mu, \nu; \lambda_0, \lambda_1) := \min_{\pi \in \mathbb{R}_+^{K_0 \times K_1}} \sum_{k_0, k_1} \pi_{k_0,k_1} \mathrm{W}_2^2(g_{k_0}, g_{k_1}) + \lambda_0 \,\mathrm{KL}(\pi \mathbf{1} | w_0) + \lambda_1 \,\mathrm{KL}(\pi^\top \mathbf{1} | w_1), \quad (21)$$

where we recall that for $a, b \in (0, +\infty)^K$, the Kullback-Leibler divergence is $\mathrm{KL}(a|b) := \sum_k a_k \log(\frac{a_k}{b_k})$. We have seen that MW_2 is a particular discrete Kantorovich problem, and likewise the unbalanced GMM-OT distance UMW_2 is simply an unbalanced discrete OT problem with a particular cost matrix.

Given the numerical challenges of optimising the weights in the balanced formulation (see the discussion in Section 3.3), we introduce this variant as a possibly more stable alternative. We suspect that the underlying geometry on the weights induced by unbalanced OT [LMS18; Chi+18b] is more amenable to optimisation. Furthermore, unbalanced OT has been shown by [Fat+21a] to be stable with respect to minibatch sampling, which is paramount for large-scale machine learning applications.

4 Illustrations and Quantitative Study of Gradient Methods

4.1 Practical Implementation

For practical implementation of the EM algorithm, some specific implementation strategies are required to ensure numerical stability, in particular when computing gradients. The first technique is applied in the E step, and consists in computing the responsibilities $\gamma_{ik}(\theta_t)$ in logarithmic space and using the so-called "log-sum-exp trick"[3] to compute the normalisation in Eq. (3). Furthermore, to stabilise the (differentiable) expression of the Gaussian density $g_{m,\Sigma}(x)$, we leverage the Cholesky decomposition of the covariance matrix Σ, which uniquely decomposes $\Sigma = LL^\top$ where $L \in \mathbb{R}^{d \times d}$ is a lower triangular matrix. In particular, the computation of the inverse is simplified by solving triangular systems, and the determinant of Σ is simply $\det \Sigma = (\prod_a L_{aa})^2$ (which we compute in logarithmic space as well).

Another important implementation aspect concerns a differentiable implementation of the matrix square root of symmetric positive semidefinite matrices. This is required in the computation of the Bures distance Eq. (16) for the MW_2 distance. Unfortunately, the naive implementation using the spectral decomposition suffers from numerical instability when eigenvalues are very close in value[4].

[3] for instance, see this blog post for an explanation of this well-known trick.

[4] as explained in the PyTorch documentation for `torch.linalg.eigh()`.

Leveraging an explicit formula for the gradient of the matrix square root (detailed in Appendix A.6), we circumvent these numerical issues by implementing our own differentiable square root function with an explicit gradient.

As is done in scikit-learn's implementation of the EM algorithm, we have an optional regularisation term $\varepsilon_r \geq 0$ for the covariance matrices Σ_k to ensure positive-definiteness. The idea is to replace the update $\Sigma_k^{(t+1)}$ with $\Sigma_k^{(t+1)} + \varepsilon_r I_d$ to enforce a minimum eigenvalue of ε_r. This regularisation was particularly crucial for numerical stability in higher-dimensional cases where the covariances were almost singular, which led to exploding gradients. In the larger-scale examples from Section 5, we chose a heuristic which sets $\varepsilon_r := 10^{-4} \times s_{\max}$, where $s_{\max}$ is the largest eigenvalue of the covariances of a GMM fitted on the target data.

4.2 Flow of EM $-$ MW_2^2 with Fixed Weights in 2D

In this section, we illustrate the use of differentiable EM for OT by numerically computing the flow (i.e. gradient descent) of the following energy:

$$\mathcal{E}_T := X \in \mathbb{R}^{n\times 2} \longmapsto \mathrm{MW}_2^2(\mu(F_X^T(\theta_0)), \nu), \tag{22}$$

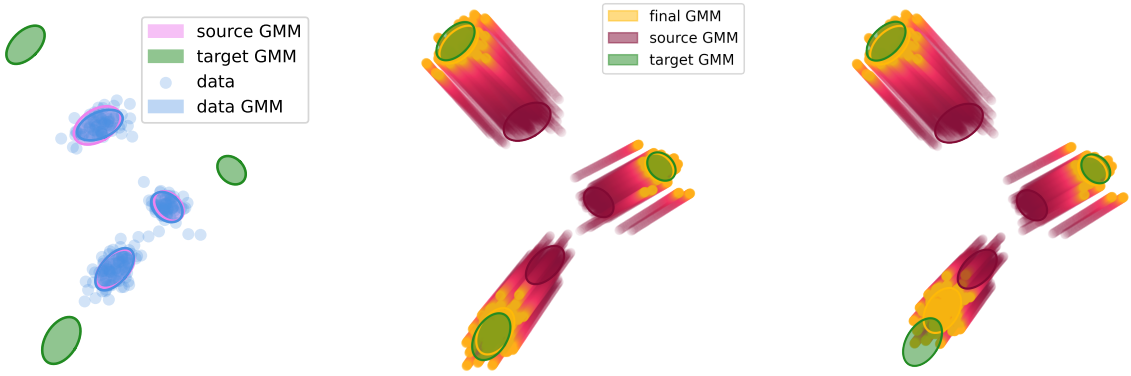
for a fixed target GMM ν, an initialisation θ_0 and a number of EM steps T. We use a variant of EM presented in Algorithm 2 that **fixes the mixture weights** in this experiment. We will compare three gradient computation methods to compute (or approximate) the gradient of $F_X^T(\theta_0)$ with respect to X, within the gradient descent of $\mathcal{E}_T$, performed using automatic differentiation. The setup is as follows: the initial dataset $X \in \mathbb{R}^{200\times 2}$ corresponds to samples of a GMM μ_0 with 3 components, and we want to displace this point cloud to match a target GMM ν with 3 components. We represent the setup in Fig. 2a and the flow for AD method in Fig. 2b.

The results for the AI method are both visually and quantitatively very close, however the experiment took six times longer to run for AI. We observe satisfactory convergence of the flow of $\mathcal{E}_T$ towards the target GMM ν. In many applications involving fixed EM weights (Algorithm 2), we observe that particles follow rectilinear trajectories, which is a similar behaviour to Wasserstein flows of W_2^2 (see [CNR25, Section 5.3]). We interpret this phenomenon as a consequence of the fixed weights, which translate to a Lagrangian viewpoint on the GMMs. In simple cases, the MW_2^2-optimal plans between the GMMs may not change during the flow, and thus the particles are moved along the induced (rectilinear) trajectories between each GMM component (see [DD20, Proposition 4]). In Fig. 2c, we show the flow with the OS method, which converges more slowly and to an unsatisfactory stationary point. This is due to the fact that OS requires a contraction assumption that is not verified for EM. OS was comparable in computation time to AD. We also experimented with the Warm-Start flow from Algorithm 3, which is a different minimisation method to minimise $\mathcal{E}_T$, yet yielded almost identical results to the AD, with a 40% lower computation time.

4.3 Flow of EM $-$ MW_2^2 in 2D: Discussion on Uniform Weights

We now consider a similar setting to Section 4.2, but without fixing the weights in the EM algorithm, i.e. using standard EM Algorithm 1. We compare two settings: the first with non-uniform weights $w_0 := (\frac{1}{5}, \frac{1}{5}, \frac{3}{5})$ for the initial GMM, and weights $w_1 := (\frac{1}{2}, \frac{3}{10}, \frac{1}{5})$ for the target GMM; and the second with uniform weights for both. In Fig. 3, we show the flow of $\mathcal{E}_T$ with the AD method. We observe in Fig. 3a that the flow for non-uniform weights converges to an unsatisfactory local minimum, with a final GMM weight of `[0.41039274, 0.2030173, 0.38658995]` instead of the target `[0.5, 0.3, 0.2]`, as shown on the simplex in Fig. 3b. In contrast, the flow for uniform weights presented in Fig. 3c converges to the target GMM and achieves a substantially lower energy, as reported in Fig. 3d. The weights stay close to uniform, with a final GMM weight of `[0.33314218, 0.30985782, 0.357]`.

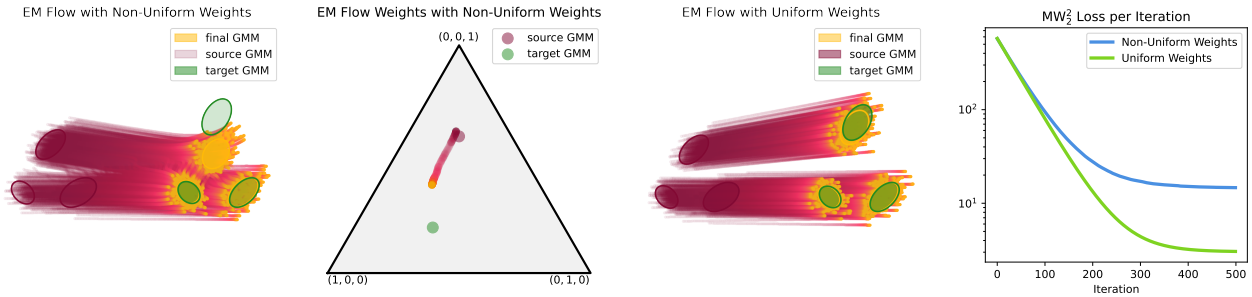
The optimisation failure in the non-uniform case is due to the essential stationary point problem illustrated in Section 3.3: intuitively, to change the weights of the current GMM, the particles need



(a) $\mathcal{E}_T$ flow setup. (b) AD method. (c) OS method.

Figure 2: Comparison of experimental setup and flows of $\mathcal{E}_T$ using different methods. The dark shades of purple correspond to earlier iterations, and the yellow shades to later iterations. (Warm-Start and AI are almost identical to AD)

to change components, but this is not possible if the components are too distant. While our framework encompasses the case of non-uniform weights, as illustrated theoretically and experimentally, it appears that the non-uniform weight setting is impractical. As a result, we recommend using uniform weights, in particular using the fixed-weights EM approach (Algorithm 2) for speed and stability.



(a) Particle flow (NU). (b) Weight evolution on the simplex $\triangle_3$ (NU). (c) Particle flow (U). (d) Energy evolutions.

Figure 3: Flow of $\mathcal{E}_T$ with the AD method and standard EM Algorithm 1. We compare two settings: one with non-uniform GMM weights (NU) and one with uniform weights (U).

4.4 Stochastic EM – MW_2^2 Flow with Fixed Weights

We consider a similar setting to Section 4.2 but introduce stochasticity in the flow at each step by performing EM only on a subsample of the optimised source point cloud and of the target point cloud. While we illustrate the technique on a toy example here, this "minibatch" stochastic gradient descent method is useful in practice when the dataset size is too large for simultaneous optimisation [Fat+20; Fat+21b; Fat+21a; Ton+24]. The same principle is applied to the image generation task in Appendix A.7.6. We observe in Fig. 4 that the general trajectory remains similar to the deterministic case. Notice that in this setting, the components are sufficiently close together to interact, yielding non-rectilinear trajectories when points are influenced by multiple components. This is amplified by the stochasticity of the method.

4.5 Quantitative Study of EM Convergence and Gradients

We study the impact of the number of points n, components K and EM iterations T on the convergence of EM iterations (to a fixed point of F), the local contractivity of F around the fixed point, and the gradient approximation methods introduced in Section 2.3. The experimental setting is as

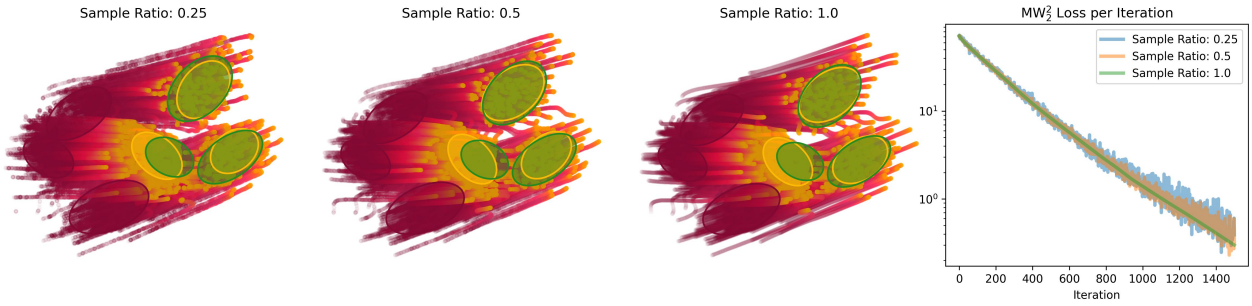


Figure 4: Stochastic Flow of $\mathcal{E}_T$ for Algorithm 2 with the full automatic differentiation method. We vary the sub-sampling ratio $r \in (0, 1]$, which corresponds to performing EM on only $[r \times n]$ random points from the current point cloud at each step.

follows: for each of the three parameters n, K, T separately (say n), we consider a range of values (say $n \in \{100, 200, 500, 1000, 1500, 2000\}$) with the others fixed. For each value of the parameter in this range, and for three different GMMs in $\mathrm{GMM}_d(K)$ with $d = 3$ (shown in Appendix A.2), we sample the data from $X \sim \mu_0^{\otimes n}$ 60 times, then run the EM algorithm for T iterations. We run the experiments on three different GMMs taken with random parameters (adding a small term $10^{-14} I_d$ to the covariances to avoid vanishing eigenvalues). Note that GMM# 1 is better conditioned than GMM# 2, and GMM# 3 is the worst-conditioned. We observe the mean squared error of the fixed point property $F(\theta_T, X) \approx \theta_T$ by evaluating $\frac{1}{p}\|\theta_T - F(\theta_T, X)\|_2^2$ with $p := K + Kd + Kd^2$, measuring the quality of convergence of the EM algorithm. To study the local contractivity of F, we compute the spectral norm $\|\partial_\theta F(\theta_T, X)\|_{\mathrm{op}}$: if this is close to 0, then locally the iterated function F_X has a tame behaviour and the OS method is expected to work well, while if it is close to or larger than 1, the local landscape is difficult and the OS method is expected to fail. Finally, we compare the OS and AI gradients (from Eqs. (10) and (11)) to the reference AD gradient by computing the relative MSEs $\frac{1}{p}\|J_{\mathrm{OS}} - J_{\mathrm{AD}}\|_2^2 / (\frac{1}{p}\|J_{\mathrm{AD}}\|_2^2)$ and $\frac{1}{p}\|J_{\mathrm{AI}} - J_{\mathrm{AD}}\|_2^2 / (\frac{1}{p}\|J_{\mathrm{AD}}\|_2^2)$, where J_{AD} is the AD gradient, which serves as a baseline (see Appendix A.3 for a discussion on this choice). Concerning the impact of the number of components K, we defer to Appendix A.7.1, since the findings are less conclusive.

Impact of the number of samples n. We begin by fixing $d = 3$, $K = 3$ and $T = 30$ and varying $n \in \{100, 200, 500, 1000, 1500, 2000\}$. The results are shown in row 1 of Fig. 5, and we observe that EM appears to converge to a fixed point for all n, albeit with a large variance in the MSE depending on the sampled GMMs. The spectral norm of the Jacobian is often close to 0.6 and has no clear trend with n, hence we expect the OS method to be a very coarse approximation of the true gradient. The quality of the OS gradient is relatively poor, and substantially worse than the AI gradient, whose median MSE is much smaller, but suffers from very high variance (in log space). Comparing GMMs shows that a precise EM convergence leads to high precision for the AI gradient.

Impact of the number of EM iterations T. Finally, we fix $n = 200, d = 3$ and $K = 3$, and vary $T \in \{1, 2, 5, 10, 15, 20, 30, 40\}$ in row 3 of Fig. 5. Reassuringly, increasing the number of iterations T leads to improved convergence of the EM algorithm to better-conditioned points. The convergence speed seems heavily dependent on the GMM, with an additional variance caused by the dataset sampling. In the favourable settings for larger T, the AI approximation substantially outperforms the OS approximation, but suffers from higher variance. Since the spectral norm of the Jacobian stabilises to values of the order of 0.5, the OS method plateaus at coarse MSEs, even for larger T.

5 Applications of Differentiable EM

In this section, we propose numerous larger-scale applications of the differentiation of the EM algorithm presented in Section 2. Our goal is to illustrate the versatility of the proposed approach, rather than to achieve state-of-the-art results on these applications. In Appendix A.7.6, we also present an

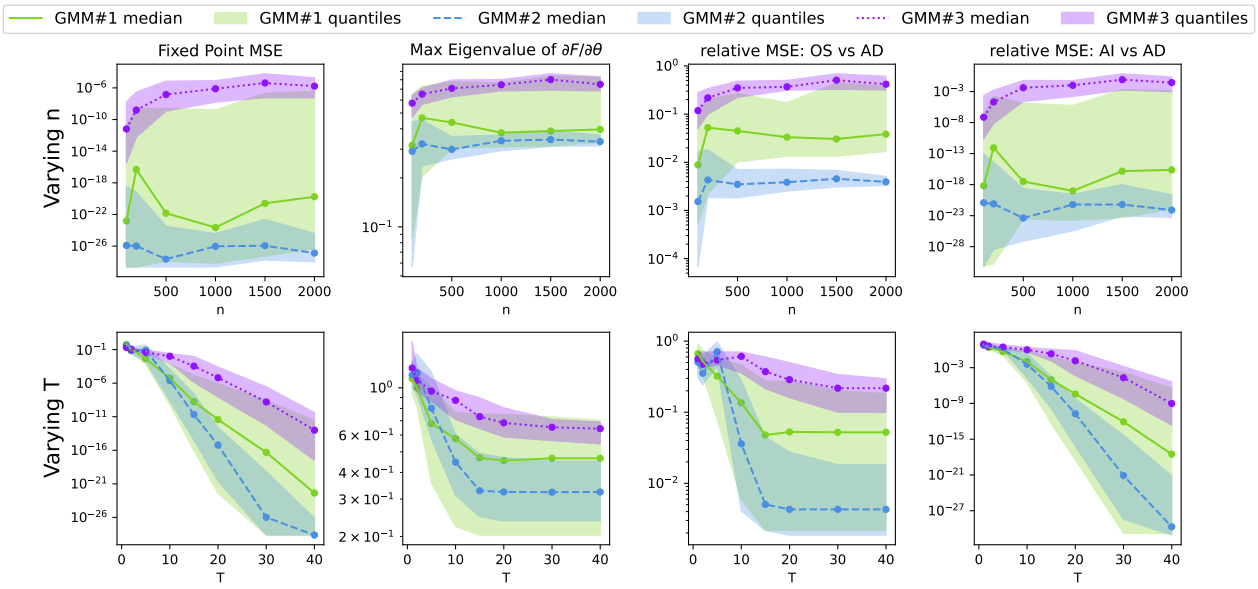


Figure 5: Varying the number of samples n and the number of iterations T, we study the convergence of EM, the local contractivity of F, and the MSEs of the OS and AI gradients against the AD gradient.

EM-MW_2^2-regularised generative model.

5.1 Barycentre Flow in 2D

Wasserstein barycentres [AC11] and their notoriously challenging computation [CD14; Álv+16; AB22; TDG24] are active fields of research. In this section, we illustrate the use of differentiable EM to flow a point cloud towards a barycentre of GMMs. Given M point clouds $Y_i \in \mathbb{R}^{n\times 2}$, our goal is to optimise a point cloud $X \in \mathbb{R}^{n\times 2}$, initialised as random normal noise, towards a barycentre (with uniform weights) of GMMs (ν_i) fitted from (Y_i). Specifically, we solve

$$\min_{X\in\mathbb{R}^{n\times 2}} \sum_{i=1}^{M} \mathrm{MW}_2^2\left(\mu(F_X^T(\theta_0)), \nu_i\right)$$

with respect to X. The GMMs ν_i are fitted beforehand, and $\mu(F_X^T(\theta_0))$ is the current EM estimate of the optimised cloud X. We illustrate the results in Fig. 6, where $M := 3$, $K := 2$ and $n := 500$. This method can be adapted to compute more general barycentres, as presented in Appendix A.7.2.

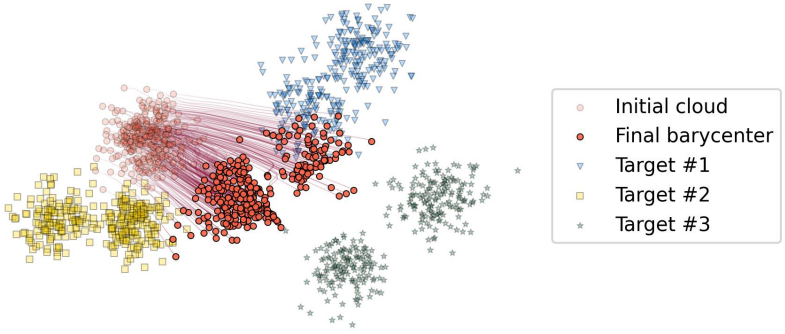


Figure 6: EM$-\mathrm{MW}_2^2$ flow displacing particles in order to make their EM output approach a barycentre of three target GMMs.

5.2 Colour Transfer

Colour transfer is a well-known imaging task where OT techniques have been used extensively [Rei+02; Del04; PK07; PKD07; PPC10; Rab+12; RFP14], it consists in transforming the RGB

colour distribution of a source image to match that of a target image. We propose the following approach: we initialise an image X as the source image, and optimise it to minimise the MW_2^2 cost between a GMM fitted on X (seen as an RGB point cloud), and a target GMM $\nu \in \mathrm{GMM}_3(K)$ fitted on the colour distribution of the target image. Specifically, we minimise $X \longmapsto \mathrm{MW}_2^2(\mu(F_X^T(\theta_0)), \nu)$ for some initialisation θ_0. *Warm-Start EM* method from Algorithm 3, choose $K := 10$ components and use fixed uniform GMM weights (Algorithm 2 for EM) to avoid being trapped in a local minimum, as seen in Section 3.3. We present some results in Fig. 7, and provide additional discussions about the optimisation choices in Appendix A.7.3. Even with only $K = 10$ components, the colour transfer preserves both details and global consistency, and in the resulting colour scheme, the source image appears to exhibit stronger contrast in this example.

Balanced OT methods may be sensitive to outliers in the distributions, leading to artifacts in the colour transfer results if colour aberrations are present in the target. Unbalanced OT methods [LMS18] can mitigate this issue [Chi+18a; Bon+24]. We now consider the unbalanced variant of the Mixture Loss defined in Section 3.4: by relaxing the constraints on marginals, it can ignore outliers in the input distributions. Specifically, in Fig. 7, we consider an illustration with a corrupted target image where a patch of red has been added, and observe that the unbalanced approach is more robust to this aberration, showing no propagation of the red artifact.

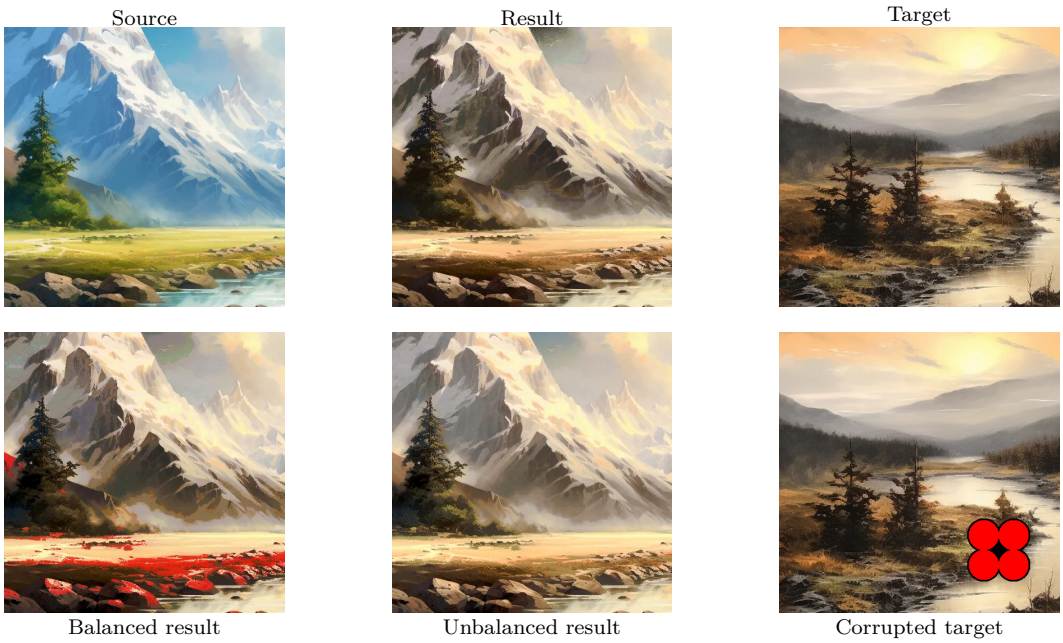


Figure 7: Colour transfer experiments. Top row: EM – MW_2^2 from source to target. Bottom row: unbalanced colour transfer with regularisations $\lambda_1 = 10$ (source) and $\lambda_2 = 0.1$ (target).

5.3 Neural Style Transfer

We apply our distance within Gatys et al.'s neural style transfer framework [GEB16]. The goal is to generate an image that combines the content of one image X_0 with the artistic style of another image Y. We rely on a pre-trained VGG-19 network [SZ15] (see Fig. 8a) to encode our image and extract relevant features from three specific layers ℓ. Given an image X in $\mathbb{R}^{3\times H\times W}$, we denote these features $\mathrm{VGG}_{1\dots\ell}(X)$. Starting with the content image as X_0, we optimise X such that its features progressively match those of the style image Y. The target mixtures $\overline{\mu}_\ell^{\text{style}}$ are fitted at the beginning of the procedure on style features $\mathrm{VGG}_{1\dots\ell}(Y)$. Notably, the target style is encoded as a low-dimensional GMM, and the reference image is not needed during training, unlike in [GEB16]. Our objective is a weighted sum of Mixture Losses between the optimised features $\mathrm{VGG}_{1\dots\ell}(X)$ and

the target $\overline{\mu}_\ell^{\text{style}}$, for each layer ℓ in $\{1, 2, 3\}$: we solve

$$\min_{X \in \mathbb{R}^{3\times H\times W}} \sum_{l=1}^{3} \lambda_\ell \, \mathrm{MW}_2^2 \left(F(\theta_{\text{init}}, \mathrm{VGG}_{1\cdots\ell}(X)), \overline{\mu}_\ell^{\text{style}} \right). \tag{23}$$

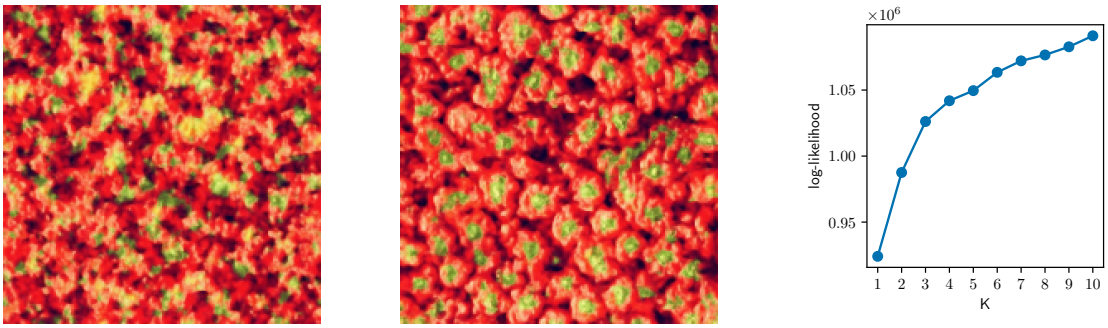
The weights λ_ℓ follow Gatys et al.'s scheme ([GEB15]) of $1/d_\ell^2$ where d_ℓ is the dimension of features in layer ℓ. We fit $K = 3$ Gaussian components at each layer. The chosen procedure for fitting Gaussian mixtures is still *Warm-Start EM*. We optimise using 100 iterations of Adam with learning rate 0.01, which takes approximately 20 seconds using CUDA on an RTX 4000. The example results shown in Fig. 8 illustrate the ability of GMM to encode (and store) an image style which, to the best of our knowledge, has never been shown. The experimental setup is further detailed in Appendix A.7.4, along with additional examples.

(a) Model architecture (b) Source im. (c) Generated im. (d) Style im.

Figure 8: Style transfer method inspired by [GEB15]: setup and example result.

5.4 Texture Synthesis

We perform texture synthesis using a novel method inspired by [GLR18; LDD23; Hou+23]. We initialise the synthesised texture using a stationary Gaussian field of the same mean and covariance as the target texture. We then optimise a weighted sum of Mixture Losses over different scales in the patch space (we refer the reader to Appendix A.7.5 for a full explanation). When doing multi-scale synthesis, we simply choose to downscale the images by a factor 2^s for s between 0 and S, so that the image downscaled by a factor 2^S has size at least 16×16. In our experiments, we choose to fit $K = 4$ Gaussian components in our mixtures. As illustrated in Fig. 9, this corresponds (roughly) to the elbow of the model's log-likelihood and gives more convincing results. As for patch sizes, notice that choosing a patch size of 1 amounts to optimising the colour intensities directly, i.e. performing standard colour transfer. In Fig. 10, we compare the results of taking 4×4 and 8×8 patches. The mono-scale variant is functional for simple textures, as presented in Appendix A.7.5. A strength of this approach is that it deals with multiple scales at once, whereas [LDD23; Hou+23] go through each scale successively. This provides a simpler approach that is less sensitive to the transition method between scales.



(a) $K = 1$ components (b) $K = 4$ components (c) Log-likelihood of fit

Figure 9: Choosing the number of components K for texture synthesis.

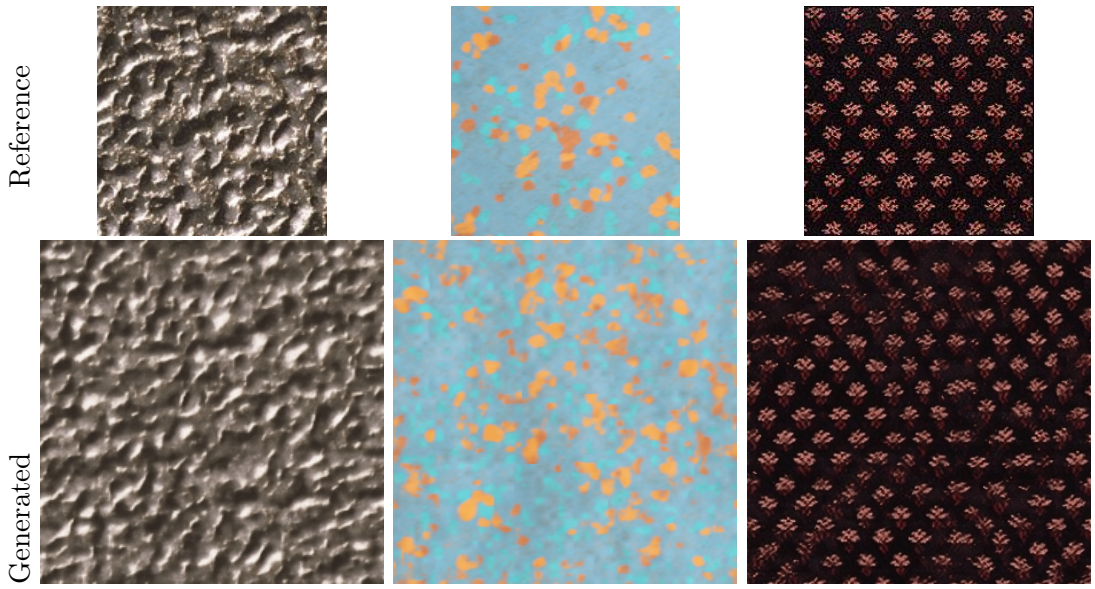


Figure 10: Multi-scale texture synthesis with $K = 4$ components for 8×8 patches

Conclusion and Outlook

In this work, we have provided multiple strategies to differentiate the Expectation-Maximisation algorithm, and illustrated their use in several applications. The overall practical message is that when possible, the Warm-Start method is to be preferred, and when not applicable, the Automatic Differentiation method is a good off-the-shelf solution. A core limitation of our methods (and EM to a lesser extent) is the computational cost when the dimension is large: our applications were limited to $d \approx 1000$. For very high dimensional cases, one could consider sparse covariance representations such as [BC25; Szw+25]. While we recommend using fixed uniform GMM weights, learnable weights remain an option. Further numerical and algorithmic tweaks could be considered to avoid issues with local optima. Finally, we focused on Gaussian Mixtures, but any mixture with differentiable densities could be considered. Note that for a seamless extension, one still requires closed-form expressions for the E and M steps.

Acknowledgements

This research was funded in part by the Agence nationale de la recherche (ANR), Grant ANR-23-CE40-0017 and by the France 2030 program, with the reference ANR-23-PEIA-0004.

References

[AC11] Martial Agueh and Guillaume Carlier. "Barycenters in the Wasserstein Space". In: *SIAM Journal on Mathematical Analysis* 43.2 (2011), pp. 904–924.

[AB22] Jason M. Altschuler and Enric Boix-Adserà. "Wasserstein Barycenters Are NP-Hard to Compute". In: *SIAM Journal on Mathematics of Data Science* 4.1 (2022), pp. 179–203.

[Álv+16] Pedro C Álvarez-Esteban, E Del Barrio, JA Cuesta-Albertos, and C Matrán. "A fixed-point approach to barycenters in Wasserstein space". In: *Journal of Mathematical Analysis and Applications* 441.2 (2016), pp. 744–762.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein generative adversarial networks". In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[BKK19] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. "Deep equilibrium models". In: *Advances in neural information processing systems* 32 (2019).

[BE94] TL Bailey and C Elkan. "Fitting a mixture model by expectation maximization to discover motifs in biopolymers." In: *Proceedings. International Conference on Intelligent Systems for Molecular Biology*. Vol. 2. 1994, pp. 28–36.

[Bec94] Thomas Beck. "Automatic differentiation of iterative processes". In: *Journal of Computational and Applied Mathematics* 50.1-3 (1994), pp. 109–118.

[Bha13] Rajendra Bhatia. *Matrix analysis*. Vol. 169. Springer Science & Business Media, 2013.

[Blo+22] Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. "Efficient and modular implicit differentiation". In: *Advances in neural information processing systems* 35 (2022), pp. 5230–5242.

[Bol+21] Jérôme Bolte, Tam Le, Edouard Pauwels, and Tony Silveti-Falls. "Nonsmooth implicit differentiation for machine-learning and optimization". In: *Advances in neural information processing systems* 34 (2021), pp. 13537–13549.

[BPV22] Jérôme Bolte, Edouard Pauwels, and Samuel Vaiter. "Automatic differentiation of nonsmooth iterative algorithms". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 26404–26417.

[BPV24] Jérôme Bolte, Edouard Pauwels, and Samuel Vaiter. "One-step differentiation of iterative algorithms". In: *Advances in Neural Information Processing Systems* 36 (2024).

[Bon+24] Clément Bonet, Kimia Nadjahi, Thibault Séjourné, Kilian Fatras, and Nicolas Courty. "Slicing Unbalanced Optimal Transport". In: *Transactions on Machine Learning Research* (2024).

[Bon+11] N. Bonneel, M. Van De Panne, S. Paris, and W. Heidrich. "Displacement interpolation using Lagrangian mass transport". In: *Proceedings of SIGGRAPH'Asia*. 2011, pp. 1–12.

[Bon13] N. Bonnotte. "Unidimensional and evolution methods for optimal transportation". PhD thesis. Paris 11, 2013.

[BC25] Charles Bouveyron and Marco Corneli. "Scaling Optimal Transport to High-Dimensional Gaussian Distributions". working paper or preprint. Jan. 2025.

[BV04] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[Boy83] Russell A Boyles. "On the convergence of the EM algorithm". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 45.1 (1983), pp. 47–50.

[CJJ05] Brian S Caffo, Wolfgang Jank, and Galin L Jones. "Ascent-based Monte Carlo expectation–maximization". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 67.2 (2005), pp. 235–251.

[Cap11] Olivier Cappé. "Online expectation maximisation". In: *Mixtures: Estimation and applications* (2011), pp. 31–53.

[CM09] Olivier Cappé and Eric Moulines. "On-line expectation–maximization algorithm for latent data models". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 71.3 (2009), pp. 593–613.

[CNR25] Sinho Chewi, Jonathan Niles-Weed, and Philippe Rigollet. *Statistical Optimal Transport. École d'Été de Probabilités de Saint-Flour XLIX – 2019*. 1st ed. Vol. 2417. Lecture Notes in Mathematics. Springer Cham, Apr. 2025, pp. XIV + 260.

[Chi+20] L. Chizat, P. Roussillon, F. Léger, F.X. Vialard, and G. Peyré. "Faster Wasserstein distance estimation with the Sinkhorn divergence". In: *Adv. Neural Inf. Process. Syst.* 33 (2020), pp. 2257–2269.

[Chi+18a] Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. "Scaling algorithms for unbalanced optimal transport problems". In: *Mathematics of computation* 87.314 (2018), pp. 2563–2609.

[Chi+18b] Lenaic Chizat, Gabriel Peyré, Bernhard Schmitzer, and François-Xavier Vialard. "Unbalanced optimal transport: Dynamic and Kantorovich formulations". In: *Journal of Functional Analysis* 274.11 (2018), pp. 3090–3123.

[Cou+17] N. Courty, R. Flamary, A. Habrard, and A. Rakotomamonjy. "Joint distribution optimal transportation for domain adaptation". In: *Adv. Neural Inf. Process. Syst.* 30 (2017).

[Cut13] Marco Cuturi. "Sinkhorn distances: Lightspeed computation of optimal transport". In: *Advances in neural information processing systems* 26 (2013).

[CD14] Marco Cuturi and Arnaud Doucet. "Fast Computation of Wasserstein Barycenters". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research. Beijing, China: PMLR, June 2014, pp. 685–693.

[Dal+23] Maxime Dalery, Genevieve Dusson, Virginie Ehrlacher, and Alexei Lozinski. *Nonlinear reduced basis using mixture Wasserstein barycenters: application to an eigenvalue problem inspired from quantum chemistry*. 2023. arXiv: 2307.15423 [math.NA].

[DH97] Peter Dayan and Geoffrey E Hinton. "Using expectation-maximization for reinforcement learning". In: *Neural Computation* 9.2 (1997), pp. 271–278.

[Del04] Julie Delon. "Midway image equalization". In: *Journal of Mathematical Imaging and Vision* 21.2 (2004), pp. 119–134.

[DD20] Julie Delon and Agnes Desolneux. "A Wasserstein-type distance in the space of Gaussian mixture models". In: *SIAM Journal on Imaging Sciences* 13.2 (2020), pp. 936–970.

[DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin. "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the royal statistical society: series B (methodological)* 39.1 (1977), pp. 1–22.

[DZS18] Ishan Deshpande, Ziyu Zhang, and Alexander G. Schwing. "Generative Modeling Using the Sliced Wasserstein Distance". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 3483–3491.

[Die+24] Jean-Frédéric Diebold, Nicolas Papadakis, Arnaud Dessein, and Charles-Alban Deledalle. "A unified framework for hard and soft clustering with regularized optimal transport". In: *2024 32nd European Signal Processing Conference (EUSIPCO)*. IEEE. 2024, pp. 1826–1830.

[ER24] Matthias J Ehrhardt and Lindon Roberts. "Analyzing inexact hypergradients for bilevel learning". In: *IMA Journal of Applied Mathematics* 89.1 (2024), pp. 254–278.

[End03] Craig K Enders. "Using the expectation maximization algorithm to estimate coefficient alpha for scales with item-level missing data." In: *Psychological methods* 8.3 (2003), p. 322.

[Fat+21a] Kilian Fatras, Thibault Séjourné, Rémi Flamary, and Nicolas Courty. "Unbalanced minibatch optimal transport; applications to domain adaptation". In: *International conference on machine learning*. PMLR. 2021, pp. 3186–3197.

[Fat+20] Kilian Fatras, Younes Zine, Rémi Flamary, Remi Gribonval, and Nicolas Courty. "Learning with minibatch Wasserstein: asymptotic and gradient properties". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 2131–2141.

[Fat+21b] Kilian Fatras, Younes Zine, Szymon Majewski, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. *Minibatch optimal transport distances; analysis and applications*. 2021. arXiv: 2101.01792 [stat.ML].

[FH02] Jeffrey A Fessler and Alfred O Hero. "Space-alternating generalized expectation-maximization algorithm". In: *IEEE Transactions on signal processing* 42.10 (2002), pp. 2664–2677.

[Fey+19] J. Feydy, P. Roussillon, A. Trouvé, and P. Gori. "Fast and scalable optimal transport for brain tractograms". In: *Proceedings of MICCAI*. Springer. 2019, pp. 636–644.

[Fri98] Nir Friedman. "The Bayesian structural EM algorithm". In: *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. 1998, pp. 129–138.

[GLR18] Bruno Galerne, Arthur Leclaire, and Julien Rabin. "A texture synthesis model based on semi-discrete optimal transport in patch space". In: *SIAM Journal on Imaging Sciences* 11.4 (2018), pp. 2456–2493.

[GTG07] Kuzman Ganchev, Ben Taskar, and João Gama. "Expectation maximization and posterior constraints". In: *Advances in neural information processing systems* 20 (2007).

[GEB16] Leon Gatys, Alexander Ecker, and Matthias Bethge. "A Neural Algorithm of Artistic Style". In: *Journal of Vision* 16.12 (Sept. 2016), p. 326.

[GEB15] Leon Gatys, Alexander S Ecker, and Matthias Bethge. "Texture synthesis using convolutional neural networks". In: *Advances in neural information processing systems* 28 (2015).

[Gen+19] Aude Genevay, Lénaic Chizat, Francis Bach, Marco Cuturi, and Gabriel Peyré. "Sample complexity of Sinkhorn divergences". In: *The 22nd international conference on artificial intelligence and statistics*. PMLR. 2019, pp. 1574–1583.

[Gil92] Jean Charles Gilbert. "Automatic differentiation and iterative processes". In: *Optimization methods and software* 1.1 (1992), pp. 13–21.

[Goo+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014).

[GVS17] Klaus Greff, Sjoerd Van Steenkiste, and Jürgen Schmidhuber. "Neural expectation maximization". In: *Advances in neural information processing systems* 30 (2017).

[GW08] Andreas Griewank and Andrea Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*. SIAM, 2008.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[Hou+23] Antoine Houdard, Arthur Leclaire, Nicolas Papadakis, and Julien Rabin. "A generative model for texture synthesis based on optimal transport between feature distributions". In: *Journal of Mathematical Imaging and Vision* 65.1 (2023), pp. 4–28.

[Kan42] Leonid V Kantorovich. "On the translocation of masses". In: *Dokl. Akad. Nauk. USSR (NS)*. Vol. 37. 1942, pp. 199–201.

[KLA19] T. Karras, S. Laine, and T. Aila. "A style-based generator architecture for generative adversarial networks". In: *Proceedings of IEEE CVPR*. 2019, pp. 4401–4410.

[Kim22] Minyoung Kim. "Differentiable Expectation-Maximization for Set Representation Learning". In: *International Conference on Learning Representations*. 2022.

[KW14] Diederik P Kingma and Max Welling. "Auto-encoding variational Bayes". In: *Int. Conf. on Learning Representations*. 2014.

[Kol+19] Soheil Kolouri, Phillip E. Pope, Charles E. Martin, and Gustavo K. Rohde. "Sliced Wasserstein Auto-Encoders". In: *International Conference on Learning Representations*. 2019.

[KC20] Jeongyeol Kwon and Constantine Caramanis. "The EM Algorithm gives Sample-Optimality for Learning Mixtures of Well-Separated Gaussians". en. In: *Proceedings of Thirty Third Conference on Learning Theory*. ISSN: 2640-3498. PMLR, July 2020, pp. 2425–2487.

[LDD23] Leclaire, Delon, and Desolneux. "Optimal Transport Between GMM for Texture Synthesis". In: *Proceedings of SSVM 2023*. 2023.

[LMS18] Matthias Liero, Alexander Mielke, and Giuseppe Savaré. "Optimal entropy-transport problems and a new Hellinger–Kantorovich distance between positive measures". In: *Inventiones mathematicae* 211.3 (2018), pp. 969–1117.

[Lin70] S Linnainmaa. "The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors (Doctoral dissertation, Master's Thesis". PhD thesis. MA thesis. University of Helsinki, 1970.

[LVD20] Jonathan Lorraine, Paul Vicol, and David Duvenaud. "Optimizing millions of hyperparameters by implicit differentiation". In: *International conference on artificial intelligence and statistics*. PMLR. 2020, pp. 1540–1552.

[Lui+18] Giulia Luise, Alessandro Rudi, Massimiliano Pontil, and Carlo Ciliberto. "Differential properties of sinkhorn approximation for learning with wasserstein distance". In: *Advances in Neural Information Processing Systems* 31 (2018).

[Luz+23] Lorenzo Luzi, Carlos Ortiz Marrero, Nile Wynar, Richard G Baraniuk, and Michael J Henry. "Evaluating generative networks using Gaussian mixtures of image features". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2023, pp. 279–288.

[MK07] Geoffrey J McLachlan and Thriyambakam Krishnan. *The EM algorithm and extensions*. John Wiley & Sons, 2007.

[MP00] Geoffrey J McLachlan and David Peel. *Finite mixture models*. John Wiley & Sons, 2000.

[MO20] Sheheryar Mehmood and Peter Ochs. "Automatic differentiation of some first-order methods in parametric optimization". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 1584–1594.

[Men+20] Gonzalo Mena, Amin Nejatbakhsh, Erdem Varol, and Jonathan Niles-Weed. *Sinkhorn EM: An Expectation-Maximization algorithm based on entropic optimal transport*. 2020. arXiv: 2006.16548 [stat.ML].

[Mia+21] Grégoire Mialon, Dexiong Chen, Alexandre d'Aspremont, and Julien Mairal. "A Trainable Optimal Transport Embedding for Feature Aggregation and its Relationship to Attention". In: *ICLR 2021-The Ninth International Conference on Learning Representations*. 2021.

[Mon81] Gaspard Monge. "Mémoire sur la théorie des déblais et des remblais". In: *Mem. Math. Phys. Acad. Royale Sci.* (1781), pp. 666–704.

[MMS24] Eduardo Fernandes Montesuma, Fred Ngolè Mboula, and Antoine Souloumiac. "Lighter, better, faster multi-source domain adaptation with gaussian mixture models and optimal transport". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2024, pp. 21–38.

[Moo96] T.K. Moon. "The expectation-maximization algorithm". In: *IEEE Signal Processing Magazine* 13.6 (1996), pp. 47–60.

[NB06] Nikolaos Nasios and Adrian G Bors. "Variational learning for Gaussian mixture models". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 36.4 (2006), pp. 849–862.

[Ng13] Shu-Kay Ng. "Recent developments in expectation-maximization methods for analyzing complex data". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 5.6 (2013), pp. 415–431.

[NW06] J Nocedal and SJ Wright. "Numerical optimization". In: *Springer Series in Operations Research and Financial Engineering* (2006).

[PPC10] Nicolas Papadakis, Edoardo Provenzi, and Vicent Caselles. "A variational model for histogram transfer of color images". In: *IEEE Transactions on Image Processing* 20.6 (2010), pp. 1682–1695.

[Pas+19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. "Py-

Torch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems* 32 (2019).

[PP08] K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook*. Version 20081110. Oct. 2008.

[PC19] G. Peyré and M. Cuturi. "Computational Optimal Transport". In: *Foundations and Trends in Machine Learning* 51.1 (2019), pp. 1–44.

[PC+19] Gabriel Peyré, Marco Cuturi, et al. "Computational optimal transport: With applications to data science". In: *Foundations and Trends® in Machine Learning* 11.5-6 (2019), pp. 355–607.

[PK07] François Pitié and Anil Kokaram. "The linear monge-kantorovitch linear colour mapping for example-based colour transfer". In: *4th European conference on visual media production*. IET. 2007, pp. 1–9.

[PKD07] François Pitié, Anil C Kokaram, and Rozenn Dahyot. "Automated colour grading using colour distribution transfer". In: *Computer Vision and Image Understanding* 107.1-2 (2007), pp. 123–137.

[RFP14] Julien Rabin, Sira Ferradans, and Nicolas Papadakis. "Adaptive color transfer with relaxed optimal transport". In: *2014 IEEE international conference on image processing (ICIP)*. IEEE. 2014, pp. 4852–4856.

[Rab+12] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. "Wasserstein barycenter and its application to texture mixing". In: *Scale Space and Variational Methods in Computer Vision: Third International Conference, SSVM 2011, Ein-Gedi, Israel, May 29–June 2, 2011, Revised Selected Papers 3*. Springer. 2012, pp. 435–446.

[RMC16] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2016.

[Ras03] Carl Edward Rasmussen. "Gaussian processes in machine learning". In: *Summer school on machine learning*. Springer, 2003, pp. 63–71.

[Rei+02] Erik Reinhard, Michael Adhikhmin, Bruce Gooch, and Peter Shirley. "Color transfer between images". In: *IEEE Computer graphics and applications* 21.5 (2002), pp. 34–41.

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[RW18] Philippe Rigollet and Jonathan Weed. "Entropic optimal transport is maximum-likelihood deconvolution". In: *Comptes Rendus. Mathématique* 356.11-12 (2018), pp. 1228–1235.

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*. Ed. by David E. Rumelhart and James L. Mcclelland. Cambridge, MA: MIT Press, 1986, pp. 318–362.

[SCR12] Rajhans Samdani, Ming-Wei Chang, and Dan Roth. "Unified expectation maximization". In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2012, pp. 688–698.

[Sha+19] Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. "Truncated backpropagation for bilevel optimization". In: *The 22nd international conference on artificial intelligence and statistics*. PMLR. 2019, pp. 1723–1732.

[SZ15] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[Szw+25] Tom Szwagier, Pierre-Alexandre Mattei, Charles Bouveyron, and Xavier Pennec. *Parsimonious Gaussian mixture models with piecewise-constant eigenvalue profiles*. 2025. arXiv: 2507.01542 [stat.ML].

[TDG24] Eloi Tanguy, Julie Delon, and Nathaël Gozlan. *Computing Barycentres of Measures for Generic Transport Costs*. 2024. arXiv: 2501.04016 [math.NA].

[TDD25] Eloi Tanguy, Agnès Desolneux, and Julie Delon. *Constrained Approximate Optimal Transport Maps*. 2025. arXiv: 2407.13445 [math.OC].

[TFD24] Eloi Tanguy, Rémi Flamary, and Julie Delon. "Properties of Discrete Sliced Wasserstein Losses". In: *Mathematics of Computation* (June 2024).

[Ton+24] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. "Improving and generalizing flow-based generative models with minibatch optimal transport". In: *Transactions on Machine Learning Research* (2024), pp. 1–34.

[VR08] Ravi Varadhan and Christophe Roland. "Simple and globally convergent methods for accelerating the convergence of any EM algorithm". In: *Scandinavian Journal of Statistics* 35.2 (2008), pp. 335–353.

[VL25] Titouan Vayer and Etienne Lasalle. *A note on the relations between mixture models, maximum-likelihood and entropic optimal transport*. 2025. arXiv: 2501.12005 [stat.ML].

[VM19] Cinzia Viroli and Geoffrey J McLachlan. "Deep Gaussian mixture models". In: *Statistics and Computing* 29.1 (2019), pp. 43–51.

[VH10] Dang Hai Tran Vu and Reinhold Haeb-Umbach. "Blind speech separation employing directional statistics in an expectation maximization framework". In: *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE. 2010, pp. 241–244.

[WB19] J. Weed and F. Bach. "Sharp asymptotic and finite-sample rates of convergence of empirical measures in Wasserstein distance". In: *Bernoulli* 25.4A (2019), pp. 2620–2648.

[Wen64] Robert Edwin Wengert. "A simple automatic derivative evaluation program". In: *Communications of the ACM* 7.8 (1964), pp. 463–464.

[Wu83] CF Jeff Wu. "On the convergence properties of the EM algorithm". In: *The Annals of statistics* (1983), pp. 95–103.

[XHM16] Ji Xu, Daniel J Hsu, and Arian Maleki. "Global analysis of expectation maximization for mixtures of two gaussians". In: *Advances in Neural Information Processing Systems* 29 (2016).

[Yua+20] Wentao Yuan, Benjamin Eckart, Kihwan Kim, Varun Jampani, Dieter Fox, and Jan Kautz. "Deepgmr: Learning latent gaussian mixture models for registration". In: *European conference on computer vision*. Springer. 2020, pp. 733–750.

[ZAC21] Marco Zaffalon, Alessandro Antonucci, and Rafael Cabañas. *Causal Expectation-Maximisation*. 2021. arXiv: 2011.02912 [cs.AI].

A Supplementary Material

A.1 Postponed Proofs

Proof of Proposition 3.1. Consider the cost matrices $C, \hat{C} \in \mathbb{R}_+^{K_0 \times K_1}$ defined by their entries at $(k_0, k_1) \in [\![1, K_0]\!] \times [\![1, K_1]\!]$:

$$C_{k_0,k_1} := \|m_{k_0}^{(0)} - m_{k_1}^{(1)}\|_2^2 + d_{\mathrm{BW}}^2(\Sigma_{k_0}^{(0)}, \Sigma_{k_1}^{(1)}),\ \hat{C}_{k_0,k_1} := \|\hat{m}_{k_0}^{(0)} - \hat{m}_{k_1}^{(1)}\|_2^2 + d_{\mathrm{BW}}^2(\hat{\Sigma}_{k_0}^{(0)}, \hat{\Sigma}_{k_1}^{(1)}),$$

and apply [TFD24, Lemma 2.1] with weights $(w_i, \overline{w_i}) := (w_i, \hat{w}_i)$ for $i \in \{0, 1\}$, and cost matrices $(M, \overline{M}) := (C, \hat{C})$: we obtain:

$$\left|\mathrm{MW}_2^2(\hat{\mu}_1, \hat{\mu}_2) - \mathrm{MW}_2^2(\mu_1, \mu_2)\right| \leq \|C - \hat{C}\|_\infty + \|C\|_\infty \left(\|w_0 - \hat{w}_0\|_1 + \|w_1 - \hat{w}_1\|_1\right).$$

First, noticing that $\forall \Sigma \in S_d^+(\mathbb{R}),\ d_{\mathrm{BW}}(0, \Sigma) = \sqrt{\mathrm{Tr}(\Sigma)}$, we apply the triangle inequality on $\|\cdot\|_2$ and d_{BW} to obtain $\|C\|_\infty \leq 4(R_m^2 + R_\Sigma^2)$. We now turn to upper-bounding $\|C - \hat{C}\|_\infty$. We will use the inequality $\forall (s,t) \in \mathbb{R}^2,\ |s^2 - t^2| \leq 2\max(|s|, |t|)|s - t|$. First, for $x, \hat{x}, y, \hat{y} \in B_{\mathbb{R}^d}(0, R_m)$, we have by the triangle inequality:

$$\left|\|x - y\|_2^2 + \|\hat{x} - \hat{y}\|_2^2\right| \leq 2\max(\|x - y\|_2, \|\hat{x} - \hat{y}\|_2)\|(x - y) - (\hat{x} - \hat{y})\|_2 \leq 4R_m \left(\|x - \hat{x}\|_2 + \|y - \hat{y}\|_2\right).$$

Similarly, for $A, \hat{A}, B, \hat{B} \in \{S \in S_d^+(\mathbb{R}) : \sqrt{\mathrm{Tr}(S)} \leq R_\sigma\}$, we compute:

$$\left|d_{\mathrm{BW}}(A, B) - d_{\mathrm{BW}}(\hat{A}, \hat{B})\right| \leq 4R_\Sigma \left(d_{\mathrm{BW}}(A, \hat{A}) + d_{\mathrm{BW}}(B, \hat{B})\right).$$

Applying our two inequalities to the entries of $C - \hat{C}$ and combining with the inequality on MW_2^2 gives Eq. (18) in expectation.

Proof of Proposition 3.2. For notational simplicity, introduce $g_k := \mathcal{N}(m_k, \Sigma_k)$ and $\hat{g}_k := \mathcal{N}(m_k, \hat{\Sigma}_k)$. Let $\eta_k := \min(\hat{w}_k, w_k)$ and the diagonal matrix $D := \mathrm{diag}(\eta)$. Our objective is to complete the non-negative matrix D into a coupling $P \in \Pi(\hat{w}, w)$. First, if $\sum_k \eta_k = 1$, then $w = \hat{w}$ and already $D \in \Pi(\hat{w}, w)$, so we can set $P := D$. If $\sum_k \eta_k \neq 1$, we introduce $Q \in \mathbb{R}_+^{K \times K}$ such that $Q_{k,\ell} = (\hat{w}_k - \eta_k)(w_\ell - \eta_\ell)/(1 - \sum_p \eta_p)$ for all (k, ℓ). The matrix Q is the independent coupling between the non-negative vectors $\hat{w} - \eta$ and $w - \eta$. We set $P := D + Q$, straightforward computation shows that $P \in \Pi(\hat{w}, w)$. We denote the pairwise cost matrix $C := (\mathrm{W}_2^2(\hat{g}_k, g_\ell))_{k,\ell}$ and use P as a candidate coupling in the definition of $\mathrm{MW}_2^2(\hat{\mu}, \mu)$:

$$\mathrm{MW}_2^2(\hat{\mu}, \mu) \leq \langle P, C\rangle = \langle D, C\rangle + \langle Q, C\rangle \leq \max_k \mathrm{W}_2^2(\hat{g}_k, g_k)(\sum_\ell \eta_\ell) + \|C\|_\infty \|Q\|_1.$$

(Note that in the case $\sum_k \eta_k = 1$, the second term vanishes.) Then since $\eta \leq w$ entry-wise, $\sum \eta_k \leq 1$, and we compute $\|Q\|_1 = 1 - \sum_k \eta_k$, and further re-write:

$$\sum_{k=1}^K \eta_k = \sum_{k=1}^K \min(\hat{w}_k, w_k) = \frac{1}{2}\sum_{k=1}^K (\hat{w}_k + w_k - |\hat{w}_k - w_k|) = 1 - \frac{\|\hat{w} - w\|_1}{2},$$

which finally yields:

$$\mathrm{MW}_2^2(\hat{\mu}, \mu) \leq \max_k \mathrm{W}_2^2(\hat{g}_k, g_k) + \frac{\|\hat{w} - w\|_1}{2} \max_{k,\ell} \mathrm{W}_2^2(\hat{g}_k, g_\ell).$$

Taking expectations and using the assumptions on the parameter estimators concludes the proof.

A.2 Specific GMMs Used in Section 3.2 and Section 4.5

In Section 3.2, we use two fixed mixtures μ and ν with $K = 3$ components in $d = 2$, shown in Fig. 11. Separation is controlled by the covariance scaling parameter σ: "low" uses $\sigma = 10$, "medium" $\sigma = 0.5$, and "high" $\sigma = 0.1$. Larger values create strong overlap between components, while smaller values yield well-separated mixtures. The same pairs (μ, ν) are used across all repetitions.

In Fig. 12, we show the three different GMMs used in our experiments, which were sampled randomly with fixed seeds. We observe that GMM #1 has relatively large variances, allowing for less numerical difficulty, while GMM #2 and (to an even larger extent) GMM #3 have some almost-degenerate covariances, leading to increased numerical instability. Note that GMM #2 appears to be the best separated, which can yield better EM convergence.

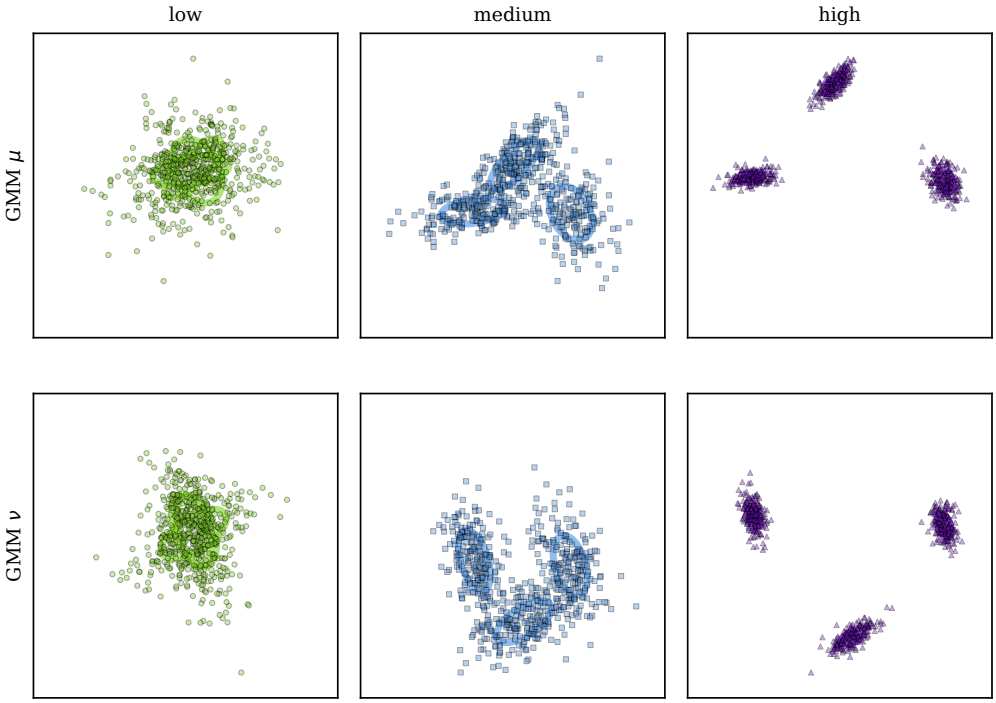


Figure 11: Two fixed GMMs μ (top) and ν (bottom) used in Section 3.2. Columns correspond to low ($\sigma = 10$), medium ($\sigma = 0.5$), and high ($\sigma = 0.1$) separation.

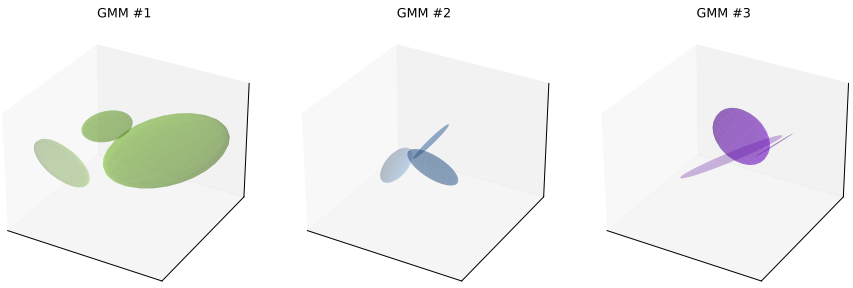


Figure 12: 3D representation of the three GMMs used in Section 4.5 to compare EM gradient methods.

A.3 Discussion on Gradient Ground Truths

As discussed in Section 2.3, the most natural baseline for the computation of a gradient of EM is the full automatic differentiation method (AD). Unfortunately, this strategy is only theoretically sound under strong assumptions [Gil92; Bec94; MO20], and may be prone to the propagation of numerical errors across iterations. Due to this latter issue in particular, one may not claim that

the AD gradient is exact. In order to discuss its use as a baseline (in particular in Section 4.5), we compare it to the gradient computed using the finite differences approximation (FD) with a step size ε_{FD}, implemented using `torch.autograd.gradcheck._get_numerical_jacobian` in Pytorch [Pas+19]. The experimental setting is as in Section 4.5: in Fig. 13, we observe the relative MSEs $\|J_{\text{FD}} - J_{\text{AD}}\|_2^2 / \|J_{\text{AD}}\|_2^2$ varying the FD step size $\varepsilon_{\text{FD}} \in \{10^{-5}, 10^{-6}, 10^{-7}\}$ on three different GMMs, taking each time 60 samples for the data X with $n = 300$, $d = 3$ and running EM with $K = 3$ and $T = 30$. Note that the FD approximation is extremely intensive numerically, prohibiting its use in practice and even for extensive testing. First, we observe that the variability between data samples is very high, indicating sensitivity of the methods to the algorithm initialisation (and in turn, convergence). Furthermore, while AD is close to FD at numerical precision for GMMs #1 and #2, the approximation is substantially coarser with a relative error of the order of 10^{-5} for GMM #3, indicating numerical instability. Given that the FD method itself is a sensitive approximation that depends strongly on ε_{FD}, we conclude that there appears to be no reliable ground truth for an exact gradient, and we resort to AD as a baseline in our experiments. We leave the challenging question of quantifying the arising numerical errors for future work.

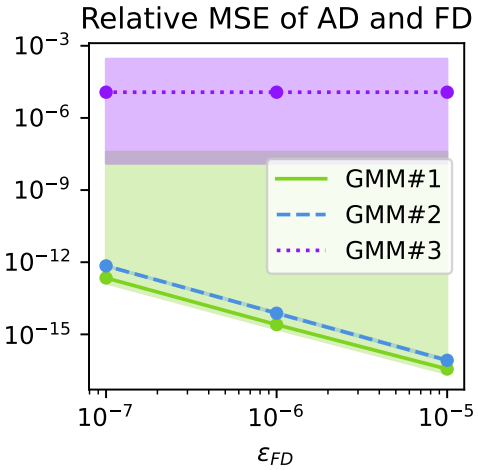


Figure 13: Relative MSE of the full automatic differentiation gradient (AD) against the finite differences approximation (FD) for three different GMMs and varying the FD step size ε_{FD}.

A.4 Explicit Differential Expressions

A.4.1 Differential of Gaussian Density

First, we compute the explicit differential of

$$g : \begin{cases} \mathbb{R}^d \times S_d^{++}(\mathbb{R}) \times \mathcal{X} & \longrightarrow & \mathbb{R}^n \\ (m, \Sigma, X) & \longmapsto & \left(\dfrac{1}{(2\pi)^{d/2} \det(\Sigma)^{1/2}} \exp\left(-\dfrac{1}{2}(x_i - m)^\top \Sigma^{-1}(x_i - m) \right) \right)_{i=1}^n \end{cases}, \tag{24}$$

where we wrote $X = (x_1, \cdots, x_n)$. We first compute the differential with respect to m:

$$\frac{\partial g}{\partial m}(m, \Sigma, X) \in \mathcal{L}(\mathbb{R}^d, \mathbb{R}^n) \simeq \mathbb{R}^{n \times d} : \left[\frac{\partial g}{\partial m}(m, \Sigma, X) \right]_{i,\cdot} = \Sigma^{-1}(x_i - m) g_{m,\Sigma}(x_i). \tag{25}$$

For the differential with respect to Σ, we remark that $T_\Sigma S_d^{++}(\mathbb{R}) = S_d(\mathbb{R})$, and use [PP08, Equation 49] stating that $\partial_A \det A = (\det A) A^{-\top}$ and [PP08, Equation 61] which states $\partial_A a^\top A^{-1} b = -A^{-\top} a b^\top A^{-\top}$, which yields:

$$\frac{\partial g}{\partial \Sigma}(m, \Sigma, X) \in \mathcal{L}(S_d(\mathbb{R}), \mathbb{R}^n) \hookrightarrow \mathbb{R}^{n \times d \times d} : \tag{26}$$

$$\left[\frac{\partial g}{\partial \Sigma}(m, \Sigma, X) \right]_{i,\cdot,\cdot} = \frac{g_{m,\Sigma}(x_i)}{2} \left(-\Sigma^{-1} + \Sigma^{-1}(x_i - m)(x_i - m)^\top \Sigma^{-1} \right). \tag{27}$$

Finally, the differential with respect to the data X reads

$$\frac{\partial g}{\partial X}(m,\Sigma,X) \in \mathcal{L}(\mathbb{R}^{n\times d},\mathbb{R}^n) \simeq \mathbb{R}^{n\times n\times d} : \left[\frac{\partial g}{\partial \Sigma}(m,\Sigma,X)\right]_{i,j,\cdot} = -\delta_{i,j} g_{m,\Sigma}(x_i)\Sigma^{-1}(x_i-m). \tag{28}$$

A.4.2 Differential of γ

We now compute the differential of the function

$$\gamma : \begin{cases} \underbrace{\triangle_K \times (\mathbb{R}^d)^K \times (S_d^{++}(\mathbb{R}))^K}_{\Theta} \times \mathcal{X} & \longrightarrow & \mathbb{R}^{n\times K} \\ \underbrace{(w,(m_k),(\Sigma_k)}_{\theta}, X) & \longmapsto & \left(\frac{w_k g_{m_k,\Sigma_k}(x_i)}{\sum_l w_l g_{m_l,\Sigma_l}(x_i)}\right)_{i,k} \end{cases}. \tag{29}$$

For notational convenience, we introduce

$$\forall (i,k) \in [\![1,n]\!]\times[\![1,K]\!],\ g_{i,k} := g_{m_k,\Sigma_k}(x_i),\ Z_i := \sum_{k'} w_{k'} g_{i,k'}. \tag{30}$$

First, the tangent space of the simplex is the same everywhere: $T_w\triangle_K = \triangle_K^0 := (\mathbb{1}_K)^\perp$, and the differential with respect to w is

$$\frac{\partial \gamma}{\partial w}(\theta,X) \in \mathcal{L}\left(\triangle_K^0, \mathbb{R}^{n\times K}\right) \hookrightarrow \mathbb{R}^{n\times K\times K} : \left[\frac{\partial \gamma}{\partial w}(\theta,X)\right]_{i,k,l} = \frac{g_{i,k}}{Z_i}\left(\delta_{k,l} - \frac{w_k g_{i,l}}{Z_i}\right). \tag{31}$$

Using Eq. (25), we compute the differential of γ w.r.t. $\mathbf{m} = (m_k)$:

$$\frac{\partial \gamma}{\partial \mathbf{m}}(\theta,X) \in \mathcal{L}\left((\mathbb{R}^d)^K, \mathbb{R}^{n\times K}\right) \simeq \mathbb{R}^{n\times K\times K\times d} : \tag{32}$$

$$\left[\frac{\partial \gamma}{\partial \mathbf{m}}(\theta,X)\right]_{i,k,l,\cdot} = \frac{w_k g_{i,k}}{Z_i}\left(\delta_{k,l} - \frac{w_l g_{i,l}}{Z_i}\right)\Sigma_l^{-1}(x_i - m_l). \tag{33}$$

Similarly, using Eq. (27), we compute the differential of γ w.r.t. $\mathbf{\Sigma} = (\Sigma_k)$:

$$\frac{\partial \gamma}{\partial \mathbf{\Sigma}}(\theta,X) \in \mathcal{L}\left((S_d(\mathbb{R}))^K, \mathbb{R}^{n\times K}\right) \hookrightarrow \mathbb{R}^{n\times K\times K\times d\times d} : \tag{34}$$

$$\left[\frac{\partial \gamma}{\partial \mathbf{\Sigma}}(\theta,X)\right]_{i,k,l,\cdot,\cdot} = \frac{w_k g_{i,k}}{2Z_i}\left(\delta_{k,l} - \frac{w_l g_{i,l}}{Z_i}\right)\left(-\Sigma_l^{-1} + \Sigma_l^{-1}(x_i-m_l)(x_i-m_l)^\top \Sigma_l^{-1}\right). \tag{35}$$

Finally, Eq. (28) allows the computation of the differential of γ w.r.t.X:

$$\frac{\partial \gamma}{\partial X}(\theta,X) \in \mathcal{L}\left(\mathbb{R}^{n\times d}, \mathbb{R}^{n\times K}\right) \simeq \mathbb{R}^{n\times K\times n\times d} : \tag{36}$$

$$\left[\frac{\partial \gamma}{\partial X}(\theta,X)\right]_{i,k,j,\cdot} = \frac{\delta_{i,j} w_k g_{i,k}}{Z_i}\left(-\Sigma_k^{-1}(x_i-m_k) + \frac{1}{Z_i}\sum_{l=1}^K w_l g_{i,l}\Sigma_l^{-1}(x_i-m_l)\right). \tag{37}$$

A.4.3 Differential of F

We introduce the coordinate notation

$$F(\theta,X) = (F_w(\theta,X), F_{\mathbf{m}}(\theta,X), F_{\mathbf{\Sigma}}(\theta,X)) \in \triangle_K \times (\mathbb{R}^d)^K \times (S_d^{++}(\mathbb{R}))^K,$$

hence the differentials of F with respect to θ and X can be written "block-wise" as follows:

$$\frac{\partial F}{\partial \theta} = \begin{pmatrix} \frac{\partial F_w}{\partial w} & \frac{\partial F_w}{\partial \mathbf{m}} & \frac{\partial F_w}{\partial \mathbf{\Sigma}} \\ \frac{\partial F_{\mathbf{m}}}{\partial w} & \frac{\partial F_{\mathbf{m}}}{\partial \mathbf{m}} & \frac{\partial F_{\mathbf{m}}}{\partial \mathbf{\Sigma}} \\ \frac{\partial F_{\mathbf{\Sigma}}}{\partial w} & \frac{\partial F_{\mathbf{\Sigma}}}{\partial \mathbf{m}} & \frac{\partial F_{\mathbf{\Sigma}}}{\partial \mathbf{\Sigma}} \end{pmatrix}, \quad \frac{\partial F}{\partial X} = \begin{pmatrix} \frac{\partial F_w}{\partial X} \\ \frac{\partial F_{\mathbf{m}}}{\partial X} \\ \frac{\partial F_{\mathbf{\Sigma}}}{\partial X} \end{pmatrix}.$$

Differentials of F_w. We now compute the differentials of F_w whose expression we remind from Eq. (4):

$$F_w : \begin{cases} \Theta \times \mathcal{X} & \longrightarrow & \triangle_K \\ (\theta, X) & \longmapsto & \left(\frac{1}{n}\sum_{i=1}^n \gamma_{i,k}(\theta, X)\right)_{k=1}^K \end{cases}. \tag{38}$$

By Eq. (31), we have

$$\frac{\partial F_w}{\partial w}(\theta, X) \in \mathcal{L}\left(\triangle_K^0, \triangle_K^0\right) \hookrightarrow \mathbb{R}^{K\times K} : \left[\frac{\partial F_w}{\partial w}(\theta, X)\right]_{k,l} = \frac{1}{n}\sum_{i=1}^n \frac{g_{i,k}}{Z_i}\left(\delta_{k,l} - \frac{w_k g_{i,l}}{Z_i}\right). \tag{39}$$

Likewise, Eq. (33) yields

$$\frac{\partial F_w}{\partial \mathbf{m}}(\theta, X) \in \mathcal{L}\left((\mathbb{R}^d)^K, \triangle_K^0\right) \hookrightarrow \mathbb{R}^{K\times K\times d} : \tag{40}$$

$$\left[\frac{\partial F_w}{\partial \mathbf{m}}(\theta, X)\right]_{k,l,\cdot} = \frac{1}{n}\sum_{i=1}^n \frac{w_k g_{i,k}}{Z_i}\left(\delta_{k,l} - \frac{w_l g_{i,l}}{Z_i}\right)\Sigma_l^{-1}(x_i - m_l). \tag{41}$$

Eq. (35) gives

$$\frac{\partial F_w}{\partial \mathbf{\Sigma}}(\theta, X) \in \mathcal{L}\left((S_d(\mathbb{R}))^K, \triangle_K^0\right) \hookrightarrow \mathbb{R}^{K\times K\times d\times d} : \tag{42}$$

$$\left[\frac{\partial F_w}{\partial \mathbf{\Sigma}}(\theta, X)\right]_{k,l,\cdot,\cdot} = \frac{1}{n}\sum_{i=1}^n \frac{w_k g_{i,k}}{2Z_i}\left(\delta_{k,l} - \frac{w_l g_{i,l}}{Z_i}\right)\left(-\Sigma_l^{-1} + \Sigma_l^{-1}(x_i - m_l)(x_i - m_l)^\top \Sigma_l^{-1}\right). \tag{43}$$

Finally, with Eq. (37), we obtain

$$\frac{\partial F_w}{\partial X}(\theta, X) \in \mathcal{L}\left(\mathbb{R}^{n\times d}, \triangle_K^0\right) \hookrightarrow \mathbb{R}^{K\times n\times d} : \tag{44}$$

$$\left[\frac{\partial F_w}{\partial X}(\theta, X)\right]_{k,i,\cdot} = \frac{w_k g_{i,k}}{nZ_i}\left(-\Sigma_k^{-1}(x_i - m_k) + \frac{1}{Z_i}\sum_{l=1}^K w_l g_{i,l}\Sigma_l^{-1}(x_i - m_l)\right). \tag{45}$$

Differentials of $F_{\mathbf{m}}$. We now turn to $F_{\mathbf{m}}$:

$$F_{\mathbf{m}} : \begin{cases} \Theta \times \mathcal{X} & \longrightarrow & \mathbb{R}^{K\times d} \\ (\theta, X) & \longmapsto & \left(\frac{\sum_{i=1}^n \gamma_{i,k}(\theta, X)x_i}{\sum_{j=1}^n \gamma_{j,k}(\theta, X)}\right)_{k=1}^K \end{cases}. \tag{46}$$

For convenience we introduce $\Gamma_k := \sum_{i=1}^n \gamma_{i,k}$. The chain rule gives

$$\frac{\partial F_{\mathbf{m}}}{\partial w}(\theta, X) \in \mathcal{L}\left(\triangle_K^0, \mathbb{R}^{K\times d}\right) \hookrightarrow \mathbb{R}^{K\times d\times K} : \tag{47}$$

$$\left[\frac{\partial F_{\mathbf{m}}}{\partial w}(\theta, X)\right]_{k,\cdot,l} = \frac{1}{\Gamma_k}\left(\sum_{i=1}^n \left[\frac{\partial \gamma}{\partial w}\right]_{i,k,l} x_i - \frac{1}{\Gamma_k}\sum_{j=1}^n \left[\frac{\partial \gamma}{\partial w}\right]_{j,k,l}\sum_{i=1}^n \gamma_{i,k}x_i\right). \tag{48}$$

We continue with the differential with respect to $\mathbf{m}$:

$$\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{m}}(\theta, X) \in \mathcal{L}\left(\mathbb{R}^{K\times d}, \mathbb{R}^{K\times d}\right) \simeq \mathbb{R}^{K\times d\times K\times d} : \tag{49}$$

$$\left[\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{m}}(\theta, X)\right]_{k,\cdot,l,\cdot} = \frac{1}{\Gamma_k}\left(\sum_{i=1}^n x_i \left[\frac{\partial \gamma}{\partial \mathbf{m}}\right]_{i,k,l,\cdot}^\top - \frac{1}{\Gamma_k}\left(\sum_{i=1}^n \gamma_{i,k}x_i\right)\sum_{j=1}^n \left[\frac{\partial \gamma}{\partial \mathbf{m}}\right]_{j,k,l,\cdot}^\top\right). \tag{50}$$

For the differential with respect to $\mathbf{\Sigma}$, we require the notion of *outer product* between two tensors, which we define below:

$$\forall A \in \mathbb{R}^{n_1 \times \cdots \times n_N},\ B \in \mathbb{R}^{m_1 \times \cdots \times m_M},\ A \otimes B := (A_{i_1,\cdots,i_N} B_{j_1,\cdots,j_M}) \in \mathbb{R}^{n_1 \times \cdots \times n_N \times m_1 \times \cdots \times m_M}. \quad (51)$$

Note that in Eq. (50), we could have written $x_i \otimes \left[\frac{\partial \gamma}{\partial \mathbf{m}}\right]_{i,k,l,\cdot}$. We will need the following differentiation rule for the outer product:

Lemma A.1. Let $A : \mathbb{R}^{p_1 \times \cdots \times p_P} \longrightarrow \mathbb{R}^{n_1 \times \cdots \times n_N}$ and $B : \mathbb{R}^{k_1 \times \cdots \times k_P} \longrightarrow \mathbb{R}^{m_1 \times \cdots \times m_M}$ be differentiable. Then the differential of $C := A \otimes B$ is:

$$\frac{\partial C}{\partial X}(X) \in \mathcal{L}(\mathbb{R}^{p_1 \times \cdots \times p_P}, \mathbb{R}^{n_1 \times \cdots \times n_N \times m_1 \times \cdots \times m_M}) \simeq \mathbb{R}^{n_1 \times \cdots \times n_N \times m_1 \times \cdots \times m_M \times p_1 \times \cdots \times p_P} : \quad (52)$$

$$\left[\frac{\partial C}{\partial X}(X)\right]_{i_1,\cdots,i_N,j_1,\cdots,j_M,k_1,\cdots,k_P} = \left[\frac{\partial A}{\partial X}(X)\right]_{i_1,\cdots,i_N,k_1,\cdots,k_P} B_{j_1,\cdots,j_M}(X) + A_{i_1,\cdots,i_N} \left[\frac{\partial B}{\partial X}(X)\right]_{j_1,\cdots,j_M,k_1,\cdots,k_P}, \quad (53)$$

$$\frac{\partial C}{\partial X}(X) = \tau\left(\frac{\partial A}{\partial X}(X) \otimes B(X)\right) + A(X) \otimes \frac{\partial B}{\partial X}(X), \quad (54)$$

where τ is the transposition $(\tau(T))_{i_1,\cdots,i_N,j_1,\cdots,j_m,k_1,\cdots,k_P} = T_{i_1,\cdots,i_N,k_1,\cdots,k_P,j_1,\cdots,j_M}$. If $B = A$, then the formula can be written as

$$\frac{\partial C}{\partial X}(X) = \tau\left(A(X) \otimes \frac{\partial A}{\partial X}(X)\right) + A(X) \otimes \frac{\partial A}{\partial X}(X). \quad (55)$$

Note that in particular, the intuitive formula $\partial(A \otimes B) = (\partial A) \otimes B + A \otimes (\partial B)$ does not hold. We now compute the differential with respect to $\mathbf{\Sigma}$:

$$\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{\Sigma}}(\theta, X) \in \mathcal{L}\left((S_d(\mathbb{R}))^K, \mathbb{R}^{K \times d}\right) \hookrightarrow \mathbb{R}^{K \times d \times K \times d \times d} : \quad (56)$$

$$\left[\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{\Sigma}}(\theta, X)\right]_{k,\cdot,l,\cdot,\cdot} = \frac{1}{\Gamma_k}\left(\sum_{i=1}^n x_i \otimes \left[\frac{\partial \gamma}{\partial \mathbf{\Sigma}}\right]_{i,k,l,\cdot,\cdot} - \frac{1}{\Gamma_k}\left(\sum_{i=1}^n \gamma_{i,k} x_i\right) \otimes \sum_{j=1}^n \left[\frac{\partial \gamma}{\partial \mathbf{\Sigma}}\right]_{j,k,l,\cdot,\cdot}\right). \quad (57)$$

The differential with respect to X is slightly different due to the product with x_i in Eq. (46):

$$\frac{\partial F_{\mathbf{m}}}{\partial X}(\theta, X) \in \mathcal{L}\left(\mathbb{R}^{n \times d}, \mathbb{R}^{K \times d}\right) \simeq \mathbb{R}^{K \times d \times n \times d} : \quad (58)$$

$$\left[\frac{\partial F_{\mathbf{m}}}{\partial X}(\theta, X)\right]_{k,\cdot,i,\cdot} = \frac{1}{\Gamma_k}\left(\gamma_{i,k} I_d + \sum_{h=1}^n x_h \left[\frac{\partial \gamma}{\partial X}\right]^\top_{h,k,i,\cdot} - \frac{1}{\Gamma_k}\left(\sum_{h=1}^n \gamma_{h,k} x_h\right) \sum_{j=1}^n \left[\frac{\partial \gamma}{\partial X}\right]^\top_{j,k,i,\cdot}\right). \quad (59)$$

Differential of $F_{\mathbf{\Sigma}}$. We finish with the computation of the differentials of

$$F_{\mathbf{\Sigma}} : \begin{cases} \Theta \times \mathcal{X} & \longrightarrow & (S_d^{++}(\mathbb{R}))^K \\ (\theta, X) & \longmapsto & \left(\frac{\sum_{i=1}^n \gamma_{i,k}(\theta, X)(x_i - F_{\mathbf{m}}(\theta, X)_k)(x_i - F_{\mathbf{m}}(\theta, X)_k)^\top}{\sum_{j=1}^n \gamma_{j,k}(\theta, X)}\right)_{k=1}^K \end{cases}. \quad (60)$$

For convenience, let $F_{m_k} := [F_{\mathbf{m}}(\theta, X)]_{k,\cdot}$. We first compute the differential with respect to w:

$$\frac{\partial F_{\mathbf{\Sigma}}}{\partial w}(\theta, X) \in \mathcal{L}\left(\triangle_K^0, (S_d(\mathbb{R}))^K\right) \hookrightarrow \mathbb{R}^{K\times d\times d\times K} : \tag{61}$$

$$\begin{aligned}\left[\frac{\partial F_{\mathbf{\Sigma}}}{\partial w}(\theta, X)\right]_{k,\cdot,\cdot,l} &= \frac{1}{\Gamma_k}\sum_{i=1}^n \left[\frac{\partial \gamma}{\partial w}\right]_{i,k,l} (x_i - F_{m_k})(x_i - F_{m_k})^\top \\ &\quad - \frac{1}{\Gamma_k^2}\sum_{j=1}^n \left[\frac{\partial \gamma}{\partial w}\right]_{j,k,l} \sum_{i=1}^n \gamma_{i,k}(x_i - F_{m_k})(x_i - F_{m_k})^\top \\ &\quad - \frac{1}{\Gamma_k}\sum_{i=1}^n \gamma_{i,k}\left(\left[\frac{\partial F_{\mathbf{m}}}{\partial w}\right]_{k,\cdot,l} (x_i - F_{m_k})^\top + (x_i - F_{m_k})\left[\frac{\partial F_{\mathbf{m}}}{\partial w}\right]_{k,\cdot,l}^\top\right).\end{aligned} \tag{62}$$

For the differential with respect to $\mathbf{m}$, we will use Lemma A.1:

$$\frac{\partial F_{\mathbf{\Sigma}}}{\partial \mathbf{m}}(\theta, X) \in \mathcal{L}\left(\mathbb{R}^{K\times d}, (S_d(\mathbb{R}))^K\right) \hookrightarrow \mathbb{R}^{K\times d\times d\times K\times d} : \tag{63}$$

$$\begin{aligned}\left[\frac{\partial F_{\mathbf{\Sigma}}}{\partial \mathbf{m}}(\theta, X)\right]_{k,\cdot,\cdot,l,\cdot} &= \frac{1}{\Gamma_k}\sum_{i=1}^n (x_i - F_{m_k})(x_i - F_{m_k})^\top \otimes \left[\frac{\partial \gamma}{\partial \mathbf{m}}\right]_{i,k,l,\cdot} \\ &\quad - \frac{1}{\Gamma_k^2}\left(\sum_{i=1}^n \gamma_{i,k}(x_i - F_{m_k})(x_i - F_{m_k})^\top\right) \otimes \sum_{j=1}^n \left[\frac{\partial \gamma}{\partial \mathbf{m}}\right]_{j,k,l,\cdot} \\ &\quad - \frac{1}{\Gamma_k}\sum_{i=1}^n \gamma_{i,k}\left\{\tau_{1,2}\left((x_i - F_{m_k}) \otimes \left[\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{m}}\right]_{k,\cdot,l,\cdot}\right) + (x_i - F_{m_k}) \otimes \left[\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{m}}\right]_{k,\cdot,l,\cdot}\right\}.\end{aligned} \tag{64}$$

In Eq. (64), the transposed term comes from the differential of the outer product Eq. (55), where $\tau_{1,2}(A)_{i,j,\dots} = A_{j,i,\dots}$. For the differential with respect to $\mathbf{\Sigma}$, we use the same method as in Eq. (64):

$$\frac{\partial F_{\mathbf{\Sigma}}}{\partial \mathbf{\Sigma}}(\theta, X) \in \mathcal{L}\left((S_d(\mathbb{R}))^K, (S_d(\mathbb{R}))^K\right) \hookrightarrow \mathbb{R}^{K\times d\times d\times K\times d\times d} : \tag{65}$$

$$\begin{aligned}\left[\frac{\partial F_{\mathbf{\Sigma}}}{\partial \mathbf{\Sigma}}(\theta, X)\right]_{k,\cdot,\cdot,l,\cdot,\cdot} &= \frac{1}{\Gamma_k}\sum_{i=1}^n (x_i - F_{m_k})(x_i - F_{m_k})^\top \otimes \left[\frac{\partial \gamma}{\partial \mathbf{\Sigma}}\right]_{i,k,l,\cdot,\cdot} \\ &\quad - \frac{1}{\Gamma_k^2}\left(\sum_{i=1}^n \gamma_{i,k}(x_i - F_{m_k})(x_i - F_{m_k})^\top\right) \otimes \sum_{j=1}^n \left[\frac{\partial \gamma}{\partial \mathbf{\Sigma}}\right]_{j,k,l,\cdot,\cdot} \\ &\quad - \frac{1}{\Gamma_k}\sum_{i=1}^n \gamma_{i,k}\left\{\tau_{1,2}\left((x_i - F_{m_k}) \otimes \left[\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{\Sigma}}\right]_{k,\cdot,l,\cdot,\cdot}\right) + (x_i - F_{m_k}) \otimes \left[\frac{\partial F_{\mathbf{m}}}{\partial \mathbf{\Sigma}}\right]_{k,\cdot,l,\cdot,\cdot}\right\}.\end{aligned} \tag{66}$$

Finally, for the differential with respect to X, the method is almost identical, with an additional term due to the presence of x_i in $(x_i - F_{m_k})(x_i - F_{m_k})^\top$.

$$\frac{\partial F_{\mathbf{\Sigma}}}{\partial X}(\theta, X) \in \mathcal{L}\left(\mathbb{R}^{n\times d}, (S_d(\mathbb{R}))^K\right) \hookrightarrow \mathbb{R}^{K\times d\times d\times n\times d} : \tag{67}$$

$$\begin{aligned}\left[\frac{\partial F_{\mathbf{\Sigma}}}{\partial X}(\theta, X)\right]_{k,\cdot,\cdot,i,\cdot} &= \frac{1}{\Gamma_k}\sum_{h=1}^n (x_h - F_{m_k})(x_h - F_{m_k})^\top \otimes \left[\frac{\partial \gamma}{\partial X}\right]_{h,k,i,\cdot} \\ &- \frac{1}{\Gamma_k^2}\left(\sum_{h=1}^n \gamma_{h,k}(x_h - F_{m_k})(x_h - F_{m_k})^\top\right) \otimes \sum_{j=1}^n \left[\frac{\partial \gamma}{\partial X}\right]_{j,k,i,\cdot} \\ &- \frac{1}{\Gamma_k}\sum_{h=1}^n \gamma_{h,k}\left\{\tau_{1,2}\left((x_h - F_{m_k}) \otimes \left[\frac{\partial F_{\mathbf{m}}}{\partial X}\right]_{k,\cdot,i,\cdot}\right) + (x_h - F_{m_k}) \otimes \left[\frac{\partial F_{\mathbf{m}}}{\partial X}\right]_{k,\cdot,i,\cdot}\right\} \\ &+ \frac{\gamma_{i,k}}{\Gamma_k}\left\{\tau_{1,2}\Big((x_i - F_{m_k}) \otimes I_d\Big) + (x_i - F_{m_k}) \otimes I_d\right\}. \end{aligned} \tag{68}$$

A.5 Local Minima in (GMM)-OT

A.5.1 Computation of the Local Minima of the Discrete 2-Wasserstein Distance

We provide an explicit local minimum of the energy $\mathcal{E}_3$ defined in Eq. (19). We illustrate the setup in Fig. 14.

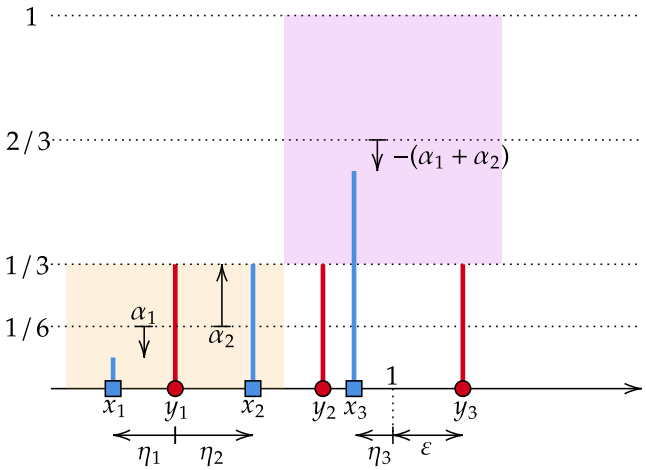


Figure 14: Setup for the local minimum of $\mathrm{W}_2^2(\mu_{\alpha,\eta}, \nu)$. The support of μ is represented with blue squares, and its weights with vertical blue lines. For the target ν, its support is red squares and its weights red lines. We consider a specific region where the points $x_1 = \eta_1$ and $x_2 = \eta_2$ stay within $(-\frac{1}{2}, \frac{1}{2})$ and the weights $a_1 = \frac{1}{6} + \alpha_1$ and $a_2 = \frac{1}{6} + \alpha_2$ stay within $[0, \frac{1}{3}]$, as represented by the orange rectangle. Likewise, the point $x_3 = 1 + \eta_3$ must stay within $(\frac{1}{2}, \frac{3}{2})$ and its weight $a_3 = \frac{2}{3} - (\alpha_1 + \alpha_2)$ must stay in $[\frac{1}{3}, 1]$, as shown with the purple rectangle.

To compute the energy $\mathcal{E}_3(\alpha, \eta)$ at $[-\frac{1}{6}, \frac{1}{6}]^2 \times (-\frac{1}{2}, \frac{1}{2})^3$, we must distinguish the two cases $\alpha_1 + \alpha_2 \geq 0$ and $\alpha_1 + \alpha_2 < 0$, which yield two different OT plans between $\mu_{\alpha,\eta}$ and ν. We will show separately that the energy $\mathcal{E}_3$ has a strict local minimum at $(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$ in both sub-regions. To break the symmetry in $(\alpha_1, \alpha_2, \eta_1, \eta_2) \leftarrow (\alpha_2, \alpha_1, \eta_2, \eta_1)$, we will impose the constraint $\eta_1 \leq \eta_2$ in the optimisation problem. To summarise, we split the optimisation problem in two as follows:

$$\min_{\substack{\alpha\in[-\frac{1}{6},\frac{1}{6}]^2 \\ \eta\in(-\frac{1}{2},\frac{1}{2})^3}} \mathcal{E}_3(\alpha,\eta) = \min(\mathcal{E}_3^+, \mathcal{E}_3^-), \quad \mathcal{E}_3^+ := \min_{\substack{\alpha\in[-\frac{1}{6},\frac{1}{6}]^2 \\ \eta\in(-\frac{1}{2},\frac{1}{2})^3 \\ \alpha_1+\alpha_2\geq 0 \\ \eta_1\leq\eta_2}} \mathcal{E}_3(\alpha,\eta), \quad \mathcal{E}_3^- := \min_{\substack{\alpha\in[-\frac{1}{6},\frac{1}{6}]^2 \\ \eta\in(-\frac{1}{2},\frac{1}{2})^3 \\ \alpha_1+\alpha_2\leq 0 \\ \eta_1\leq\eta_2}} \mathcal{E}_3(\alpha,\eta). \tag{69}$$

Case $\alpha_1 + \alpha_2 \geq 0$. For any $(\alpha, \eta) \in [-\frac{1}{6}, \frac{1}{6}]^2 \times (-\frac{1}{2}, \frac{1}{2})^3$ such that $\alpha_1 + \alpha_2 \geq 0$ and $\eta_1 \leq \eta_2$, the optimal plan between $\mu_{\alpha,\eta}$ and ν is given by:

$$\pi^+(\alpha, \eta) = \begin{pmatrix} \frac{1}{6} + \alpha_1 & 0 & 0 \\ \frac{1}{6} - \alpha_1 & \alpha_1 + \alpha_2 & 0 \\ 0 & \frac{1}{3} - \alpha_1 - \alpha_2 & \frac{1}{3} \end{pmatrix},$$

and thus the energy $\mathcal{E}_3(\alpha, \eta)$ is given by:

$$\mathcal{E}_3(\alpha, \eta) = \left(\tfrac{1}{6} + \alpha_1\right)\eta_1^2 + \left(\tfrac{1}{6} - \alpha_1\right)\eta_2^2 + (\alpha_1 + \alpha_2)\,(1 - \varepsilon - \eta_2)^2 + \left(\tfrac{1}{3} - \alpha_1 - \alpha_2\right)(\eta_3 + \varepsilon)^2 + \tfrac{1}{3}(\eta_3 - \varepsilon)^2. \tag{70}$$

Taking dual variables $\lambda_1, \lambda_2 \in \mathbb{R}$, the Lagrangian of the problem $\mathcal{E}_3^+$ defined in Eq. (69) is given by:

$$\mathcal{L}^+(\alpha, \eta, \lambda_1, \lambda_2) = \mathcal{E}_3(\alpha, \eta) + \lambda_1(-\alpha_1 - \alpha_2) + \lambda_2(\eta_1 - \eta_2). \tag{71}$$

To find solutions of the problem $\mathcal{E}_3^+$, we study the necessary KKT system (see [BV04, Section 5.5.3], or [NW06, Theorem 12.1]). We begin with the stationarity condition, expressed at the point $(\alpha, \eta) \in [-\frac{1}{6}, \frac{1}{6}]^2 \times (-\frac{1}{2}, \frac{1}{2})^3$:

$$\begin{aligned}
0 &= \frac{\partial \mathcal{L}^+}{\partial \alpha_1} = \eta_1^2 - \eta_2^2 + (1 - \varepsilon - \eta_2)^2 - (\eta_3 + \varepsilon)^2 - \lambda_1; \\
0 &= \frac{\partial \mathcal{L}^+}{\partial \alpha_2} = (1 - \varepsilon - \eta_2)^2 - (\eta_3 + \varepsilon)^2 - \lambda_1; \\
0 &= \frac{\partial \mathcal{L}^+}{\partial \eta_1} = 2\left(\tfrac{1}{6} + \alpha_1\right)\eta_1 + \lambda_2; \\
0 &= \frac{\partial \mathcal{L}^+}{\partial \eta_2} = 2\left(\tfrac{1}{6} - \alpha_1\right)\eta_2 - 2(\alpha_1 + \alpha_2)(1 - \varepsilon - \eta_2) - \lambda_2; \\
0 &= \frac{\partial \mathcal{L}^+}{\partial \eta_3} = 2\left(\tfrac{1}{3} - \alpha_1 - \alpha_2\right)(\eta_3 + \varepsilon) + \tfrac{2}{3}(\eta_3 - \varepsilon).
\end{aligned}$$

The remaining KKT conditions are the primal / dual feasibility conditions, which are given by:

$$\alpha_1 + \alpha_2 \geq 0,\ \eta_1 \leq \eta_2,\ \lambda_1 \geq 0,\ \lambda_2 \geq 0,$$

and the complementary slackness conditions:

$$\lambda_1(-\alpha_1 - \alpha_2) = 0,\ \lambda_2(\eta_1 - \eta_2) = 0.$$

It is easy to see that the point $(\alpha, \eta, \lambda_1, \lambda_2) = (0_{\mathbb{R}^2}, 0_{\mathbb{R}^3}, 1 - 2\varepsilon, 0)$ is a solution of the KKT system (note that $\varepsilon < \frac{1}{2}$).

We now check that our critical point is a local minimum. For $\alpha \in [-\frac{1}{6}, \frac{1}{6}]^2$, $\eta \in (-\frac{1}{2}, \frac{1}{2})^3$ verifying $\alpha_1 + \alpha_2 \geq 0$, we compute the Hessian of $\mathcal{E}_3$:

$$H^+(\alpha, \eta) = \begin{pmatrix} 0 & 0 & 2\eta_1 & 2\varepsilon - 2 & -2\eta_3 - 2\varepsilon \\ 0 & 0 & 0 & 2\eta_2 + 2\varepsilon - 2 & -2\eta_3 - 2\varepsilon \\ 2\eta_1 & 0 & 2\alpha_1 + \frac{1}{3} & 0 & 0 \\ 2\varepsilon - 2 & 2\eta_2 + 2\varepsilon - 2 & 0 & 2\alpha_2 + \frac{1}{3} & 0 \\ -2\eta_3 - 2\varepsilon & -2\eta_3 - 2\varepsilon & 0 & 0 & -2\alpha_1 - 2\alpha_2 + \frac{4}{3} \end{pmatrix}. \tag{72}$$

Using numerical solvers, we obtain that at $(\alpha, \eta) = (0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$, the Hessian verifies the following property:

$$H^+(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})v = 0,\ v := (1, -1, 0, 0, 0),\ \forall w \in v^\perp,\ w^\top H^+(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})w > 0. \tag{73}$$

Adding the vector tv to (α, η) for $|t|$ small enough corresponds to adding t to α_1 and subtracting t from α_2, and we notice that at $(\alpha, \eta) = (0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$, since $\eta_1 = \eta_2$, this operation does not change the cost: $\mathcal{E}_3((t, -t), 0_{\mathbb{R}^3}) = \mathcal{E}_3(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$. We conclude that $(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$ is a local minimum for the problem $\mathcal{E}_3^+$ defined in Eq. (69).

Case $\alpha_1 + \alpha_2 \leq 0$. For any $(\alpha, \eta) \in [-\frac{1}{6}, \frac{1}{6}]^2 \times (-\frac{1}{2}, \frac{1}{2})^3$ such that $\alpha_1 + \alpha_2 \leq 0$ and $\eta_1 \leq \eta_2$, the optimal plan between $\mu_{\alpha,\eta}$ and ν and the associated energy $\mathcal{E}_3$ are given by:

$$\pi^-(\alpha, \eta) = \begin{pmatrix} \frac{1}{6} + \alpha_1 & 0 & 0 \\ \frac{1}{6} + \alpha_2 & 0 & 0 \\ -\alpha_1 - \alpha_2 & \frac{1}{3} & \frac{1}{3} \end{pmatrix},$$

$$\mathcal{E}_3(\alpha, \eta) = \left(\tfrac{1}{6} + \alpha_1\right) \eta_1^2 + \left(\tfrac{1}{6} + \alpha_2\right) \eta_2^2 - (\alpha_1 + \alpha_2)\,(1 + \eta_3)^2 + \tfrac{1}{3}\,(\eta_3 + \varepsilon)^2 + \tfrac{1}{3}(\eta_3 - \varepsilon)^2.$$

We deduce the expression of the Lagrangian of the problem $\mathcal{E}_3^-$ defined in Eq. (69):

$$\mathcal{L}^-(\alpha, \eta, \lambda_1, \lambda_2) = \mathcal{E}_3(\alpha, \eta) + \lambda_1(\alpha_1 + \alpha_2) + \lambda_2(\eta_1 - \eta_2). \tag{74}$$

The KKT stationarity condition writes then:

$$\begin{aligned}
0 &= \frac{\partial \mathcal{L}^-}{\partial \alpha_1} = \eta_1^2 - (1 + \eta_3)^2 + \lambda_1; \\
0 &= \frac{\partial \mathcal{L}^-}{\partial \alpha_2} = \eta_2^2 - (1 + \eta_3)^2 + \lambda_1; \\
0 &= \frac{\partial \mathcal{L}^-}{\partial \eta_1} = 2\left(\tfrac{1}{6} + \alpha_1\right) \eta_1 + \lambda_2; \\
0 &= \frac{\partial \mathcal{L}^-}{\partial \eta_2} = 2\left(\tfrac{1}{6} - \alpha_1\right) \eta_2 - \lambda_2; \\
0 &= \frac{\partial \mathcal{L}^-}{\partial \eta_3} = -2\,(\alpha_1 + \alpha_2)\,(1 + \eta_3) + \tfrac{4}{3}\eta_3.
\end{aligned}$$

Again the primal / dual feasibility conditions read:

$$\alpha_1 + \alpha_2 \leq 0,\ \eta_1 \leq \eta_2,\ \lambda_1 \geq 0,\ \lambda_2 \geq 0,$$

and the complementary slackness conditions:

$$\lambda_1(\alpha_1 + \alpha_2) = 0,\ \lambda_2(\eta_1 - \eta_2) = 0.$$

Likewise, the point $(\alpha, \eta, \lambda_1, \lambda_2) = (0_{\mathbb{R}^2}, 0_{\mathbb{R}^3}, 1, 0)$ is a solution of the KKT system.

To prove local minimality, we compute the Hessian of $\mathcal{E}_3$ at $\alpha \in [-\frac{1}{6}, \frac{1}{6}]^2$, $\eta \in (-\frac{1}{2}, \frac{1}{2})^3$ verifying $\alpha_1 + \alpha_2 \leq 0$:

$$H^-(\alpha, \eta) = \begin{pmatrix} 0 & 0 & 2\eta_1 & 0 & -2\eta_3 - 2 \\ 0 & 0 & 0 & 2\eta_2 & -2\eta_3 - 2 \\ 2\eta_1 & 0 & 2\alpha_1 + \frac{1}{3} & 0 & 0 \\ 0 & 2\eta_2 & 0 & 2\alpha_2 + \frac{1}{3} & 0 \\ -2\eta_3 - 2 & -2\eta_3 - 2 & 0 & 0 & -2\alpha_1 - 2\alpha_2 + \frac{4}{3} \end{pmatrix}.$$

The property of $H^+(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$ from Eq. (73) is also satisfied by $H^-(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$, and we conclude likewise that $(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$ is a local minimum for the problem $\mathcal{E}_3^-$ defined in Eq. (69).

We conclude that $(\alpha, \eta) = (0_{\mathbb{R}^2}, 0_{\mathbb{R}^3})$ is a minimum of $\mathcal{E}_3$ on the set $[-\frac{1}{6}, \frac{1}{6}]^2 \times (-\frac{1}{2}, \frac{1}{2})^3$, with value $\mathcal{E}_3(0_{\mathbb{R}^2}, 0_{\mathbb{R}^3}) = \frac{2}{3}\varepsilon^2 > 0$ (use Eq. (70) for example).

A.5.2 Discrete 2-Wasserstein Distance for $n = 2$: Proof of Unique Local Minimum

Unique Local Minimum of the Discrete 2-Wasserstein Distance for $n = 2$. We consider the minimisation of the quadratic Wasserstein cost $\mu \longmapsto \mathrm{W}_2^2(\mu, \nu)$ when μ is restricted to two Dirac atoms,

$$\mu_{x,\alpha} = \alpha\delta_{x_1} + (1 - \alpha)\delta_{x_2}, \qquad (x, \alpha) \in \mathbb{R}^{2d} \times [0, 1],$$

against the fixed target $\nu = \gamma\delta_{y_1} + (1-\gamma)\delta_{y_2}$ with $\gamma \in (0,1)$ and $y_1 \neq y_2$. Denoting

$$\mathcal{E}_2(x,\alpha) := \mathrm{W}_2^2(\mu_{x,\alpha}, \nu), \tag{75}$$

we show that inside the domain $\mathcal{D} := \{(x,\alpha) : x_1 \neq x_2,\ \alpha \in (0,1)\}$ every local minimiser of $\mathcal{E}_2$ is such that $\mu_{x,\alpha} = \nu$, and is thus a global minimiser. We show this result more generally, for costs $c(x,y) := \phi(x-y)$ where ϕ is convex, $\mathcal{C}^1$, $\phi(v) = 0 \iff v = 0$ and $\nabla\phi(v) = 0 \iff v = 0$. Additionally, boundary stationary points can occur when atoms merge, i.e. $\alpha \in \{0,1\}$ or $x_1 = x_2$; they are local but not global minima. When these weights arise from EM, we can prevent such degeneracies by fixing weights or imposing a hard lower bound on them throughout the iterations.

We now determine all the local minimisers of the energy $\mathcal{E}_2$ defined in Eq. (75) in the domain $\mathcal{D} := \{(x,\alpha) : x_1 \neq x_2,\ \alpha \in (0,1)\}$. Fix $\gamma \in (0,1)$, and let x_1, x_2, y_1 and y_2 in $\mathbb{R}^d$, with $y_1 \neq y_2$. For $\alpha \in (0,1)$, we consider the measures:

$$\mu(x_1, x_2, \alpha) := \alpha\delta_{x_1} + (1-\alpha)\delta_{x_2}, \quad \nu := \gamma\delta_{y_1} + (1-\gamma)\delta_{y_2}.$$

Consider the ground cost $c := (x,y) \in \mathbb{R}^d \times \mathbb{R}^d \longmapsto \phi(x-y)$, with:

- (H1) $\phi : \mathbb{R}^d \to \mathbb{R}_+$ convex and $\mathcal{C}^1$,
- (H2) $\phi(v) = 0 \iff v = 0$,
- (H3) $\nabla\phi(v) = 0 \iff v = 0$.

For instance, ϕ could be $\|\cdot\|_p^p$ with $p > 1$. Every feasible coupling between $\mu(x_1, x_2, \alpha)$ and ν can be written under the form:

$$\pi(\alpha, t) := \begin{pmatrix} \alpha - t & t \\ \gamma - \alpha + t & 1 - \gamma - t \end{pmatrix},\ t \in [t_-(\alpha), t_+(\alpha)],\ t_-(\alpha) := \max(0, \alpha-\gamma),\ t_+(\alpha) := \min(\alpha, 1-\gamma).$$

The interval $[t_-(\alpha), t_+(\alpha)]$ is non-empty for all $\alpha \in (0,1)$, since $\gamma \in (0,1)$. The transport cost associated to a plan $\pi(\alpha,t)$ writes:

$$\mathcal{F}(x,\alpha,t) := (\alpha - t)c(x_1,y_2) + (\gamma - \alpha + t)c(x_2,y_2) + tc(x_1,y_2) + (1-\gamma-t)c(x_2,y_2). \tag{76}$$

Since $\mathcal{E}_2(x_1,x_2,\alpha) = \min_{t\in[t_-(\alpha),t_+(\alpha)]} \mathcal{F}(x_1,x_2,\alpha,t)$ and that $\mathcal{F}$ is linear in t, we obtain:

$$\begin{aligned} \mathcal{E}_2(x_1,x_2,\alpha) &= \min\left(\mathcal{E}^-(x_1,x_2,\alpha), \mathcal{E}^+(x_1,x_2,\alpha)\right), \\ \mathcal{E}^-(x_1,x_2,\alpha) &:= \mathcal{F}(x_1,x_2,\alpha,t_-(\alpha)),\ \mathcal{E}^+(x_1,x_2,\alpha) := \mathcal{F}(x_1,x_2,\alpha,t_+(\alpha)). \end{aligned} \tag{77}$$

In particular, any local optimum of $\mathcal{E}_2$ in $\mathcal{D} := \{(x_1,x_2,\alpha) \in \mathbb{R}^d \times \mathbb{R}^d \times (0,1) :\ x_1 \neq x_2\}$ must be a local optimum of $\mathcal{E}^-$ or $\mathcal{E}^+$. Straightforward computation yields the following symmetrical expression:

$$\forall x_1, x_2 \in \mathbb{R}^d,\ \forall \alpha \in (0,1),\ \forall t \in \mathbb{R},\ \mathcal{F}(x_1,x_2,\alpha,t) = \mathcal{F}(x_2,x_1,1-\alpha,1-\gamma-t),$$

which can be understood as exchanging the roles of x_1 and x_2. For any $\alpha \in (0,1)$, we have $1-\gamma-t_-(\alpha) = t_+(\alpha)$, concluding a symmetrical relationship between $\mathcal{E}^-$ and $\mathcal{E}^+$:

$$\forall x_1, x_2 \in \mathbb{R}^d,\ \forall \alpha \in (0,1),\ \mathcal{E}^+(x_2,x_1,1-\alpha) = \mathcal{E}^-(x_1,x_2,\alpha). \tag{78}$$

As a consequence, any local optimum $(x_1,x_2,\alpha) \in \mathcal{D}$ of $\mathcal{E}^+$ is such that $(x_2,x_1,1-\alpha)$ is a local optimum of $\mathcal{E}^-$, and conversely. To determine the local minima of $\mathcal{E}_2$ in $\mathcal{D}$, we can focus on the local minima of $\mathcal{E}^-$ in $\mathcal{D}$. We split $\mathcal{D}$ intro three sub-regions wherein $\mathcal{E}^-$ has an explicit expression: consider

$$\mathcal{R}_< := \{0 < \alpha < \gamma\} \cap \mathcal{D}, \quad \mathcal{R}_= := \{\alpha = \gamma\} \cap \mathcal{D}, \quad \mathcal{R}_> := \{\gamma < \alpha < 1\} \cap \mathcal{D},$$

we have the following expressions for $\mathcal{E}^-$ at $(x_1,x_2,\alpha) \in \mathcal{D}$:

$$\mathcal{E}^-(x_1,x_2,\alpha) = \begin{cases} \alpha c(x_1,y_1) + (\gamma-\alpha)c(x_2,y_1) + (1-\gamma)c(x_2,y_2) & \text{if } (x_1,x_2,\alpha) \in \mathcal{R}_< \cup \mathcal{R}_= \\ \gamma c(x_1,y_1) + (\alpha-\gamma)c(x_1,y_2) + (1-\alpha)c(x_2,y_2) & \text{if } (x_1,x_2,\alpha) \in \mathcal{R}_= \cup \mathcal{R}_> \end{cases}. \tag{79}$$

No local minimum in $\mathcal{R}_<$. For $(x_1, x_2, \alpha) \in \mathcal{R}_<$, consider

$$\mathcal{E}^-_<(x_1, x_2, \alpha) := \alpha c(x_1, y_1) + (\gamma - \alpha)c(x_2, y_1) + (1 - \gamma)c(x_2, y_2).$$

A local optimum $(x_1, x_2, \alpha) \in \mathcal{R}_<$ of $\mathcal{E}^-$ must satisfy the stationarity conditions:

$$\begin{aligned}
0 = \frac{\partial \mathcal{E}^-_<}{\partial x_1} &= \alpha \nabla\phi(x_1 - y_1) && \underset{\text{(H3)}}{\Longrightarrow} && x_1 = y_1, \\
0 = \frac{\partial \mathcal{E}^-_<}{\partial \alpha} &= \underbrace{c(x_1, y_1)}_{=0 \text{ by (H2)}} - c(x_2, y_1) && \underset{\text{(H2)}}{\Longrightarrow} && x_2 = y_1. \\
0 = \frac{\partial \mathcal{E}^-_<}{\partial x_2} &= (\gamma - \alpha) \underbrace{\nabla\phi(x_2 - y_1)}_{=0 \text{ by (H2)}} + (1 - \gamma)\nabla\phi(\underbrace{x_2}_{=y_1} - y_2) = (1 - \gamma)\nabla\phi(y_1 - y_2).
\end{aligned}$$

By (H3), since $y_1 \neq y_2$, we have $\nabla c(y_1, y_2) \neq 0$, hence $\mathcal{E}_-$ has no local minimum in $\mathcal{R}_<$.

No local minimum in $\mathcal{R}_>$. For $(x_1, x_2, \alpha) \in \mathcal{R}_>$, consider

$$\mathcal{E}^-_>(x_1, x_2, \alpha) := \gamma c(x_1, y_1) + (\alpha - \gamma)c(x_1, y_2) + (1 - \alpha)c(x_2, y_2).$$

Likewise, a local minimum $(x_1, x_2, \alpha) \in \mathcal{R}_>$ of $\mathcal{E}^-$ must verify:

$$\begin{aligned}
0 = \frac{\partial \mathcal{E}^-_>}{\partial x_1} &= (1 - \alpha)\nabla\phi(x_2 - y_2) && \underset{\text{(H3)}}{\Longrightarrow} && x_2 = y_2, \\
0 = \frac{\partial \mathcal{E}^-_>}{\partial \alpha} &= c(x_1, y_2) - \underbrace{c(x_2, y_2)}_{=0 \text{ by (H2)}} && \underset{\text{(H2)}}{\Longrightarrow} && x_1 = y_2. \\
0 = \frac{\partial \mathcal{E}^-_>}{\partial x_2} &= \gamma\nabla\phi(\underbrace{x_1}_{=y_2} - y_1) + (\alpha - \gamma) \underbrace{\nabla\phi(x_2 - y_2)}_{=0 \text{ by (H2)}} = \gamma\nabla\phi(y_2 - y_1) \neq 0.
\end{aligned}$$

As before, this system cannot hold, and thus $\mathcal{E}_-$ has no local minimum in $\mathcal{R}_>$.

Analysis on $\mathcal{R}_=$. For $(x_1, x_2, \alpha) \in \mathcal{R}_=$, we have:

$$\mathcal{E}^-(x_1, x_2, \alpha) = \mathcal{E}^-_=(x_1, x_2) := \gamma c(x_1, y_1) + (1 - \gamma)c(x_2, y_2),$$

thus a local minimum $(x_1, x_2, \alpha) \in \mathcal{R}_=$ of $\mathcal{E}^-$ must satisfy the stationarity conditions:

$$\begin{aligned}
0 = \frac{\partial \mathcal{E}^-_=}{\partial x_1} &= \gamma\nabla\phi(x_1 - y_1) && \underset{\text{(H3)}}{\Longrightarrow} && x_1 = y_1, \\
0 = \frac{\partial \mathcal{E}^-_=}{\partial x_2} &= (1 - \gamma)\nabla\phi(x_2 - y_2) && \underset{\text{(H3)}}{\Longrightarrow} && x_2 = y_2,
\end{aligned}$$

Since $(x_1, x_2, \alpha) = (y_1, y_2, \gamma)$ is a global minimum of $\mathcal{E}^-_=$, we conclude that the only local minimum of $\mathcal{E}^-$ in $\mathcal{D}$ is (y_1, y_2, γ).

Using Eq. (78), we deduce that the only local minimum of $\mathcal{E}^+$ in $\mathcal{D}$ is $(y_2, y_1, 1-\gamma)$, which is a global minimum of $\mathcal{E}_2$ in $\mathcal{D}$. Finally, there are two local minima of $\mathcal{E}_2$ in $\mathcal{D}$, which are the two global minima corresponding to $\mu(x_1, x_2, \alpha) = \nu$.

Local minimum for a single-Dirac source, and how to avoid it. The constraint $(x_1, x_2, \alpha) \in \mathcal{D}$ imposes in particular that $\mu(x_1, x_2, \alpha)$ is composed of two distinct Dirac masses. We now focus on the pathological case where $\alpha = 0$, showing the existence of a local optimum. For simplicity, we consider $d = 1$ and the cost $c(x, y) := |x - y|^2$. Set

$$z^\star := (x_1^\star, x_2^\star, \alpha^\star) := (-1, 1 - \gamma, 0),\ y_1 := 0,\ y_2 := 1,\ \gamma \in (0, 1).$$

For (x_1, x_2, α) in an open vicinity of $z^\star$, the cost simplifies to

$$\mathcal{E}_2(x_1, x_2, \alpha) = \alpha x_1^2 + (\gamma - \alpha)x_2^2 + (1-\gamma)(x_2 - 1)^2.$$

To show that $z^\star$ is a local minimum, we compute the gradient of $\mathcal{E}_2$ at $z^\star$:

$$\frac{\partial \mathcal{E}_2}{\partial x_1}(z^\star) = 2\alpha^\star x_1^\star = 0,$$

$$\frac{\partial \mathcal{E}_2}{\partial x_2}(z^\star) = 2\left(x_2^\star - (1-\gamma)\right) = 0,$$

$$\frac{\partial \mathcal{E}_2}{\partial \alpha}(z^\star) = x_1^2 - x_2^2 = 1 - (1-\gamma)^2 > 0.$$

Consider a perturbation $h := (h_1, h_2, h_3) \in \mathbb{R}^3$ with $h_3 > 0$ (as $\alpha \geq 0$). For $z := z^\star + h$, we have:

$$\mathcal{E}_2(z) = \mathcal{E}_2(z^\star) + \underbrace{\frac{\partial \mathcal{E}_2}{\partial \alpha}(z^\star)h_3}_{>0} + \mathcal{O}(\|h\|^2),$$

so every feasible perturbation increases the objective. Thus $z^\star$ is a strict local minimiser, but is not global. When α arises as a mixture weight in EM, two remedies are possible: fixing uniform weights, or enforcing $\alpha \geq \varepsilon > 0$: both methods sidestep the degenerate local minimum above.

A.5.3 Essential Stationary Points for the MW2-EM Loss: Computations

We illustrate a phenomenon of points where the gradients are infinitesimally small on a simple example with two Gaussian components, and studying a variant of the EM algorithm that does not update the covariances for simplicity. We fix $\varepsilon > 0$ and the following parametrised input GMM:

$$\forall \alpha \in (0,1),\ \forall m_1, m_2 \in \mathbb{R},\ \mu(\alpha, m_1, m_2) := \alpha\mathcal{N}(m_1, \varepsilon^2) + (1-\alpha)\mathcal{N}(m_2, \varepsilon^2),$$

and for $w := \frac{2}{3}$, the target GMM: $\nu := w\mathcal{N}(0, \varepsilon^2) + (1-w)\mathcal{N}(1, \varepsilon^2)$. We consider the particular dataset $X_\varepsilon \in \mathbb{R}^{6\times 1}$ defined by:

$$X_\varepsilon := (x_1, \cdots, x_6),\ x_1 := -\varepsilon,\ x_2 := \varepsilon,\ x_3 := x_5 := m^\star - \varepsilon,\ x_4 := x_6 := m^\star + \varepsilon.$$

We define the GMM $\mu^\star := (1-w)\mathcal{N}(0, \varepsilon^2) + w\mathcal{N}(m^\star, \varepsilon^2)$ associated to the vanishing gradient point, and introduce $\theta^\star := ((1-w), 0, m^\star)$ its parameters. For any $X \in \mathbb{R}^{6\times 1}$, we denote by $\theta(X)$ the parameters of the GMM fitted by the EM algorithm in one iteration on X with initialisation $\theta^\star$: $\theta(X) := F(\theta^\star, X)$ (we remind that in this section, we consider a simplified EM that does not update covariances). We shall first see that $\theta(X_\varepsilon) \approx \theta^\star$, then that the energy

$$\mathcal{E}_{\mathrm{EM}-\mathrm{MW}_2^2} := X \longmapsto \mathrm{MW}_2^2(\mu(\theta(X)), \nu), \quad \mu(\theta(X)) := \alpha(X)\mathcal{N}(m_1(X), \varepsilon^2) + (1-\alpha(X))\mathcal{N}(m_2(X), \varepsilon^2),$$

verifies $\partial_X \mathcal{E}_{\mathrm{EM}-\mathrm{MW}_2^2}(X_\varepsilon) \approx 0$. We summarise our setting in Fig. 15.

Showing that $\theta(X_\varepsilon) \approx \theta^\star$. Using the expressions of the EM update from Eq. (3), we obtain that responsibilities $\gamma(X_\varepsilon)$ computed with initialisation $\theta^\star$ verify:

$$\forall i \in \{1,2\},\ \gamma_{i,1}(X_\varepsilon) = 1 + \mathcal{O}(e^{-1/\varepsilon^2}),\ \forall i \in \{3,4,5,6\},\ \gamma_{i,2}(X_\varepsilon) = 1 + \mathcal{O}(e^{-1/\varepsilon^2}),$$

as $\varepsilon \longrightarrow 0^+$. This means that the two points x_1, x_2 are considered as belonging to the first component $\mathcal{N}(0, \varepsilon^2)$ of $\mu^\star$, and the four points x_3, x_4, x_5, x_6 to the second component $\mathcal{N}(m^\star, \varepsilon^2)$ of $\mu^\star$. As for the weight $\alpha(X_\varepsilon)$ and means $m(X_\varepsilon)$ of the GMM $F(\theta^\star, X^\star)$, we use Eq. (4) to deduce that:

$$\alpha(X_\varepsilon) = (1-w) + \mathcal{O}(e^{-1/\varepsilon^2}),\ m_1(X_\varepsilon) = 0 + \mathcal{O}(e^{-1/\varepsilon^2}),\ m_2(X_\varepsilon) = m^\star + \mathcal{O}(e^{-1/\varepsilon^2}). \tag{80}$$

In this sense, we have $\theta(X_\varepsilon) \approx \theta^\star$ as $\varepsilon \longrightarrow 0^+$.

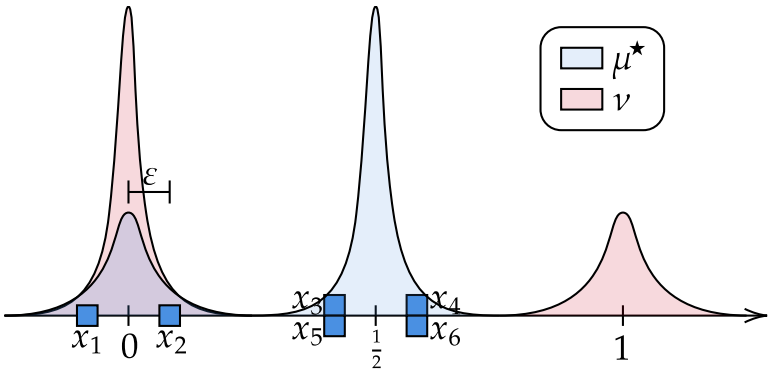


Figure 15: With the data $X_\varepsilon := (x_1, \cdots, x_6)$, one iteration of the EM algorithm initialised at $\theta^\star$ (corresponding to the GMM $\mu^\star$) yields approximately the same parameters $\theta^\star$. We shall see that the gradient of the energy $\mathcal{E}_{\mathrm{EM-MW}_2^2}$ at X_ε is approximately zero, illustrating the vanishing gradient phenomenon.

Vanishing gradient of the energy $\mathcal{E}_{\mathrm{EM-MW}_2^2}$ at X_ε. For $\varepsilon > 0$ sufficiently small and X in a sufficiently small open vicinity of X_ε, it follows by regularity of F that $\theta(X) = (\alpha(X), m_1(X), m_2(X))$ will be sufficiently close to $\theta(X_\varepsilon)$ to ensure that $\alpha(X) < w$ and that $m_1(X) < m_2(X)$, which yields (by property on one-dimensional OT) the following expression for the energy:

$$\mathcal{E}_{\mathrm{EM-MW}_2^2}(X) := \mathrm{MW}_2^2(\mu(\theta(X)), \nu) = \mathcal{F}(\alpha(X), m_1(X), m_2(X)),$$

where $\mathcal{F} : (0,1) \times \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$ is the function defined by:

$$\forall (\alpha, m_1, m_2) \in (0,1) \times \mathbb{R} \times \mathbb{R},\ \mathcal{F}(\alpha, m_1, m_2) := \alpha m_1^2 + (w - \alpha) m_2^2 + (1 - w)(m_2 - 1)^2.$$

To determine the gradient of $\mathcal{E}_{\mathrm{EM-MW}_2^2}$ at X_ε, we use the chain rule:

$$\nabla \mathcal{E}_{\mathrm{EM-MW}_2^2}(X_\varepsilon) = \frac{\partial \mathcal{F}}{\partial \alpha}(\theta(X_\varepsilon)) \frac{\partial \alpha}{\partial X}(X_\varepsilon) + \frac{\partial \mathcal{F}}{\partial m_1}(\theta(X_\varepsilon)) \frac{\partial m_1}{\partial X}(X_\varepsilon) + \frac{\partial \mathcal{F}}{\partial m_2}(\theta(X_\varepsilon)) \frac{\partial m_2}{\partial X}(X_\varepsilon).$$

Differentiating $\mathcal{F}$ and evaluating at $\theta(X_\varepsilon)$ which verifies the properties given in Eq. (80) yields:

$$\frac{\partial \mathcal{F}}{\partial \alpha}(\theta(X_\varepsilon)) = -m^\star + \mathcal{O}(e^{-1/\varepsilon^2}) = \mathcal{O}(1),\ \frac{\partial \mathcal{F}}{\partial m_1}(\theta(X_\varepsilon)) = \mathcal{O}(e^{-1/\varepsilon^2}),\ \frac{\partial \mathcal{F}}{\partial m_2}(\theta(X_\varepsilon)) = \mathcal{O}(e^{-1/\varepsilon^2}).$$

Using the expression of $\frac{\partial \alpha}{\partial X}$ from Eq. (45) with $X := X_\varepsilon$ and $\theta := \theta^\star$ yields $\frac{\partial \alpha}{\partial X}(X_\varepsilon) = \mathcal{O}(\varepsilon^{-2} e^{-1/\varepsilon^2})$.

With the expressions of $\frac{\partial m_1}{\partial X}$ and $\frac{\partial m_2}{\partial X}$ computed in Eq. (59), we obtain $\frac{\partial m_1}{\partial X}(X_\varepsilon) = \mathcal{O}(1)$ and $\frac{\partial m_2}{\partial X}(X_\varepsilon) = \mathcal{O}(1)$. Putting everything together, we conclude that the gradient vanishes:

$$\nabla \mathcal{E}_{\mathrm{EM-MW}_2^2}(X_\varepsilon) = \mathcal{O}(\varepsilon^{-2} e^{-1/\varepsilon^2}).$$

A.6 Differentiating the Matrix Square Root

We consider the (differentiable) matrix square root function:

$$R := \left\{ \begin{array}{ccc} S_d^{++}(\mathbb{R}) & \longrightarrow & S_d^{++}(\mathbb{R}) \\ A & \longmapsto & \sqrt{A} \end{array} \right. ,$$

and provide a formula for its differential which is useful for numerical automatic differentiation.

Proposition A.2. Let $A \in S_d^{++}(\mathbb{R})$ and $H \in S_d(\mathbb{R})$. Then the differential of the matrix square

root at A in the direction H is given by the following matrix in $S_d(\mathbb{R})$:

$$\mathrm{d}_A R(H) = PGP^\top, \quad [G]_{i,j} := \frac{[P^\top HP]_{i,j}}{\sqrt{\lambda_i} + \sqrt{\lambda_j}},$$

where the orthonormal decomposition of A is given by $A = P\,\mathrm{diag}(\lambda_1, \cdots, \lambda_d)P^\top$.

Proof. For any $A \in S_d^{++}(\mathbb{R})$, we have by definition $R(A)R(A) = A$, and differentiating this identity at A in the direction H yields that $\mathrm{d}_A R(H)$ is a solution of the following Sylvester equation:

$$R(A)X + XR(A) = H. \tag{81}$$

By [Bha13, Theorem VII.2.1], Eq. (81) has a unique solution in $\mathbb{R}^{d\times d}$, which is therefore $\mathrm{d}_A R(H)$. Using the notation of the result statement, we consider the symmetric matrix $X := PGP^\top$ and notice that by definition of G, we have:

$$\forall i, j \in [\![1, d]\!],\ \sqrt{\lambda_i} G_{i,j} + G_{i,j}\sqrt{\lambda_j} = [P^\top HP]_{i,j},$$

introducing $D := \mathrm{diag}(\lambda_1, \cdots, \lambda_d)$, we deduce that $R(D)G + GR(D) = P^\top HP$ and then that X is indeed a solution of Eq. (81), hence by uniqueness $X = \mathrm{d}_A R(H)$. □

Below, we provide a PyTorch [Pas+19] implementation of this gradient computation, allowing for automatic gradient propagation.

```
import torch

class MatrixSquareRoot(torch.autograd.Function):
    @staticmethod
    def forward(ctx, A):
        A_sym = .5 * (A + A.transpose(-2, -1))
        L, P = torch.linalg.eigh(A_sym)
        S = L.clamp_min(0).sqrt()
        R = (P * S.unsqueeze(-2)) @ P.transpose(-2,-1)
        ctx.save_for_backward(P, S)
        return R

    @staticmethod
    def backward(ctx, H):
        P, S = ctx.saved_tensors
        H_sym = .5 * (H + H.transpose(-2, -1))
        D = S.unsqueeze(-1) + S.unsqueeze(-2)
        G = (P.transpose(-2,-1) @ H_sym @ P) / D
        G = G.masked_fill(D == 0, 0)
        return P @ G @ P.transpose(-2, -1)
```

A.7 Experimental Details and Additional Results

A.7.1 Impact of the Number of Components for the Gradient Methods

In this section, we present the impact of the parameter K in the setting of Section 4.5. We fix $n = 2000, d = 3$ and $T = 30$, varying $K \in \{3, 5, 8, 10, 12, 15\}$ in Fig. 16. The results suggest that the number of components K greatly impacts the difficulty of the EM algorithm to converge to a "stable" fixed point, incurring worsening gradient approximations. In certain settings, the spectral norm of the Jacobian can be orders of magnitude larger than 1, completely invalidating the OS method. In this experiment, we required a regularisation of $10^{-8} I_d$ for the covariances for numerical stability.

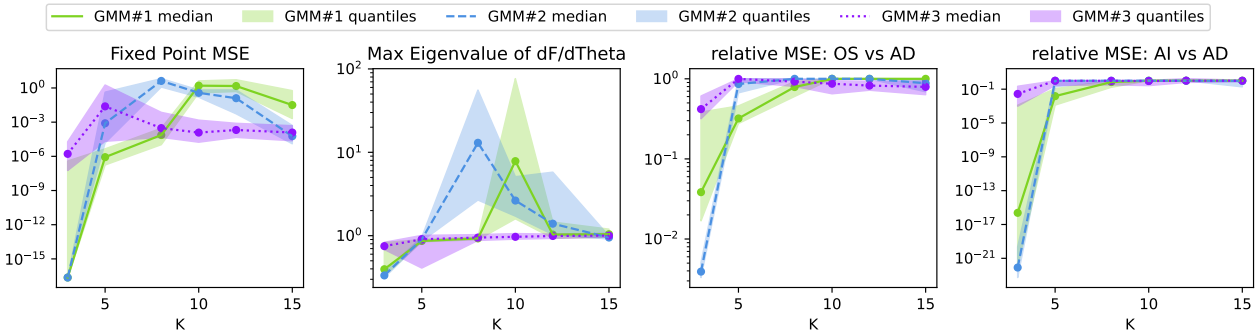


Figure 16: Varying the number of components K, we study the convergence of EM, the local contractivity of F, and the MSEs of the OS and AI gradients.

A.7.2 Barycentres

We consider a barycentre problem similar to Section 5.1, with more complex datasets: we take three two-dimensional images I_1, I_2 and I_3, we randomly sample $n = 500$ points $Y_i \in \mathbb{R}^{n\times 2}$ from each. In Fig. 17, we flow a point cloud initialised as random normal noise, towards a barycentre of $K = 15$ GMMs fitted from (Y_i).

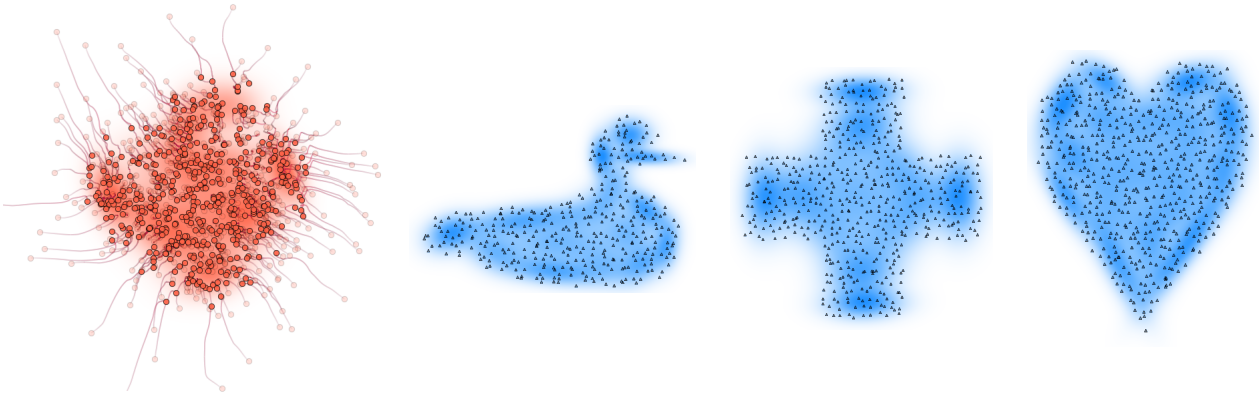


Figure 17: Left: EM – MW_2^2-Barycentre flow; right: input point clouds.

We can also compute a generalised barycentre X in $\mathbb{R}^{n\times 3}$, such that every projection $P_i(X) \in \mathbb{R}^{n\times 2}$ orthogonal to the canonical direction e_i coincides with Y_i. Specifically, we solve

$$\min_{X\in\mathbb{R}^{n\times 2}} \sum_{i=1}^{3} \mathrm{MW}_2^2 \left(P_i\#\mu(F_X^T(\theta_0)), \nu_i\right).$$

The GMMs $(\nu_i) \in \mathrm{GMM}_2(40)^3$ are fitted beforehand with the point clouds (Y_i), and μ_X is the running EM estimation of optimised cloud X. The projected GMM $P_i\#\mu(F_X^T(\theta_0))$ is defined by projecting the means and covariances of the three-dimensional GMM $\mu(F_X^T(\theta_0))$. We sample $n = 1000$ points and fit $K = 100$ Gaussian components in each cloud. Example results are shown in Fig. 18. We obtained very similar results with an alternative loss which estimates three GMMs in $\mathbb{R}^2$ instead of one in $\mathbb{R}^3$. In all our barycenter experiments, we set $\varepsilon_r = 10^{-3}$.

A.7.3 Colour Transfer

Optimiser choice. We use gradient descent with fixed step size. Indeed, optimisers such as Adam have per-parameter learning rates that are adapted dynamically. As a result, points are treated differently and move at different speeds. For instance, imagine that we independently sample points $x_1, \cdots, x_n$ in $\mathbb{R}^d$ following the law $\mathcal{N}(0_d, I_d)$ and want to match them to $\mathcal{N}(0_d, 2I_d)$ using the Mixture Loss. Optimisers like Adam might focus on moving the points on the boundary to make the cloud's

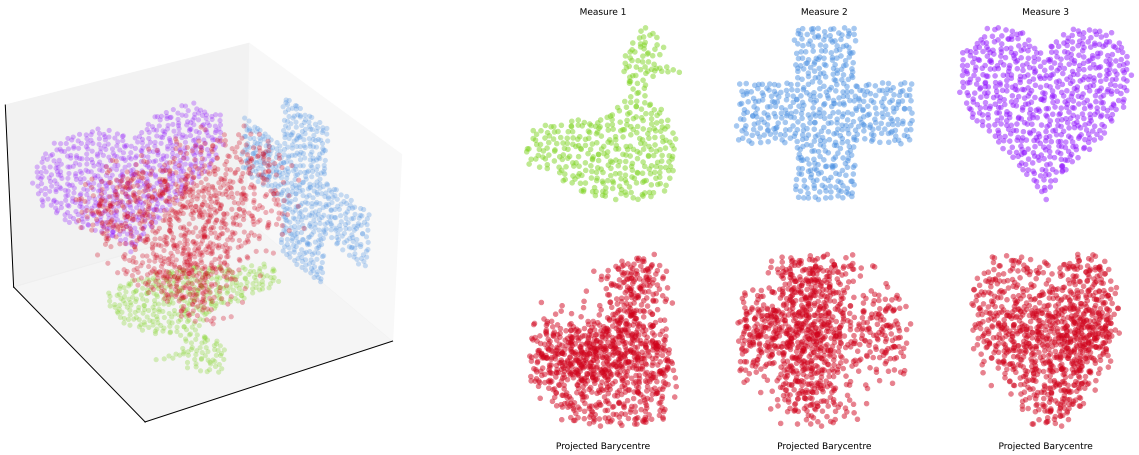
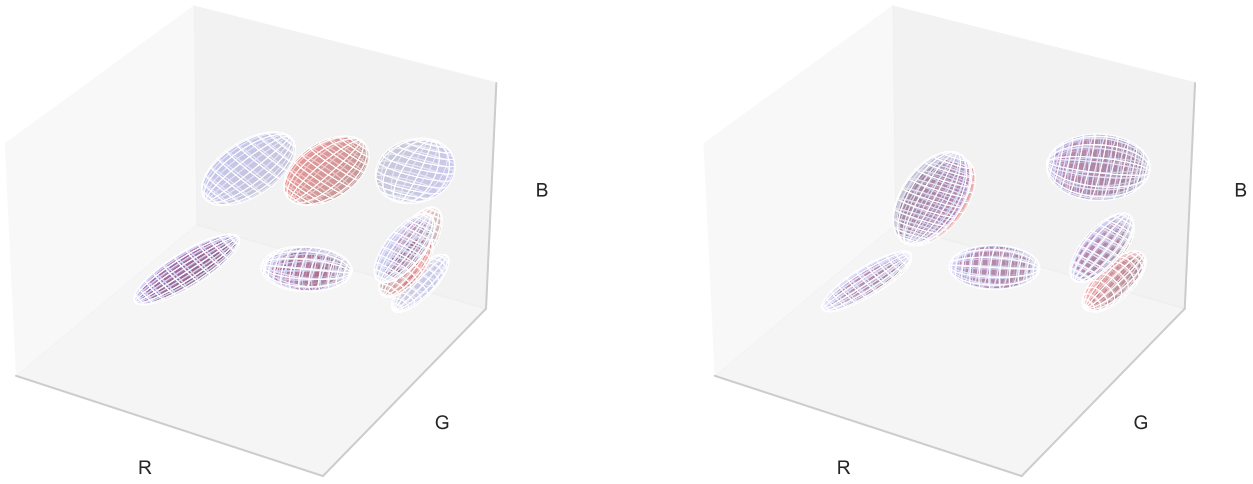


Figure 18: 3D barycentre (left) and its projections (right).

variance larger. Fixed-step gradient descent, on the contrary, will treat each point equally, and will rescale the cloud as a whole.

Fixing mixture weights. As discussed in Section 3.3, fixing the mixture weights to be uniform (using Algorithm 2) avoids local minima, as illustrated in Fig. 19. For a colour transfer task with $K = 6$ components, when weights are allowed to vary, some components of the optimised mixture (in red) are trapped between two target components (in blue).



(a) (b)

Figure 19: Final GMMs in RGB space with (a) variable weights and (b) fixed weights. Target is in blue and optimised mixture is in red. In (a) we are stuck in a local minimum, in (b) we converged.

Choice of the number of components K. We choose $K = 10$ Gaussian components in our colour transfer experiments. Taking $K = 1$, i.e. one Gaussian per source and target, is equivalent to applying the following affine map to source pixels:

$$T : x \in \mathbb{R}^d \longmapsto m_t + A(x - m_s),$$

where m_s and m_t are the empirical means of source and target colour distributions, and where

$$A = \Sigma_s^{-1/2} \left(\Sigma_s^{1/2} \Sigma_t \Sigma_s^{1/2} \right)^{1/2} \Sigma_s^{-1/2} = A^\top.$$

This map performs a coarser colour transfer than the optimisation with $K = 10$, as compared in Fig. 20: the contrast is better preserved with higher values of K. We always set $\varepsilon_r = 10^{-3}$ in this experiment.

(a) Result with $K = 1$ (b) Result with $K = 10$ (c) Target

Figure 20: Colour transfer with (a) $K = 1$ components and (b) $K = 10$ components.

A.7.4 Neural Style Transfer

We note that the optimiser choice and EM variant (between Algorithms 1 and 2) do not change the results qualitatively. In our experiments, we use Adam and use standard EM Algorithm 1.

Choice of the number of components K. We choose $K = 3$ in our experiments. When using $K = 1$ Gaussian component, the style transfer objective in Eq. (23) simplifies to

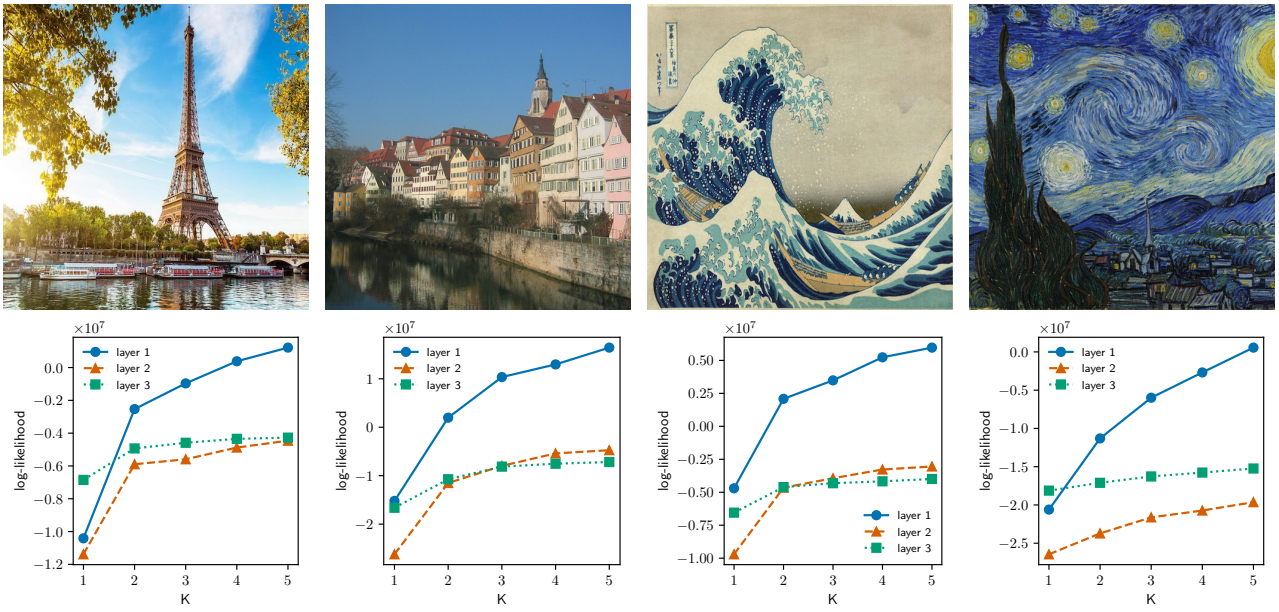
$$\min_{X \in \mathbb{R}^{3\times H\times W}} \sum_{\ell=1}^{3} \lambda_\ell \left(\|m_{s,\ell}(X) - m_{t,\ell}\|_2^2 + d_{\mathrm{BW}}^2 \left(\Sigma_{s,\ell}(X), \Sigma_{t,\ell}\right) \right),$$

where $m_{s,\ell}(X)$ and $\Sigma_{s,\ell}(X)$ are the empirical means and covariances of source features $\mathrm{VGG}_{1\cdots\ell}(X)$. We similarly define $m_{t,\ell}$ and $\Sigma_{t,\ell}$ for the target image Y. As there are only one source and target Gaussian, there is no need for an EM algorithm, we simply estimate the means and covariance and apply Gaussian OT. To evaluate the influence of K, we take two content images (the Eiffel tower and Gatys' [GEB15] picture of Tuebingen), and two style images (*The Great Wave* and *The Starry Night*). We compute their features corresponding to first layers $1, \cdots, \ell$ of VGG, for $\ell \in \{1, \cdots, 3\}$. We fit Gaussian mixtures on these features with varying number of components K, and we evaluate corresponding log-likelihoods as a measure of model quality. Results are presented in Fig. 21: most of them elbow around $K = 3$, the value we retain in our experiments. We set $\varepsilon_r = 0.1$ (as the dominant eigenvalue of the target covariance matrices empirically lies between 10^2 and 10^3).

See Fig. 22 for a comparison of style transfer with $K = 1$ and $K = 3$. Taking higher values of K does not yield significant improvement in the results. Yet, for Fig. 22a, the sky is less uniform with $K = 1$ and the bottom-left corner is more blurry. For Fig. 22c, the sky differs a bit between $K = 1$ and $K = 3$. In Fig. 23, we illustrate some gradient descent iterations, and observe the progressive refinement of the stylisation. It appears that the colour correspondance is achieved quite early, with stylistic details appearing later in the optimisation.

A.7.5 Texture Synthesis

Consider a (space-periodic) target texture $u \in [0,1]^{h\times w\times C}$. Our objective is to produce a (space-periodic) texture $x \in [0,1]^{H\times W\times C}$. To initialise, we sample a stationary Gaussian field $Z \in \mathbb{R}^{H\times W\times C}$ of i.i.d. entries of law $\mathcal{N}(0, I_C)$. The initialisation is then a stationary Gaussian field with the same statistics as u, defined as $\mathbb{T}_{H,W}$ by $x_0 := m + u \star Z$, where $m \in \mathbb{R}^C$ is the mean of u and $\star$ denotes the discrete convolution with periodic boundary conditions.



(a) Eiffel Tower (b) Tuebingen (c) The Great Wave (d) The Starry Night

Figure 21: Four images and their corresponding log-likelihood vs. K plots, for VGG layers $\ell \in \{1, \cdots, 3\}$.

Given a texture $v \in \mathbb{R}^{H \times W \times C}$, we consider $P_p(v) \subset [0,1]^{C \times p^2}$ the set of its $p \times p$ patches (with periodic boundary conditions and indexing from the top-left corner). We also define the downsampling operator D_{s_i} that shrinks the image by 2^{s_i}. The patch distribution of a texture v is noted $\mu_p^v = \frac{1}{|P_p(v)|} \sum_{y \in P_p(v)} \delta_y$.

For a given list of scales $\mathcal{S} = \{(p_i, s_i)\}_{1 \leq i \leq L}$, we aim to make the patch distributions $\mu_p(x)$ of the optimised texture x match the targets $\mu_p(v)$. Each μ_p^v is approximated by a Gaussian mixture $\hat{\mu}_p^v$. The target mixture $\hat{\mu}_p^u$ is fitted once by EM and fixed. We use the *Warm-Start EM* (see Algorithm 3) variant during optimisation, and take $\varepsilon_r = 10^{-3}$. We solve the problem

$$\min_x \sum_{i=1}^{L} 2^{2s_i} \, \mathrm{MW}_2^2 \left(\hat{\mu}_{p_i}^{D_{s_i} x}, \, \hat{\mu}_{p_i}^{D_{s_i} u} \right).$$

At the end, we perform a nearest neighbour projection on the largest scale: each patch in the generated x is matched to its nearest patch in the target u. Then, each pixel is reconstructed by averaging the corresponding patches. In Fig. 24 we observe an example of this process for a single scale. For this simple example, even the lighter mono-scale model produces satisfactory results.

A.7.6 Image Generation

As a proof of concept, we train a Generative Adversarial Network (GAN) [Goo+14; ACB17; DZS18] using the Mixture–Wasserstein distance as a regularisation. The idea is to encourage the generator to produce images which have similar features to real images: while this does not suffice to produce realistic images, it can guide the training optimisation out of spurious local minima induced by the notoriously unstable GAN training ([ACB17]). Given a batch size b and a latent dimension ℓ, we independently sample $x_1, \cdots, x_b \sim \mathcal{N}(0_\ell, I_\ell)$, and we draw $y_1, \cdots, y_b$ from our dataset of real images. The aim of our *generator* $\mathtt{G} : \mathbb{R}^\ell \longrightarrow \mathbb{R}^{C \times H \times W}$ is to map these Gaussian samples to real images of height H, width W and C channels. We also introduce a *feature extractor* $\mathtt{F} : \mathbb{R}^{C \times H \times W} \longrightarrow \mathbb{R}^f$ which learns to map images to relevant features of dimension f, and a *discriminator* $\mathtt{D} : \mathbb{R}^f \longrightarrow \mathbb{R}$ whose goal is to discriminate real and fake images by producing different vectors of features. The

(a) Eiffel → The Great Wave (b) Tuebingen → The Starry Night (c) Tuebingen → The Scream

Figure 22: Taking $K = 1$ (top) gives results comparable to $K = 3$ (bottom).

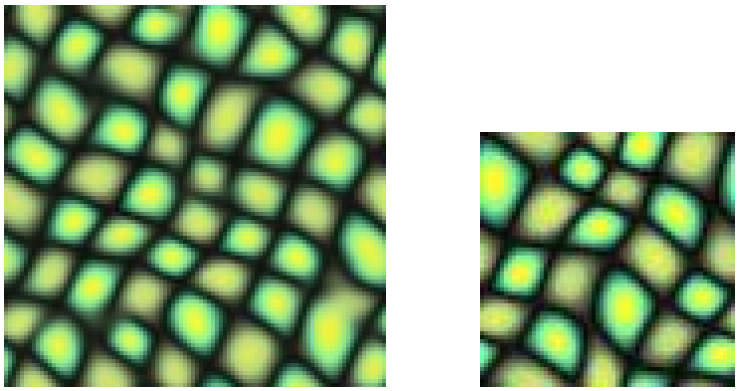
(a) Iteration 10 (b) Iteration 30 (c) Iteration 50 (d) Iteration 70 (e) Iteration 90

Figure 23: Evolution of the style transfer across iterations.

full adversarial objective on a noise batch X and a real image batch Y is then defined as follows:

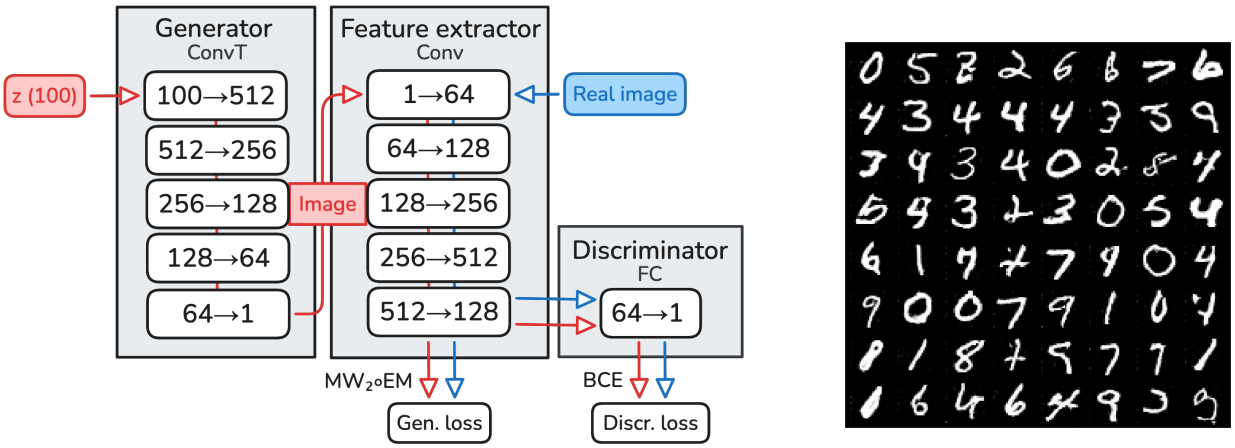
$$\min_{\mathtt{G}} \left[\mathrm{MW}_2^2 \left(F^T(\theta_0, \mathtt{F} \circ \mathtt{G}(X)), F^T(\theta_0', \mathtt{F}(Y)) \right) + \max_{\mathtt{F},\mathtt{D}} \left(\sum_{i=1}^{b} \log \mathtt{D} \circ \mathtt{F}\left(y_i\right) + \log\left(1 - \mathtt{D} \circ \mathtt{F} \circ \mathtt{G}(x_i)\right) \right) \right], \tag{82}$$

where θ_0 and θ_0' are chosen using *k-means* initialisation. Note that the feature extractor $\mathtt{F}$ appears in the MW_2^2 term, but gradients from this term are not used to update $\mathtt{F}$. We optimise these losses using Stochastic Gradient Descent and *Full Automatic Differentiation* (see Section 2.3). For every sampled batch, we alternate one step on $\mathtt{F}$ and $\mathtt{D}$, then one step on $\mathtt{G}$. The generator is encouraged to produce images with similar features to the real ones, and aims to fool the discriminator. The feature extractor $\mathtt{F}$ attempts to extract features such that $\mathtt{D}$ is able to discriminate real from fake images. The full network architecture is described in Appendix A.7.6. Note that as in other GANs [Goo+14; ACB17; DZS18], we do not optimise over the full dataset and rather optimise by sampling mini-batches at each step, incurring a seldom-discussed bias [Fat+20; Fat+21b; Fat+21a; Ton+24] for computational efficiency.



(a) (b)

Figure 24: (a) Texture synthesis with 12×12 patches, $K = 4$, mono-scale, (b) reference.



(a) GAN architecture (adapted from DCGAN [RMC16]) (b) MNIST generation with $K = 10$.

Figure 25: Image generation with the MW_2^2-GAN from Eq. (82).