Nonequispaced Fast Fourier Transform Boost for the Sinkhorn Algorithm

Rajmadan Lakshmanan*†   Alois Pichler*‡§   Daniel Potts*‡¶

February 13, 2023

Abstract

This contribution features an accelerated computation of the Sinkhorn's algorithm, which approximates the Wasserstein transportation distance, by employing nonequispaced fast Fourier transforms (NFFT). The algorithm proposed allows approximations of the Wasserstein distance by involving not more than $O(n \log n)$ operations for probability measures supported by n points. Furthermore, the proposed method avoids expensive allocations of the characterizing matrices. With this numerical acceleration, the transportation distance is accessible to probability measures out of reach so far. Numerical experiments using synthetic and real data affirm the computational advantage and superiority.

Mathematics Subject Classifications: 90C08, 90C15, 60G07
Keywords: Sinkhorn's divergence · optimal transport · NFFT · entropy

1 Introduction

In optimal transport theory, the Wasserstein distance – often referred to as the Monge-Kantorovich distance – is used to define and quantify optimal transitions between probability measures. The lowest (or cheapest) average costs to fully transfer one probability measure into another characterizes the distance. In most applications, costs correspond to the distance between locations. For a comprehensive discussion of the Wasserstein distance from mathematical perspective we may refer to Villani [40].

The concept of entropy regularization of the Wasserstein distance, proposed by Cuturi [9], is an important touchstone, which improves the computational process of traditional methods. This entropy regularized Wasserstein problem is efficiently solved using the Sinkhorn's algorithm (cf. Sinkhorn [36]). In today's data-driven world, the powerful and growing relationship between optimization and data science utilizes the Wasserstein distance, e.g., for text classification (cf.

---

*Faculty of Mathematics, University of Technology, Chemnitz, Germany
†rajmadan.lakshmanan@math.tu-chemnitz.de
‡DFG, German Research Foundation – Project-ID 416228727 – SFB 1410.
§ https://orcid.org/0000-0001-8876-2429
¶ https://orcid.org/0000-0003-3651-4364

Kusner et al. [19]), clustering (cf. Chakraborty et al. [7]), image classification (cf. Tai et al. [38]) or domain adaptation (cf. Courty et al. [8]). Notably, most of the applications rely on *discrete* measures. However, some significant contributions are also presented in literature to support the arguments of semi-discrete and/ or continuous measures (cf. Mensch and Peyré [22]). The constructive line of research on the entropy regularization method to approximate the Wasserstein distance proposes many significant algorithms to increase the computational efficiency, as well as to stabilize the approximation accuracy (cf. Dvurechensky et al. [10], Lin et al. [20] or Schmitzer [34]). However, this article addresses the efficient computation of standard Sinkhorn's algorithm in terms of time and memory allocation to approximate the Wasserstein distance in a simple personal computer, especially in case of large data volume.

Related works.   In the past decade, based on the well-known standard (equispaced) fast Fourier transform (FFT) method, many approaches have been proposed to find efficient data representation for various problems. The family of standard FFT algorithms has been applied in many areas, such as face recognition (cf. Hao et al. [15]), autonomous vehicles (cf. Bilik et al. [5]), voice assistants (cf. Revay and Teschke [30]), etc., and have achieved notable performance. The standard FFT algorithm improves the computational operations from $O(n^2)$ to $O(n \log n)$, where n denotes the number of data points, this process involves equispaced sampling. However, in some cases, the equispaced sampling is one of the root causes of failure to meet accuracy (cf. Platte et al. [27], Plonka et al. [28, Chapter 7]). We recognize that the optimal transport (OT) communities use the idea of standard FFT to speed up the Sinkhorn's iterations in some places (cf. Papadakis et al. [25]). The standard FFT methods utilize equispaced convolution, which is a setback, when we consider the stability of the computation and approximation accuracy (cf. Peyré and Cuturi [26, Remark 4.18]). To overcome this challenge we present a *non*-equispaced convolution below, and it is also achievable in $O(n \log n)$ arithmetic operations. Furthermore, for faster computation, low-rank factorization techniques are considered as a popular argument among OT communities (cf. Altschuler et al. [2], Scetbon and Cuturi [32], Altschuler and Boix-Adsera [3]). As a consequence of the line of research on Low-Rank Factorization, Scetbon et al. [33] have developed an algorithm to efficiently solve the regularized OT problem, which depends on *low-rank couplings*. As well, the method can be employed to accelerate problems involving multi marginals, cf. Ba and Quellmalz [4].

Contribution.   We improve the computational time and memory allocation of the standard entropy regularization approach to approximate Wasserstein distance with negligible or no compromise of accuracy. The technique we present here is a fast summation method, and it is based on the nonequispaced fast Fourier transform (NFFT), see Plonka et al. [28, Chapter 7]. Using NFFT, we boost the performance of standard entropy regularization of Wasserstein distance with stable computation and high (machine) accuracy. Additionally, we explicitly provide the bounds for the approximation of the Wasserstein distance. We experimentally substantiate the computational efficiency of our proposed algorithm, and we validate the accuracy via numerical results.

Outline of the paper. This paper is organized as follows. Initially, in Section 2, we discuss the necessary notations and definitions of Wasserstein distance. Section 3 introduces the entropy regularization approach to approximate the Wasserstein distance (Primal problem) and its dual formulation. Additionally, we show the convergence properties of Sinkhorn's iteration and recall the Sinkhorn divergence. A fast summation technique based on NFFT, which is utilized in this paper, is introduced in Section 4. In Section 4.2, we propose the NFFT-accelerated Sinkhorn's algorithm and schematically explain the operations. Section 5 contains the demonstration of performance of our proposed algorithm on synthetic as well as real data sets. Finally, Section 6 summarizes and concludes the paper.

2 Preliminaries

In this section, we provide a short review of the Monge–Kantorovich or the Wasserstein distance.

On a space of probability measures, Wasserstein distances offer a natural metric. Intuitively, the Wasserstein distance measures the minimum, average amount of transporting cost required to transform one distribution into another.

Definition 2.1 (Wasserstein distance). Let $(\mathcal{X}, d)$ be a Polish space and P and $\tilde{P} \in \mathcal{P}(\mathcal{X})$ be two probability measures on the Borel sets of $\mathcal{X}$. The Wasserstein distance of order $r \geq 1$ of the probability measures P and $\tilde{P}$ for a given cost or distance function $d \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is

$$W_r(P, \tilde{P}) := w_r(P, \tilde{P})^{1/r},$$

where

$$w_r(P, \tilde{P}) := \inf_{\pi \in \Pi(P, \tilde{P})} \iint_{\mathcal{X} \times \mathcal{X}} d(x, \tilde{x})^r \, \pi(\mathrm{d}x, \mathrm{d}\tilde{x}). \tag{1}$$

Here, $\Pi(P, \tilde{P}) \subset \mathcal{P}(\mathcal{X}^2)$ is the set of bivariate probability measures on $\mathcal{X} \times \mathcal{X}$ with marginals P and $\tilde{P}$, respectively; that is, $\pi(A \times \mathcal{X}) = P(A)$ and $\pi(\mathcal{X} \times B) = \tilde{P}(B)$ for all measureables sets A and $B \subset \mathcal{X}$.

Wasserstein distances metrize the weak* topology on measures with finite rth moment. In the discrete setting considered below and all regular situations, the infimum in (1) is attained (cf. Villani [39, Theorem 7.12]).

Discrete framework. Concrete implementations of the Wasserstein problem rely on discrete measures of the form*

$$P(\cdot) = \sum_{i=1}^{n} p_i \, \delta_{x_i}(\cdot)$$

with $p_i \geq 0$ and $\sum_{i=1}^{n} p_i = 1$. These measures are dense in $\mathcal{P}(\mathcal{X})$ with respect to the weak* topology, see Bolley [6].

---

*$\delta_x(A) := \mathbf{1}_A(x) = \begin{cases} 1 & \text{if } x \in A, \\ 0 & \text{else} \end{cases}$ is the Dirac measure located at $x \in \mathcal{X}$.

For two discrete probability measures

$$P = \sum_{i=1}^{n} p_i \delta_{x_i} \quad \text{and} \quad \tilde{P} = \sum_{j=1}^{\tilde{n}} \tilde{p}_j \, \delta_{\tilde{x}_j},$$

the bivariate measure $\pi = \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \delta_{(x_i, \tilde{x}_j)} \in \mathcal{P}(\mathcal{X}^2)$ solves the Wasserstein problem (1), provided that the matrix $\pi = (\pi_{ij}) \in \mathbb{R}^{n \times \tilde{n}}$ is the solution of the optimization problem

$$w_r(P, \tilde{P}) = \min \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij} \, d_{ij}^r, \tag{2a}$$

where

$$\sum_{j=1}^{\tilde{n}} \pi_{ij} = p_i \text{ for } i = 1, \dots, n, \tag{2b}$$

$$\sum_{i=1}^{n} \pi_{ij} = \tilde{p}_j \text{ for } j = 1, \dots, \tilde{n} \text{ and} \tag{2c}$$

$$\pi_{ij} \geq 0 \text{ for all } i = 1, \dots, n \text{ and } j = 1, \dots, \tilde{n} \tag{2d}$$

and $d_{ij} = d(x_i, x_j)$ is the distance matrix. The problem (2a)–(2d) is a linear optimization problem, occasionally referred to as *Kantorovich problem*. In what follows, the optimal matrix is denoted π^w.

Complexity. For $n \approx \tilde{n}$, the linear optimization problem (2a)–(2d) can be solved by straightforward computation involving $O(n^3)$ multiplications.

In the following Section 3 we recall the popular approach based on entropy regularization to reduce the computational burden of the optimization problem (2a)–(2d). This approach is efficiently tackled by an iteration process, which is popularly known as *Sinkhorn's algorithm*. This algorithm is also known as matrix scaling type algorithm (cf. Rote and Zachariasen [31]).

Notations. Throughout this article, $\| \cdot \|$ stands for Euclidean norm or 2-norm and $\| \cdot \|_1$ stands for 1-norm. The vector of all ones and zeros denote as $\mathbf{1}_n := (1, \dots, 1)^\top \in \mathbb{R}^n$ and $\mathbf{0}_n := (0, \dots, 0)^\top \in \mathbb{R}^n$. For any probability vectors a and b, the Kullback–Leibler divergence is

$$D_{KL}(a \mid b) := \sum_{i=1}^{n} a_i \log \left(\frac{a_i}{b_i} \right).$$

3 Entropy regularization and Sinkhorn divergences

This section considers the entropy-regularization of the Wasserstein problem, and characterizes its duality. Furthermore, we recall Sinkhorn's algorithm which permits a considerably faster implementation.

3.1 Entropy regularization of the Wasserstein problem

The Entropy Regularized Wasserstein (ERW) problem involves the entropic regularization term

$$H(\pi) := -\sum_{i,j} \pi_{ij} \log \pi_{ij}$$

to the linear optimization problem (2a)–(2d).

Definition 3.1 (ERW distance). The ERW distance with regularization parameter $\lambda > 0$ is the minimal value of the optimization problem

$$s_{r;\lambda}(P, \tilde{P}) := \min \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij}\, d_{ij}^{r} - \frac{1}{\lambda} H(\pi), \tag{3a}$$

where

$$\sum_{j=1}^{\tilde{n}} \pi_{ij} = p_i \text{ for } i = 1, \ldots, n, \tag{3b}$$

$$\sum_{i=1}^{n} \pi_{ij} = \tilde{p}_j \text{ for } j = 1, \ldots, \tilde{n} \text{ and} \tag{3c}$$

$$\pi_{ij} > 0 \text{ for all } i = 1, \ldots, n \text{ and } j = 1, \ldots, \tilde{n}. \tag{3d}$$

The matrix minimizing (3a) subject to the constraintes (3b)–(3d) is denoted $\pi^s \in \mathbb{R}^{n \times \tilde{n}}$. Further, we set

$$\tilde{s}_{r;\lambda} := s_{r;\lambda} + \frac{1}{\lambda} H(\pi^s) = \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij}^{s}\, d_{ij}^{r}. \tag{4}$$

The non-negativity constraint (2d) is notably not active in the constraints (3b)–(3d), as the function $\varphi(x) := x \log x$ is strictly convex in $[0, 1]$ with $\varphi'(0) = -\infty$, and the optimal solution consequently satisfies $\pi_{ij} > 0$. The regularizing term $\frac{1}{\lambda} H(\cdot)$ is strictly convex, so that the solution of the problem (3a)–(3d) exists and is unique.

Remark 3.2 (Regularizing term). To surpass the difficulty of numerical computation of the linear optimization problem (2a)–(2d), the entropy regularization approach was originally proposed in Cuturi [9]. We also refer to Gasnikov et al. [12], which comprises the argument of efficient numerical methods for entropy linear programming problems.

Choice of the regularization parameter λ. In general, the selection of the regularization parameter λ plays a crucial role to obtain a good approximation of the Wasserstein distance. From (3a) and the arguments below, we infer that if $\lambda \to \infty$, we obtain the standard Wasserstein distance in the limit. We refer to Neumayer and Steidl [24], who study the regularization parameter. The constructive line of research by Feydy [11] affirms that when the regularization parameter λ is not sufficiently large, the transportation plan and the regularized Wasserstein distance may be inconsiderable. However, from the literature, we infer that the choice $\lambda \geq 20$ is a good bargain between accuracy and computational speed (cf. Genevay [13, Figure 1.2, Figures 3.1–3.3], Scetbon et al. [33, Figure 2], Neumayer and Steidl [24, Section 7]).[†] To acquire a better approximation

---

[†]Cf. https://marcocuturi.net/SI.html

accuracy, we can increase the regularization parameter $\lambda > 20$ with the price of relatively more arithmetic operations.

In some applications it is crucial to estimate the Wasserstein distance with given accuracy. This can be accomplished by choosing the regularization parameter λ large enough. The following Lemma 3.3 gives a precise instruction how to choose λ to obtain a prescribed accuracy.

Lemma 3.3 (Quality of the Sinkhorn Approximation)**.** *For $\varepsilon > 0$ it holds that*

$$s_{r;\lambda}(P, \tilde{P}) \leq w_r(P, \tilde{P})^r \leq \tilde{s}_{r;\lambda}(P, \tilde{P}) \leq s_{r;\lambda}(P, \tilde{P}) + \varepsilon, \tag{5}$$

provided that

$$\lambda \geq \frac{H(P) + H(\tilde{P})}{\varepsilon}.$$

Here, $H(P) = -\sum_{i=1}^{n} p_i \log p_i$ ($H(\tilde{P}) = -\sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{p}_j$, resp.) is the entropy of the measure P ($\tilde{P}$, resp.). Further, the entropies are bounded by $H(P) + H(\tilde{P}) \leq \log n + \log \tilde{n}$.

Proof. The first inequality in (5) follows by substituting the matrix π^w in (3a), as $H(\pi^w) > 0$. Further, with the matrix π^s, it holds that $w_{r;\lambda}(P, \tilde{P}) \leq \tilde{s}_{r;\lambda}(P, \tilde{P})$, which is the second inequality using (4).

Now let π be any matrix with marginals p (cf. (2b)) and $\tilde{p}$ (cf. (2c)). It follows with the log sum inequality (or Gibbs' inequality)

$$\sum_{i,j} \pi_{ij} \log \frac{\pi_{ij}}{p_i \, \tilde{p}_j} \geq 0$$

that

$$\sum_{i,j} \pi_{ij} \log \pi_{ij} \geq \sum_{i,j} \pi_{ij} \log p_i + \sum_{i,j} \pi_{ij} \log \tilde{p}_j = \sum_{i} p_i \log p_i + \sum_{j} \tilde{p}_j \log \tilde{p}_j$$

$$= \sum_{i,j} p_i \tilde{p}_j \log p_i + \sum_{i,j} p_i \tilde{p}_j \log \tilde{p}_j = \sum_{i,j} p_i \tilde{p}_j \log(p_i \tilde{p}_j)$$

$$= \sum_{i} p_i \log p_i + \sum_{j} \tilde{p}_j \log \tilde{p}_j,$$

that is, $H(\pi) \leq H(P) + H(\tilde{P})$ and thus $\frac{1}{\lambda} H(\pi) \leq \frac{H(P) + H(\tilde{P})}{\lambda} \leq \varepsilon$ for the parameter λ large enough as in the assumption.

The remaining inequality follows from $\tilde{s}_{r;\lambda}(P, \tilde{P}) = s_{r;\lambda}(P, \tilde{P}) + \frac{1}{\lambda} H(\pi^s) \leq s_{r;\lambda}(P, \tilde{P}) + \varepsilon$.

The inequality $H(P) \leq \log n$ follows by applying Gibb's inequality to the measures with weights p ($\tilde{p}$, resp.) and the constant weights $(1/n, \ldots, 1/n)$ $((1/\tilde{n}, \ldots, 1/\tilde{n})$, resp.). $\qquad\square$

Remark 3.4. Note, that the choice $\lambda \geq \frac{\log n + \log \tilde{n}}{\varepsilon}$ is independent of the probability measure, but only depends on their granularity or dimension n ($\tilde{n}$, resp.). We may also refer to Luise et al. [21, Proposition 1] and references therein for further, related inequalities for continuous measures.

3.1.1 Dual representation of entropy-regularized Wasserstein distance

We restate the optimization problem of ERW in the following dual formulation.

Proposition 3.5 (cf. Peyré and Cuturi [26, Remark 4.28]). *For $\lambda > 0$, the ERW (3a), (3b)–(3c) admits the following dual representation*

$$d(\alpha, \tilde{\alpha}) := \max_{\alpha \in \mathbb{R}^n, \, \tilde{\alpha} \in \mathbb{R}^{\tilde{n}}} \frac{1}{\lambda} + \frac{1}{\lambda} \sum_{i=1}^{n} p_i \log \alpha_i + \frac{1}{\lambda} \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j - \frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \alpha_i \, e^{-\lambda d_{ij}^r} \, \tilde{\alpha}_j, \quad (6)$$

which is strictly concave dual function.

Proof. The Lagrangian of the ERW problem (3a) with dual parameters β (for the constraint (2b)) and γ (for (2c)) is

$$L(\pi; \beta, \gamma) = \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} d_{ij}^r \pi_{ij} + \frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij} \log \pi_{ij} + \sum_{i=1}^{n} \beta_i \Big(p_i - \sum_{j=1}^{\tilde{n}} \pi_{ij} \Big) + \sum_{j=1}^{\tilde{n}} \gamma_j \Big(\tilde{p}_j - \sum_{i=1}^{n} \pi_{ij} \Big).$$

The optimal measure π^* satisfying the first order constraint

$$0 = \frac{\partial L}{\partial \pi_{ij}} = d_{ij}^r + \frac{1}{\lambda} (\log \pi_{ij} + 1) - \beta_i - \gamma_j$$

is

$$\pi_{ij}^* = \exp\big(-\lambda(d_{ij}^r - \beta_i - \gamma_j) - 1 \big). \quad (7)$$

The measure π^* minimizes the Lagrangian L for β and γ fixed, and reveals the dual function

$$d(\beta, \gamma) = \inf_{\pi} L(\pi; \beta, \gamma) = L(\pi^*; \beta, \gamma)$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} d_{ij}^r \pi_{ij}^* + \frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij}^* \big(-\lambda(d_{ij}^r - \beta_i - \gamma_j) - 1 \big)$$

$$+ \sum_{i=1}^{n} \beta_i \Big(p_i - \sum_{j=1}^{\tilde{n}} \pi_{ij}^* \Big) + \sum_{j=1}^{\tilde{n}} \gamma_j \Big(\tilde{p}_j - \sum_{i=1}^{n} \pi_{ij}^* \Big)$$

$$= -\frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij}^* + \sum_{i=1}^{n} \beta_i \, p_i + \sum_{j=1}^{\tilde{n}} \gamma_j \, \tilde{p}_j$$

$$= \sum_{i=1}^{n} p_i \, \beta_i + \sum_{j=1}^{\tilde{n}} \tilde{p}_j \, \gamma_j - \frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} e^{-\lambda(d_{ij}^r - \beta_i - \gamma_j) - 1}$$

explicitly. Now substitute $\alpha_i = e^{\lambda \beta_i - 1/2}$ and $\tilde{\alpha}_j = e^{\lambda \gamma_j - 1/2}$, then the dual function is

$$d(\alpha, \tilde{\alpha}) = \frac{1}{\lambda} \sum_{i=1}^{n} p_i \Big(\frac{1}{2} + \log \alpha_i \Big) + \frac{1}{\lambda} \sum_{j=1}^{\tilde{n}} \tilde{p}_j \Big(\frac{1}{2} + \log \tilde{\alpha}_j \Big) - \frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \alpha_i \, e^{-\lambda d_{ij}^r} \, \tilde{\alpha}_j. \quad (8)$$

The assertion of the proposition thus follows, as $\sum_{i=1}^{n} p_i = 1$ and $\sum_{j=1}^{\tilde{n}} \tilde{p}_j = 1$ and as the duality gap vanishes for the strictly convex objective function (3a). $\square$

Scaling variables and kernel matrix. Indeed, first of all, we notice that optimal measure (7) can be obtained in terms of the scaling variables α_i and $\tilde{\alpha}_j$ by

$$\pi^* = \operatorname{diag}(\alpha)\, \mathrm{e}^{-\lambda\, d^r}\, \operatorname{diag}(\tilde{\alpha}). \tag{9}$$

The aforementioned dual problem (6) can be solved by a matrix scaling algorithm, which is popularly known as Sinkhorn's algorithm. Further, the derivate of (6) with respect to α_i ($\tilde{\alpha}_i$, resp.) gives the first order conditions, which is the basis for Sinkhorn's iteration and it is expressed as

$$\alpha_i := \frac{p_i}{\sum_{j=1}^{\tilde{n}} \mathrm{e}^{-\lambda\, d_{ij}^r}\, \tilde{\alpha}_j} \quad\text{and}\quad \tilde{\alpha}_j := \frac{p_j}{\sum_{i=1}^{n} \mathrm{e}^{-\lambda\, d_{ij}^r}\, \alpha_i}. \tag{10}$$

The main computational bottleneck of Sinkhorn's iterations is the matrix-vector multiplication in (10), which requires $O(n \cdot \tilde{n})$ arithmetic operations. In our study, we relax the computational burden by taking advantage of special structure of the matrix

$$k_{ij} := \mathrm{e}^{-\lambda\, d_{ij}^r} \quad \in \mathbb{R}^{n \times \tilde{n}},$$

which is called *Gibbs kernel* or kernel matrix.

The following discussion explicitly details the Sinkhorn's algorithm and its properties.

3.2 Sinkhorn's Algorithm

Algorithm 1: Sinkhorn's algorithm

Input: distance d_{ij} given in (19), probability vectors $p \in \mathbb{R}_{\geq 0}^n$, $\tilde{p} \in \mathbb{R}_{\geq 0}^{\tilde{n}}$, regularization parameter $\lambda > 0$, $r \geq 1$, threshold ϵ and starting value $\tilde{\alpha} = (\tilde{\alpha}_1, \ldots, \tilde{\alpha}_{\tilde{n}})$

Set
$$k_{ij} = \exp\left(-\lambda\, d_{ij}^r\right), \ \alpha^{(0)} := \mathbf{1}_n, \text{ and } \tilde{\alpha}^{(0)} := \mathbf{1}_{\tilde{n}}.$$

while $\|E^\Delta\| > \epsilon$ **do**

    if Δ *is odd* **then**

$$\alpha_i^\Delta \leftarrow \frac{p_i}{\sum_{j=1}^{\tilde{n}} k_{ij}\, \tilde{\alpha}_j^{\Delta-1}}, \quad i = 1, \ldots, n;$$

$$\tilde{\alpha}_j^\Delta \leftarrow \tilde{\alpha}_j^{\Delta-1}, \qquad\qquad j = 1, \ldots, \tilde{n}; \tag{11}$$

    else

$$\tilde{\alpha}_j^\Delta \leftarrow \frac{\tilde{p}_j}{\sum_{i=1}^{n} k_{ij}\, \alpha_i^{\Delta-1}}, \quad j = 1, \ldots, \tilde{n};$$

$$\alpha_i^\Delta \leftarrow \alpha_i^{\Delta-1}, \qquad\qquad i = 1, \ldots, n; \tag{12}$$

    increment $\Delta \leftarrow \Delta + 1$

Result:

$$s_{r;\lambda}(P, \tilde{P}) = \frac{1}{\lambda} + \frac{1}{\lambda}\sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} + \frac{1}{\lambda}\sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*} - \frac{1}{\lambda}\sum_{i=1}^{n}\sum_{j=1}^{\tilde{n}} \alpha_i^{\Delta^*}\, \mathrm{e}^{-\lambda\, d_{ij}^r}\, \tilde{\alpha}_j^{\Delta^*}$$

The matrix $\pi^{\Delta^*} = \operatorname{diag}(\alpha^{\Delta^*})\, k\, \operatorname{diag}(\tilde{\alpha}^{\Delta^*})$ can also be computed, which is the proximate solution of ERW problem (3a) (3b)–(3c).

In this section, we illustrate the *iteration process* and stopping criteria of Sinkhorn's Algorithm 1 to compute ERW distance.

The *iteration counts* of Algorithm 1 are denoted $\Delta \in \mathbb{N}$ and the final iteration count is Δ^*. Algorithm 1 alternately determines α^Δ and $\tilde{\alpha}^\Delta$ with

$$\begin{cases} \tilde{\alpha}^\Delta = \tilde{\alpha}^{\Delta-1}, & \text{if } \Delta \text{ is odd;} \\ \alpha^\Delta = \alpha^{\Delta-1}, & \text{if } \Delta \text{ is even.} \end{cases}$$

Sinkhorn's theorem (cf. Sinkhorn [36], Sinkhorn and Knopp [37] and Section 3.2.1 below) for the matrix scaling ensures that iterating (10) converges and the vectors α^Δ and $\tilde{\alpha}^\Delta$ are unique up to a scalar. From Algorithm 1, the resultant matrix $\pi^{\Delta^*} = \text{diag}(\alpha^{\Delta^*})\, k\, \text{diag}(\tilde{\alpha}^{\Delta^*})$ can be computed, which is the proximate solution of ERW problem (3a), (3b)–(3c).

Stopping criteria. The iteration process reveals the matrices $\left(\pi^\Delta\right)_{\Delta \in \mathbb{N}}$, which are defined as

$$\pi^\Delta := \text{diag}(\alpha^\Delta)\, k\, \text{diag}(\tilde{\alpha}^\Delta), \quad \Delta \in \mathbb{N}.$$

The norm of residuals E^Δ, which measure the error of the iteration, is

$$\|E^\Delta\| := \|\pi^\Delta \mathbf{1}_{\tilde{n}} - p\| + \|(\pi^\Delta)^\top \mathbf{1}_n - \tilde{p}\|,$$

where $\mathbf{1}_n := (1, \ldots, 1)^\top \in \mathbb{R}^n$ and $\mathbf{1}_{\tilde{n}} := (1, \ldots, 1)^\top \in \mathbb{R}^{\tilde{n}}$. If Δ is odd, $\|(\pi^\Delta)^\top \mathbf{1}_n - \tilde{p}\| = 0$ and if Δ is even, $\|\pi^\Delta \mathbf{1}_{\tilde{n}} - p\| = 0$. The stopping criteria for Algorithm 1, i.e., $\|E^{\Delta^*}\| \le \epsilon$, implies that

$$\|\pi^{\Delta^*} \mathbf{1}_{\tilde{n}} - p\| + \|(\pi^{\Delta^*})^\top \mathbf{1}_n - \tilde{p}\| \le \epsilon.$$

Algorithm 1 consolidates the individual steps again.

Stabilized Sinkhorn's algorithm. The standard Sinkhorn's Algorithm 1 significantly reduces the complexity of the traditional methods. However, the thirst of larger λ among few applications raises the problem of numerical instabilities. More precisely, for larger λ, the elementwise exponential matrix $k = e^{-\lambda\, d^r}$ suffers numerical underflow. This side effect has increased the need among the OT community to compromise for a slower algorithm, which is known as log-domain stabilized Sinkhorn's algorithm.

The log-domain stabilized Sinkhorn's Algorithm 2 scales dual variables (β, γ) instead of exponentiated scaling variables $(\alpha, \tilde{\alpha})$, and it utilizes the famous trick among machine learning community called *log-sum-exp trick*. This log-domain computation and the log-sum-exp trick tackle the numerical underflow.

Algorithm 2: Sinkhorn's algorithm (log-domain stabilized)

Input: distance d_{ij} given in (19), probability vectors $p \in \mathbb{R}^n_{\geq 0}$, $\tilde{p} \in \mathbb{R}^{\tilde{n}}_{\geq 0}$, regularization parameter $\lambda > 0$, $r \geq 1$, threshold ϵ and starting value $\gamma = (\gamma_1, \dots, \gamma_{\tilde{n}})$

Set
$$k_{ij} = \exp\left(-\lambda\, d^r_{ij}\right),\ \beta^{(0)} := \mathbf{0}_n,\ \text{and}\ \gamma^{(0)} := \mathbf{0}_{\tilde{n}}.$$

while $\|E^\Delta\| > \epsilon$ **do**

  if Δ *is odd* **then**

$$\beta^\Delta_i \leftarrow \frac{1}{\lambda}\left(\log p_i - \log\Big(\sum_{j=1}^{\tilde{n}} k_{ij}\, e^{\lambda\,\gamma^{\Delta-1}_j - 1/2}\Big)\right),\quad i = 1, \dots, n;$$

$$\gamma^\Delta_j \leftarrow \gamma^{\Delta-1}_j,\quad j = 1, \dots, \tilde{n};$$

  else

$$\gamma^\Delta_j \leftarrow \frac{1}{\lambda}\left(\log \tilde{p}_j - \log\Big(\sum_{i=1}^{n} k_{ij}\, e^{\lambda\,\beta^{\Delta-1}_i - 1/2}\Big)\right),\quad j = 1, \dots, \tilde{n};$$

$$\beta^\Delta_i \leftarrow \beta^{\Delta-1}_i,\quad i = 1, \dots, n;$$

  increment $\Delta \leftarrow \Delta + 1$

Result:
$$s_{r;\lambda}(P, \tilde{P}) = \sum_{i=1}^{n} p_i\, \beta_i + \sum_{j=1}^{\tilde{n}} \tilde{p}_j\, \gamma_j - \frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} e^{-\lambda(d^r_{ij} - \beta_i - \gamma_j) - 1}$$

The matrix $\pi^{\Delta^*} = \operatorname{diag}(e^{\lambda\,\beta^{\Delta^*} - 1/2})\, k\, \operatorname{diag}(e^{\lambda\,\gamma^{\Delta^*} - 1/2})$ can be computed, which is the proximate solution of ERW problem (3a), (3b)–(3c)

Algorithm 2 encapsulates the individual steps again.

3.2.1 Convergence properties of Sinkhorn's iteration

The aim of this section is to demonstrate the convergence properties of Sinkhorn's iteration. The following proofs, which summarize the convergence of Sinkhorn's iteration, are applied in many contexts (cf. Altschuler et al. [1], Dvurechensky et al. [10], Khalil Abid and Gower [18]). We consider the following auxiliary lemmas to substantiate the objective of Algorithm 1 (i.e., the approximation of the Wasserstein distance) from a theoretical standpoint. The dual formulation of the ERW problem relates the function $d(\alpha, \tilde{\alpha})$ (cf. (8)) and

$$f(\alpha, \tilde{\alpha}) = \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \alpha_i\, \tilde{k}_{ij}\, \tilde{\alpha}_j - \sum_{i=1}^{n} p_i \log \alpha_i - \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j, \tag{13}$$

where $\tilde{k} := \frac{k}{\|k\|_1}$, $k = \exp\left(-\lambda\, d^r\right)$, and $p \in \mathbb{R}^n_{\geq 0}$, $\tilde{p} \in \mathbb{R}^{\tilde{n}}_{\geq 0}$ are the probability vectors, which satisfy

$$p^\top \mathbf{1}_n = \tilde{p}^\top \mathbf{1}_{\tilde{n}} = 1.$$

Remark 3.6 (Normalization of the kernel matrix k). In the literature, the approach of normalization of the kernel matrix k is widely used (cf. Altschuler et al. [1], Khalil Abid and Gower [18], Kalantari et al. [16]) for theoretical and numerical analysis. Without loss of generality, we utilize this approach only to substantiate the convergence properties. For numerical experiments, we consider the standard matrix k.

The following Lemma 3.7 describes the evolution of the objective function (13) to the target marginals $(p, \tilde{p})$ of Sinkhorn's iteration.

Lemma 3.7 (cf. Kalantari et al. [16, Lemma 3.1]). *The iterates α^Δ and $\tilde{\alpha}^\Delta$ of Algorithm 1 satisfy*

$$f(\alpha^\Delta, \tilde{\alpha}^\Delta) - f(\alpha^{\Delta+1}, \tilde{\alpha}^{\Delta+1}) = D_{KL}(p \mid \pi^\Delta \mathbf{1}_{\tilde{n}}) + D_{KL}(\tilde{p} \mid (\pi^\Delta)^\top \mathbf{1}_n).$$

Proof. First, we assume $\Delta \geq 1$ is even. By equation (13), it follows that

$$f(\alpha^\Delta, \tilde{\alpha}^\Delta) - f(\alpha^{\Delta+1}, \tilde{\alpha}^{\Delta+1}) = \sum_{ij} (\alpha_i^\Delta \tilde{k}_{ij} \tilde{\alpha}_j^\Delta - \alpha_i^{\Delta+1} \tilde{k}_{ij} \tilde{\alpha}_j^{\Delta+1}) \qquad (14)$$

$$+ \sum_i p_i \big(\log(\alpha_i^{\Delta+1}) - \log(\alpha_i^\Delta) \big)$$

$$+ \sum_j \tilde{p}_j \big(\log(\tilde{\alpha}_j^{\Delta+1}) - \log(\tilde{\alpha}_j^\Delta) \big).$$

The first component of equation (14) turns into

$$\sum_{ij} (\alpha_i^\Delta \tilde{k}_{ij} \tilde{\alpha}_j^\Delta - \alpha_i^{\Delta+1} \tilde{k}_{ij} \tilde{\alpha}_j^{\Delta+1}) = 0,$$

since

$$(\alpha^\Delta)^\top \tilde{k} \tilde{\alpha}^\Delta = \mathbf{1}_n^\top \pi^\Delta \mathbf{1}_{\tilde{n}} = \tilde{p}^\top \mathbf{1}_{\tilde{n}} = 1,$$

similarly

$$(\alpha^{\Delta+1})^\top \tilde{k} \tilde{\alpha}^{\Delta+1} = \mathbf{1}_n^\top \pi^{\Delta+1} \mathbf{1}_{\tilde{n}} = \mathbf{1}_n^\top p = 1,$$

see (11)–(12). For this reason, the equation (14) becomes

$$\sum_i p_i \big(\log(\alpha_i^{\Delta+1}) - \log(\alpha_i^\Delta) \big) + \sum_j p_j \big(\log(\tilde{\alpha}_j^{\Delta+1}) - \log(\tilde{\alpha}_j^\Delta) \big). \qquad (15)$$

By Equation (11), and (12), the above Equation (15) becomes

$$D_{KL}(p|\pi^\Delta \mathbf{1}_{\tilde{n}}) + D_{KL}(\tilde{p}|(\pi^\Delta)^\top \mathbf{1}_n),$$

and $D_{KL}(\tilde{p}|(\pi^\Delta)^\top \mathbf{1}_n) = 0$, since $\tilde{p} = (\pi^\Delta)^\top \mathbf{1}_n$. This completes the proof of lemma for Δ *even* case.

A similar argument applies to the case of *odd* Δ. $\qquad \square$

In the following Lemma 3.8, we consider the gap between $f(\mathbf{1}_n, \mathbf{1}_{\tilde{n}})$ and $f(\alpha^{\Delta^*}, \tilde{\alpha}^{\Delta^*})$. We know that

$$f(\mathbf{1}_n, \mathbf{1}_{\tilde{n}}) := f(\alpha^{(0)}, \tilde{\alpha}^{(0)}),$$

since $\alpha^{(0)} = \mathbf{1}_n$, and $\tilde{\alpha}^{(0)} = \mathbf{1}_{\tilde{n}}$, which is a starting value of Algorithm 1.

Lemma 3.8 (cf. Kalantari et al. [16, Lemma 4.1, Lemma 4.2]). *It holds that*

$$f(\mathbf{1}_n, \mathbf{1}_{\tilde{n}}) - f(\alpha^{\Delta^*}, \tilde{\alpha}^{\Delta^*}) \leq \log(\kappa/\jmath),$$

where κ is the sum of the entries of matrix π^{Δ^*}, and $\jmath := \min_{ij} \tilde{k}_{ij}$.

Proof. Let $(\alpha^{\Delta^*}, \tilde{\alpha}^{\Delta^*})$ be the minimizer of the objective function (13), and we set

$$\kappa := (\alpha^{\Delta^*})^\top \tilde{k} \, \tilde{\alpha}^{\Delta^*}.$$

Equation (13) rewrites as

$$f(\alpha^{\Delta^*}, \tilde{\alpha}^{\Delta^*}) = \kappa - \sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} - \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*},$$

and

$$f(\mathbf{1}_n, \mathbf{1}_{\tilde{n}}) = \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \tilde{k}_{ij} = \kappa.$$

Now we have

$$f(\mathbf{1}_n, \mathbf{1}_{\tilde{n}}) - f(\alpha^{\Delta^*}, \tilde{\alpha}^{\Delta^*}) = \sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} + \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*}. \tag{16}$$

Without loss of generality we assume that each entry of $\tilde{k}$ is at least $\jmath > 0$, then one has

$$\jmath \Big(\sum_{i=1}^{n} \alpha_i^{\Delta^*} \Big) \Big(\sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j^{\Delta^*} \Big) \leq (\alpha^{\Delta^*})^\top \tilde{k} \, \tilde{\alpha}^{\Delta^*} = \kappa. \tag{17}$$

Taking the log of both sides of equation (17) produces

$$\log \Big(\sum_{i=1}^{n} \alpha_i^{\Delta^*} \Big) + \log \Big(\sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j^{\Delta^*} \Big) \leq \log(\kappa/\jmath). \tag{18}$$

To complete the proof, we consider equations (16), (18), and the log-sum inequality. Now we have

$$\begin{aligned}
f(\mathbf{1}_n, \mathbf{1}_{\tilde{n}}) - f(\alpha^{\Delta^*}, \tilde{\alpha}^{\Delta^*}) &= \sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} + \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*} \\
&\leq \sum_{i=1}^{n} p_i \log \Big(\sum_{l=1}^{n} \alpha_l^{\Delta^*} \Big) + \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \Big(\sum_{m=1}^{\tilde{n}} \tilde{\alpha}_m^{\Delta^*} \Big) \\
&= \log \Big(\sum_{l=1}^{n} \alpha_l^{\Delta^*} \Big) + \log \Big(\sum_{m=1}^{\tilde{n}} \tilde{\alpha}_m^{\Delta^*} \Big) \\
&\leq \log(\kappa/\jmath).
\end{aligned}$$

This completes the proof of the lemma. □

Remark 3.9 (Complexity of Sinkhorn's iteration). The complexity of Sinkhorn's iteration is a well studied aspect of regularized Wasserstein problems (cf. Altschuler et al. [1], Dvurechensky et al. [10], Khalil Abid and Gower [18]). Approximately, Sinkhorn's iteration requires $O(\log n + \|d^r\|_\infty \lambda)$ arithmetic operations to converge (cf. Dvurechensky et al. [10]). This means that when $\lambda \to \infty$ corresponding to $\|d^r\|_\infty$ and n, number of iteration will be increased.

Complexity of ERW.   For $n \approx \tilde{n}$, using Algorithm 1 the ERW problem (3a), (3b)–(3c) can be solved by involving $O(n^2 \log n + \|d^r\|_\infty \lambda)$ arithmetic operations (note, that n^2 operations are needed to perform the matrix-vector multiplication).

Remark 3.10 (Entropy bias and Sinkhorn divergence). Regardless of the computational advancement of the ERW problem, it is biased. That is,

$$s_{r;\lambda}(P, P) \neq 0.$$

The quantity $s_{r;\lambda}$ is not a distance, more specifically, it violates the axiom of definiteness of the distance function. To overcome this difficulty, Ramdas et al. [29] introduce the *Sinkhorn divergence* as

$$sd_{r;\lambda}(P, \tilde{P}) := s_{r;\lambda}(P, \tilde{P}) - \frac{1}{2}s_{r;\lambda}(P, P) - \frac{1}{2}s_{r;\lambda}(\tilde{P}, \tilde{P}),$$

which is a natural normalization (or debias) of the quantity. The key properties of Sinkhorn divergence include
  (i) non-negativity,
 (ii) $\lim_{\lambda \to \infty} sd_{r;\lambda}(P, \tilde{P}) = w_r(P, \tilde{P})$ and
(iii) $sd_{r;\lambda}(P, P) = 0$ for all $\lambda > 0$.

4   Nonequispaced Fast Fourier Transform (NFFT)

Generally, for a faster computation of the matrix-vector multiplication with the distance matrix d^r, equispaced convolution is used. The most common algorithm used to compute equispaced convolution is the standard FFT algorithm. Our research, in contrast, promotes the *non*equispaced convolution, which is approximated by the *non*equispaced fast Fourier transform (NFFT) to accelerate the computation of the Sinkhorn's Algorithm 1. More precisely, the matrix-vector multiplications of Sinkhorn's iteration (10), which is the main computational bottleneck, are tackled by *fast summation* based on NFFT in $O(n \log n)$ arithmetic operations. Moreover, this fast summation technique has better stability and is accomplished with machine precision.

4.1   NFFT-based fast summation

This subsection succinctly describes the fast summation based on NFFT.

The fast summation method based on NFFT takes advantage of the special structure of the *Euclidean* distance matrix. The distance matrix $d \in \mathbb{R}^{n \times \tilde{n}}$ in (2a) has entries

$$d(x_i, \tilde{x}_j) := \|x_i - \tilde{x}_j\|, \tag{19}$$

which is the distances of all combinations of states, and we recall that $\|\cdot\|$ denotes the Euclidean norm or 2-norm.

Approximation of matrix-vector multiplication of Sinkhorn's iteration. The fast summation technique based on NFFT takes advantage of the particular form of the sums

$$t_i :\approx (k\,\tilde{\alpha})_i = \sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j \, e^{-\lambda \|x_i - \tilde{x}_j\|^r}, \quad i = 1, \ldots, n, \tag{20}$$

as well as of the sums of the 'transposed',

$$\tilde{t}_j :\approx (k\,\alpha)_j = \sum_{i=1}^{n} \alpha_i \, e^{-\lambda \|x_i - \tilde{x}_j\|^r}, \quad j = 1, \ldots, \tilde{n}, \tag{21}$$

since these summations are the bottleneck of Sinkhorn's iteration.

An overview of NFFT. For *equispaced* points x_i and $\tilde{x}_j$, the summation of (20) and (21) correspond to the multiplication of a Toeplitz matrix with a vector, respectively. In this case, we immediately obtain a fast algorithm based on embedding the matrix into a circulant matrix and then diagonalize the matrix by the Fourier matrix, see Plonka et al. [28, Theorem 3.31], such that we end up with $O(n \log n)$ operations using the FFT, see Plonka et al. [28, pp. 141–142]. A fast algorithm with *arbitrary* points follows the similar ideas, but based on the NFFT, see Plonka et al. [28, Chapter 7] and the related software in Keiner et al. [17] for details.

4.1.1 The ansatz of NFFT based fast summation

The core idea of *fast summation* based on NFFT is to accurately approximate the radial kernel function

$$\mathcal{K}(y) := e^{-\lambda \|y\|^r}.$$

In general, this approximation of the kernel function $\mathcal{K}(y)$ accommodates, when the entries of the matrix k are in the form

$$k_{ij} = \mathcal{K}(x_i - \tilde{x}_j).$$

In terms of fast summation, the goal of the NFFT is to accurately approximate $\mathcal{K}(y)$ by a h-periodic trigonometric polynomial $\mathcal{K}_{RK}(y)$,

$$\mathcal{K}(y) \approx \mathcal{K}_{RK}(y) := \sum_{k \in \mathcal{I}_N} b_k \, e^{2\pi i k y/h}, \quad \mathcal{I}_N := \left\{ -\frac{N}{2}, -\frac{N}{2} + 1, \ldots, -1, 0, \ldots, \frac{N}{2} - 1 \right\}^d, \tag{22}$$

with appropriate Fourier coefficients $b_k \in \mathbb{C}$ and bandwidth $N \in 2\mathbb{N}$. For example, when considering the Gaussian kernel function ($r = 2$), we have

$$(k\,\alpha)_j = \sum_{i=1}^{n} \alpha_i \, e^{-\lambda \|x_i - \tilde{x}_j\|^2}, \quad j = 1, \ldots, \tilde{n}. \tag{23}$$

Now we rewrite Equation (23) by involving the kernel function $\mathcal{K}(y) = e^{-\lambda \|y\|^2}$ as

$$(k\,\alpha)_j := \sum_{i=1}^{n} \alpha_i \, \mathcal{K}(x_i - \tilde{x}_j), \quad j = 1, \ldots, \tilde{n}. \tag{24}$$

14

For the efficient computation of (24), the fast summation technique based on NFFT approximates $\mathcal{K}$ by the trigonometric polynomial $\mathcal{K}_{RK}$.

From Equation (22), we notice that $\mathcal{K}_{RK}(y)$ are h-periodic functions, although the kernel $\mathcal{K}(y)$ is not h-periodic. Therefore, we regularize $\mathcal{K}(y)$ to obtain a h-periodic smooth kernel function $\tilde{\mathcal{K}}(y)$, which is $p-1$ times continuously differentiable in the periodic setting, where $p \in \mathbb{N}$ is the degree of smoothness, and the Fourier coefficients decay quickly.

Regularization of $\mathcal{K}(y)$. Assume that we have $\|x_j\| \le \frac{L}{2}$, i.e., $\|x_i - \tilde{x}_j\| \le L$, for some $L > 0$. We define the multivariate, h-periodic regularized kernel function $\tilde{\mathcal{K}} \colon [-\frac{h}{2}, \frac{h}{2}]^d \to \mathbb{R}$ with $h \ge 2L$ by

$$\tilde{\mathcal{K}}(y) := \begin{cases} \mathcal{K}(\|y\|) & \text{if } \|y\| \le L, \\ \mathcal{K}_{\mathrm{B}}(\|y\|) & \text{if } L < \|y\| \le \frac{h}{2}, \\ \mathcal{K}_{\mathrm{B}}(\frac{h}{2}) & \text{if } y \in [-\frac{h}{2}, \frac{h}{2}]^d \text{ and } \|y\| > \frac{h}{2}, \end{cases}$$

where K_{B} is an appropriately chosen univariate polynomial, which is constructed using two-point Taylor interpolation, see Figure 1. For a detailed interpretation of this approach, we refer to Plonka et al. [28, Chapter 7.5].
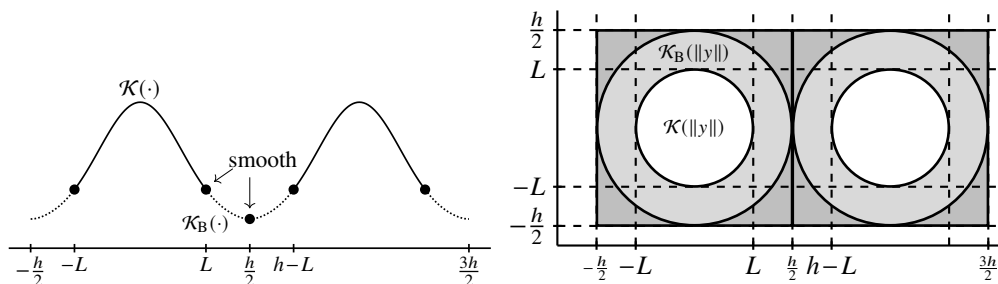

Figure 1: The regularized periodic function — dimension one (left) and the dimensions two (right), see Nestler [23, Figure 3.14].

Approximation of smooth periodic function $\tilde{\mathcal{K}}$. In the univariate case, we are now able to approximate the smooth periodic function $\tilde{\mathcal{K}}$ by a Fourier series to obtain

$$\tilde{t}_j := \sum_{i=1}^{n} \alpha_i \tilde{\mathcal{K}}(x_i - \tilde{x}_j) \approx \sum_{i=1}^{n} \alpha_i \mathcal{K}_{RK}(x_i - \tilde{x}_j), \quad j = 1, \dots, \tilde{n}.$$

By using (22) and interchanging the order of summation, as well as utilizing the outstanding property

$$e^{2\pi i (x_i - \tilde{x}_j)/h} = e^{2\pi i x_i/h}\, e^{-2\pi i \tilde{x}_j/h},$$

we obtain

$$t_i \approx \sum_{k \in \mathcal{I}_N} b_k \left(\sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j e^{-2\pi i k \tilde{x}_j/h} \right) e^{2\pi i k x_i/h}, \quad i = 1, \dots, n. \tag{25}$$

15

We compute the inner sum for each k $\in \mathcal{I}_N$ using the NFFT in $O(N \log N + \tilde{n})$ arithmetical operations and the outer sum with $O(N \log N + n)$.

This simple idea works very well, if the function $\mathcal{K}$ is smooth and can be approximated by a short Fourier series $\tilde{\mathcal{K}}$,i.e., by a small polynomial of degree N. This is especially true for the case $r = 2$, where the method is also known as *fast Gaussian transform*.

Remark 4.1. We note as well that large values of λ corresponds to a localization of the support points of the measure. In this setting, the set of support points can be ordered in reduced operations (as mentioned below), so that matrix-vector operations are eligible in the same time as our implementation. For this reason, our algorithm is primarily adapted for small values of λ. Nevertheless, it renders stable computation for sufficiently large values of λ.

Remark 4.2 (Arithmetic complexity). For $\lambda > 0$, the kernel approximation (22) is independent of n ($\tilde{n}$, resp.) data points, therefore we can appropriately fix the polynomial degree N. Thus, the approximation ends up with $O(n + \tilde{n})$ arithmetic operations. Furthermore, for $r = 1$, we need additional near-field regularisation at the point $y = 0$. In this case, we end up with an arithmetic complexity of $O(n \log n + \tilde{n} \log \tilde{n})$, see Plonka et al. [28, Chapter 7.5] for a detailed interpretation.
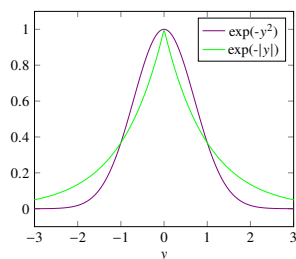

Figure 2: Wasserstein Distance for $r = 1$ (green) and $r = 2$ (violet).

4.2   NFFT boost for the Sinkhorn's algorithm

This section presents the NFFT-accelerated standard and log-domain Sinkhorn's algorithms. The NFFT-accelerated Sinkhorn's algorithms propose a novel method using non-equispaced convolution to approximate the Wasserstein distance. The algorithms below describe the operations of our proposed method schematically.

The NFFT-accelerated ERW distance (lower bound) including the entropy is denoted by

$$s_{r;\lambda;\text{NFFT}}(P, \tilde{P}),$$

which can be computed using Algorithm 3 and 4. This quantity is an approximation of $s_{r;\lambda}(P, \tilde{P})$, i.e., $s_{r;\lambda}(P, \tilde{P}) \approx s_{r;\lambda;\text{NFFT}}(P, \tilde{P})$. Furthermore, the ERW distance (upper bound) is computed by

$$\tilde{s}_{r;\lambda;\text{NFFT}}(P, \tilde{P}) = s_{r;\lambda;\text{NFFT}}(P, \tilde{P}) + \frac{H(P) + H(\tilde{P})}{\lambda}.$$

The NFFT-accelerated Sinkhorn divergence is computed by

$$sd_{r;\lambda;\text{NFFT}}(P, \tilde{P}) := s_{r;\lambda}(P, \tilde{P}) - \frac{1}{2} s_{r;\lambda;\text{NFFT}}(P, P) - \frac{1}{2} s_{r;\lambda;\text{NFFT}}(\tilde{P}, \tilde{P}).$$

16

Arithmetic complexity. For simplicity, we assume that $n = \tilde{n}$. As mentioned earlier, the evaluation of sums in Algorithm 3 and 4 take only $O(n)$ arithmetic operations for $r = 2$ and $O(n \log n)$ for $r = 1$. From Remark 3.9, we know that Sinkhorn's iteration process requires $O(\log n + \|d^r\|_\infty \lambda)$. Therefore, for $r = 2$, our proposed algorithms require only

$$O(n \log n + \|d^r\|_\infty \lambda),$$

and for $r = 1$

$$O(n (\log n)^2 + \|d^r\|_\infty \lambda).$$

Algorithm 3: NFFT-accelerated Sinkhorn's algorithm

Input: support nodes x_i and $\tilde{x}_j$, probability vectors $p \in \mathbb{R}^n_{\geq 0}$, $\tilde{p} \in \mathbb{R}^{\tilde{n}}_{\geq 0}$, regularization parameter $\lambda > 0$, threshold ε and
 starting value $\tilde{\alpha} = (\tilde{\alpha}_1, \dots, \tilde{\alpha}_{\tilde{n}})$
while $\|E^\Delta\| > \varepsilon$ **do**
 Set
$$\alpha^{(0)} := \mathbf{1}_n, \text{ and } \tilde{\alpha}^{(0)} := \mathbf{1}_{\tilde{n}}.$$

 if Δ *is odd* **then**
  compute
$$t_i^{\Delta-1} \leftarrow \sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j^{\Delta-1} e^{-\lambda\|x_i - \tilde{x}_j\|^r}, \quad i = 1, \dots, n,$$

  by employing the fast summation (25) and set

$$\alpha_i^\Delta \leftarrow \frac{p_i}{t_i^{\Delta-1}}, \quad i = 1, \dots, n;$$

$$\tilde{\alpha}_j^\Delta \leftarrow \tilde{\alpha}_j^{\Delta-1}, \quad j = 1, \dots, \tilde{n}.$$

 else
  compute
$$\tilde{t}_j^{\Delta-1} \leftarrow \sum_{i=1}^{n} e^{-\lambda\|x_i - \tilde{x}_j\|^r} \alpha_i^{\Delta-1}, \quad j = 1, \dots, \tilde{n},$$

  by employing the fast summation (25) and set

$$\tilde{\alpha}_j^\Delta \leftarrow \frac{\tilde{p}_j}{\tilde{t}_j^{\Delta-1}}, \quad j = 1, \dots, \tilde{n};$$

$$\alpha_i^\Delta \leftarrow \alpha_i^{\Delta-1}, \quad i = 1, \dots, n.$$

 increment $\Delta \leftarrow \Delta + 1$
Result: The ERW distance (cf. (6)) approximating the Wasserstein distance $W_r(P, \tilde{P})$ is

$$s_{r;\lambda;\text{NFFT}}(P, \tilde{P}) := \frac{1}{\lambda} + \frac{1}{\lambda}\sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} + \frac{1}{\lambda}\sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*} - \frac{1}{\lambda}\sum_{j=1}^{\tilde{n}} \tilde{t}_j^{\Delta^*} \tilde{\alpha}_j^{\Delta^*}.$$

Remark 4.3 (Optimal transition matrix π^{Δ^*}). The NFFT-accelerated Sinkhorn's Algorithm 3 and 4 bypass the allocations of the matrices d, k and π^{Δ^*} and returns the objective of the Sinkhorn's algorithm, i.e., the ERW distance $s_{r;\lambda}(P, \tilde{P})$ of the measures P and $\tilde{P}$. Our proposed algorithms provide the optimal exponentiated dual variables $(\alpha, \tilde{\alpha})$ (Algorithm 3) and optimal

---

Algorithm 4: NFFT-accelerated Sinkhorn's algorithm (log-domain)

Input: support nodes x_i and $\tilde{x}_j$, probability vectors $p \in \mathbb{R}^n_{\geq 0}$, $\tilde{p} \in \mathbb{R}^{\tilde{n}}_{\geq 0}$, regularization parameter $\lambda > 0$, threshold ε and starting value $\gamma = (\gamma_1, \ldots, \gamma_{\tilde{n}})$

while $\|E^\Delta\| > \varepsilon$ **do**

    Set

$$\beta^{(0)} := \mathbf{0}_n, \text{ and } \gamma^{(0)} := \mathbf{0}_{\tilde{n}}.$$

    if Δ *is odd* **then**

        compute

$$t_i^{\Delta-1} \leftarrow \sum_{j=1}^{\tilde{n}} e^{\lambda \gamma_j^{\Delta-1}-1/2} e^{-\lambda\|x_i-\tilde{x}_j\|^r}, \quad i = 1, \ldots, n,$$

        by employing the fast summation (25) and set

$$\beta_i^\Delta \leftarrow \frac{1}{\lambda}\Big(\log p_i - \log t_i^{\Delta-1}\Big), \quad i = 1, \ldots, n;$$
$$\gamma_j^\Delta \leftarrow \gamma_j^{\Delta-1}, \quad j = 1, \ldots, \tilde{n}.$$

    else

        compute

$$\tilde{t}_j^{\Delta-1} \leftarrow \sum_{i=1}^{n} e^{-\lambda\|x_i-\tilde{x}_j\|^r} e^{\lambda \beta_i^{\Delta-1}-1/2}, \quad j = 1, \ldots, \tilde{n},$$

        by employing the fast summation (25) and set

$$\gamma_j^\Delta \leftarrow \frac{1}{\lambda}\Big(\log \tilde{p}_j - \log \tilde{t}_j^{\Delta-1}\Big), \quad j = 1, \ldots, \tilde{n};$$
$$\beta_i^\Delta \leftarrow \beta_i^{\Delta-1}, \quad i = 1, \ldots, n.$$

    increment $\Delta \leftarrow \Delta + 1$

Result: The ERW distance (cf. (6)) approximating the Wasserstein distance $W_r(P, \tilde{P})$ is

$$s_{r;\lambda}(P, \tilde{P}) \approx \sum_{i=1}^{n} p_i \beta_i + \sum_{j=1}^{\tilde{n}} \tilde{p}_j \gamma_j - \frac{1}{\lambda} \sum_{j=1}^{\tilde{n}} \tilde{t}_j^{\Delta^*} e^{\lambda \gamma_j^{\Delta^*}-1/2}.$$

---

dual variables $(\beta^{\Delta^*}, \gamma^{\Delta^*})$ (Algorithm 4). Hence, the optimal transition matrix π^{Δ^*} can still be computed with (9). However, this – as mentioned – requires $O(n \cdot \tilde{n})$ operations, which would increase the performance time and thus is avoided.

The NFFT fast summation technique splendidly adapts to the Sinkhorn's algorithms. As mentioned eariler, this technique guarantees fast and memory efficient computation with machine accuracy.

5 Numerical Experiments

This section demonstrates the performance and accuracy of NFFT-accelerated Sinkhorn's Algorithm 3 using synthetic as well as real data sets. All runtime measurements were performed on a standard desktop computer with Intel(R) Core(TM) i7-7700 CPU and 15.0 GB of RAM. The source code of our implementation of standard Sinkhorn's Algorithm 1, log-domain stabilized Sinkhorn's Algorithm 2, NFFT accelerated Sinkhorn's Algorithm 3, NFFT accelerated log-domain Sinkhorn's Algorithm 4 and linear programming solver to compute $w_r(P, \tilde{P})$, which can be used

to reproduce the following results, are available in online.[†] The implementation of our proposed algorithms are based on the freely available repository 'NFFT3.jl'.[‡]

5.1 Synthetic data

We test in one-dimension (Section 5.1.1 below), and for two-dimensional data (Section 5.1.2) to demonstrate the performance of our proposed algorithm which still delivers results, which are out of reach for tradtional implementations.

5.1.1 NFFT-accelerated Sinkhorn's algorithm in one dimension

Consider a measure P on $\mathbb{R}$ with quantiles s_i, i.e.,

$$P\big((-\infty, s_i]\big) = \frac{i}{\tilde{n} + 1}, \qquad i = 1, \ldots, \tilde{n},$$

and corresponding weights

$$p_i := P\left(\left(\frac{s_{i-1} + s_i}{2}, \frac{s_i + s_{i+1}}{2}\right]\right), \qquad i = 1, \ldots, \tilde{n}.$$

The measure

$$\tilde{P}_{\tilde{n}} := \sum_{i=1}^{\tilde{n}} p_i \, \delta_{s_i}$$

is the best discrete approximation of P in Wasserstein distance (cf. Graf and Luschgy [14]).

To demonstrate the performance of Algorithm 3, we consider independent and identically distributed observations $X_i \in \mathbb{R}$, $i = 1, \ldots, n$, from the measure P, and the corresponding empirical measure

$$\hat{P}_n := \frac{1}{n} \sum_{i=1}^{n} \delta_{X_i}.$$

Table 1 compares the computation time of Sinkhorn's Algorithm 1, and the NFFT-accelerated Sinkhorn's Algorithm 3.

$n = \tilde{n}$:	1000	10 000	10 000	100 000	1 000 000	10 000 000
Sinkhorn's Algorithm 1	0.49 s	23.72 s	41.16 s	*out of memory or > 1 hour*		
NFFT-accelerated Sinkhorn 3	0.28 s	0.39 s	2.08 s	2.31 s	9.38 s	62.4 s

Table 1: Dimension 1: Comparison of computation times for $r = 2$ and $\lambda = 20$

The table demonstrates that the NFFT-accelerated Sinkhorn's Algorithm 3 easily delivers results for problem sizes, which are out of reach for the traditional Sinkhorn's Algorithm 1.

[†]Cf. https://github.com/rajmadan96/NFFT-Sinkhorn-Wasserstein_distance/
[‡]Cf. https://github.com/NFFT/NFFT3.jl

5.1.2 NFFT-accelerated Sinkhorn's in two dimension

We demonstrate next performance of the NFFT-accelerated Sinkhorn's Algorithm 3 by approximating the Wasserstein distance for empirical measures

$$P = \frac{1}{n} \sum_{i=1}^{n} \delta_{(U_i^1, U_i^2)} \text{ and } \tilde{P} = \frac{1}{\tilde{n}} \sum_{j=1}^{\tilde{n}} \delta_{(\tilde{U}_j^1, \tilde{U}_j^2)}$$

on $\mathbb{R} \times \mathbb{R}$, where $U_i^1, U_i^2, i = 1, \ldots, n$, and $\tilde{U}_j^1, \tilde{U}_j^2, j = 1, \ldots, \tilde{n}$, are independent samples from the uniform distribution.

Table 2 displays execution times for the uniform distribution on $[0, 1] \times [0, 1]$. While computation time and memory allocations are already critical for $n, \tilde{n} \approx 100\,000$, the NFFT-accelerated Sinkhorn's algorithm still performs in reasonable time.

$n = \tilde{n}$:	1000	10 000	100 000	1 000 000	10 000 000
Sinkhorn's Algorithm 1	0.39 s	22.9 s	*out of memory or > 1 hour*		
NFFT-accelerated Sinkhorn 3	0.31 s	0.42 s	2.0 s	7.2 s	59.4 s

Table 2: Two dimensions: comparison of computation times for $r = 2$ and $\lambda = 20$

5.2 Benchmark datasets

This section validates the regularization parameter λ, and demonstrates the performance and the accuracy of NFFT-accelerated Sinkhorn's Algorithm 3 using real datasets. We use a dataset called DOTmark (see Figure 3); DOT stands for *discrete optimal transport*. This benchmark dataset is specially designed to effectively test and compare optimal transport methods (cf. Schrieber et al. [35]). It has gray level representation of the images in the resolution of 32×32 to 512×512, and it consists of 10 subsets of dataset, ranging from smooth to rough structure.

Transformation of images to probability vectors. A grayscale digital image can be represented as a matrix, where each entry represents a pixel in the image and the value of the pixel is the image's gray scale level in the range $[0, 1]$ (see Figure 4). In order to convert the grayscale image matrices into probability vectors, we vectorize and normalize the matrices. Furthermore, intensities of background pixels are the ℓ_1 distance between pixels i and j of the respective grids $(32 \times 32, \ldots, 512 \times 512)$.
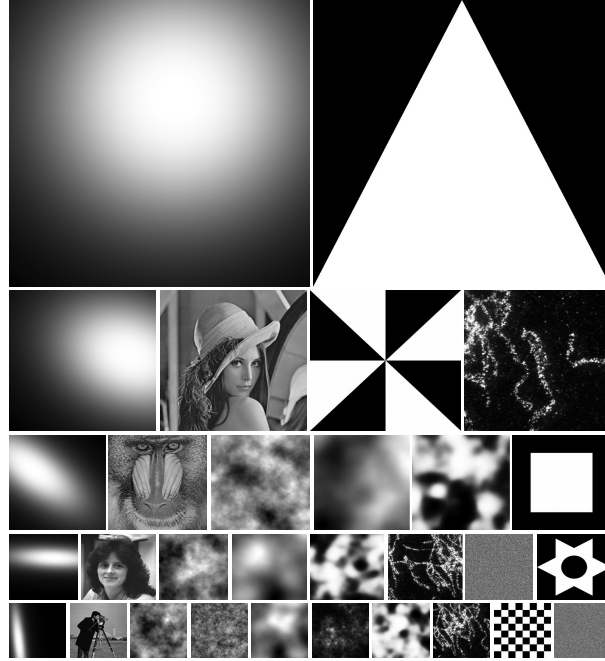
20

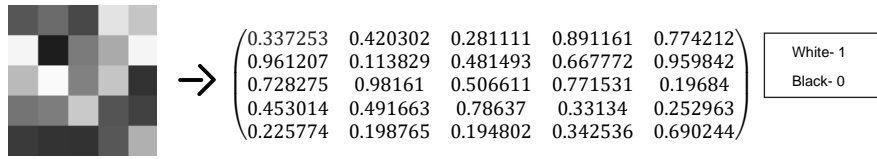Figure 3: Examples of the DOTmark database (from low to high resolution images)

$$\begin{pmatrix} 0.337253 & 0.420302 & 0.281111 & 0.891161 & 0.774212 \\ 0.961207 & 0.113829 & 0.481493 & 0.667772 & 0.959842 \\ 0.728275 & 0.98161 & 0.506611 & 0.771531 & 0.19684 \\ 0.453014 & 0.491663 & 0.78637 & 0.33134 & 0.252963 \\ 0.225774 & 0.198765 & 0.194802 & 0.342536 & 0.690244 \end{pmatrix}$$

White- 1

Black- 0

Figure 4: Grayscale image (5×5) is represented as a matrix

5.2.1 Validation of the regularization parameter λ

In this section we capture the behaviors of the lower and upper bounds (cf. Lemma 3.3) and Sinkhorn divergence, i.e., $s_{r;\lambda}(P, \tilde{P})$, $\tilde{s}_{r;\lambda}(P, \tilde{P})$, $sd_{r;\lambda}(P, \tilde{P})$ and $s_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$, $\tilde{s}_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$, $sd_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$ for different values of the entropy regularization parameter $\lambda \in \{10, 20, 50, \ldots, 200\}$.
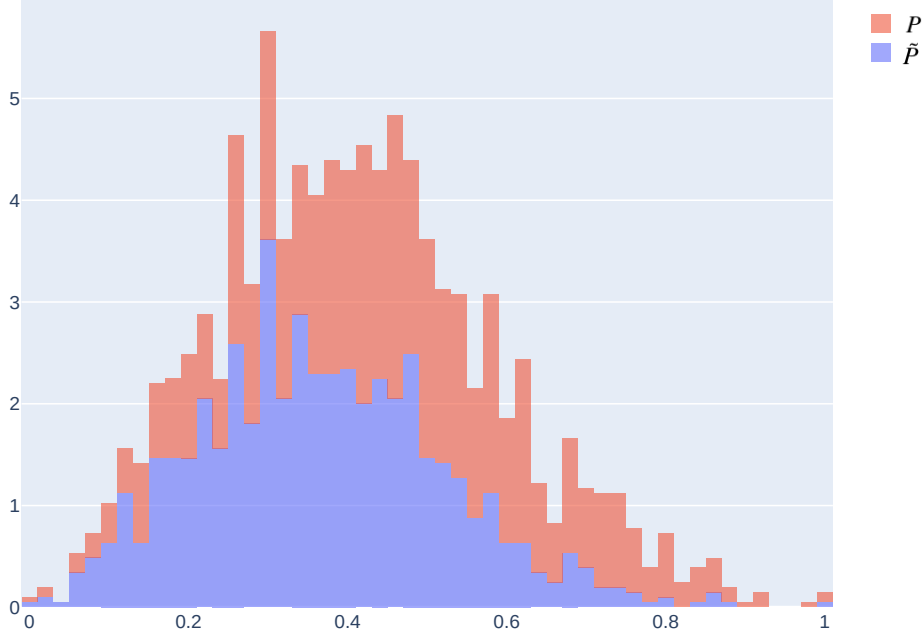
Figure 5: Histogram of the measures P and $\tilde{P}$ of the dataset GRFrough; n ($\tilde{n}$, resp.) = 1024

We use the 'GRFrough' dataset, which is a subset of images from the DOTmark dataset. Notably, it has a rough structure, relative to the other subset of images (see Figure 5). Figure 6 below investigate these quantities with respect to increasing λ. We infer that $s_{r;\lambda}(P, \tilde{P})$, $s_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$ converge slowly, for λ increasing, to $w_r(P, \tilde{P})$. However, $\tilde{s}_{r;\lambda}(P, \tilde{P})$ and $\tilde{s}_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$ converge quickly to $w_r(P, \tilde{P})$, and Sinkhorn divergence $\left(sd_{r;\lambda}(P, \tilde{P}), sd_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})\right)$ also converge quicker, in comparison to $s_{r;\lambda}(P, \tilde{P})$, $s_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$. The argument behind these behaviors is, for larger λ, the weightage of the entropy in the objective function (3a) decreases, and the matrices π^s and π^w coincide. Furthermore, the NFFT approximation is stable for different values of regularization parameter λ.

5.2.2 Performance analysis

This section extensively substantiates the performance of NFFT-accelerated Sinkhorn's Algorithm 3 in terms of time and memory allocation. For the experiments, we use the DOTmark dataset, ranging from 32×32 to 512×512 pixels in size, and we consider transports between two different images of equal size.

Table 3 and 4 compare the time and memory allocation of Sinkhorn's Algorithm 1 and NFFT
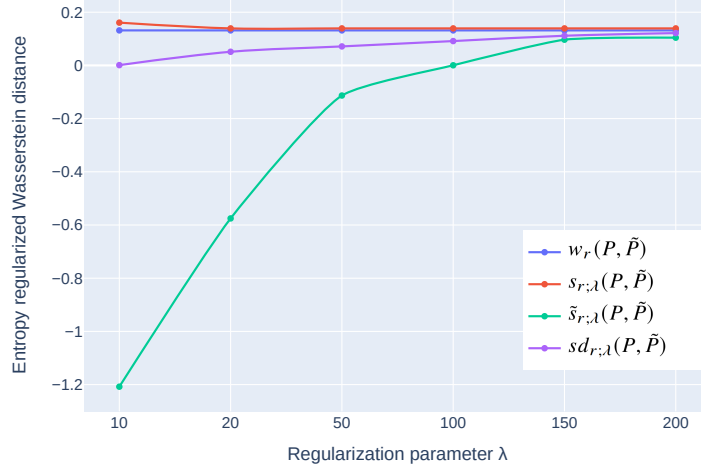
(a) Numerical results of Algorithm 1
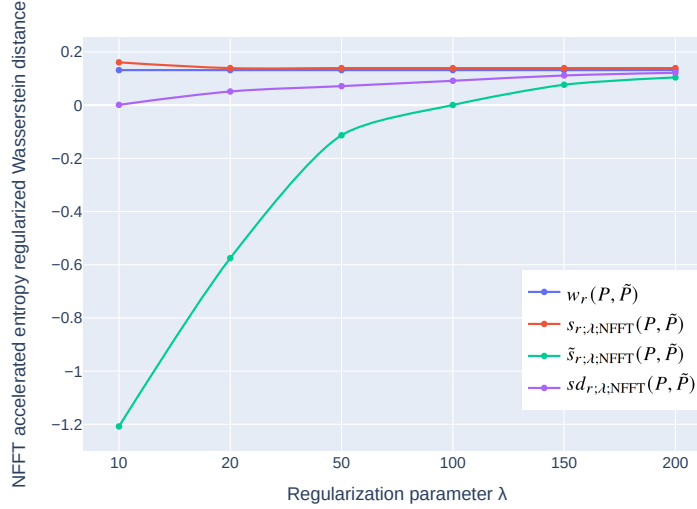
(b) Numerical results of Algorithm 3

Figure 6: Approximation of the Wasserstein distance using $s_{r;\lambda}(P, \tilde{P})$, $\tilde{s}_{r;\lambda}(P, \tilde{P})$, $sd_{r;\lambda}(P, \tilde{P})$ and $s_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$, $\tilde{s}_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$, $sd_{r;\lambda;\mathrm{NFFT}}(P, \tilde{P})$. The parameters are $\lambda \in \{10, 20, 50, \ldots, 200\}$ and $r = 2$.

accelarated Sinkhorn's Algorithm 3. Among all the tests, the computational time and memory allocation of our proposed algorithm is significantly better.

From the performance analysis, we infer that NFFT-accelerated Sinkhorn's Algorithm 3 utilizes lesser time and memory, in spite of high number of points n ($\tilde{n}$, resp.) for computations, and our device runs out of memory for Sinkhorn's Algorithm 1, when the problems are sized larger than 16384 n ($\tilde{n}$, resp.).

$n = \tilde{n}$	1024		4096		16 384		65 536		262 144	
Dataset: **DOTmark**	Alg. 1	Alg. 3	Alg. 1	Alg. 3	Alg. 1	Alg. 3	Alg. 1	Alg. 3	Alg. 1	Alg. 3
CauchyDensity	0.79 s	**0.32** s	3.91 s	**0.34** s	75.0 s	**1.08** s	-	**1.11** s	-	**3.70** s
ClassicImages	0.75 s	**0.29** s	3.53 s	**0.36** s	69.0 s	**1.11** s	-	**1.34** s	-	**3.69** s
GRFmoderate	1.51 s	**0.41** s	4.15 s	**0.53** s	93.16 s	**1.26** s	-	**1.11** s	-	**3.36** s
GRFrough	0.89 s	**0.39** s	3.72 s	**0.46** s	79.0 s	**1.31** s	-	**1.64** s	-	**3.81** s
GRFsmooth	1.92 s	**0.43** s	5.15 s	**0.61** s	102.13 s	**1.46** s	-	**1.81** s	-	**3.52** s
LogGRF	2.19 s	**0.52** s	5.31 s	**0.69** s	105.10 s	**1.78** s	-	**2.11** s	-	**4.71** s
LogitGRF	2.01 s	**0.49** s	5.22 s	**0.63** s	104.20 s	**1.58** s	-	**2.01** s	-	**4.62** s
MicroscopyImages	0.53 s	**0.21** s	3.34 s	**0.23** s	67.9 s	**0.72** s	-	**0.91** s	-	**1.86** s
Shapes	0.61 s	**0.26** s	3.64 s	**0.29** s	72.4 s	**0.92** s	-	**1.01** s	-	**2.56** s
WhiteNoise	0.63 s	**0.29** s	3.82 s	**0.31** s	73.0 s	**1.02** s	-	**1.07** s	-	**2.70** s

Table 3: Comparison of computational time allocation of Sinkhorn's Algorithm 1 and NFFT-accelerated Sinkhorn's Algorithm 3; $\lambda = 20$ and $r = 2$

$n = \tilde{n}$	1024		4096		16 384		65 536		262 144	
Dataset	Alg. 1	Alg. 3	Alg. 1	Alg. 3	Alg. 1	Alg. 3	Alg. 1	Alg. 3	Alg. 1	Alg. 3
DOTmark	(MB)		(MB)		(MB)		(MB)		(MB)	
CauchyDensity	363.63	**37.11**	924.74	**51.94**	9032.14	**105.99**	-	**289.99**	-	**321.23**
ClassicImages	353.14	**35.31**	913.16	**49.14**	9000.11	**104.19**	-	**274.29**	-	**311.29**
GRFmoderate	412.12	**41.21**	941.23	**65.54**	9420.35	**135.17**	-	**301.15**	-	**333.52**
GRFrough	341.34	**33.11**	912.13	**43.12**	8909.13	**99.12**	-	**263.21**	-	**301.21**
GRFsmooth	442.13	**51.21**	981.41	**85.34**	9740.31	**149.19**	-	**331.19**	-	**373.12**
LogGRF	463.63	**57.11**	1124.14	**91.94**	9831.34	**155.99**	-	**359.19**	-	**391.95**
LogitGRF	451.23	**52.41**	1023.11	**89.14**	9800.54	**151.49**	-	**349.29**	-	**384.13**
MicroscopyImages	250.11	**19.41**	912.17	**35.17**	8611.39	**55.12**	-	**221.74**	-	**243.21**
Shapes	311.17	**29.41**	922.12	**41.13**	8751.27	**95.13**	-	**241.21**	-	**283.61**
WhiteNoise	321.12	**31.41**	932.13	**42.11**	8800.12	**97.11**	-	**253.22**	-	**299.31**

Table 4: Comparison of computational memory allocation of Sinkhorn's Algorithm 1 and NFFT-accelerated Sinkhorn's Algorithm 3; $\lambda = 20$ and $r = 2$

5.2.3 Accuracy analysis

We validate the computational accuracy of NFFT-accelerated Sinkhorn's Algorithm 3. Throughout the accuracy analysis, we use $\tilde{s}_{r;\lambda}(P, \tilde{P})$ and $\tilde{s}_{r;\lambda;\text{NFFT}}(P, \tilde{P})$, since it is a better approximation of Wasserstein distance (see Section 5.2.1). Initially, we perform the accuracy analysis using the low resolution images from the DOTmark dataset. From Table 5, we notice that NFFT-accelerated Sinkhorn's Algorithm 3 has achieved stable approximation without compromise in accuracy.

Now, we move on to high resolution images of the 'GRFrough' dataset. Table 6 comprises the list of values that enables us to understand the approximation accuracy, as we move from low to high resolution images. We compare the results of Sinkhorn's Algorithm 1 with NFFT-accelerated Sinkhorn's Algorithm 3 for the problems sized up to 16 384 n ($\tilde{n}$, resp.), and we infer that there is no compromise in accuracy. We recognize that advancing to high resolution images does not

$n \times \tilde{n} = 1024 \times 1024$	Wasserstein	Sinkhorn 1	NFFT-accelerated Sinkhorn 3
Dataset: **DOTmark**	$w_r(P, \tilde{P})$	$\tilde{s}_{r;\lambda}(P, \tilde{P})$	$\tilde{s}_{r;\lambda;\text{NFFT}}(P, \tilde{P})$
CauchyDensity	0.120 498 1	0.120 499 2	0.120 499 2
ClassicImages	0.062 984 7	0.062 984 8	0.062 984 8
GRFmoderate	0.060 086 1	0.060 962 4	0.060 962 4
GRFrough	0.032 428 6	0.032 714 9	0.032 714 9
GRFsmooth	0.142 234 6	0.145 611 8	0.145 611 8
LogGRF	0.126 029 4	0.126 735 6	0.126 735 6
LogitGRF	0.107 569 8	0.107 778 4	0.107 778 4
MicroscopyImages	0.092 038 2	0.092 417 9	0.092 417 9
Shapes	0.143 584 7	0.144 727 9	0.144 727 9
WhiteNoise	0.020 609 3	0.020 690 0	0.020 690 0

Table 5: Comparison of accuracy to compute or approximate the Wasserstein distance in low-resolution images; $\lambda = 20$ and $r = 2$

Dataset: GRFrough	Wasserstein		Sinkhorn 1		NFFT-accelerated Sinkhorn 3	
$n \times \tilde{n}$	$w_r(P, \tilde{P})$	time	$\tilde{s}_{r;\lambda}(P, \tilde{P})$	time	$\tilde{s}_{r;\lambda;\text{NFFT}}(P, \tilde{P})$	time
1024×1024	0.032 428 6	72.34 s	0.032 714 9	0.89 s	0.032 714 9	0.39 s
4096×4096	*out of memory*		0.087 294 6	3.72 s	0.087 294 6	0.46 s
16384×16384	*out of memory*		0.124 975 1	79.0 s	0.124 975 1	1.31 s
65536×65536	*out of memory*		*out of memory*		0.743 219 2	1.64 s
262144×262144	*out of memory*		*out of memory*		0.911 743 6	3.81 s

Table 6: Comparison of computation times and accuracy to compute or approximate the Wasserstein distance, from low to high resolution images; $\lambda = 20$ and $r = 2$

affect the stability of approximation. Due to the break of Sinkhorn's Algorithm 1, it cannot be used as comparison factor when the size of the problem is beyond 16 384 n ($\tilde{n}$, resp.). However, our proposed algorithm computes the largest problem available in the DOTmark dataset, which is of size 262 144 n ($\tilde{n}$, resp.).

5.3 Comparisons and further steps

In this section, we substantiate the historical evolution of the prominent algorithms, which approximate the Wasserstein distance. Furthermore, we discuss the supremacy and the direction of further development of our proposed algorithms.

Historical remarks. The approach of entropy regularization of the Wasserstein distance by Cuturi [9] is a well-known path breaking approach to approximate the Wasserstein distance, which is effectively computed by Sinkhorn's algorithm. Later on, many constructive approaches and/ or analyses were contributed to improve and/ or support the entropy regularization approach

(cf. Altschuler et al. [1], Dvurechensky et al. [10]). In 2019, the approach of log-domain stabilization and truncated kernel of the Sinkhorn's algorithm was proposed by Schmitzer [34]. The log-domain stabilization method satisfies the demand for larger regularization parameters λ, and the truncated kernel reduces memory demand and also accelerates the iterations. In the same article, a multi-scale scheme was also proposed, which enables more efficient computations of the kernel truncated approach. As discussed in Remark 3.10, these prominent approaches still suffer by the entropy bias. In order to remove/ reduce the bias, Sinkhorn divergence was proposed by Ramdas et al. [29].

These prominent approaches affirm the progressive improvement of the algorithm, which approximates the Wasserstein distance. However, notably, these approaches still deteriorate by the matrix-vector operations, which is the bottleneck of the algorithms.

Name of algorithm / method	Algorithm	denotement
Standard Sinkhorn (Cuturi [9])	Algorithm 1	Std. Sinkhorn
Stabilized log-domain Sinkhorn (Schmitzer [34])	Algorithm 2	Stb. log Sinkhorn
Sinkhorn divergence (Ramdas et al. [29])	Algorithm 1	$sd_{r;\lambda}(P,\tilde{P})$
Multi-scale Sinkhorn (Schmitzer [34])	Algorithm ‡‡	Multi Sinkhorn
NFFT-accelerated Sinkhorn	Algorithm 3	NFFT Sinkhorn
NFFT-accelerated log-domain Sinkhorn	Algorithm 4	NFFT log Sinkhorn
NFFT-accelerated Sinkhorn divergence	Algorithm 3	$sd_{r;\lambda;\mathrm{NFFT}}(P,\tilde{P})$

Table 7: List of algorithms

$n = \tilde{n}$	1024	4096	16 384	65 536	262 144
Algorithm	(MB)	(MB)	(MB)	(MB)	(MB)
Std. Sinkhorn	27.44	381.10	6644.10	*out of memory*	
Stb. log Sinkhorn	103.10	487.48	6943.12	*out of memory*	
$sd_{r;\lambda}(P,\tilde{P})$	51.34	391.49	7139.12	*out of memory*	
Multi Sinkhorn	1.9	19.49	32.13	*out of memory*	
NFFT Sinkhorn	**1.7**	**2.06**	**8.98**	**37.5**	**132.0**
NFFT log Sinkhorn	**1.79**	**2.10**	**9.31**	**43.1**	**142.7**
$sd_{r;\lambda;\mathrm{NFFT}}(P,\tilde{P})$	**2.3**	**3.19**	**12.78**	**45.29**	**162.0**

Table 8: Comparison of computational memory allocations of algorithms in Table 7; $\lambda = 20$ and $r = 2$

Now, we compare our proposed algorithms with prominent algorithms, which are discussed so far (see Table 7). We would like to emphasize that our proposed algorithms are compatible even with low-threshold applications, this does not require expensive hardware or having access

---

‡‡Cf. https://github.com/OTGroupGoe/MultiScaleOT.jl
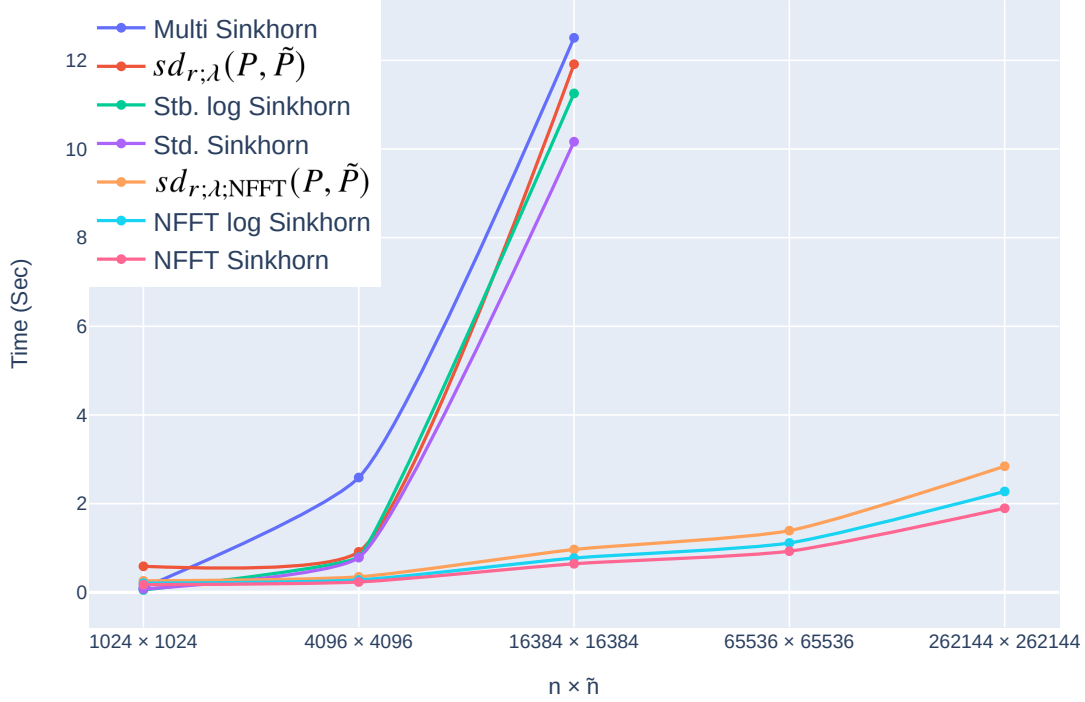
Figure 7: Comparison of computational time allocations of algorithms in Table 7; $\lambda = 20$ and $r = 2$

to supercomputers. All the algorithms involved in the comparison including our proposed algorithms follow Central Processing Unit (CPU) implementation paradigms. We follow the same experimentally setup utilized in preceding Section 5.2.3, and we use the 'GRFrough' dataset. From Figure 7 and Table 8, it is evident that our proposed algorithms perform significantly better in terms of time and memory allocations. Our device runs out of memory for all algorithms/ methods, except our proposed algorithms, when the problems are sized larger than $16\,384$ n ($\tilde{n}$, resp.). In terms of memory allocations, the Multi Sinkhorn algorithm shows significant performance, and the results are closer to our proposed algorithms. However, it requires more computational time, and it breaks due to the kernel matrix formation, when the problems are sized larger than $16\,384$ n ($\tilde{n}$, resp.).

Faster computation. In general, for faster computations, Graphics Processing Unit (GPU) implementations are used. The 'GeomLoss' package is a clever GPU implementation to approximate the Wasserstein distance or to solve the OT problem. We refer to Feydy [11] and the corresponding GitHub repository for further information on the implementation. As mentioned in Section 1, for a fast computation, low rank approximation techniques are also considered.

However, the 'GeomLoss' is the prominent contribution in terms of fast computation.

Further steps. Our proposed algorithms surpass the burden of time and memory allocations, and it is also flexible to adapt to the log-domain implementation. However, further research directions will be focused on applying our algorithms to suitable applications, and incorporation of possible extensions. In order to reach wider audiences, the GPU implementation of our proposed algorithms can be considered as one of the possible extensions as well. We would like to mention that GPU implementation of NFFT algorithm is readily available in corresponding GitHub repository.[§]

6 Summary

The nonequispaced fast Fourier transform, as presented in this article, allows computing a proper approximation of the Wasserstein distance in $O(n \log n)$ arithmetic operations. NFFT-accelerated Sinkhorn's Algorithm 3 performs significantly better than standard Sinkhorn's Algorithm 1, in terms of computational time and memory allocations. Our numerical results demonstrate the effectiveness of the new method as well as the tightness of our theoretical bounds. We believe that our algorithms can be widely used in data sciene applications for handling large-scale dataset.

References

[1] J. Altschuler, J. Weed, and P. Rigollet. Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1961–1971. Curran Associates Inc., 2017. URL https://arxiv.org/abs/1705.09634. 10, 12, 26

[2] J. Altschuler, F. Bach, A. Rudi, and J. Niles-Weed. Massively scalable sinkhorn distances via the nyström method. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/f55cadb97eaff2ba1980e001b0bd9842-Paper.pdf. 2

[3] J. M. Altschuler and E. Boix-Adsera. Polynomial-time algorithms for multimarginal optimal transport problems with structure. *arXiv preprint arXiv:2008.03006*, 2020. URL https://arxiv.org/abs/2008.03006. 2

[4] F. A. Ba and M. Quellmalz. Accelerating the Sinkhorn algorithm for sparse multi-marginal optimal transport by fast Fourier transforms, 2022. URL https://arXiv.org/abs/2208.03120. 2

[5] I. Bilik, O. Longman, S. Villeval, and J. Tabrikian. The rise of radar for autonomous vehicles: Signal processing solutions and future research directions. *IEEE signal processing Magazine*, 36(5):20–31, 2019. doi:10.1109/MSP.2019.2926573. 2

[§]Cf. https://github.com/sukunis/CUNFFT/tree/master/src

[6] F. Bolley. Separability and completeness for the Wasserstein distance. In C. Donati-Martin, M. Émery, A. Rouault, and C. Stricker, editors, *Séminaire de Probabilités XLI*, volume 1934 of *Lecture Notes in Mathematics*, pages 371–377. Springer, Berlin, Heidelberg, 2008. doi:10.1007/978-3-540-77913-1. 3

[7] S. Chakraborty, D. Paul, and S. Das. Hierarchical clustering with optimal transport. *Statistics & Probability Letters*, 163:108781, 2020. doi:10.1016/j.spl.2020.108781. 2

[8] N. Courty, R. Flamary, and D. Tuia. Domain adaptation with regularized optimal transport. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 274–289. Springer, 2014. doi:10.1007/978-3-662-44848-9. 2

[9] M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013. URL https://proceedings.mlr.press/v89/feydy19a.html. 1, 5, 25, 26

[10] P. Dvurechensky, A. Gasnikov, and A. Kroshnin. Computational optimal transport: Complexity by accelerated gradient descent is better than by Sinkhorn's algorithm. In *International conference on machine learning*, pages 1367–1376. PMLR, 2018. URL http://proceedings.mlr.press/v80/dvurechensky18a.html. 2, 10, 12, 26

[11] J. Feydy. *Geometric data analysis, beyond convolutions*. PhD thesis, Université Paris-Saclay Gif-sur-Yvette, France, 2020. URL https://www.math.ens.psl.eu/~feydy/geometric_data_analysis_draft.pdf. 5, 27

[12] A. V. Gasnikov, E. Gasnikova, Y. E. Nesterov, and A. Chernov. Efficient numerical methods for entropy-linear programming problems. *Computational Mathematics and Mathematical Physics*, 56(4):514–524, 2016. URL https://link.springer.com/content/pdf/10.1134/S0965542516040084.pdf. 5

[13] A. Genevay. *Entropy-regularized optimal transport for machine learning*. PhD thesis, Paris Sciences et Lettres (ComUE), 2019. URL https://www.theses.fr/2019PSLED002. 5

[14] S. Graf and H. Luschgy. *Foundations of Quantization for Probability Distributions*, volume 1730 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 2000. doi:10.1007/BFb0103945. 19

[15] N. Hao, M. E. Kilmer, K. Braman, and R. C. Hoover. Facial recognition using tensor-tensor decompositions. *SIAM Journal on Imaging Sciences*, 6(1):437–463, 2013. doi:10.1137/110842570. URL https://doi.org/10.1137/110842570. 2

[16] B. Kalantari, I. Lari, F. Ricca, and B. Simeone. On the complexity of general matrix scaling and entropy minimization via the ras algorithm. *Mathematical Programming*, 112 (2):371–401, 2008. doi:10.1007/s10107-006-0021-4. 10, 11

[17] J. Keiner, S. Kunis, and D. Potts. NFFT 3.5, C subroutine library. http://www.tu-chemnitz.de/~potts/nfft. Contributors: F. Bartel, M. Fenn, T. Görner, M. Kircheis, T. Knopp, M. Quellmalz, M. Schmischke, T. Volkmer, A. Vollrath. 14

[18] B. Khalil Abid and R. M. Gower. Greedy stochastic algorithms for entropy-regularized optimal transport problems. *arXiv e-prints*, pages arXiv–1803, 2018. URL http://proceedings.mlr.press/v84/abid18a/abid18a.pdf. 10, 12

[19] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger. From word embeddings to document distances. In *International conference on machine learning*, pages 957–966. PMLR, 2015. URL https://proceedings.mlr.press/v37/kusnerb15.html. 2

[20] T. Lin, N. Ho, and M. I. Jordan. On the efficiency of Sinkhorn and Greenkhorn and their acceleration for optimal transport. *arXiv preprint arXiv:1906.01437*, 2019. URL https://proceedings.mlr.press/v97/lin19a.html. 2

[21] G. Luise, A. Rudi, M. Pontil, and C. Ciliberto. Differential properties of sinkhorn approximation for learning with wasserstein distance. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/3fc2c60b5782f641f76bcefc39fb2392-Paper.pdf. 6

[22] A. Mensch and G. Peyré. Online sinkhorn: Optimal transport distances from sample streams. *Advances in Neural Information Processing Systems*, 33: 1657–1667, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/123650dd0560587918b3d771cf0c0171-Abstract.html. 2

[23] F. Nestler. *Efficient Computation of Electrostatic Interactions in Particle Systems Based on Nonequispaced Fast Fourier Transforms*. Dissertation. Universitätsverlag Chemnitz, 2018. ISBN 978-3-96100-054-8. URL http://nbn-resolving.de/urn:nbn:de:bsz:ch1-qucosa2-233760. 15

[24] S. Neumayer and G. Steidl. From optimal transport to discrepancy. *Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging: Mathematical Imaging and Vision*, pages 1–36, 2021. URL https://link.springer.com/content/pdf/10.1007/978-3-030-03009-4_95-1.pdf. 5

[25] N. Papadakis, G. Peyré, and E. Oudet. Optimal transport with proximal splitting. *SIAM Journal on Imaging Sciences*, 7(1):212–238, 2014. doi:10.1137/130920058. 2

[26] G. Peyré and M. Cuturi. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5–6):355–607, 2019. URL https://ieeexplore.ieee.org/document/8641476. 2, 7

[27] R. B. Platte, L. N. Trefethen, and A. B. Kuijlaars. Impossibility of fast stable approximation of analytic functions from equispaced samples. *SIAM review*, 53(2):308–318, 2011. doi:10.1137/090774707. 2

[28] G. Plonka, D. Potts, G. Steidl, and M. Tasche. *Numerical Fourier Analysis*. Applied and Numerical Harmonic Analysis. Birkhäuser, 2018. ISBN 978-3-030-04305-6. doi:10.1007/978-3-030-04306-3. 2, 14, 15, 16

[29] A. Ramdas, N. García Trillos, and M. Cuturi. On wasserstein two-sample testing and related families of nonparametric tests. *Entropy*, 19(2):47, 2017. URL https://www.mdpi.com/1099-4300/19/2/47. 13, 26

[30] S. Revay and M. Teschke. Multiclass language identification using deep learning on spectral images of audio signals. *arXiv preprint arXiv:1905.04348*, 2019. URL https://arxiv.org/abs/1905.04348. 2

[31] G. Rote and M. Zachariasen. Matrix scaling by network flow. In N. Bansal, K. Pruhs, and C. Stein, editors, *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 848–854. SIAM, 2007. URL http://dl.acm.org/citation.cfm?id=1283383.1283474. 4

[32] M. Scetbon and M. Cuturi. Linear time sinkhorn divergences using positive features. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13468–13480. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/9bde76f262285bb1eaeb7b40c758b53e-Paper.pdf. 2

[33] M. Scetbon, M. Cuturi, and G. Peyré. Low-rank sinkhorn factorization. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9344–9354. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/scetbon21a.html. 2, 5

[34] B. Schmitzer. Stabilized sparse scaling algorithms for entropy regularized transport problems. *SIAM Journal on Scientific Computing*, 41(3):A1443–A1481, 2019. doi:10.1137/16M1106018. 2, 26

[35] J. Schrieber, D. Schuhmacher, and C. Gottschlich. Dotmark–a benchmark for discrete optimal transport. *IEEE Access*, 5:271–282, 2016. doi:10.1109/ACCESS.2016.2639065. 20

[36] R. Sinkhorn. Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, 74(4):402, 1967. doi:10.2307/2314570. 1, 9

[37] R. Sinkhorn and P. Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21:343–348, 1967. ISSN 0030-8730. 9

[38] K. S. Tai, P. D. Bailis, and G. Valiant. Sinkhorn label allocation: Semi-supervised classification via annealed self-training. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10065–10075. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/tai21a.html. 2

[39] C. Villani. *Topics in Optimal Transportation*, volume 58 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2003. ISBN 0-821-83312-X. doi:10.1090/gsm/058. URL http://books.google.com/books?id=GqRXYFxe0l0C. 3

[40] C. Villani. *Optimal transport, old and new*, volume 338 of *Grundlehren der Mathematischen Wissenschaften*. Springer, Berlin, 2009. doi:10.1007/978-3-540-71050-9. 1