Electronic Transactions on Numerical Analysis.
Volume 58, pp. 289–315, 2023.
Copyright © 2023, Kent State University.
ISSN 1068–9613.
DOI: 10.1553/etna_vol58s289

ETNA
Kent State University and
Johann Radon Institute (RICAM)

NONEQUISPACED FAST FOURIER TRANSFORM BOOST FOR THE SINKHORN ALGORITHM[*]

RAJMADAN LAKSHMANAN[†], ALOIS PICHLER[†‡§], AND DANIEL POTTS[†‡¶]

Abstract. This contribution features an accelerated computation of Sinkhorn's algorithm, which approximates the Wasserstein transportation distance, by employing nonequispaced fast Fourier transforms (NFFT). The proposed algorithm allows approximations of the Wasserstein distance by involving not more than $\mathcal{O}(n \log n)$ 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.

Key words. Sinkhorn's divergence, optimal transport, NFFT, entropy

AMS subject classifications. 90C08, 90C15, 60G07

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, the costs correspond to the distance between locations. For a comprehensive discussion of the Wasserstein distance from a mathematical perspective, we may refer to [40].

The concept of entropy regularization of the Wasserstein distance, proposed by [9], is an important touchstone, which improves the computational process of traditional methods. This entropy-regularized Wasserstein problem is efficiently solved using Sinkhorn's algorithm (cf. [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 [19], clustering [7], image classification [38], or domain adaptation [8]. Notably, most of the applications rely on *discrete* measures. However, some significant contributions are also presented in the literature to support the arguments of semi-discrete and/or continuous measures (cf. [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. [10, 20, 34]). However, this article addresses the efficient computation of the standard Sinkhorn's algorithm in terms of time and memory allocation to approximate the Wasserstein distance on 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 representations for various problems. The family of standard FFT algorithms has been applied in many areas such as face recognition [15], autonomous vehicles [5], voice assistance [30], etc., and they have achieved notable performance. The standard FFT algorithm improves the computational operations from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$, where n denotes the number of data points; this process involves equispaced sampling. However, in some cases, the equispaced

[*]Received August 15, 2022. Accepted February 13, 2023. Published online on March 21, 2023. Recommended by L. Reichel.

[†]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 sampling is one of the root causes of failure to achieve high accuracy (cf. [27], [28, Chapter 7]). We recognize that the optimal transport (OT) communities use the idea of standard FFT to speed up Sinkhorn's iterations in some places [25]. The standard FFT methods utilize equispaced convolution, which is a drawback when we consider the stability of the computation and approximation accuracy (cf. [26, Remark 4.18]). To overcome this challenge we present a *non*-equispaced convolution below, and it is also achievable in $\mathcal{O}(n\log n)$ arithmetic operations. Furthermore, for faster computation, low-rank factorization techniques are considered a popular issue among OT communities (cf. [1, 3, 32]). As a consequence of the line of research on low-rank factorization, the authors in [33] have developed an algorithm to efficiently solve the regularized OT problem, which depends on *low-rank couplings*. This method can also be employed to accelerate problems involving multi-marginals; cf. [4].

Contribution. We improve the computational time and memory allocation of the standard entropy-regularization approach to approximate the 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 [28, Chapter 7]. Using NFFT, we boost the performance of the standard entropy regularization of the Wasserstein distance with stable computation and high (machine) accuracy. Additionally, we explicitly provide 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 notation and definitions of the 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 a demonstration of the 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 Wasserstein distance.

On a space of probability measures, Wasserstein distances offer a natural metric. Intuitively, the Wasserstein distance measures the minimum average amount of the 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*

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

where

$$w_r(P,\tilde{P}) \coloneqq \inf_{\pi\in\Pi(P,\tilde{P})} \iint_{\mathcal{X}\times\mathcal{X}} d(x,\,\tilde{x})^r\,\pi(\mathrm{d}x,\mathrm{d}\tilde{x}). \tag{2.1}$$

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 measurable sets A and $B\subset\mathcal{X}$.

Wasserstein distances metricize the weak* topology on measures with finite rth moment. In the discrete setting considered below and all regular situations, the infimum in (2.1) is attained (cf. [39, Theorem 7.12]).

Discrete framework. Concrete implementations of the Wasserstein problem rely on discrete measures of the form[1]

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

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 [6].

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 $\pi = \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij} \delta_{(x_i, \tilde{x}_j)} \in \mathcal{P}(\mathcal{X}^2)$ solves the Wasserstein problem (2.1) provided that the matrix $\pi = (\pi_{ij}) \in \mathbb{R}^{n \times \tilde{n}}$ is the solution of the optimization problem

$$w_r(P, \tilde{P}) = \min \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \pi_{ij}\, d_{ij}^r, \qquad \text{where} \tag{2.2a}$$

$$\sum_{j=1}^{\tilde{n}} \pi_{ij} = p_i \qquad \text{for } i = 1, \dots, n, \tag{2.2b}$$

$$\sum_{i=1}^{n} \pi_{ij} = \tilde{p}_j \qquad \text{for } j = 1, \dots, \tilde{n} \;\text{ and} \tag{2.2c}$$

$$\pi_{ij} \geq 0 \qquad \text{for all } i = 1, \dots, n \text{ and } j = 1, \dots, \tilde{n} \tag{2.2d}$$

and $d_{ij} = d(x_i, x_j)$ is the distance matrix. The problem (2.2a)–(2.2d) is a linear optimization problem, occasionally referred to as the *Kantorovich problem*. In what follows, the optimal matrix is denoted by π^w.

Complexity. For $n \approx \tilde{n}$, the linear optimization problem (2.2a)–(2.2d) can be solved by straightforward computation involving $\mathcal{O}(n^3)$ multiplications.

In the following Section 3, we recall the popular approach based on entropy regularization to reduce the computational burden of the optimization problem (2.2a)–(2.2d). 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. [31]).

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

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

[1]The Dirac measure located at $x \in \mathcal{X}$ is defined by $\delta_x(A) \coloneqq \mathbf{1}_A(x) = 1$ if $x \in A$ and 0 else.

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

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

added to the linear optimization problem (2.2a)–(2.2d).

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

$$s_{r;\lambda}(P,\tilde{P}) := \min \sum_{i=1}^{n}\sum_{j=1}^{\tilde{n}} \pi_{ij}\, d_{ij}^r - \frac{1}{\lambda} H(\pi), \qquad \textit{where} \tag{3.1a}$$

$$\sum_{j=1}^{\tilde{n}} \pi_{ij} = p_i \quad \textit{for } i = 1,\dots,n, \tag{3.1b}$$

$$\sum_{i=1}^{n} \pi_{ij} = \tilde{p}_j \quad \textit{for } j = 1,\dots,\tilde{n} \ \textit{and} \tag{3.1c}$$

$$\pi_{ij} > 0 \quad \textit{for all } i = 1,\dots,n \textit{ and } j = 1,\dots,\tilde{n}. \tag{3.1d}$$

The matrix minimizing (3.1a) subject to the constraints (3.1b)–(3.1d) is denoted as $\pi^s \in \mathbb{R}^{n\times\tilde{n}}$. Further, we set

$$\tilde{s}_{r;\lambda} := s_{r;\lambda} + \frac{1}{\lambda} H(\pi^s) = \sum_{i=1}^{n}\sum_{j=1}^{\tilde{n}} \pi_{ij}^s\, d_{ij}^r. \tag{3.2}$$

The non-negativity constraint (2.2d) is notably not active in the constraints (3.1b)–(3.1d), 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 (3.1a)–(3.1d) exists and is unique.

REMARK 3.2 (Regularizing term). To circumvent the difficulty of the numerical computation of the linear optimization problem (2.2a)–(2.2d), the entropy regularization approach was originally proposed in [9]. We also refer to [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 (3.1a) and the arguments below, we infer that for $\lambda \to \infty$, we obtain the standard Wasserstein distance in the limit. We refer to [24], where the regularization parameter is studied. The constructive line of research by [11] affirms that when the regularization parameter λ is not sufficiently large, the transportation plan and the regularized Wasserstein distance may be incompatible. However, from the literature, we infer that the choice $\lambda \geq 20$ is a good compromise between accuracy and computational speed (cf. [13, Figure 1.2, Figures 3.1–3.3], [33, Figure 2], [24, Section 7])[2]. To acquire a better approximation accuracy, we can increase the regularization parameter $\lambda > 20$ with the cost of more arithmetic operations.

[2]cf. https://marcocuturi.net/SI.html

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 describes how to choose λ to obtain a prescribed accuracy.

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

$$s_{r;\lambda}(P,\tilde{P}) \le w_r(P,\tilde{P})^r \le \tilde{s}_{r;\lambda}(P,\tilde{P}) \le s_{r;\lambda}(P,\tilde{P}) + \varepsilon, \tag{3.3}$$

provided that

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

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

Proof. The first inequality in (3.3) follows by substituting the matrix π^w into (3.1a), as $H(\pi^w) > 0$. Further, with the matrix π^s, it holds that $w_{r;\lambda}(P,\tilde{P}) \le \tilde{s}_{r;\lambda}(P,\tilde{P})$, which is the second inequality using (3.2).

Now let π be any matrix with marginals p (cf. (2.2b)) and $\tilde{p}$ (cf. (2.2c)). It follows from the log-sum inequality (or Gibbs' inequality)

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

that

$$\begin{aligned}\sum_{i,j} \pi_{ij} \log \pi_{ij} &\ge \sum_{i,j} \pi_{ij} \log p_i + \sum_{i,j} \pi_{ij} \log \tilde{p}_j = \sum_i p_i \log p_i + \sum_j \tilde{p}_j \log \tilde{p}_j \\ &= \sum_{i,j} p_i \tilde{p}_j \log p_i + \sum_{i,j} p_i \tilde{p}_j \log \tilde{p}_j = \sum_{i,j} p_i \tilde{p}_j \log(p_i \tilde{p}_j) \\ &= \sum_i p_i \log p_i + \sum_j \tilde{p}_j \log \tilde{p}_j,\end{aligned}$$

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

The remaining inequality follows from

$$\tilde{s}_{r;\lambda}(P,\tilde{P}) = s_{r;\lambda}(P,\tilde{P}) + \frac{1}{\lambda} H(\pi^s) \le s_{r;\lambda}(P,\tilde{P}) + \varepsilon.$$

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

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

3.1.1. Dual representation of the entropy-regularized Wasserstein distance.

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

PROPOSITION 3.5 (cf. [26, Remark 4.28]). *For $\lambda > 0$, the ERW problem (3.1a)–(3.1c) admits the following dual representation*

$$\begin{aligned} d(\alpha,\tilde\alpha) := \max_{\alpha\in\mathbb{R}^n,\ \tilde\alpha\in\mathbb{R}^{\tilde n}} \frac{1}{\lambda} + \frac{1}{\lambda}\sum_{i=1}^n p_i \log\alpha_i \\ + \frac{1}{\lambda}\sum_{j=1}^{\tilde n} \tilde p_j \log\tilde\alpha_j - \frac{1}{\lambda}\sum_{i=1}^n\sum_{j=1}^{\tilde n} \alpha_i\, \mathrm{e}^{-\lambda\, d_{ij}^r}\, \tilde\alpha_j, \end{aligned} \tag{3.4}$$

which is a strictly concave dual function.

Proof. The Lagrangian of the ERW problem (3.1a) with dual parameters β (for the constraint (2.2b)) and γ (for (2.2c)) is

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

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

$$\pi_{ij}^* = \exp\big(-\lambda(d_{ij}^r - \beta_i - \gamma_j) - 1\big). \tag{3.5}$$

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

$$\begin{aligned} d(\beta,\gamma) &= \inf_\pi L(\pi;\beta,\gamma) = L(\pi^*;\beta,\gamma) \\ &= \sum_{i=1}^n\sum_{j=1}^{\tilde n} d_{ij}^r\,\pi_{ij}^* + \frac{1}{\lambda}\sum_{i=1}^n\sum_{j=1}^{\tilde n}\pi_{ij}^*\big(-\lambda(d_{ij}^r-\beta_i-\gamma_j)-1\big) \\ &\qquad + \sum_{i=1}^n \beta_i\Big(p_i - \sum_{j=1}^{\tilde n}\pi_{ij}^*\Big) + \sum_{j=1}^{\tilde n}\gamma_j\Big(\tilde p_j - \sum_{i=1}^n\pi_{ij}^*\Big) \\ &= -\frac{1}{\lambda}\sum_{i=1}^n\sum_{j=1}^{\tilde n}\pi_{ij}^* + \sum_{i=1}^n\beta_i\,p_i + \sum_{j=1}^{\tilde n}\gamma_j\,\tilde p_j \\ &= \sum_{i=1}^n p_i\,\beta_i + \sum_{j=1}^{\tilde n}\tilde p_j\,\gamma_j - \frac{1}{\lambda}\sum_{i=1}^n\sum_{j=1}^{\tilde n}\mathrm{e}^{-\lambda(d_{ij}^r-\beta_i-\gamma_j)-1} \end{aligned}$$

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

$$d(\alpha,\tilde\alpha) = \frac{1}{\lambda}\sum_{i=1}^n p_i\left(\frac12 + \log\alpha_i\right) + \frac{1}{\lambda}\sum_{j=1}^{\tilde n}\tilde p_j\left(\frac12+\log\tilde\alpha_j\right) - \frac{1}{\lambda}\sum_{i=1}^n\sum_{j=1}^{\tilde n}\alpha_i\,\mathrm{e}^{-\lambda\, d_{ij}^r}\,\tilde\alpha_j. \tag{3.6}$$

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 (3.1a). ∎

Scaling variables and kernel matrix. First of all, we notice that the optimal measure (3.5) can be obtained in terms of the scaling variables α_i and $\tilde{\alpha}_j$ by

$$\pi^* = \operatorname{diag}(\alpha)\, \mathrm{e}^{-\lambda\, d^r}\, \operatorname{diag}(\tilde{\alpha}). \tag{3.7}$$

The aforementioned dual problem (3.4) can be solved by a matrix-scaling algorithm, which is popularly known as Sinkhorn's algorithm. Further, the derivative of (3.4) with respect to α_i (resp. $\tilde{\alpha}_i$) gives first-order conditions, which are the basis for Sinkhorn's iteration and are expressed as

$$\alpha_i := \frac{p_i}{\sum_{j=1}^{\tilde{n}} \mathrm{e}^{-\lambda\, d_{ij}^r}\, \tilde{\alpha}_j} \qquad \text{and} \qquad \tilde{\alpha}_j := \frac{p_j}{\sum_{i=1}^{n} \mathrm{e}^{-\lambda\, d_{ij}^r}\, \alpha_i}. \tag{3.8}$$

The main computational bottleneck of Sinkhorn's iterations is the matrix-vector multiplication in (3.8), which requires $\mathcal{O}(n \cdot \tilde{n})$ arithmetic operations. In our study, we reduce the computational burden by taking advantage of the 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 Sinkhorn's algorithm and its properties.

3.2. Sinkhorn's algorithm.

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

The *iteration counts* of Algorithm 1 are denoted by $\Delta \in \mathbb{N}$ and the final iteration count by 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. [36, 37] and Section 3.2.1 below) for the matrix scaling ensures that the iterations (3.8) converge and that the vectors α^Δ and $\tilde{\alpha}^\Delta$ are unique up to a scalar. From Algorithm 1, the resultant matrix $\pi^{\Delta^*} = \operatorname{diag}(\alpha^{\Delta^*})\, k\, \operatorname{diag}(\tilde{\alpha}^{\Delta^*})$ can be computed, which is an aproximate solution of the ERW problem (3.1a), (3.1b)–(3.1c).

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

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

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

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

where $\mathbf{1}_n := (1,\dots,1)^\top \in \mathbb{R}^n$ and $\mathbf{1}_{\tilde{n}} := (1,\dots,1)^\top \in \mathbb{R}^{\tilde{n}}$. If Δ is odd, then it holds that $\|(\pi^\Delta)^\top \mathbf{1}_n - \tilde{p}\| = 0$, and if Δ is even, then $\|\pi^\Delta \mathbf{1}_{\tilde{n}} - p\| = 0$. The stopping criterion 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 describes the individual steps.

Algorithm 1: Sinkhorn's algorithm.

```
Input: Distance $d_{ij}$ given in (4.1), 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 $\epsilon$, and a starting value
       $\tilde{\alpha} = (\tilde{\alpha}_1, \dots, \tilde{\alpha}_{\tilde{n}})$.
Set
    $k_{ij} = \exp\big(-\lambda\, d_{ij}^r\big), \quad \alpha^{(0)} := \mathbf{1}_n, \quad \text{and} \quad \tilde{\alpha}^{(0)} := \mathbf{1}_{\tilde{n}}.$
while $\|E^\Delta\| > \epsilon$ do
    if $\Delta$ is odd then
        (3.9)   $\alpha_i^\Delta \leftarrow \frac{p_i}{\sum_{j=1}^{\tilde{n}} k_{ij}\, \tilde{\alpha}_j^{\Delta-1}}, \quad i = 1, \dots, n;$
                $\tilde{\alpha}_j^\Delta \leftarrow \tilde{\alpha}_j^{\Delta-1}, \quad j = 1, \dots, \tilde{n};$
    else
        (3.10)  $\tilde{\alpha}_j^\Delta \leftarrow \frac{\tilde{p}_j}{\sum_{i=1}^{n} k_{ij}\, \alpha_i^{\Delta-1}}, \quad j = 1, \dots, \tilde{n};$
                $\alpha_i^\Delta \leftarrow \alpha_i^{\Delta-1}, \quad i = 1, \dots, n;$
    increment $\Delta \leftarrow \Delta + 1$.
Result:
    $s_{r;\lambda}(P, \tilde{P}) = \frac{1}{\lambda} + \frac{1}{\lambda}\sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} + \frac{1}{\lambda}\sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*} - \frac{1}{\lambda}\sum_{i=1}^{n}\sum_{j=1}^{\tilde{n}} \alpha_i^{\Delta^*} e^{-\lambda\, d_{ij}^r}\, \tilde{\alpha}_j^{\Delta^*}.$
    The matrix $\pi^{\Delta^*} = \operatorname{diag}(\alpha^{\Delta^*})\, k\, \operatorname{diag}(\tilde{\alpha}^{\Delta^*})$ can also be computed, which is
    an aproximate solution of the ERW problem (3.1a)–(3.1c).
```

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

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

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. [2, 10, 18]). We consider the following auxiliary lemmas to substantiate the objective of

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

```
Input: Distance $d_{ij}$ given in (4.1), 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 $\epsilon$, and a starting value
       $\gamma = (\gamma_1, \ldots, \gamma_{\tilde{n}})$.
Set
       $k_{ij} = \exp\big(-\lambda\, d_{ij}^r\big), \quad \beta^{(0)} := \mathbf{0}_n, \quad \text{and} \quad \gamma^{(0)} := \mathbf{0}_{\tilde{n}}.$
while $\|E^\Delta\| > \epsilon$ do
    if $\Delta$ is odd then
        $\beta_i^\Delta \leftarrow \frac{1}{\lambda}\Big(\log p_i - \log\big(\sum_{j=1}^{\tilde{n}} k_{ij}\, \mathrm{e}^{\lambda\, \gamma_j^{\Delta-1} - 1/2}\big)\Big), \quad i = 1, \ldots, n;$
        $\gamma_j^\Delta \leftarrow \gamma_j^{\Delta-1}, \quad j = 1, \ldots, \tilde{n};$
    else
        $\gamma_j^\Delta \leftarrow \frac{1}{\lambda}\Big(\log \tilde{p}_j - \log\big(\sum_{i=1}^{n} k_{ij}\, \mathrm{e}^{\lambda\, \beta_i^{\Delta-1} - 1/2}\big)\Big), \quad j = 1, \ldots, \tilde{n};$
        $\beta_i^\Delta \leftarrow \beta_i^{\Delta-1}, \quad i = 1, \ldots, n;$
    increment $\Delta \leftarrow \Delta + 1$.
Result:
       $s_{r;\lambda}(P, \tilde{P}) = \sum_{i=1}^{n} p_i\, \beta_i + \sum_{j=1}^{\tilde{n}} \tilde{p}_j\, \gamma_j - \frac{1}{\lambda} \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \mathrm{e}^{-\lambda(d_{ij}^r - \beta_i - \gamma_j) - 1}$
       The matrix $\pi^{\Delta^*} = \operatorname{diag}(\mathrm{e}^{\lambda\, \beta^{\Delta^*} - 1/2})\, k\, \operatorname{diag}(\mathrm{e}^{\lambda\, \gamma^{\Delta^*} - 1/2})$ can be computed,
       which is an aproximate solution of the ERW problem (3.1a)–(3.1c).
```

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. (3.6)) and

$$f(\alpha, \tilde{\alpha}) = \sum_{i=1}^{n} \sum_{j=1}^{\tilde{n}} \alpha_i\, \tilde{k}_{ij}\, \tilde{\alpha}_j - \sum_{i=1}^{n} p_i \log \alpha_i - \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j, \tag{3.11}$$

where $\tilde{k} := \frac{k}{\|k\|_1}$, $k = \exp\big(-\lambda\, d^r\big)$, 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. [2, 16, 18]) 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 (3.11) to the target marginals $(p, \tilde{p})$ of Sinkhorn's iteration.

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

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

Proof. First, we assume that $\Delta\geq 1$ is even. By equation (3.11), it follows that

$$\begin{aligned}f(\alpha^\Delta,\tilde\alpha^\Delta)-f(\alpha^{\Delta+1},\tilde\alpha^{\Delta+1})=&\sum_{ij}(\alpha_i^\Delta\,\tilde k_{ij}\,\tilde\alpha_j^\Delta-\alpha_i^{\Delta+1}\,\tilde k_{ij}\,\tilde\alpha_j^{\Delta+1})\\&+\sum_i p_i\big(\log(\alpha_i^{\Delta+1})-\log(\alpha_i^\Delta)\big)\\&+\sum_j \tilde p_j\big(\log(\tilde\alpha_j^{\Delta+1})-\log(\tilde\alpha_j^\Delta)\big).\end{aligned}\tag{3.12}$$

The first component on the right-hand side of (3.12) 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

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

and 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 (3.9)–(3.10). For this reason, the right-hand side of (3.12) becomes

$$\sum_i p_i\big(\log(\alpha_i^{\Delta+1})-\log(\alpha_i^\Delta)\big)+\sum_j p_j\big(\log(\tilde\alpha_j^{\Delta+1})-\log(\tilde\alpha_j^\Delta)\big).\tag{3.13}$$

By (3.9) and (3.10), the above expression (3.13) 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 the lemma for the case of *even* Δ. A similar argument applies to the case of *odd* Δ. □

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

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

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

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

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

where κ is the sum of the entries of the matrix π^{Δ^*} and $\jmath\coloneqq\min_{ij}\tilde k_{ij}$.

Proof. Let $(\alpha^{\Delta^*},\tilde\alpha^{\Delta^*})$ be the minimizer of the objective function (3.11), and we set

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

Equation (3.11) 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

$$f(\mathbf{1}_n, \mathbf{1}_{\tilde{n}}) - f(\alpha^{\Delta^*}, \tilde{\alpha}^{\Delta^*}) = \sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} + \sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*}. \tag{3.14}$$

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

$$\jmath \Big(\sum_{i=1}^{n} \alpha_i^{\Delta^*} \Big) \Big(\sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j^{\Delta^*} \Big) \leq (\alpha^{\Delta^*})^\top \tilde{k}\, \tilde{\alpha}^{\Delta^*} = \kappa. \tag{3.15}$$

Taking the log of both sides of equation (3.15) produces

$$\log \Big(\sum_{i=1}^{n} \alpha_i^{\Delta^*} \Big) + \log \Big(\sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j^{\Delta^*} \Big) \leq \log \Big(\frac{\kappa}{\jmath} \Big). \tag{3.16}$$

To complete the proof, we consider equations (3.14), (3.16), and the log-sum inequality. Now we have

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

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. [2, 10, 18]). Approximately, Sinkhorn's iteration requires $\mathcal{O}(\log n + \|d^r\|_\infty \lambda)$ arithmetic operations to converge (cf. [10]). This means that when $\lambda \to \infty$ for fixed $\|d^r\|_\infty$ and n, the number of iteration will be increased.

Complexity of the ERW problem. For $n \approx \tilde{n}$, using Algorithm 1, the ERW problem (3.1a), (3.1b)–(3.1c) can be solved by involving $\mathcal{O}(n^2 \log n + \|d^r\|_\infty \lambda)$ 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, in [29], the *Sinkhorn divergence* is introduced as

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

which is a natural normalization (or debias) of the quantity. The key properties of the 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 to compute equispaced convolution is the standard FFT algorithm. Our research, in contrast, promotes *non*equispaced convolution, which is approximated by the *non*equispaced fast Fourier transform (NFFT) to accelerate the computation of Sinkhorn's Algorithm 1. More precisely, the matrix-vector multiplications of Sinkhorn's iteration (3.8), which is the main computational bottleneck, are tackled by *fast summation* based on NFFT in $\mathcal{O}\big(n \log n\big)$ arithmetic operations. Moreover, this fast summation technique has better stability and is accomplished with machine precision.

4.1. NFFT-based fast summation.

This section succinctly describes 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 (2.2a) has entries

$$d(x_i, \tilde{x}_j) \coloneqq \|x_i - \tilde{x}_j\|, \tag{4.1}$$

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

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

$$t_i \coloneqq\approx \big(k\,\tilde{\alpha}\big)_i = \sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j \,\mathrm{e}^{-\lambda\|x_i-\tilde{x}_j\|^r}, \quad i = 1, \dots, n, \tag{4.2}$$

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

$$\tilde{t}_j \coloneqq\approx \big(k\,\alpha\big)_j = \sum_{i=1}^{n} \alpha_i \,\mathrm{e}^{-\lambda\|x_i-\tilde{x}_j\|^r}, \quad j = 1, \dots, \tilde{n}, \tag{4.3}$$

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 (4.2) and (4.3) corresponds 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 [28, Theorem 3.31]) such that we end up with $\mathcal{O}(n \log n)$ operations using the FFT; see [28, pp. 141–142]. A fast algorithm with *arbitrary* points follows the similar ideas but is based on the NFFT; see [28, Chapter 7] and the related software in [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

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

In general, this approximation of the kernel function $\mathcal{K}(y)$ is appropriate when the entries of the matrix k are of 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 an h-periodic trigonometric polynomial $\mathcal{K}_{RK}(y)$,

$$\begin{aligned} \mathcal{K}(y) \approx \mathcal{K}_{RK}(y) &:= \sum_{\mathrm{k} \in \mathcal{I}_N} b_{\mathrm{k}} \, \mathrm{e}^{2\pi \mathrm{i} \mathrm{k} y/h}, \\ \mathcal{I}_N &:= \left\{ -\frac{N}{2}, -\frac{N}{2}+1, \ldots, -1, 0, \ldots, \frac{N}{2}-1 \right\}^d, \end{aligned} \tag{4.4}$$

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

$$\left(k\,\alpha\right)_j = \sum_{i=1}^{n} \alpha_i \, \mathrm{e}^{-\lambda \|x_i - \tilde{x}_j\|^2}, \quad j = 1, \ldots, \tilde{n}. \tag{4.5}$$

Now we rewrite equation (4.5) by involving the kernel function $\mathcal{K}(y) = \mathrm{e}^{-\lambda \|y\|^2}$ as

$$\left(k\,\alpha\right)_j := \sum_{i=1}^{n} \alpha_i \, \mathcal{K}(x_i - \tilde{x}_j), \quad j = 1, \ldots, \tilde{n}. \tag{4.6}$$

For the efficient computation of (4.6), the NFFT-based fast summation technique approximates $\mathcal{K}$ by the trigonometric polynomial $\mathcal{K}_{RK}$.

From equation (4.4), 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 an 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 $\|x_j\| \leq \frac{L}{2}$, i.e., $\|x_i - \tilde{x}_j\| \leq L$, 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 $h \geq 2L$ by

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

where K_{B} is an appropriately chosen univariate polynomial, which is constructed using two-point Taylor interpolation; see Figure 4.1. For a detailed interpretation of this approach, we refer to [28, Chapter 7.5].

Approximation of the 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

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

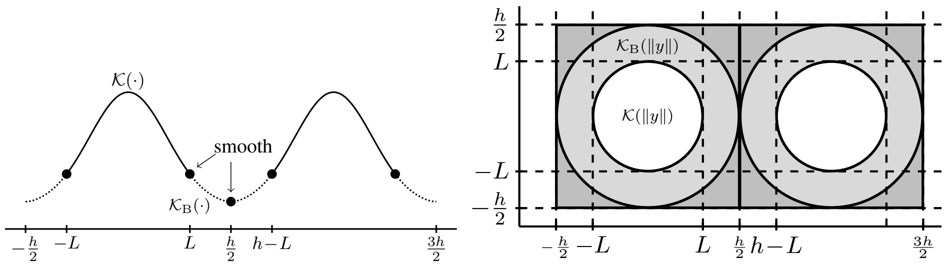

$\mathcal{K}(\cdot)$
smooth
$\mathcal{K}_{\mathrm{B}}(\cdot)$
$-\frac{h}{2}$ $-L$ L $\frac{h}{2}$ $h-L$ $\frac{3h}{2}$
$\mathcal{K}_{\mathrm{B}}(\|y\|)$
$\mathcal{K}(\|y\|)$
$\frac{h}{2}$ L $-L$ $-\frac{h}{2}$

FIG. 4.1. *The regularized periodic function in dimension one (left) and dimensions two (right); see [23, Figure 3.14].*

By using (4.4) and interchanging the order of summation as well as utilizing the property

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

we obtain

$$t_i \approx \sum_{\mathrm{k}\in\mathcal{I}_N} b_{\mathrm{k}} \left(\sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j \mathrm{e}^{-2\pi\mathrm{i}\mathrm{k}\tilde{x}_j/h}\right) \mathrm{e}^{2\pi\mathrm{i}\mathrm{k}x_i/h}, \quad i=1,\dots,n. \tag{4.7}$$

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

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 polynomial of low 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 correspond 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 can be carried out 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 (4.4) is independent of the n (resp. $\tilde{n}$) data points. Therefore we can appropriately fix the polynomial degree N. Thus, the approximation ends up with $\mathcal{O}(n+\tilde{n})$ arithmetic operations. Furthermore, for $r=1$, we need an additional near-field regularization at the point $y=0$. In this case, we end up with an arithmetic complexity of $\mathcal{O}(n\log n+\tilde{n}\log\tilde{n})$; see [28, Chapter 7.5] for a detailed interpretation.

4.2. NFFT boost for Sinkhorn's algorithm. This section presents the NFFT-accelerated standard and the log-domain Sinkhorn's algorithm. The NFFT-accelerated Sinkhorn's algorithm propose a novel method using nonequispaced 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 Algorithms 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

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

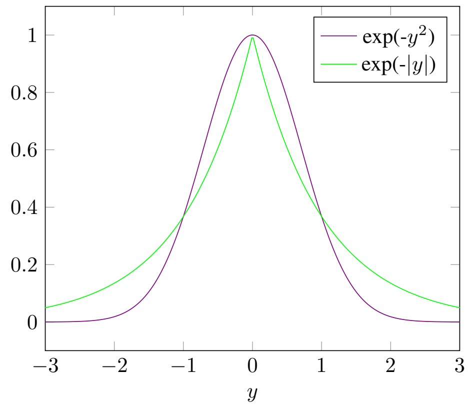


FIG. 4.2. *Wasserstein distance for $r = 1$ (green) and $r = 2$ (violet).*

The NFFT-accelerated Sinkhorn divergence is computed by

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

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

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

and for $r = 1$

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

REMARK 4.3 (Optimal transition matrix π^{Δ^*}). The NFFT-accelerated Sinkhorn's Algorithms 3 and 4 bypass the allocations of the matrices d, k, and π^{Δ^*} and return 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 the optimal dual variables $(\beta^{\Delta^*},\gamma^{\Delta^*})$ (Algorithm 4). Hence, the optimal transition matrix π^{Δ^*} can still be computed with (3.7). However, this—as mentioned—requires $\mathcal{O}(n\tilde{n})$ operations, which would increase the performance time and thus is avoided.

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

5. Numerical experiments. This section demonstrates the performance and accuracy of the 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 an Intel(R) Core(TM) i7-7700 CPU and 15.0 GB of RAM. The source code of our implementation of the standard Sinkhorn's Algorithm 1, the log-domain stabilized Sinkhorn's Algorithm 2, the NFFT-accelerated Sinkhorn's Algorithm 3, the NFFT-accelerated log-domain Sinkhorn's Algorithm 4, and the linear programming solver to compute $w_r(P,\tilde{P})$, which can be used to

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 $\varepsilon$ and starting value
       $\tilde{\alpha} = (\tilde{\alpha}_1, \dots, \tilde{\alpha}_{\tilde{n}})$.
while $\|E^\Delta\| > \varepsilon$ do
    Set
        $$\alpha^{(0)} := \mathbf{1}_n \quad \text{and} \quad \tilde{\alpha}^{(0)} := \mathbf{1}_{\tilde{n}}.$$
    if $\Delta$ is odd then
        compute
            $$t_i^{\Delta-1} \leftarrow \sum_{j=1}^{\tilde{n}} \tilde{\alpha}_j^{\Delta-1} \mathrm{e}^{-\lambda\|x_i-\tilde{x}_j\|^r}, \quad i = 1, \dots, n,$$
        by employing the fast summation (4.7), 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
            $$\tilde{t}_j^{\Delta-1} \leftarrow \sum_{i=1}^{n} \mathrm{e}^{-\lambda\|x_i-\tilde{x}_j\|^r} \alpha_i^{\Delta-1}, \quad j = 1, \dots, \tilde{n},$$
        by employing the fast summation (4.7), 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. (3.4)) approximating the Wasserstein distance
        $W_r(P, \tilde{P})$ is
    $$s_{r;\lambda;\text{NFFT}}(P, \tilde{P}) := -\frac{1}{\lambda} + \frac{1}{\lambda}\sum_{i=1}^{n} p_i \log \alpha_i^{\Delta^*} + \frac{1}{\lambda}\sum_{j=1}^{\tilde{n}} \tilde{p}_j \log \tilde{\alpha}_j^{\Delta^*} - \frac{1}{\lambda}\sum_{j=1}^{\tilde{n}} \tilde{t}_j^{\Delta^*} \tilde{\alpha}_j^{\Delta^*}.$$
```

reproduce the following results, are available online[3]. The implementation of our proposed algorithms are based on the freely available repository 'NFFT3.jl'[4].

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

[3]cf. https://github.com/rajmadan96/NFFT-Sinkhorn-Wasserstein_distance/
[4]cf. https://github.com/NFFT/NFFT3.jl

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 $\varepsilon$ and starting value
       $\gamma = (\gamma_1, \dots, \gamma_{\tilde{n}})$.
while $\|E^\Delta\| > \varepsilon$ do
    Set
        $$\beta^{(0)} := \mathbf{0}_n \quad \text{and} \quad \gamma^{(0)} := \mathbf{0}_{\tilde{n}}.$$
    if $\Delta$ is odd then
        compute
            $$t_i^{\Delta-1} \leftarrow \sum_{j=1}^{\tilde{n}} \mathrm{e}^{\lambda \gamma_j^{\Delta-1} - 1/2} \, \mathrm{e}^{-\lambda \|x_i - \tilde{x}_j\|^r}, \quad i = 1, \dots, n,$$
        by employing the fast summation (4.7), and set
            $$\beta_i^\Delta \leftarrow \frac{1}{\lambda}\Big(\log p_i - \log t_i^{\Delta-1}\Big), \quad i = 1, \dots, n;$$
            $$\gamma_j^\Delta \leftarrow \gamma_j^{\Delta-1}, \quad j = 1, \dots, \tilde{n};$$
    else
        compute
            $$\tilde{t}_j^{\Delta-1} \leftarrow \sum_{i=1}^{n} \mathrm{e}^{-\lambda \|x_i - \tilde{x}_j\|^r} \, \mathrm{e}^{\lambda \beta_i^{\Delta-1} - 1/2}, \quad j = 1, \dots, \tilde{n},$$
        by employing the fast summation (4.7), and set
            $$\gamma_j^\Delta \leftarrow \frac{1}{\lambda}\Big(\log \tilde{p}_j - \log \tilde{t}_j^{\Delta-1}\Big), \quad j = 1, \dots, \tilde{n};$$
            $$\beta_i^\Delta \leftarrow \beta_i^{\Delta-1}, \quad i = 1, \dots, n;$$
    increment $\Delta \leftarrow \Delta + 1$.
Result: The ERW distance (cf. (3.4)) approximating the Wasserstein distance
        $W_r(P, \tilde{P})$ is
            $$s_{r;\lambda}(P, \tilde{P}) \approx \sum_{i=1}^{n} p_i \, \beta_i + \sum_{j=1}^{\tilde{n}} \tilde{p}_j \, \gamma_j - \frac{1}{\lambda} \sum_{j=1}^{\tilde{n}} \tilde{t}_j^{\Delta^*} \, \mathrm{e}^{\lambda \gamma_j^{\Delta^*} - 1/2}.$$
```

5.1.1. NFFT-accelerated Sinkhorn's algorithm in one dimension.

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

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

and corresponding weights

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

The measure

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

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

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

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

Table 5.1 provides a comparison of the computation time of Sinkhorn's Algorithm 1 and the NFFT-accelerated Sinkhorn's Algorithm 3.

TABLE 5.1
Dimension 1: comparison of computation times for $r = 2$ and $\lambda = 20$.

$n = \tilde{n}$:	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's Alg. 3	0.28 s	0.39 s	2.08 s	2.31 s	9.38 s	62.4 s

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

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

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

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

Table 5.2 displays the execution times for the uniform distribution on $[0, 1] \times [0, 1]$. While the computation time and the memory allocations are already critical for n, $\tilde{n} \approx 100\,000$, the NFFT-accelerated Sinkhorn's algorithm still performs in reasonable time.

TABLE 5.2
Two dimensions: comparison of computation times for $r = 2$ and $\lambda = 20$.

$n = \tilde{n}$:	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-acc. Sinkhorn's Alg. 3	0.31 s	0.42 s	2.0 s	7.2 s	59.4 s

5.2. Benchmark datasets. This section validates the regularization parameter λ and demonstrates the performance and the accuracy of the NFFT-accelerated Sinkhorn's Algorithm 3 using real datasets. We use a dataset called DOTmark (see Figure 5.1); DOT stands for *discrete optimal transport*. This benchmark dataset is specially designed to effectively test and compare optimal transport methods (cf. [35]). It involves gray-level representations of images in a resolution of 32×32 to 512×512, and it consists of 10 subsets of the dataset, ranging from smooth to rough structures.

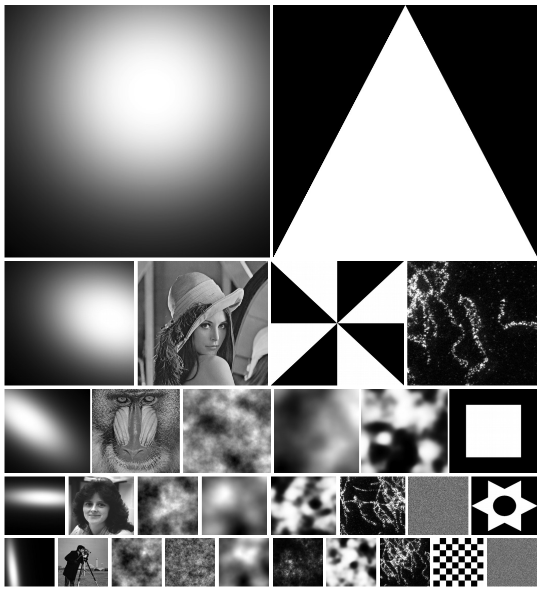
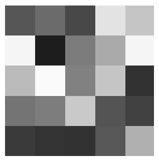


FIG. 5.1. *Examples of the DOTmark database (from low- to high-resolution images).*



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

White- 1
Black- 0

FIG. 5.2. *A gray-scale image* (5×5) *is represented as a matrix.*

Transformation of images to probability vectors. A gray-scale 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 5.2). In order to convert the gray-scale image matrices into probability vectors, we vectorize and normalize the matrices. Furthermore, the intensities of background pixels are the ℓ_1-distance between the pixels i and j of the respective grids ($32 \times 32, \ldots, 512 \times 512$).

5.2.1. Validation of the regularization parameter $\boldsymbol{\lambda}$. In this section we capture the behaviors of the lower and upper bounds (cf. Lemma 3.3) and the Sinkhorn divergence, i.e., $s_{r;\lambda}(P, \tilde{P})$, $\tilde{s}_{r;\lambda}(P, \tilde{P})$, $sd_{r;\lambda}(P, \tilde{P})$, and $s_{r;\lambda;\text{NFFT}}(P, \tilde{P})$, $\tilde{s}_{r;\lambda;\text{NFFT}}(P, \tilde{P})$, $sd_{r;\lambda;\text{NFFT}}(P, \tilde{P})$

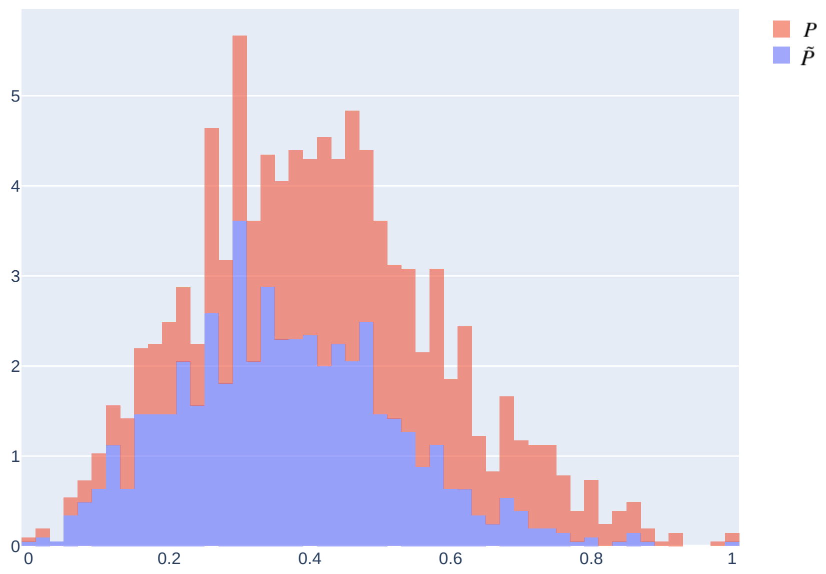


FIG. 5.3. *Histogram of the measures P and $\tilde{P}$ of the dataset GRFrough; n (resp. $\tilde{n}$) $= 1024$.*

for different values of the entropy-regularization parameter $\lambda \in \{10, 20, 50, \dots, 200\}$.

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 5.3). In Figure 5.4 below we investigate these quantities with respect to increasing λ.

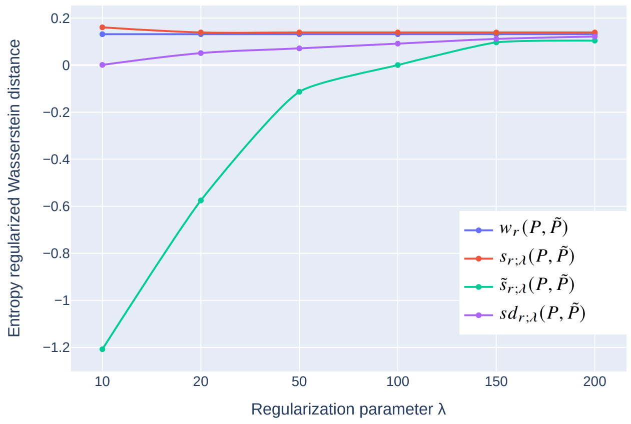
We infer that $s_{r;\lambda}(P, \tilde{P})$, $s_{r;\lambda;\text{NFFT}}(P, \tilde{P})$ converge slowly to $w_r(P, \tilde{P})$ for increasing λ. However, $\tilde{s}_{r;\lambda}(P, \tilde{P})$ and $\tilde{s}_{r;\lambda;\text{NFFT}}(P, \tilde{P})$ converge quickly to $w_r(P, \tilde{P})$, and the Sinkhorn divergence $\big(sd_{r;\lambda}(P, \tilde{P}), sd_{r;\lambda;\text{NFFT}}(P, \tilde{P})\big)$ also converge quicker compared to $s_{r;\lambda}(P, \tilde{P})$, $s_{r;\lambda;\text{NFFT}}(P, \tilde{P})$. The argument behind these behaviors is that, for larger λ, the weight of the entropy in the objective function (3.1a) decreases, and the matrices π^s and π^w are close. Furthermore, the NFFT approximation is stable for different values of the regularization parameter λ.

5.2.2. Performance analysis. This section extensively substantiates the performance of the 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.

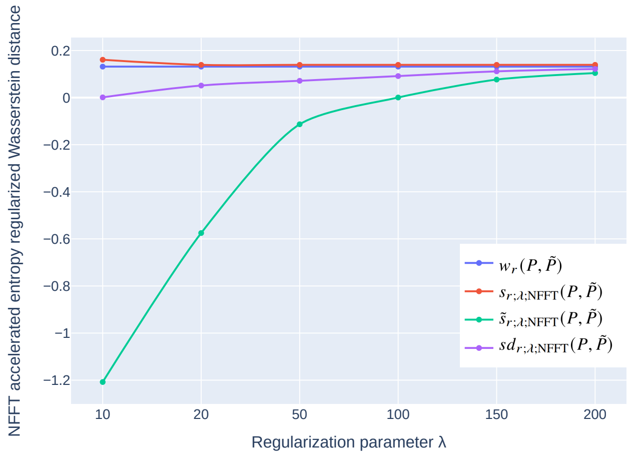
Tables 5.3 and 5.4 provide a comparison of the time and memory allocation of Sinkhorn's Algorithm 1 and the NFFT-accelerated Sinkhorn's Algorithm 3. Among all the tests, the computational time and memory allocation of our proposed algorithm is significantly better.

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

5.2.3. Accuracy analysis. We validate the computational accuracy of the NFFT-accelerated Sinkhorn's Algorithm 3. Throughout this analysis, we use $\tilde{s}_{r;\lambda}(P, \tilde{P})$ and $\tilde{s}_{r;\lambda;\text{NFFT}}(P, \tilde{P})$



(a) Numerical results of Algorithm 1.



(b) Numerical results of Algorithm 3.

FIG. 5.4. *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* $s_{r;\lambda;NFFT}(P,\tilde{P})$, $\tilde{s}_{r;\lambda;NFFT}(P,\tilde{P})$, $sd_{r;\lambda;NFFT}(P,\tilde{P})$. *The parameters are* $\lambda \in \{10, 20, 50, \dots, 200\}$ *and* $r = 2$.

since these are more accurate approximations of the Wasserstein distance (see Section 5.2.1). Initially, we perform the accuracy analysis using the low-resolution images from the DOTmark dataset. From Table 5.5, we notice that the NFFT-accelerated Sinkhorn's Algorithm 3 has achieved stable approximations without a compromise in accuracy.

Now, we move on to high-resolution images of the 'GRFrough' dataset. Table 5.6 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 the NFFT-accelerated Sinkhorn's Algorithm 3 for the problems sized up to $16\,384$ n (resp. $\tilde{n}$), and we infer that there is no compromise in accuracy. We recognize that advancing to high-resolution images does not affect the stability of approximation. Due to the breakdown of Sinkhorn's Algorithm 1, it cannot be used as comparison factor when the size of the problem is beyond $16\,384$ n (resp. $\tilde{n}$). However, our proposed algorithm computes the largest problem available in the DOTmark dataset, which is of size $262\,144$ n (resp. $\tilde{n}$).

TABLE 5.3
Comparison of computational time allocation of Sinkhorn's Algorithm 1 and the NFFT-accelerated Sinkhorn's Algorithm 3; $\lambda = 20$ and $r = 2$.

$n = \tilde{n}$	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 5.4
Comparison of computational memory allocation of Sinkhorn's Algorithm 1 and the 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 5.5
Comparison of accuracy to compute or approximate the Wasserstein distance in low-resolution images; $\lambda = 20$ and $r = 2$.

$n \times \tilde{n} = 1024 \times 1024$	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

5.3. Comparisons and further steps. In this section, we substantiate the historical evolution of the prominent algorithms that approximate the Wasserstein distance. Furthermore,

TABLE 5.6
Comparison of computation times and accuracy to compute or approximate the Wasserstein distance from low- to high-resolution images; $\lambda = 20$ and $r = 2$.

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

we discuss the advantages and the direction of further development of our proposed algorithms.

Historical remarks. The approach of entropy regularization of the Wasserstein distance by [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 carried out to improve and/or support the entropy-regularization approach (cf. [2, 10]). In 2019, the approach of log-domain stabilization and the truncated kernel of Sinkhorn's algorithm was proposed in [34]. The log-domain stabilization method satisfies the demand for larger regularization parameters λ, and the truncated kernel reduces the memory demand and also accelerates the iterations. In the same article, a multi-scale scheme was proposed, which enables more efficient computations of the kernel-truncated approach. As discussed in Remark 3.10, these approaches suffer from the entropy bias. In order to remove/ reduce the bias, Sinkhorn divergence was proposed in [29].

These approaches affirm the progressive improvement of the algorithms that approximate the Wasserstein distance. However, notably, these approaches still require the standard matrix-vector operations, which is the bottleneck of the algorithms.

TABLE 5.7
List of algorithms.

Name of algorithm / method	Algorithm	denotation
Standard Sinkhorn ([9])	Algorithm 1	Std. Sinkhorn
Stabilized log-domain Sinkhorn ([34])	Algorithm 2	Stb. log Sinkhorn
Sinkhorn divergence ([29])	Algorithm 1	$sd_{r;\lambda}(P, \tilde{P})$
Multi-scale Sinkhorn ([34])	Algorithm[5]	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	$sd_{r;\lambda;\text{NFFT}}(P, \tilde{P})$

Now, we compare our proposed algorithms with the algorithms that are discussed so far (see Table 5.7). 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 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 the preceding Section 5.2.3, and we use the "GRFrough" dataset. From Figure 5.5 and Table 5.8, it is evident that our proposed algorithms perform significantly better in terms of time and memory allocations. Our device runs out of memory

[5]cf. https://github.com/OTGroupGoe/MultiScaleOT.jl

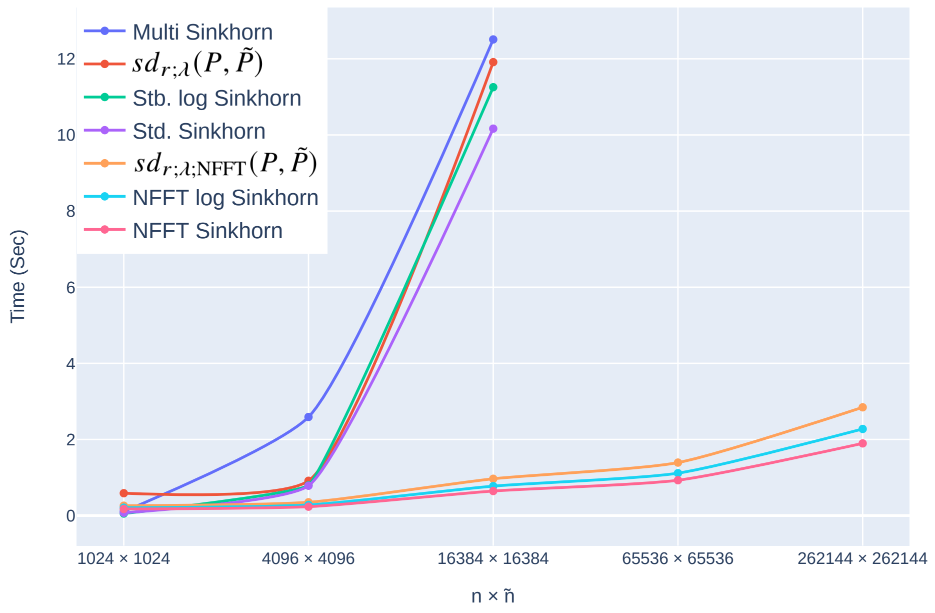


FIG. 5.5. *Comparison of computational time allocations of the algorithms in Table 5.7;* $\lambda = 20$ *and* $r = 2$.

TABLE 5.8
Comparison of computational memory allocations of the algorithms in Table 5.7; $\lambda = 20$ *and* $r = 2$.

$n = \tilde{n}$	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

for all algorithms/ methods except for our proposed algorithms when the problems are sized larger than 16 384 n (resp. $\tilde{n}$). 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 down due to the kernel-matrix formation when the problem size is larger than 16 384 n (resp. $\tilde{n}$).

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 [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” package is the prominent contribution in terms of fast computation.

Further steps. Our proposed algorithms reduces the burden of time and memory allocations, and it is also flexible to adapt to the log-domain implementation. Further research directions will be focused on applying our algorithms to suitable applications and the 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 a GPU implementation of the NFFT algorithm is readily available in the corresponding GitHub repository[6].

6. Summary.

The nonequispaced fast Fourier transform, as presented in this article, allows computing a proper approximation of the Wasserstein distance in $\mathcal{O}(n \log n)$ arithmetic operations. The NFFT-accelerated Sinkhorn's Algorithm 3 performs significantly better than the 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 science applications for handling large-scale datasets.

REFERENCES

[1] J. Altschuler, F. Bach, A. Rudi, and J. Niles-Weed, *Massively scalable Sinkhorn distances via the Nyström method*, in Proceedings of the 33rd International Conference on Neural Information Processing Systems NIPS 2019, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., Adv. Neural. Inf. Process. Syst. 32, Curran Assoc., Red Hook, 2019, pp. 4427–4437.

[2] 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 NIPS 2017, U. von Luxburg, I. Guyon, S. Bengio, H. Wallach, and R. Fergus, eds., Adv. Neural. Inf. Process. Syst. 30, Curran Assoc., Red Hook, 2017, pp. 1961–1971.

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

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

[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 Proc. Mag., 36 (2019), pp. 20–31.

[6] F. Bolley, *Separability and completeness for the Wasserstein distance*, in Séminaire de Probabilités XLI, C. Donati-Martin, M. Émery, A. Rouault, and C. Stricker, eds., Vol. 1934 of Lecture Notes in Math., Springer, Berlin, 2008, pp. 371–377.

[7] S. Chakraborty, D. Paul, and S. Das, *Hierarchical clustering with optimal transport*, Statist. Probab. Lett., 163 (2020), Paper No. 108781, 7 pages.

[8] N. Courty, R. Flamary, and D. Tuia, *Domain adaptation with regularized optimal transport*, in Machine Learning and Knowledge Discovery in Databases ECML PKDD 2014, T. Calders, F. Esposito, E. Hüllermeier, and R. Meo, eds., Lecture Notes in Comput. Sci., vol 8724, Springer, Berlin, 2014, pp. 274–289.

[9] M. Cuturi, *Sinkhorn distances: lightspeed computation of optimal transport*, in Proceedings of the 26th International Conference on Neural Information Processing Systems NIPS 2013, C. J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds., Adv. Neural. Inf. Process. Syst. 26, Curran Assoc., Red Hook, 2023, 9 pages.

[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 2018, J. Dy and A. Krause, eds., PMLR, Vol. 80, 2018, pp. 1367–1376.

[11] J. Feydy, *Geometric Data Analysis, Beyond Convolutions*, Ph.D. Thesis, Université Paris-Saclay, Gif-sur-Yvette, 2020.

[12] A. V. Gasnikov, E. B. Gasnikova, Y. E. Nesterov, and A. V. Chernov, *Efficient numerical methods for entropy-linear programming problems*, Comput. Math. Math. Phys., 56 (2016), pp. 514–524.

[13] A. Genevay, *Entropy-Regularized Optimal Transport for Machine Learning*, Ph.D. Thesis, University Paris Sciences et Lettres (ComUE), Paris, 2019.

[6] cf. https://github.com/sukunis/CUNFFT/tree/master/src

[14] S. Graf and H. Luschgy, *Foundations of Quantization for Probability Distributions*, Springer, Berlin, 2000.
[15] N. Hao, M. E. Kilmer, K. Braman, and R. C. Hoover, *Facial recognition using tensor-tensor decompositions*, SIAM J. Imaging Sci., 6 (2013), pp. 437–463.
[16] B. Kalantari, I. Lari, F. Ricca, and B. Simeone, *On the complexity of general matrix scaling and entropy minimization via the RAS algorithm*, Math. Program., 112 (2008), pp. 371–401.
[17] J. Keiner, S. Kunis, and D. Potts, *NFFT 3.5, C subroutine library*, Software. http://www.tu-chemnitz.de/~potts/nfft.
[18] B. Khalil Abid and R. M. Gower, *Greedy stochastic algorithms for entropy-regularized optimal transport problems*, in International Conference on Artificial Intelligence and Statistics 2018, A. Storkey and F. Perez-Cruz, eds., PMLR, Vol. 84, 2018, pp. 1505–1512.
[19] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, *From word embeddings to document distances*, in International Conference on Machine Learning 2015, F. Bach and D. Blei, eds., PMLR, Vol. 37, 2015, pp. 957–966.
[20] T. Lin, N. Ho, and M. I. Jordan, *On the efficiency of Sinkhorn and Greenkhorn and their acceleration for optimal transport*, in International Conference on Machine Learning 2019, K. Chaudhuri and R. Salakhutdinov, eds., PMLR, Vol. 97, 2019, pp. 3982–3991.
[21] G. Luise, A. Rudi, M. Pontil, and C. Ciliberto, *Differential properties of Sinkhorn approximation for learning with Wasserstein distance*, in Proceedings of the 32nd International Conference on Neural Information Processing Systems NIPS 2018, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds., Adv. Neural. Inf. Process. Syst. 31, Curran Assoc., Red Hook, 2018, pp. 5864–5874.
[22] A. Mensch and G. Peyré, *Online Sinkhorn: optimal transport distances from sample streams*, in Proceedings of the 34th International Conference on Neural Information Processing Systems NIPS 2020., H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, eds., Adv. Neural. Inf. Process. Syst. 33, Curran Assoc., Red Hook, 2020, pp. 1657–1667.
[23] F. Nestler, *Efficient Computation of Electrostatic Interactions in Particle Systems Based on Nonequispaced Fast Fourier Transforms*, Ph.D. Thesis, Universitätsverlag Chemnitz, Chemnitz, 2018.
[24] S. Neumayer and G. Steidl, *From optimal transport to discrepancy*, in Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging, K. Chen, C.-B. Schönlieb, X.-C. Tai, and L. Younces, eds., Springer, Cham, 2021, 36 pages.
[25] N. Papadakis, G. Peyré, and E. Oudet, *Optimal transport with proximal splitting*, SIAM J. Imaging Sci., 7 (2014), pp. 212–238.
[26] G. Peyré and M. Cuturi, *Computational optimal transport: with applications to data science*, Found. Trends Machine Learning, 11 (2019), pp. 355–607.
[27] R. B. Platte, L. N. Trefethen, and A. B. J. Kuijlaars, *Impossibility of fast stable approximation of analytic functions from equispaced samples*, SIAM Rev., 53 (2011), pp. 308–318.
[28] G. Plonka, D. Potts, G. Steidl, and M. Tasche, *Numerical Fourier Analysis*, Birkhäuser/Springer, Cham, 2018.
[29] A. Ramdas, N. García Trillos, and M. Cuturi, *On Wasserstein two-sample testing and related families of nonparametric tests*, Entropy, 19 (2017), Paper No. 47, 15 pages.
[30] S. Revay and M. Teschke, *Multiclass language identification using deep learning on spectral images of audio signals*, Preprint on arXiv, 2019. https://arxiv.org/abs/1905.04348
[31] G. Rote and M. Zachariasen, *Matrix scaling by network flow*, in Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, ACM, New York, 2007, pp. 848–854.
[32] M. Scetbon and M. Cuturi, *Linear time Sinkhorn divergences using positive features*, in Proceedings of the 34th International Conference on Neural Information Processing Systems NIPS 2020, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, eds., Adv. Neural. Inf. Process. Syst. 33, Curran Assoc., Red Hook, 2020, pp. 13468–13480.
[33] M. Scetbon, M. Cuturi, and G. Peyré, *Low-rank Sinkhorn factorization*, in International Conference on Machine Learning 2021, M. Meila and T. Zhang, eds., PMLR, Vol. 139, 2021, pp. 9344–9354.
[34] B. Schmitzer, *Stabilized sparse scaling algorithms for entropy regularized transport problems*, SIAM J. Sci. Comput., 41 (2019), pp. A1443–A1481.
[35] J. Schrieber, D. Schuhmacher, and C. Gottschlich, *Dotmark–a benchmark for discrete optimal transport*, IEEE Access, 5 (2016), pp. 271–282.
[36] R. Sinkhorn, *Diagonal equivalence to matrices with prescribed row and column sums*, Amer. Math. Monthly, 74 (1967), pp. 402–405.
[37] R. Sinkhorn and P. Knopp, *Concerning nonnegative matrices and doubly stochastic matrices*, Pacific J. Math., 21 (1967), pp. 343–348.
[38] K. S. Tai, P. D. Bailis, and G. Valiant, *Sinkhorn label allocation: semi-supervised classification via annealed self-training*, in International Conference on Machine Learning 2021, M. Meila and T. Zhang, eds., PMLR, Vol. 139, 2021, pp. 10065–10075.

[39] C. VILLANI, *Topics in Optimal Transportation*, AMS, Providence, 2003.
[40] ———, *Optimal Transport*, Springer, Berlin, 2009.