MULTILEVEL OPTIMAL TRANSPORT: A FAST APPROXIMATION OF WASSERSTEIN-1 DISTANCES*

JIALIN LIU†, WOTAO YIN†, WUCHEN LI†, AND YAT TIN CHOW‡

Abstract. We propose a fast algorithm for the calculation of the Wasserstein-1 distance, which is a particular type of optimal transport distance with homogeneous of degree one ground metric. Our algorithm is built on multilevel primal-dual algorithms. Several numerical examples and complexity analysis are provided to demonstrate its computational speed. On some commonly used image examples of size 512×512, the proposed algorithm gives solutions within 0.5 seconds on a single CPU, which is much faster than the state-of-the-art algorithms.

Key words. Multilevel algorithms; Optimal transport; Wasserstein-1 distance; Primal-dual algorithm.

AMS subject classifications. 49M25; 49M30; 90C90

1. Introduction.

Optimal transport (OT) plays crucial roles in many areas, including fluid dynamics [45], image processing [39, 40], machine learning [1, 20] and control [11, 12]. It is a well-posed distance measuring two probability distributions over a given domain. The distance is often named Earth Mover's distance (EMD) or the Wasserstein distance. Plenty of theories on OT have been introduced [3, 4, 21, 32, 45]. Despite the theoretical development, computing the distance is still challenging since the OT problems usually do not have closed-form solutions. Fast numerical algorithms are essential for the related applications.

Recently, a particular class of OT, named the Wasserstein-1 distance, has been widely used in machine learning problems [1, 23, 37]. It gains rising interests in the computational mathematics community [25, 29, 2, 44]. The Wasserstein-1 distance is named as its ground metric is homogeneous of degree one. In this paper, we focus on numerically computing Wasserstein-1 distances.

In literature, many numerical schemes have been proposed for the OT problem. [27, 39, 30, 36, 35, 31, 2] modeled the OT problem as a linear programming (LP) with specific structures. They utilized these structures to develop efficient solvers. [33, 38, 5, 24, 29, 28, 41] modeled OT as a nonsmooth convex optimization problem and introduced iterative algorithms to solve it. [14, 6, 43, 18, 9, 19, 15] studied the OT problems with regularizers and proposed efficient algorithms to solve them. In particular, some algorithms have been developed for calculating the Wasserstein-1 distance and its variants. Ling and Okada [30] exploited the structure of the problem to improve the transportation simplex algorithm [27] and proposed Tree-EMD. Pele and Werman [35, 36] proposed and solved EMD with a thresholded ground metric. Li et al. [29] studied a primal-dual algorithm for calculating Wasserstein-1 distances that is friendly to parallel programming and has an implementation on CUDA. Jacobs et al. [28] introduced the proximal PDHG method, whose number of iterations is independent of the grid size. Bassetti et al. [2] studied the connections between the Wasserstein-1 distance and the uncapacitated minimum cost flow problem and applied

*The codes will be released to: https://github.com/liujl11git/multilevelOT.

Funding: The research is supported by AFOSR MURI proposal number 18RT0073, ONR N000141712162 and NSF DMS-1720237. The Titan Xp used for this research were donated by the NVIDIA Corporation.

†Department of Mathematics, University of California, Los Angeles. (liujl11@math.ucla.edu; wotaoyin@math.ucla.edu; wcli@math.ucla.edu)

‡Department of Mathematics, University of California, Riverside. (yattinc@ucr.edu)

the network simplex algorithm to solve it.

Motivations and our contributions. Although many numerical algorithms [30, 2, 29, 28] have been proposed to calculate the Wasserstein-1 distance, there is still some room to speed them up, especially for large-scale problems, for example, a grid of 512×512. Motivated by the success of multigrid methods [46] for calculating Wasserstein-p ($p>1$) distance [31, 24], we apply the cascadic multilevel method [7] to calculate Wasserstein-1 distances. We compute the distances on different grid levels and use the solutions on the coarse grids to initialize the calculation of solutions on the finer grids. We use this method to speed up the state-of-the-art algorithms [29, 28], dramatically reducing the computational expense on the finest grids and lessening the total time consumption by $2\sim 200$ times. The speedup effect depends on the size of the problem. It is significant for large-scale problems.

The rest of this paper is organized as follows. In Section 2, we briefly review the Wasserstein-1 distance. In Section 3, we demonstrate our multilevel algorithms and provide a complexity analysis in Section 4. In Section 5, we numerically validate the assumptions used in Section 4. Finally, in Section 6, we present several numerical examples.

2. Problem description. Given a domain $\Omega\subset\Re^d$, the EMD, or the Wasserstein distance, is a commonly-used metric to measure the distance between two probability distributions defined on Ω: $\rho^0,\rho^1:\Omega\to\Re$. In the 1D case ($d=1$), the Wasserstein Distance has a closed-form solution [45]. With two or higher dimensions ($d\geq 2$), the distance is no longer given in a closed form, and it is obtained via iterative algorithms. In this paper, we consider the following Wasserstein-1 distance:

$$\begin{aligned}
&\underset{\pi:\Omega\times\Omega\to\Re}{\text{minimize}} \int_{x,y\in\Omega}\|x-y\|_p\pi(x,y)dxdy\\
&\text{subject to } \int_{y\in\Omega}\pi(x,y)dy=\rho^0(x),\quad \forall x\in\Omega\\
&\qquad\qquad \int_{x\in\Omega}\pi(x,y)dx=\rho^1(y),\quad \forall y\in\Omega,\\
&\qquad\qquad \pi(x,y)\geq 0,\quad \forall x,y\in\Omega,
\end{aligned}\tag{2.1}$$

where $\|\cdot\|_p, 1\leq p\leq\infty$, is the "ground metric" of the Wasserstein distance. The minimization variable π is a joint distribution $\pi:\Omega\times\Omega\to\Re$ whose marginal distributions are ρ^0,ρ^1. The dual problem of (2.1), also named the Kantorovich dual problem, is:

$$\begin{aligned}
&\underset{\phi^0,\phi^1:\Omega\to\Re}{\text{maximize}} \int_{x\in\Omega}\phi^0(x)\rho^0(x)dx-\int_{y\in\Omega}\phi^1(y)\rho^1(y)dy\\
&\text{subject to } \phi^0(x)-\phi^1(y)\leq\|x-y\|_p,\quad \forall x,y\in\Omega,
\end{aligned}\tag{2.2}$$

where ϕ^0,ϕ^1 are (Kantorovich) dual variables.

2.1. Problem settings. In this paper, we focus on an equivalent and simpler form of (2.2). Since $\|\cdot\|_p$ is homogeneous of degree one, by [45], there is an equivalent form of (2.2), where $\phi^0=\phi^1=\phi$. In other words,

$$\begin{aligned}
&\underset{\phi:\Omega\to\Re}{\text{maximize}} \int_{x\in\Omega}\phi(x)(\rho^0(x)-\rho^1(x))dx\\
&\text{subject to } \|\nabla\phi(x)\|_q\leq 1,\quad \forall x\in\Omega,
\end{aligned}\tag{2.3}$$

where $1/p + 1/q = 1$ and $1 \le q \le \infty$. The following minimization problem, which is the dual problem of (2.3), is also considered in this paper:

$$\begin{aligned} &\underset{m:\Omega\to\Re^d}{\text{minimize}} \int_{x\in\Omega} \|m(x)\|_p dx \\ &\text{subject to } \operatorname{div}(m(x)) = \rho^0(x) - \rho^1(x), \quad \forall x \in \Omega, \\ &\qquad m(x)\cdot n(x) = 0, \quad \forall x \in \partial\Omega \end{aligned} \tag{2.4}$$

where "div" denotes the divergence operator $\operatorname{div}(m(x)) = \sum_{i=1}^d \frac{\partial m_i}{\partial x_i}(x)$ and $n(x)$ is normal to $\partial\Omega$. Here m is a d dimensional field satisfying the zero flux boundary condition [3], the solution of (2.4) m^* is called "the optimal flux".

2.2. Discretization. We set $\Omega = [0,1]^d$. Let Ω^h be a grid on Ω with step size $h > 0$:

$$\Omega^h = \{0, h, 2h, 3h, \cdots, 1\}^d.$$

Let $N = 1/h$ be the grid size. Any $x \in \Omega^h$ is a d dimensional tensor, of which the value of the i^{th} component x_i is chosen from: $x_i \in \{0, h, 2h, 3h, \cdots, 1\}$. The discretized distributions ρ_h^0, ρ_h^1 are $(N+1)^d$ tensors, and the discretized flux m_h is a $(N+1)^d \times d$ tensor, which represents a map $\Omega^h \to \Re^d$: $\rho_h^0 = \{\rho^0(x)\}_{x\in\Omega^h}$, $\rho_h^1 = \{\rho^1(x)\}_{x\in\Omega^h}$, and $m_h = \{m(x)\}_{x\in\Omega^h}$. The discretized version of (2.4) can be written as

$$\begin{aligned} &\underset{m_h:\Omega^h\to\Re^d}{\text{minimize}} \sum_{x\in\Omega^h} \|m_h(x)\|_p h^d \\ &\text{subject to} \quad \operatorname{div}^h(m_h(x)) = \rho_h^0(x) - \rho_h^1(x), \quad \forall x \in \Omega^h, \end{aligned} \tag{2.5}$$

where the discrete divergence operator is:

$$\begin{aligned} \operatorname{div}^h(m_h(x)) &= \sum_{i=1}^d D_{h,i} m(x), \\ D_{h,i}m(x) &= \begin{cases} (m_{h,i}(x_{-i}, x_i))/h, & x_i = 0 \\ (m_{h,i}(x_{-i}, x_i) - m_{h,i}(x_{-i}, x_i - h))/h, & 0 < x_i < 1 \\ (-m_{h,i}(x_{-i}, x_i - h))/h, & x_i > 1 \end{cases} \end{aligned}$$

In the definition of div^h, $m_h(x) \in \Re^d$ means the flow at point x, $m_{h,i}(x) \in \Re$ is the i^{th} component of $m_h(x)$. The notion "$-i$" refers to all the components excluding i: $x_{-i} = \{x_j : j \in \{1, 2, \cdots, d\}, j \ne i\}$.

To simplify our notation, we rewrite the above problem (2.5) as:

$$\begin{aligned} &\underset{m_h:\Omega^h\to\Re^d}{\text{minimize}} \quad f(m_h) \\ &\text{subject to} \quad A_h m_h = \rho_h. \end{aligned} \tag{2.6}$$

where $f(\cdot)$ denotes a norm of m_h, A_h denotes the divergence operator, which is linear, and $\rho_h = \rho_h^0 - \rho_h^1$.

The dual problem of (2.6), which is also the discrete version of (2.3), is:

$$\begin{aligned} &\underset{\phi_h:\Omega^h\to\Re}{\text{minimize}} \quad \sum_{x\in\Omega^h} \phi_h(x)\rho_h(x)h^d \\ &\text{subject to} \quad \|A_h^*\phi_h(x)\|_q \le 1, \ \forall x \in \Omega^h, \end{aligned} \tag{2.7}$$

where $\phi_h : \Omega^h \to \Re$ is the *Kantorovich potential*: $\phi_h = \{\phi(x)\}_{x\in\Omega^h}$. The adjoint operator of A_h, A_h^*, denotes the gradient operator.

In this paper, we solve (2.6) and (2.7) jointly by primal-dual algorithms.

Define some norms on Ω^h:

$$\|m_h\|_2^2 = \sum_{x\in\Omega^h} \|m(x)\|_2^2, \quad \|m_h\|_{L^2}^2 = \sum_{x\in\Omega^h} \|m(x)\|_2^2 h^d,$$
$$\|\phi_h\|_2^2 = \sum_{x\in\Omega^h} \phi^2(x), \quad \|\phi_h\|_{L^2}^2 = \sum_{x\in\Omega^h} \phi^2(x) h^d,$$
$$\|\varphi_h\|_2^2 = \sum_{x\in\Omega^h} \|\varphi(x)\|_2^2, \quad \|\varphi_h\|_{L^2}^2 = \sum_{x\in\Omega^h} \|\varphi(x)\|_2^2 h^d.$$

Define inner products on Ω^h:

$$\langle \phi_h, \phi_h' \rangle = \sum_{x\in\Omega^h} \phi_h(x)\phi_h'(x),$$

$$\langle \phi_h, \phi_h' \rangle_h = \sum_{x\in\Omega^h} \phi_h(x)\phi_h'(x) h^d.$$

3. Algorithm description. In this section, we review two recent primal-dual algorithms designed for (2.6) and (2.7). We apply a multilevel framework (Section 3.2) to further accelerate these algorithms.

3.1. Two recent algorithms for (2.6) and (2.7).

Algorithm 1 (Li et al. [29]). Problems (2.6) and (2.7) can be jointly solved by the following min-max problem:

$$\min_{m_h} \max_{\phi_h} L(m_h, \phi_h) = f(m_h) + \langle \phi_h, A_h m_h - \rho_h \rangle_h. \tag{3.1}$$

Inspired by the Chambolle-Pock Algorithm [10], the authors of [29] proposed the following algorithm to solve (3.1):

$$\begin{aligned} m_h^{k+1} &= \arg\min_{m_h} L(m_h, \phi_h^k) + \frac{1}{2\mu}\|m_h - m_h^k\|_{L^2}^2, \\ \bar{m}_h^{k+1} &= 2m_h^{k+1} - m_h^k, \\ \phi^{k+1} &= \arg\max_{\phi_h} L(\bar{m}_h^{k+1}, \phi_h) - \frac{1}{2\tau}\|\phi_h - \phi_h^k\|_{L^2}^2. \end{aligned} \tag{3.2}$$

Parameters $\mu, \tau > 0$ need to be tuned. If $\mu\tau\|A_h\|^2 < 1$, then we have the iteration $(m_h^k, \phi_h^k) \to (m_h^*, \phi_h^*)$, which is the solution of (3.1). In this paper, we use[1] $\mu = \tau = 1/(2\|A_h\|)$. The iteration stops when the following fixed point residual (FPR) R^k falls below a threshold:

$$\mathrm{R}_h^k := \frac{1}{\mu}\|m_h^{k+1} - m_h^k\|_{L^2}^2 + \frac{1}{\tau}\|\phi_h^{k+1} - \phi_h^k\|_{L^2}^2 - 2\big\langle \phi_h^{k+1} - \phi_h^k, \mathrm{div}^h(m_h^{k+1} - m_h^k)\big\rangle_h. \tag{3.3}$$

The algorithm is summarized in Algorithm 1.

[1]The parameter choice $\mu = \tau = 1/(2\|A_h\|)$ is convenient for complexity analysis. Practically, $\mu = \tau = 1/\|A_h\|$ is better although it does not guarantee convergence theoretically.

Algorithm 1: A primal-dual algorithm for EMD [29]

Input: Distributions ρ^0, ρ^1, grid step size h, initial point m^0, ϕ^0, tolerance ε.
while $R_h^k < \varepsilon$ *is not satisfied* **do**
  Execute (3.2).
end
Output: m^K, ϕ^K

Algorithm 2 (Jacobs et al. [28]). Problem (2.6) can be written as:

$$\min_{m_h, u_h} \max_{\varphi_h} f(u_h) + \delta_{A_h m_h = \rho_h}(m_h) + \langle \varphi_h, m_h - u_h \rangle_h,$$

which is equivalent with

$$\min_{m_h} \max_{\varphi_h} \tilde{L}(m_h, \varphi_h) = \delta_{A_h m_h = \rho_h}(m_h) - f^*(\varphi_h) + \langle \varphi_h, m_h \rangle_h. \tag{3.4}$$

In the above formula, $\varphi_h : \Omega^h \to \Re^d$ is the dual variable, that is the gradient of the Kantorovich potential: $\varphi_h = A_h^* \phi_h$. Function $\delta_{A_h m_h = \rho_h}$ is the indicator function of $A_h m_h = \rho_h$:

$$\delta_{A_h m_h = \rho_h}(m_h) = \begin{cases} 0, & \text{if } A_h m_h = \rho_h, \\ +\infty, & \text{if } A_h m_h \neq \rho_h. \end{cases}$$

Function f^* is the convex conjugate of f:

$$f^*(\varphi_h) = \sup_{u_h} \langle \varphi_h, u_h \rangle_h - f(u_h)$$

The authors of [28] solve (3.4) in the following way:

$$\begin{aligned} m_h^{k+1} &= \arg\min_{m_h} \tilde{L}(m_h, \bar{\varphi}_h^k) + \frac{1}{2\mu} \|m_h - m_h^k\|_{L^2}^2 \\ \varphi_h^{k+1} &= \arg\max_{\varphi_h} \tilde{L}(m_h^{k+1}, \varphi_h) - \frac{1}{2\tau} \|\varphi_h - \varphi_h^k\|_{L^2}^2 \\ \bar{\varphi}_h^{k+1} &= 2\varphi_h^{k+1} - \varphi_h^k, \end{aligned} \tag{3.5}$$

where the first subproblem solving m_h^{k+1} requires computing a projection onto the affine space $\{m_h | A_h m_h = \rho_h\}$. Since the discrete Laplacian inverse $((A_h)^* A_h)^{-1}$ can be easily computed by FFT, the projection could be efficiently calculated [28].

Parameters $\mu, \tau > 0$ need to be tuned. As long as $\mu\tau < 1$, we have the iteration $(m_h^k, \varphi_h^k) \to (m_h^*, \varphi_h^*)$, which is the solution of (3.4). In this paper, we choose[2] $\mu = \tau = 1/2$. The stopping condition is to have the following fixed point residual G^k small enough:

$$\mathrm{G}_h^k = \frac{1}{\mu} \|m_h^{k+1} - m_h^k\|_{L^2}^2 + \frac{1}{\tau} \|\varphi_h^{k+1} - \varphi_h^k\|_{L^2}^2 + 2\langle \varphi_h^{k+1} - \varphi_h^k, m_h^{k+1} - m_h^k \rangle_h. \tag{3.6}$$

With φ_h^* in hand, the Kantorovich potential ϕ_h^* can be easily solved by the method given in Appendix B.

The algorithm is listed in Algorithm 2.

[2]The parameter choice $\mu = \tau = 1/2$ is convenient for complexity analysis. Practically, $\mu = \tau = 1$ is better.

Algorithm 2: Prox-PDHG for EMD [28]

Input: Distributions ρ^0, ρ^1, grid step size h, initial point m^0, φ^0, tolerance ε.
while $G_h^k < \varepsilon$ *is not satisfied* **do**
  Execute (3.5).
end
Output: m^K, φ^K

3.2. A framework: multilevel initialization. In this subsection, we describe a framework, inspired by the cascadic multilevel method [7], to substantially speed up Algorithms 1 and 2. With the multilevel framework, Algorithms 1 and 2 lead to Algorithms 1M and 2M respectively.

Suppose we have L levels of grids with step sizes $h_1, h_2, \cdots, h_L$ respectively. The step sizes satisfy

$$h_1 > h_2 > \cdots > h_{L-1} > h_L = h.$$

The finest step size $h_L = h$. On each level, the space Ω is respectively discretized as

$$\Omega^{h_1}, \cdots, \Omega^{h_{L-1}}, \Omega^{h_L}.$$

If h is the power of $1/2$, we take $h_l = 2^{L-l}h$. Then we have

$$\Omega^{h_1} \subset \Omega^{h_2} \subset \cdots \Omega^{h_{L-1}} \subset \Omega^h.$$

On the l^{th} level, the optimal flux problem (2.6) is

$$\begin{aligned} &\underset{m_{h_l}:\Omega^{h_l}\to\Re^d}{\text{minimize}} f(m_{h_l}) \\ &\text{subject to } A_{h_l} m_{h_l} = \rho_{h_l} \end{aligned} \tag{3.7}$$

We apply the cascadic multilevel technique [7] to the OT problem. We use 0 initial solution on the level $l = 1$ and solve a sequence of minimization problem (3.7) with one pass from the coarsest level $l = 1$ to the finest level $l = L$. On each level, we use Algorithm 1 or Algorithm 2 that is stopped as the iterate is accurate enough ($R_{h_1}^k < \varepsilon$ for Algorithm 1, $G_{h_1}^k < \varepsilon$ for Algorithm 2). The obtained solution is denoted by $(m_{h_1}^K, \phi_{h_1}^K)$ or $(m_{h_1}^K, \varphi_{h_1}^K)$. After that, we interpolate the obtained solutions to the next level $l = 2$ and treat them as the initial solutions of level $l = 2$. The process is repeated for $l = 3, \cdots, L$. Algorithms 1M and 2M are the multilevel versions of Algorithms 1 and 2 respectively.

Practically, the solution on a coarse level is a good estimate of that on a finer level. Thus, the cascadic multilevel method works well.

3.3. Cross-level interpolation. In this subsection, we describe the cross-level interpolations in Algorithms 1M and 2M in detail.

Interpolation of potentials ϕ_h. For any $x \in \Omega^{h_l}$ on level l, we partition the set of the components x_j into two subsets, depending on whether they also belong to the grid on the coarser level $l-1$:

$$\begin{aligned} J =& \big\{ j : x_j \in \{0, h_{l-1}, 2h_{l-1}, \cdots, 1\} \big\} \\ \bar{J} =& \big\{ j : x_j \in \{0, h_l, 2h_l, \cdots, 1\}, x_j \notin \{0, h_{l-1}, 2h_{l-1}, \cdots, 1\} \big\} \end{aligned} \tag{3.8}$$

Algorithm 1M Multilevel version of Algorithm 1

Input : Distributions ρ^0, ρ^1, grid step size h, tolerance ε.
Initialization: Let $m^K_{h_0} = 0, \phi^K_{h_0} = 0$.
for $l = 1, 2, \cdots, L$ **do**
Initialize the current level:

$$m^0_{h_l} = \text{Interpolate}(m^K_{h_{l-1}}), \quad \phi^0_{h_l} = \text{Interpolate } (\phi^K_{h_{l-1}})$$

Call Algorithm 1:

$$(m^K_{h_l}, \phi^K_{h_l}) = \text{Algorithm 1}(\rho^0, \rho^1, h_l, m^0_{h_l}, \phi^0_{h_l}, \varepsilon)$$

end
Output: $m^K_{h_L}, \phi^K_{h_l}$

Algorithm 2M Multilevel version of Algorithm 2

Input : Distributions ρ^0, ρ^1, grid step size h, tolerance ε.
Initialization: Let $m^K_{h_0} = 0, \varphi^K_{h_0} = 0$.
for $l = 1, 2, \cdots, L$ **do**
Initialize the current level:

$$m^0_{h_l} = \text{Interpolate } (m^K_{h_{l-1}}), \quad \varphi^0_{h_l} = \text{Interpolate } (\varphi^K_{h_{l-1}})$$

Call Algorithm 2:

$$(m^K_{h_l}, \varphi^K_{h_l}) = \text{Algorithm 2}(\rho^0, \rho^1, h_l, m^0_{h_l}, \varphi^0_{h_l}, \varepsilon)$$

end
Output: $m^K_{h_L}, \phi^K_{h_l}$ (Obtain $\phi^K_{h_l}$ from $\varphi^K_{h_l}$, see Appendix B)

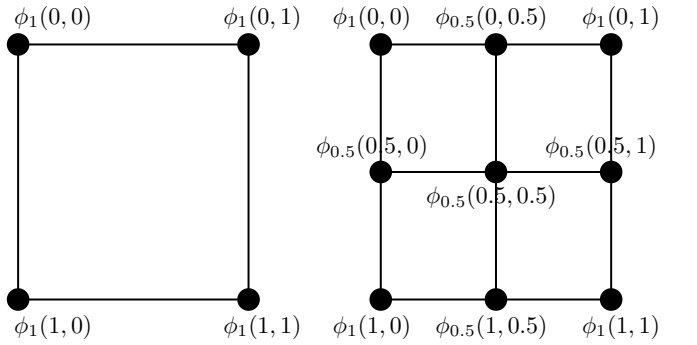
Let the elements in $\bar{J}$ be denoted as $j_1, j_2, \cdots, j_{|\bar{J}|}$. The mapping $\phi_{h_l} = \text{Interpolate } (\phi_{h_{l-1}})$ is defined pointwisely as the average value of a neighborhood. For $x \in \Omega^{h_l}$,

(3.9)
$$\phi_{h_l}(x) = \frac{1}{2^{|\bar{J}|}} \sum_{|y_{j_1} - x_{j_1}| \le h_l} \sum_{|y_{j_2} - x_{j_2}| \le h_l} \cdots \sum_{|y_{j_{|\bar{J}|}} - x_{j_{|\bar{J}|}}| \le h_l} \phi_{h_{l-1}}(x_J, y_{j_1}, y_{j_2}, \cdots, y_{j_{|\bar{J}|}}).$$

For example, if $d = 2$ (2D case) and $h_l = h_{l-1}/2$, (3.9) can be written as:

$$\phi_{h_l}(x_1, x_2) = \begin{cases} \phi_{h_{l-1}}(x_1, x_2), & \text{if } x_1/h_l, x_2/h_l \text{ are even} \\ \Big(\phi_{h_{l-1}}(x_1, x_2 - h_l) + \phi_{h_{l-1}}(x_1, x_2 + h_l)\Big)/2, & \text{if } x_1/h_l \text{ is even} \\ \Big(\phi_{h_{l-1}}(x_1 - h_l, x_2) + \phi_{h_{l-1}}(x_1 + h_l, x_2)\Big)/2, & \text{if } x_2/h_l \text{ is even} \\ \Big(\phi_{h_{l-1}}(x_1 - h_l, x_2 - h_l) + \phi_{h_{l-1}}(x_1 - h_l, x_2 + h_l)... \\ +\phi_{h_{l-1}}(x_1 + h_l, x_2 - h_l) + \phi_{h_{l-1}}(x_1 + h_l, x_2 + h_l)\Big)/4, & \text{otherwise} \end{cases}$$

Figure 1 gives an illustration of this 2D interpolation.



$$\phi_1, h = 1 \qquad \phi_{0.5} = \text{Interpolate}(\phi_1), h = 0.5$$

$$\phi_{0.5}(0.5,1) = \tfrac{\phi_1(0,1)+\phi_1(1,1)}{2} \qquad \phi_{0.5}(0,0.5) = \tfrac{\phi_1(0,0)+\phi_1(0,1)}{2}$$

$$\phi_{0.5}(1,0.5) = \tfrac{\phi_1(1,0)+\phi_1(1,1)}{2} \qquad \phi_{0.5}(0.5,0) = \tfrac{\phi_1(0,0)+\phi_1(1,0)}{2}$$

$$\phi_{0.5}(0.5,0.5) = \tfrac{\phi_1(0,0)+\phi_1(0,1)+\phi_1(1,0)+\phi_1(1,1)}{4}$$

FIGURE 1. *An illustration of (3.9) (2D case): from* 1×1 *grid to* 2×2 *grid*

Interpolation of flux m_h. Due to the zero-flux boundary condition for (2.4), interpolating m is different from ϕ. The flow m can be viewed as "edge weights" on the grid [29, 2], as in Figure 2. With the definition of J (3.8), $m_{h_l} = \text{Interpolate}\ (m_{h_{l-1}})$ is pointwisely defined in (3.10). For $x \in \Omega^{h_l}, i = 1, 2, \cdots, d$,

$$m_{h_l,i}(x) = \begin{cases} \frac{1}{2^{|J|}} \sum_{|y_{j_1} - x_{j_1}| \le h_l} \cdots \sum_{|y_{j_{|\bar{J}|}} - x_{j_{|\bar{J}|}}| \le h_l} \\ \qquad m_{h_{l-1},i}(x_J, y_{j_1}, y_{j_2}, \cdots, y_{j_{|\bar{J}|}}), \quad i \in J \\ \frac{1}{2^{|J|}} \sum_{|y_{j_1} - x_{j_1}| \le h_l} \cdots \sum_{|y_{j_{|\bar{J}|}} - x_{j_{|\bar{J}|}}| \le h_l} \\ \qquad m_{h_{l-1},i}(x_J, y_i, y_{j_1}, y_{j_2}, \cdots, y_{j_{|\bar{J}|}}), \quad i \notin J, \end{cases} \tag{3.10}$$

where y_i is an element in $\{0, h_{l-1}, 2h_{l-1}, \cdots, 1\}$ which is the nearest to x_i.

For example, if $d = 2$ (2D case) and $h_l = h_{l-1}/2$, (3.10) can be written as:

$$m_{h_l,1}(x_1,x_2) = \begin{cases} m_{h_{l-1},1}(x_1,x_2), & \text{if } x_1/h_l, x_2/h_l \text{ are even} \\ m_{h_{l-1},1}(x_1 - h_l, x_2), & \text{if } x_2/h_l \text{ is even} \\ \Big(m_{h_{l-1},1}(x_1,x_2-h_l) + m_{h_{l-1},1}(x_1,x_2+h_l)\Big)/2, & \text{if } x_1/h_l \text{ is even} \\ \Big(m_{h_{l-1},1}(x_1-h_l,x_2-h_l) + m_{h_{l-1},1}(x_1-h_l,x_2+h_l)\Big)/2, & \\ & \text{otherwise} \end{cases}$$

$$m_{h_l,2}(x_1,x_2) = \begin{cases} m_{h_{l-1},2}(x_1,x_2), & \text{if } x_1/h_l, x_2/h_l \text{ are even} \\ m_{h_{l-1},2}(x_1, x_2 - h_l), & \text{if } x_1/h_l \text{ is even} \\ \Big(m_{h_{l-1},2}(x_1-h_l,x_2) + m_{h_{l-1},2}(x_1+h_l,x_2)\Big)/2, & \text{if } x_2/h_l \text{ is even} \\ \Big(m_{h_{l-1},2}(x_1-h_l,x_2-h_l) + m_{h_{l-1},2}(x_1+h_l,x_2-h_l)\Big)/2, & \\ & \text{otherwise} \end{cases}$$

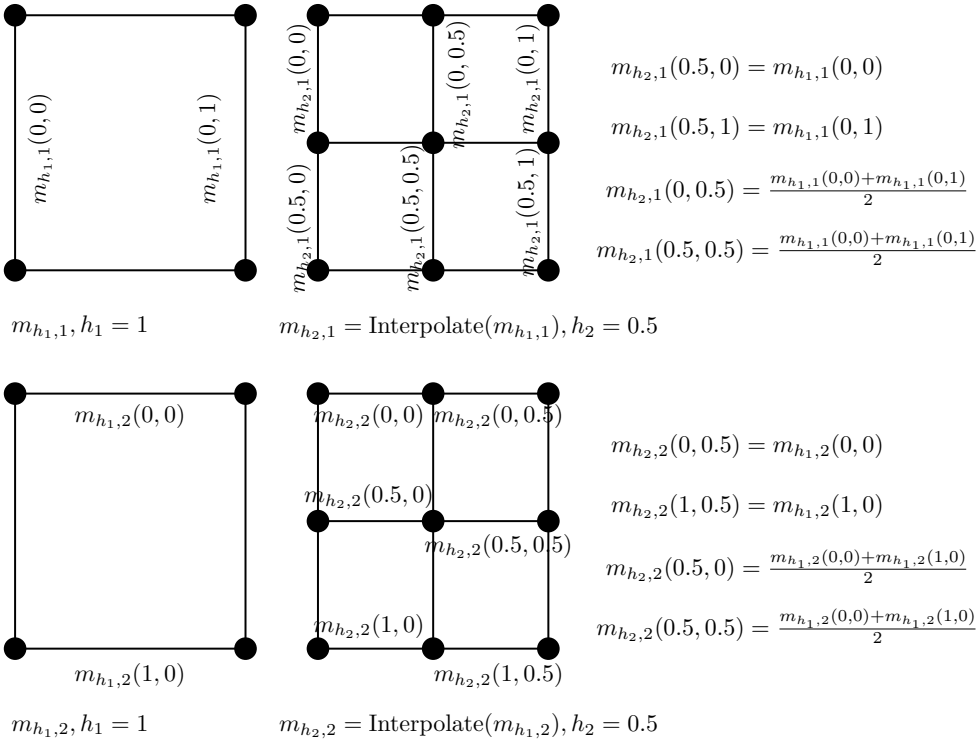


FIGURE 2. *An illustration of (3.10) (2D case): from* 1×1 *grid to* 2×2 *grid*

Figure 2 illustrates the above formula.

Interpolation of φ_h. Since φ_h has the same dimension with m_h, the interpolation of φ_h is the same with interpolation of flux m_h (3.10).

4. Analysis of computational costs. In this section, we provide complexity analysis of Algorithms 1, 2, 1M and 2M.

4.1. Analysis of Algorithms 1 and 1M. Let $z_h = (m_h, \phi_h)$ and $Z^*_{h_l}$ be the solution set of the l^{th} level min-max problem:

$$Z^*_{h_l} = \left\{ (m^*_{h_l}, \phi^*_{h_l}) \middle| (m^*_{h_l}, \phi^*_{h_l}) \text{ is a saddle point of } L(m_{h_l}, \phi_{h_l}) \right\},$$

where L is defined in (3.1).

ASSUMPTION 1. *The solution sets on all the levels are nonempty and bounded, i.e.,*

$$\|z^*_{h_l}\|^2_{L^2} \le C_1, \quad \forall z^*_{h_l} \in Z^*_{h_l}, \quad \forall l = 1, 2, \cdots, L \tag{4.1}$$

Assumption 1 is mild. Since $z^*_{h_l} = (m^*_{h_l}, \phi^*_{h_l})$, the norm of $z^*_{h_l}$ can be decomposed as $\|z^*_{h_l}\|^2_{L^2} = \|m^*_{h_l}\|^2_{L^2} + \|\phi^*_{h_l}\|^2_{L^2}$. The dual solution $\phi^*_{h_l}$, by the definition in (2.7), has the property: $\|A^*_h \phi^*_{h_l}(x)\|_q \le 1, \forall x \in \Omega^{h_l}$, where A^*_h is the gradient operator defined on Ω^{h_l}. It implies that all the dual solutions $\phi^*_{h_l}$ are Lipschitz continuous uniformly on the compact domain $\Omega = [0,1]^d$. Thus, all the dual solutions $\phi^*_{h_l}$ are

uniformly bounded as long as they are kept zero-meaned. Actually keeping $\phi^*_{h_l}$ to be zero-meaned is not difficult, see [45]. The primal solution $m^*_{h_l}$, by definition, is the solution of minimization problem (3.7). Thus, $f(m^*_{h_l})$ must be uniformly bounded. Although the L^2 norm may not be controlled by $f(m^*_{h_l}) = \sum_{x\in\Omega^{h_l}} \|m^*_{h_l}(x)\|_p h_l^d$, on commonly used examples, we numerically validated Assumption 1 in Table 3 and observed that C_1 exists and is independent of grid size.

ASSUMPTION 2. *For any optimal solution $z^*_{h_l} \in Z^*_{h_l}$ on level l, there exists an optimal solution $z^*_{h_{l+1}} \in Z^*_{h_{l+1}}$ on the finer level $l+1$ such that*

$$\|\text{Interpolate}(z^*_{h_l}) - z^*_{h_{l+1}}\|^2_{L^2} \le C_2(h_l)^r, \quad \forall l = 1, 2, \cdots, L-1, \tag{4.2}$$

where $r > 0$ depends on the smoothness of the solution $z^*_{h_l}$, the interpolation method we choose and the properties of $\rho^0_{h_l}$ and $\rho^1_{h_l}$ on each of the levels.

Assumption 2 requires the solution sets between two consecutive levels are close to each other. We are not able to show (4.2) holds theoretically, because different density $\rho = \rho^0 - \rho^1$ lead to different r. However, Assumption 2 holds on commonly used examples. We numerically validated it in Section 5.2 with $d = 2$ and $p = 1, 2, \infty$. Figure 3 gives a visualized example of the multiple solutions on different levels. Table 4 quantifies $\|\text{Interpolate}(z^*_{h_l}) - z^*_{h_{l+1}}\|^2_{L^2}$ and shows that r is approximately $1 \le r \le 2$.

In the following theorem, we consider only the case of $r < d+1$. Actually $r < d+1$ is a worse case compared with $r \ge d+1$. If $r \ge d+1$ holds, our multilevel method is so efficient that the complexity of Algorithm 1M is even unrelated with h because the complexity is no longer dominated by the calculation on the finest level. Practically $r \ge d+1$ rarely happens, and we ignore this case.

THEOREM 4.1. *Given ρ^0, ρ^1, h, if Assumptions 1,2 hold and $h_l = 2^{L-l}h$, then it holds that:*[3]

1. *Given 0 as the initialization, Algorithm 1 takes $O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h})$ iterations to stop.*
2. *Given 0 as the initialization, the complexity of Algorithm 1 is $O(\frac{1}{\varepsilon}\frac{d^{3/2}}{h^{d+1}})$.*
3. *Algorithm 1M takes $O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h^{1-r}})$ iterations on the finest level if $L \ge 2$.*
4. *If $r < d+1$ and L large enough, calculation on the finest level L is the dominant term in Algorithm 1M, the complexity of the algorithm is $O(\frac{1}{\varepsilon}\frac{d^{3/2}}{h^{d+1-r}})$.*

This theorem shows why and how much Algorithm 1M helps speed up Algorithm 1. As long as the optimal solution on the coarse level is close to one of the optimal solutions on the finer level, the multilevel technique is able to reduce the number of iterations on the finer level. If the distance between the coarse solution and the fine solution is controlled by $O(h^r)$, the order of the complexity of Algorithm 1 can be reduced by h^r. Table 7 demonstrates the number of iterations and calculation time are significantly reduced.

Proof. **Step 1:** Analyzing how many iterations Algorithm 1 takes. Define

$$M_h = h^d \begin{bmatrix} I/\mu & -(A_h)^* \\ -A_h & I/\tau \end{bmatrix}.$$

The fixed point residual can be written as $R^k_h = \|z^{k+1}_h - z^k_h\|^2_{M_h}$. Then Chambolle-Pock is equivalent with proximal point algorithm (PPA) with M_h-metric (Theorem 1

[3] In this article, $O(\cdot)$ denotes the asymptotic rate as $\varepsilon \to 0$ and $h \to 0$.

in [29]). Since $\mu\tau\|A_h\|^2 < 1$ is satisfied, we have the following conclusions [26]:

$$R_h^k \le \frac{1}{k}\|z_h^0 - z_h^*\|_{M_h}^2, \quad \forall z_h^* \in Z_h^*, \tag{4.3}$$

FPR is monotone:

$$R_h^{k+1} \le R_h^k, \quad \forall k, \tag{4.4}$$

and the global convergence holds in the sense:

$$z_h^k \to z_h^*, \quad \text{for some } z_h^* \in Z_h^*, \text{ as } k \to \infty. \tag{4.5}$$

By Cauchy-Swartz, we obtain

$$2\langle \phi_h^{k+1} - \phi_h^k, A_h(m_h^{k+1} - m_h^k)\rangle \le \|A_h\| \cdot \|\phi_h^{k+1} - \phi_h^k\|_2^2 + \|A_h\| \cdot \|m_h^{k+1} - m_h^k\|_2^2.$$

The above inequality and the parameter choice $\mu = \tau = 1/(2\|A_h\|)$ lead to

$$\|z_h^0 - z_h^*\|_{M_h}^2 \le \|A_h\|\|z_h^0 - z_h^*\|_{L^2}^2.$$

The norm $\|A_h\|$ is the square root of the largest sigular-value of $(A_h)^*A_h$, which is the discrete Laplacian with grid step size h. By the Gershgorin circle theorem [22], $\sigma_{\max}\big((A_h)^*A_h\big) \le 4\frac{d}{h^2}$ and, thus, $\|A_h\| \le 2\frac{\sqrt{d}}{h}$, which imples

$$\|z_h^0 - z_h^*\|_{M_h}^2 \le 2\frac{\sqrt{d}}{h}\|z_h^0 - z_h^*\|_{L^2}^2.$$

Since we take zero as the initialization $z_h^0 = 0$, based on Assumption 1 and conclusion (4.3), as long as $k > 2C_1\frac{1}{\varepsilon}\frac{\sqrt{d}}{h}$, we have

$$R_h^k \le \frac{1}{k}\|z_h^0 - z_h^*\|_{M_h}^2 \le 2\sqrt{d}\frac{1}{kh}\|z_h^0 - z_h^*\|_{L^2}^2 \le 2\sqrt{d}\frac{1}{kh}C_1 < \varepsilon,$$

That is, within $(2C_1\frac{1}{\varepsilon}\frac{\sqrt{d}}{h}) \approx O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h})$ iterations, the stopping condition of Algorithm 1 is satisfied.

Step 2: Analyzing how many iterations Algorithm 1M take.

Level 1: Using the similar argument in Step 1, we conclude that Algorithm 1M takes $O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h_1})$ iterations to stop on level $l = 1$.

Level 2: The calculation of level 2 is initialized by the result of level 1. Conclusions (4.4) and (4.5) also hold for h_1. By the global convergence of $\{z_{h_1}^k\}_k$ (4.5), there exists a $\bar{K}$ such that $\|z_{h_1}^k - z_{h_1}^*\|_{L^2}^2 \le C_2(h_1)^r$ for all $k \ge \bar{K}$. Now we set $\bar{\varepsilon} = R_{h_1}^{\bar{K}}$. Then, by the monotonicity of FPR (4.4), we conclude that, as long as $\varepsilon < \bar{\varepsilon}$, when Level 1 stops, the final iteration K satisfies $K \ge \bar{K}$, $\|z_{h_1}^K - z_{h_1}^*\|_{L^2}^2 \le C_2(h_1)^r$ is achieved.

Lemma A.1 (Appendix A) implies $\|\text{Interpolate }(z_{h_1}^K - z_{h_1}^*)\|_{L^2}^2 \le \|z_{h_1}^K - z_{h_1}^*\|_{L^2}^2$.

As long as $k > 2^{r+3}C_2\frac{1}{\varepsilon}\frac{\sqrt{d}}{h_2^{1-r}}$, we have

$$
\begin{aligned}
&R_{h_2}^k \le \frac{1}{k}\|z_{h_2}^0 - z_{h_2}^*\|_{M_{h_2}}^2 \le 2\sqrt{d}\frac{1}{kh_2}\|z_{h_2}^0 - z_{h_2}^*\|_{L^2}^2\\
=&2\sqrt{d}\frac{1}{kh_2}\|\text{Interpolate }(z_{h_1}^K) - z_{h_2}^*\|_{L^2}^2\\
\le&2\sqrt{d}\frac{1}{kh_2}\Big(2\|\text{Interpolate}(z_{h_1}^K) - \text{Interpolate}(z_{h_1}^*)\|_{L^2}^2 + 2\|\text{Interpolate}(z_{h_1}^*) - z_{h_2}^*\|_{L^2}^2\Big)\\
=&2\sqrt{d}\frac{1}{kh_2}\Big(2\|\text{Interpolate }(z_{h_1}^K - z_{h_1}^*)\|_{L^2}^2 + 2\|\text{Interpolate }(z_{h_1}^*) - z_{h_2}^*\|_{L^2}^2\Big)\\
\le&2\sqrt{d}\frac{1}{kh_2}\Big(2\|z_{h_1}^K - z_{h_1}^*\|_{L^2}^2 + 2\|\text{Interpolate }(z_{h_1}^*) - z_{h_2}^*\|_{L^2}^2\Big)\\
\le&2\sqrt{d}\frac{1}{kh_2}\Big(2C_2(h_1)^r + 2C_2(h_1)^r\Big)\\
=&2\sqrt{d}\frac{1}{kh_2}\Big(2C_2(2h_2)^r + 2C_2(2h_2)^r\Big) = \frac{2^{r+3}C_2}{k}\frac{\sqrt{d}}{h_2^{1-r}} < \varepsilon.
\end{aligned}
$$

In the above arguments, K represents the final iteration of level 1, k means the k^{th} iteration of level 2. Consequently, within $(2^{r+3}C_2\frac{1}{\varepsilon}\frac{\sqrt{d}}{h_2^{1-r}}) \approx O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h_2^{1-r}})$ iterations, Level 2 stops.

With the same proof line, we have, for Levels $2, 3, \cdots, L$, the number of iterations are $O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h_2^{1-r}})$, $O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h_3^{1-r}}), \cdots O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h^{1-r}})$, respectively. Point 3 of Theorem 4.1 is proved.

Step 3: Analyzing complexities of Algorithm 1 and 1M.

First, we consider the case where $p = 1$ or $p = 2$.

For Algorithm 1, the complexity is a product of the iterations, the complexity is asymptotically "iterations × single step complexity." In each step of Algorithm 1, the dominant calculation is computing $A_h m_h$ or $(A_h)^* \phi_h$ [29], which has a complexity of $O(d\frac{1}{h^d})$. Thus, the total complexity of Algorithm 1 is:

$$O(\frac{1}{\varepsilon}\frac{\sqrt{d}}{h}) \times O(d\frac{1}{h^d}) = O(\frac{1}{\varepsilon}\frac{d^{3/2}}{h^{d+1}}).$$

For Algorithm 1M, the complexity is a product of two parts: iterations on all levels and the interpolations between the levels. Let us first consider the former part. Similar to Algorithm 1, the complexity of level 1 is $O(\frac{1}{\varepsilon}\frac{d^{3/2}}{h_1^{d+1}})$. The complexity of Level $l (2 \le l \le L)$ is $O(\frac{1}{\varepsilon}\frac{d^{3/2}}{h_l^{d+1-r}})$. Since $h_L = h, h_{L-1} = 2h, h_{L-2} = 2^2 h, \cdots$, we have

$$
\begin{aligned}
&\sum_{l=2}^{L} O\Big(\frac{1}{\varepsilon}\frac{d^{3/2}}{h_l^{d+1-r}}\Big) + O\Big(\frac{1}{\varepsilon}\frac{d^{3/2}}{h_1^{d+1}}\Big)\\
=&O\Big(\frac{1}{\varepsilon}\frac{d^{3/2}}{h^{d+1-r}}\Big)\Big(\sum_{i=0}^{L-2} 2^{-i(d+1-r)} + \frac{C_1}{C_2}2^{-(d+1)(L-1)}h^{-r}\Big).
\end{aligned}
$$

As L large enough, $2^{-(d+1)(L-1)}h^{-r} \le 1$ holds. As $r < d+1$, $\sum_{i=0}^{L-2} 2^{-i(d+1-r)} < \infty$ holds. Thus, the above complexity is asymptotically $O(\frac{1}{\varepsilon}\frac{d^{3/2}}{h^{d+1-r}})$ if $r < d+1$.

Now let us consider the second part, the complexity of interpolations between the levels l and $l+1$. Each node on level $l+1$ is obtained by no more than 2^d nodes,

totally we have $O(d/h_{l+1}^d)$ nodes, so the complexity of interpolation between the levels l and $l+1$ is $O(d2^d/h_{l+1}^d)$. The complexity of interpolations in Algorithm 1M is

$$O(\frac{d2^d}{h^d})(1+\frac{1}{2^d}+\frac{1}{2^{2d}}+\cdots+\frac{1}{2^{(L-1)d}})=O(\frac{d2^d}{h^d}).$$

As long as ε is small enough, $\frac{1}{\varepsilon}\frac{d^{3/2}}{h^{d+1-r}} \gg \frac{d2^d}{h^d}$, i.e. the calculation of Algorithm 1 on all the levels dominates the calculation in Algorithm 1M. The complexity of Algorithm 1M is $O(\frac{1}{\varepsilon}\frac{d^{3/2}}{h^{d+1-r}})$.

For $p=\infty$, the dominant calculation in a single step includes two parts. One is computing $A_h m_h$ or $(A_h)^*\phi_h$, which we analyze in the case of $p=1,2$. The other is calculating ℓ_∞ shrinkage operator. By the Moreau decomposition [34], computing an ℓ_∞ shrinkage operator is equivalent with computing a projection onto an ℓ_1 ball. By [16], the complexity of the latter is $O(d)$. We need to project all the points $x\in\Omega^h$. In total, there are $O(N^d)=O(1/h^d)$ points, so the single step complexity is $O(\frac{d}{h^d})$. Following the above argument, we obtain the complexities of Algorithm 1 and Algorithm 1M as $p=\infty$ has the same asymptotic rate as $p=1,2$. ∎

4.2. Analysis of Algorithms 2 and 2M. Let $y_h=(m_h,\varphi_h)$. Let $Y_{h_l}^*$ be the solution set of the l^{th} level min-max problem:

$$Y_{h_l}^*=\left\{(m_{h_l}^*,\varphi_{h_l}^*)\middle|(m_{h_l}^*,\varphi_{h_l}^*)\text{ is a saddle point of }\tilde{L}(m_{h_l},\varphi_{h_l})\right\},$$

where $\tilde{L}$ is defined in (3.4).

ASSUMPTION 3. *The solution sets on all the levels are nonempty and bounded, i.e.,*

$$\|y_{h_l}^*\|_{L^2}^2\le C_3,\quad \forall y_{h_l}^*\in Y_{h_l}^*,\ \forall l=1,2,\cdots,L \tag{4.6}$$

Assumption 3 is mild. By $y_{h_l}^*=(m_{h_l}^*,\varphi_{h_l}^*)$, we have $\|y_{h_l}^*\|_{L^2}^2=\|m_{h_l}^*\|_{L^2}^2+\|\varphi_{h_l}^*\|_{L^2}^2$. The dual optimal solution $\varphi_{h_l}^*$, by the definition in (3.4), has the property: $\varphi_{h_l}^*=A_h^*\phi_{h_l}^*$. Since $\|A_h^*\phi_{h_l}^*(x)\|_q\le 1,\forall x\in\Omega^{h_l}$, we have $\|\varphi_{h_l}^*(x)\|_q\le 1,\forall x\in\Omega^{h_l}$, which implies all the dual solutions $\phi_{h_l}^*$ are uniformly bounded:[4]

$$\|\varphi_{h_l}^*(x)\|_{L^2}^2=\sum_{x\in\Omega^{h_l}}\|\varphi_{h_l}^*(x)\|_2^2h_l^d\le\sum_{x\in\Omega^{h_l}}d\|\varphi_{h_l}^*(x)\|_q^2h_l^d\le d.$$

The primal solution $m_{h_l}^*$ in this assumption shares the same properties with that in Assumption 1. We validated Assumption 3 numerically in Table 5, we can see that C_3 is independent of the grid size.

ASSUMPTION 4. *For any optimal solution $y_{h_l}^*\in Y_{h_l}^*$ on level l, there exists an optimal solution $y_{h_{l+1}}^*\in Y_{h_{l+1}}^*$ on the finer level $l+1$ such that*

$$\|\text{Interpolate}(y_{h_l}^*)-y_{h_{l+1}}^*\|_{L^2}^2\le C_4(h_l)^\nu,\quad \forall l=1,2,\cdots,L-1. \tag{4.7}$$

where $\nu>0$ depends on the smoothness of the solution $y_{h_l}^*$, the interpolation method we choose and the properties of $\rho_{h_l}^0$ and $\rho_{h_l}^1$ on each of the levels.

[4]This bound is due to the fact that $\|a\|_2\le\sqrt{d}\|a\|_q$ for all $a\in\Re^d$ and $1\le q\le\infty$.

Similar to Assumption 2, this assumption is numerically validated in Section 5.4 with $d = 2$ and $p = 1, 2, \infty$. Figures 5 provides a visualization and Table 6 quantifies $\|\text{Interpolate}(y^*_{h_l}) - y^*_{h_{l+1}}\|^2_{L^2}$ and shows that ν is approximately 1 on the commonly used examples.

In the following theorem, we only consider the case of $\nu < d$. Actually $\nu < d$ is a case worse to deal with than $\nu \geq d$. If $\nu \geq d$ holds, our multilevel method is so efficient that the complexity of Algorithm 2M is not even related with h because the complexity is no longer dominated by the calculation on the finest level. Practically $\nu \geq d$ rarely happens, so we ignore this case.

THEOREM 4.2. *Given ρ^0, ρ^1, h, if Assumptions 3,4 hold and $h_l = 2^{L-l}h$, then it holds that*

1. *Given 0 as initialization, Algorithm 2 takes $O(\frac{1}{\varepsilon})$ iterations to stop.*
2. *Given 0 as initialization, the complexity of Algorithm 2 is $O(\frac{1}{\varepsilon}\frac{d}{h^d}\log(\frac{1}{h}))$.*
3. *Algorithm 2M takes $O(\frac{1}{\varepsilon}h^\nu)$ iterations on the finest level if $L \geq 2$.*
4. *If $\nu < d$ and L is large enough, calculation on the finest level L is the dominant term in Algorithm 2M, the complexity of the algorithm is $O(\frac{1}{\varepsilon}\frac{d}{h^{d-\nu}}\log(\frac{1}{h}))$.*

Similar to Theorem 4.1, this theorem shows Algorithm 2M helps speed up Algorithm 2 when the solution on the coarse level is a good estimate of that on a finer level. Table 9 numerically validates the theorem: the number of iterations and calculation time are largely reduced.

Proof. **Step 1:** Analyzing number of iterations Algorithms 2 and 2M require.

Let

$$\tilde{M} = \begin{bmatrix} I/\mu & I \\ I & I/\tau \end{bmatrix}.$$

Similar to Algorithm 1, Algorithm 2 is equivalent with PPA with $\tilde{M}$-metric.

Just follow the same proof line of step 1 in the proof of Theorem 4.1. Substituting $-A_h$ with I, we obtain: Algorithm 2 takes $O(1/\varepsilon)$ iterations to stop. The number of iterations is not related with grid step size h. This conclusion is consistent with the results in [28]. Moreover, Algorithm 2M takes $O(\frac{1}{\varepsilon})$ iterations for level $l = 1$ and $O(\frac{1}{\varepsilon}(h_l)^\nu)$ iterations for level $l, 2 \leq l \leq L$.

Step 2: Analyzing complexities.

First, we consider the case where $p = 1$ or $p = 2$.

For Algorithm 2, the complexity can be estimated by "iterations $\times$ single step complexity." In each step of Algorithm 2, the dominant calculation is conducting d dimensional FFT on $\bar{\varphi}^k_h$ [28], which have complexity of $O(N^d \log(N^d)$. Since $N = 1/h$, the complexity is $O(d\frac{1}{h^d}\log(\frac{1}{h}))$ [17]. Then the complexity of Algorithm 1 is:

$$O(\frac{1}{\varepsilon}) \times O(d\frac{1}{h^d}\log(\frac{1}{h})) = O(\frac{1}{\varepsilon}\frac{d}{h^d}\log(\frac{1}{h})).$$

For Algorithm 2M, we first analyze the complexity of the calculation on each level.

The complexity of level $l = 1$ is $O(\frac{1}{\varepsilon}\frac{d}{h_1^d}\log(\frac{1}{h_1}))$. The complexity of level $l(2 \leq$

$l \le L$) is $O(\frac{1}{\varepsilon}\frac{d}{h_l^{d-\nu}}\log(\frac{1}{h_l}))$. Since $h_L = h, h_{L-1} = 2h, h_{L-2} = 2h, \cdots$, we have

$$\begin{aligned}&\sum_{l=2}^{L} O\Big(\frac{1}{\varepsilon}\frac{d}{h_l^{d-\nu}}\log(\frac{1}{h_l})\Big) + O\Big(\frac{1}{\varepsilon}\frac{d}{h_1^{d}}\log(\frac{1}{h_1})\Big)\\ \le& O\Big(\frac{1}{\varepsilon}\frac{d}{h^{d-\nu}}\log(\frac{1}{h})\Big)\Big(\sum_{i=0}^{L-2} 2^{-i(d-\nu)} + 2^{-d(L-1)}h^{-\nu}\Big).\end{aligned}$$

As L large enough, $2^{-d(L-1)}h^{-\nu} \le 1$ holds. As $\nu < d$, $\sum_{i=0}^{L-2} 2^{-i(d-\nu)} < \infty$ holds. Thus, the complexity of the calculation on all levels is asymptotically $O(\frac{1}{\varepsilon}\frac{d}{h^{d-\nu}}\log(\frac{1}{h}))$.

Using similar argument of that in Theorem 4.1, as ε small enough, the above complexity is much larger than that of interpolation, i.e., $\frac{1}{\varepsilon}\frac{d}{h^{d-\nu}} \gg \frac{d2^d}{h^d}$. The complexity of Algorithm 2M is $O(\frac{1}{\varepsilon}\frac{d}{h^{d-\nu}}\log(\frac{1}{h}))$. Moreover, in the case of $p = \infty$, with the same argument in Step 3 of the proof of Theorem 4.1, we obtain the conclusions in Theorem 4.2. ∎

4.3. Summary of complexities. Tables 1 and 2 summarize the complexities. Complexity results of Algorithms 1, 2, 1M and 2M are given in Table 1. Let $N = 1/h$, Table 1 can be directly obtained by Theorems 4.1 and 4.2. In the case of $d = 2$ (2D case), we compare Algorithms 1, 2, 1M and 2M with other EMD algorithms [30, 2] in Table 2. By [30], their algorithm has complexity of $O((N^d)^2)$. As $d = 2$, it is $O(N^4)$. The algorithm in [2] constructs a graph and solves the uncapacitated minimum cost flow problem on the graph. The worst case complexity is $O\big(|V|\log(|V|)(|V|\log(|V|) + |E|)\big)$, where $|V|$ is the number of nodes in the created graph and $|E|$ is the number of edges. As $p = 1$ or $p = \infty$, $|V| = O(N^2), |E| = O(N^2)$, the complexity is $O(N^4\log^2(N))$; as $p = 2$, $|V| = O(N^2), |E| = O(N^4)$, the complexity is $O(N^6\log(N))$.

TABLE 1
Complexities of Algorithms 1, 2, 1M and 2M. The parameters r, ν depend on the interpolation accuracy (Assumptions 2, 4).

$p = 1, 2, \infty$	
Algorithm 1 [29]	$O(\frac{1}{\varepsilon}d^{3/2}N^{d+1})$
Algorithm 2 [28]	$O(\frac{1}{\varepsilon}dN^d\log(N))$
Algorithm 1M	$O(\frac{1}{\varepsilon}d^{3/2}N^{d+1-r})$
Algorithm 2M	$O(\frac{1}{\varepsilon}dN^{d-\nu}\log(N))$

5. Numerical validation of the assumptions. We numerically validated Assumptions 1, 2, 3 and 4 in the case of dimension $d = 2$ and $p \in \{1, 2, \infty\}$. We implemented Algorithms 1M and 2M in MATLAB to validate our assumptions.

5.1. Validation of Assumption 1. Since the EMD generally does not have a closed-form solution in the 2D case, we numerically estimate $\|z^*_{h_l}\|^2_{L^2}$ to validate Assumption 2. By Theorem 1 in [29], $z^k_{h_l} \to z^*_{h_l}$ as $k \to \infty$ for all l. Consequently, as long as the stopping tolerance ε is small enough, we could use $\|z^K_{h_l}\|^2_{L^2}$ obtained by Algorithm 1M to estimate $\|z^*_{h_l}\|^2_{L^2}$. In this subsection, we set $\varepsilon = 10^{-8}$.

Table 3 reports the averaged quantity of $\|z^*_{h_l}\|^2_{L^2}$ on the DOTmark dataset [42]. The results show that $\|z^*_{h_l}\|^2_{L^2}$ is clearly bounded by a constant independent of grid

TABLE 2
Complexity analysis of 2D case ($d = 2$): Tree-EMD and Min-cost flow are exact algorithms, but they are computational expensive for large grid sizes. Algorithms 1 and 2 are inexact algorithms with tolerance ε, they are more efficient for large-scale problems. With multilevel initialization, Algorithms 1M and 2M enjoy cheaper complexities than Algorithms 1 and 2 respectively.

	$p=1$	$p=2$	$p=\infty$
Tree-EMD [30]	$O(N^4)$	-	-
Min-cost flow[2][5]	$O(N^4\log^2(N))$	$O(N^6\log(N))$	$O(N^4\log^2(N))$
Algorithm 1 [29]	$O(\frac{1}{\varepsilon}N^3)$	$O(\frac{1}{\varepsilon}N^3)$	$O(\frac{1}{\varepsilon}N^3)$
Algorithm 2 [28]	$O(\frac{1}{\varepsilon}N^2\log(N))$	$O(\frac{1}{\varepsilon}N^2\log(N))$	$O(\frac{1}{\varepsilon}N^2\log(N))$
Algorithm 1M	$O(\frac{1}{\varepsilon}N^{3-r})$	$O(\frac{1}{\varepsilon}N^{3-r})$	$O(\frac{1}{\varepsilon}N^{3-r})$
Algorithm 2M	$O(\frac{1}{\varepsilon}N^{2-\nu}\log(N))$	$O(\frac{1}{\varepsilon}N^{2-\nu}\log(N))$	$O(\frac{1}{\varepsilon}N^{2-\nu}\log(N))$

TABLE 3
Validation of Assumption 1 on the DOTmark dataset, $L = 6$

Averaged $\|z^*_{h_l}\|^2_{L^2}$						
	$l=1$ $h_l=\frac{1}{16}$	$l=2$ $h_l=\frac{1}{32}$	$l=3$ $h_l=\frac{1}{64}$	$l=4$ $h_l=\frac{1}{128}$	$l=5$ $h_l=\frac{1}{256}$	$l=6$ $h_l=\frac{1}{512}$
$p=1$	0.115	0.100	0.093	0.090	0.089	0.088
$p=2$	0.068	0.060	0.057	0.056	0.055	0.055
$p=\infty$	0.060	0.055	0.053	0.051	0.051	0.051

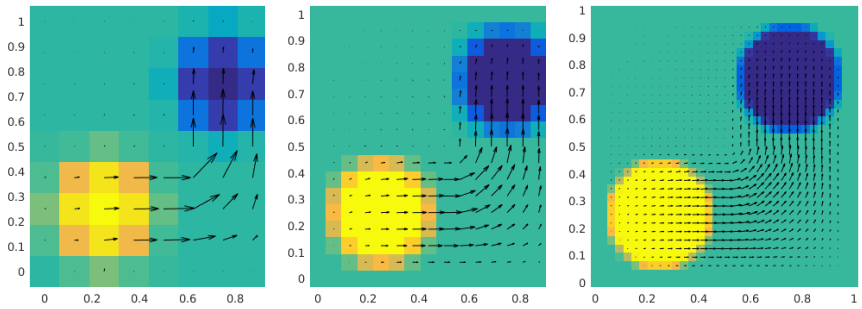
step size h_l. There may be multiple solutions in $Z^*_{h_l}$ on each level l, while Table 3 demonstrates that the solutions $z^*_{h_l}$ obtained by Algorithm 1M satisfy Assumption 1.

5.2. Validation of Assumption 2. Similar to the validation of Assumption 1 in Section 5.1, we use $\|\text{Interpolate}(z^K_{h_l}) - z^K_{h_{l+1}}\|^2_{L^2}$ get by Algorithm 1M to estimate $\|\text{Interpolate}(z^*_{h_l}) - z^*_{h_{l+1}}\|^2_{L^2}$. In this subsection, we also set $\varepsilon = 10^{-8}$.

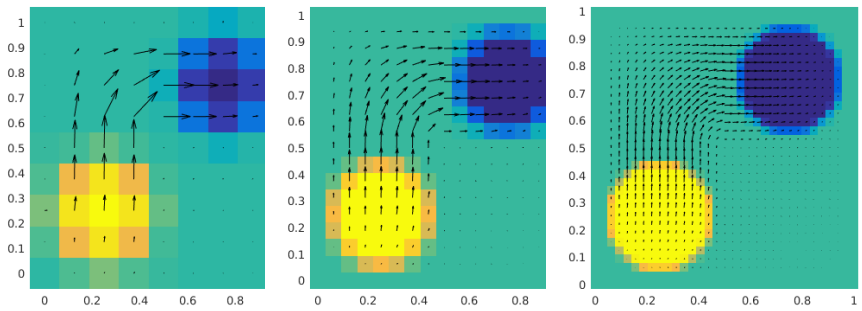
Visualization of $z^*_{h_l}$. The solution set $Z^*_{h_l}$ may have multiple solutions on each level. What we want to show in this paragraph is that, for each coarse level solution $z^*_{h_l} \in Z^*_{h_l}$, there is a finer level solution $z^*_{h_{l+1}} \in Z^*_{h_{l+1}}$ that is close to $z^*_{h_l}$. Here we set $p = 1$. Figures 3 and 4 illustrate this point. First, we the primal solution consider $m^*_{h_l}$. With different initializations, we obatin two different optimal $m^*_{h_l}$s on level $l = 1$: Figures 3(a) and 3(d). With the results in Figures 3(a) and 3(d) as initializations, we obtain the solutions $m^*_{h_l}$ on level 2: Figures 3(b) and 3(e). The flux in Figure 3(b) is close to that in Figure 3(a); Figure 3(e) is close to Figure 3(d). Thus, Assumption 2 is meaningful when there are multiple solutions on each level: for a solution $m^*_{h_l}$ on level l, there is a solution $m^*_{h_{l+1}}$ on level $l+1$ similar to $m^*_{h_l}$. Secondly, we consider the dual solution $\phi^*_{h_l}$. As $p = 1$, $\phi^*_{h_l}$ is unique upto a constant for each level l. The $\phi^*_{h_l}$ on level l is close to that on the finer level $\phi^*_{h_{l+1}}$. Figure 4 demonstrates this point.

Quantitative validation of $\|\text{Interpolate}(z^*_{h_l}) - z^*_{h_{l+1}}\|^2_{L^2}$. In Table 4, We report the averaged $\|\text{Interpolate}(z^*_{h_l}) - z^*_{h_{l+1}}\|^2_{L^2}$ on the "classic images" in DOTmark dataset [42] with different choices of $p \in \{1, 2, \infty\}$. By the results in Table 4, $\|\text{Interpolate}(z^*_{h_l}) - z^*_{h_{l+1}}\|^2_{L^2} \le O(h_l)^r$ is numerically satisfied and, approximately, $1 \le r \le 2$.

[5]The complexity of solving the minimum cost flow problem is the upper bound for the worst case. In practice, their algorithm has better performance than the theoretical bound. Numerical results are reported in Table 11.

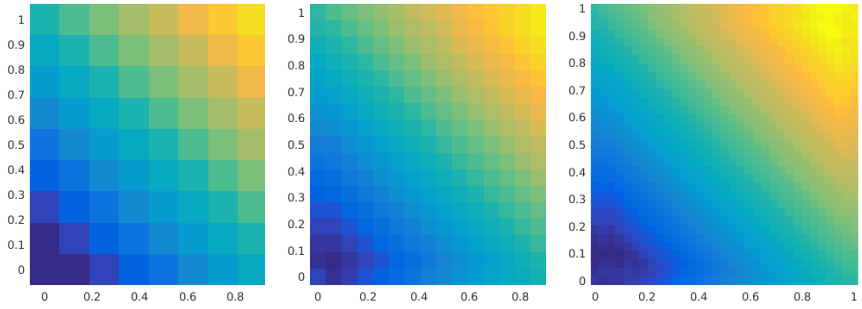


(a) One $m^*_{h_1}$, level 1, 8×8 grid (b) One $m^*_{h_2}$, level 2, 16×16 grid (c) One $m^*_{h_3}$, level 3, 32×32 grid



(d) Another $m^*_{h_1}$, level 1, 8×8 grid (e) Another $m^*_{h_2}$, level 2, 16×16 grid (f) Another $m^*_{h_3}$, level 3, 32×32 grid

FIGURE 3. *Visualization of Assumption 2: the black flux represents* $m_{h_l} : \Omega^{h_l} \to \Re^2$*, the two circles represent* $\rho^0_{h_l}, \rho^1_{h_l}$ *respectively. There are multiple solutions on each level. For every solution* $m^*_{h_l}$ *on level* l*, there is a solution* $m^*_{h_{l+1}}$ *on the finer level* $l+1$ *that is similar to* $m^*_{h_l}$*.*



(a) $\phi^*_{h_1}$, level 1, 8×8 grid (b) $\phi^*_{h_2}$, level 2, 16×16 grid (c) $\phi^*_{h_3}$, level 3, 32×32 grid

FIGURE 4. *Visualization of Assumption 2: dual solution (Kantorovich potential)* $\phi^*_{h_l} : \Omega^{h_l} \to \Re$ *on each level. The dual solution* $\phi^*_{h_{l+1}}$ *on level* $l+1$ *is close to* $\phi^*_{h_l}$ *on the coarser level* l*.*

TABLE 4
Validation of Assumption 2 on the DOTmark dataset, $L = 6$

Averaged $\|\text{Interpolate}(z^*_{h_l}) - z^*_{h_{l+1}}\|^2_{L^2}$					
	$l=1$ $h_l = 1/16$	$l=2$ $h_l = 1/32$	$l=3$ $h_l = 1/64$	$l=4$ $h_l = 1/128$	$l=5$ $h_l = 1/256$
$p=1$	1.59×10^{-3}	5.45×10^{-4}	1.55×10^{-4}	2.43×10^{-5}	2.91×10^{-6}
$p=2$	5.17×10^{-4}	1.53×10^{-4}	4.57×10^{-5}	1.16×10^{-5}	2.77×10^{-6}
$p=\infty$	8.40×10^{-4}	3.06×10^{-4}	6.65×10^{-5}	2.08×10^{-5}	6.41×10^{-6}

5.3. Validation of Assumption 3. Similar to the validation of Assumption 1, we use $\|y^K_{h_l}\|^2_{L^2}$ in Algorithm 2M to estimate $\|y^*_{h_l}\|^2_{L^2}$. In this subsection, we set $\varepsilon = 10^{-8}$.

Table 5 reports the averaged quantity of $\|y^*_{h_l}\|^2_{L^2}$ on the DOTmark dataset [42]. The results show that $\|y^*_{h_l}\|^2_{L^2}$ is clearly bounded by a constant independent of grid step size h_l.

TABLE 5
Validation of Assumption 3 on the DOTmark dataset, $L = 6$

Averaged $\|y^*_{h_l}\|^2_{L^2}$						
	$l=1$ $h_l = \frac{1}{16}$	$l=2$ $h_l = \frac{1}{32}$	$l=3$ $h_l = \frac{1}{64}$	$l=4$ $h_l = \frac{1}{128}$	$l=5$ $h_l = \frac{1}{256}$	$l=6$ $h_l = \frac{1}{512}$
$p=1$	2.072	2.018	1.984	1.972	1.967	1.965
$p=2$	1.044	1.016	1.003	0.997	0.995	0.993
$p=\infty$	0.984	0.966	0.960	0.963	0.968	0.972

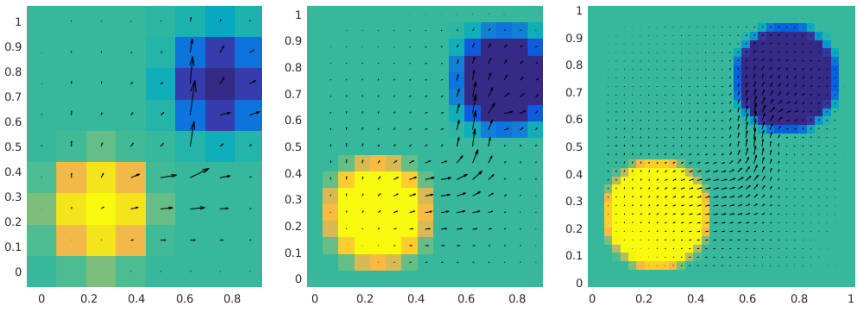
5.4. Validation of Assumption 4. Similar to the validation of Assumption 3 in Section 5.3, we use $\|\text{Interpolate}(y^K_{h_l}) - y^K_{h_{l+1}}\|^2_{L^2}$ in Algorithm 2M to estimate $\|\text{Interpolate}(y^*_{h_l}) - y^*_{h_{l+1}}\|^2_{L^2}$. In this subsection, we also set $\varepsilon = 10^{-8}$.

Visualization of $y^*_{h_l}$. Similar to the validation of Assumption 2, we set $p = 1$ and get the results in Figures 5 and 6. Figure 5 shows that: for a solution $m^*_{h_l}$ on level l, there is a solution $m^*_{h_{l+1}}$ on level $l+1$, which is similar to $m^*_{h_l}$. On this specific numerical example, the dual variable $\varphi^*_{h_l}$ is unique for each level l. Figure 6 demonstrates that $\varphi^*_{h_l}$ on level l is close to $\varphi^*_{h_{l+1}}$ on level $l+1$.

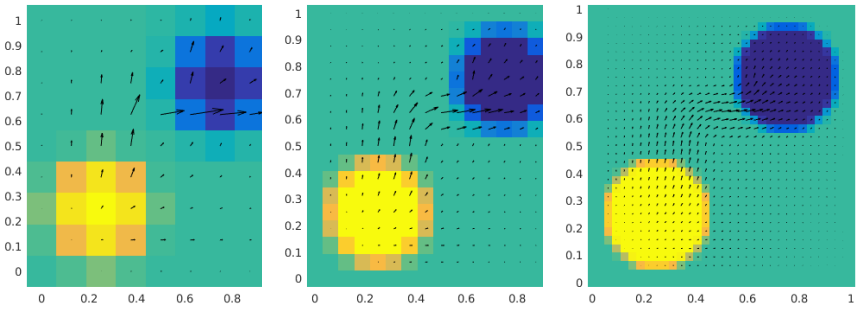
Quantitative validation of $\|\text{Interpolate}(y^*_{h_l}) - y^*_{h_{l+1}}\|^2_{L^2}$. In Table 6, We report the averaged $\|\text{Interpolate}(y^*_{h_l}) - y^*_{h_{l+1}}\|^2_{L^2}$ on the "classic images" in DOTmark dataset [42] with different choices of $p \in \{1, 2, \infty\}$. By the results in Table 6, $\|\text{Interpolate}(y^*_{h_l}) - y^*_{h_{l+1}}\|^2_{L^2} \le O(h_l)^\nu$ is numerically satisfied and, approximately, $\nu \approx 1$.

6. Numerical results. In this section, we numerically study why and how much our Algorithms 1M and 2M speed up Algorithms 1 and 2. The conclusions in Theorems 4.1 and 4.2 are validated. Moreover, we compare our algorithms with other EMD solvers [30, 2, 29, 28]. We implemented Algorithms 1M and 2M as $d = 2$ in MATLAB. All the experiments were conducted on a single CPU (Intel i7-2600 CPU @ 3.40GHz).

6.1. The effect of multilevel initialization. In this subsection, we study why multilevel initialization helps speed up Algorithms 1 and 2. All the results are obtained on the "cat" example which is also used as a benchmark in [29, 28].

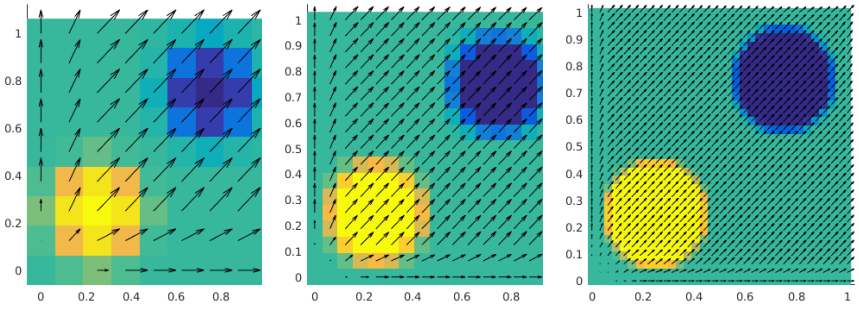


(a) One $m^*_{h_1}$, level 1, 8×8 grid (b) One $m^*_{h_2}$, level 2, 16×16 grid (c) One $m^*_{h_3}$, level 3, 32×32 grid



(d) Another $m^*_{h_1}$, level 1, 8×8 grid (e) Another $m^*_{h_2}$, level 2, 16×16 grid (f) Another $m^*_{h_3}$, level 3, 32×32 grid

FIGURE 5. *Visualization of Assumption 4: the black flux represents* $m_{h_l} : \Omega^{h_l} \to \Re^2$*, the the two circles represent* $\rho^0_{h_l}, \rho^1_{h_l}$ *respectively. There are multiple solutions on each level. For every solution* $m^*_{h_l}$ *on level* l*, there is a solution* $m^*_{h_{l+1}}$ *on level* $l+1$ *that is close to* $m^*_{h_l}$*.*



(a) $\varphi^*_{h_1}$, level 1, 8×8 grid (b) $\varphi^*_{h_2}$, level 2, 16×16 grid (c) $\varphi^*_{h_3}$, level 3, 32×32 grid

FIGURE 6. *Visualization of Assumption 4: dual solution* $\varphi^*_{h_l} : \Omega^{h_l} \to \Re^2$ *on each level. The dual solution* $\varphi^*_{h_{l+1}}$ *on level* l *is close to* $\varphi^*_{h_l}$ *on level* $l+1$*.*

TABLE 6
Validation of Assumption 4 on DOTmark, $L = 6$

Averaged $\|\text{Interpolate}(y^*_{h_l}) - y^*_{h_{l+1}}\|^2_{L^2}$					
	$l = 1$	$l = 2$	$l = 3$	$l = 4$	$l = 5$
	$h_l = 1/16$	$h_l = 1/32$	$h_l = 1/64$	$h_l = 1/128$	$h_l = 1/256$
$p = 1$	2.53×10^{-1}	1.32×10^{-1}	4.38×10^{-2}	1.02×10^{-2}	3.36×10^{-3}
$p = 2$	6.10×10^{-2}	2.91×10^{-2}	1.30×10^{-2}	4.67×10^{-3}	1.13×10^{-3}
$p = \infty$	8.49×10^{-2}	4.71×10^{-2}	2.14×10^{-2}	8.86×10^{-3}	3.45×10^{-3}

Algorithm 1M. We report the results of Algorithms 1 and 1M in Tables 7 and 8. If $L = 1$, Algorithm 1M reduces to Algorithm 1, which takes 5264 iterations to stop as Table 7 shows. If $L = 2$, we first conduct 2790 iterations on a coarse grid 256×256. The obtained result is used to initialize the algorithm on the fine grid 512×512. With this initialization, Algorithm 1M only takes 49 iterations to stop on the fine grid. Although extra calculations on 256×256 grid are required, the merit of fewer iterations on the fine grid overcomes the extra calculation cost. Thus, the total calculation time is reduced from 95.25 seconds to 10.64 seconds. When the number of levels L get even larger, the computing time could be further reduced. As $L = 4$, the computing time is reduced to 2.924 seconds. The results support Theorem 4.1: as ε small enough and L large enough, the calculation on the finest level 512×512 is dominant. Since the multilevel algorithm is able to dramatically reduce the calculation expense on the finest level, it consumes much less computing time.

TABLE 7
The effect of level number L in Algorithm 1M: 512×512, $p = 1$, tolerance $\varepsilon = 10^{-6}$. "Iters" is the number of iterations, and "Time" is in second. The results support Theorem 4.1.

Number	Calculation cost on each level								Total
	64×64		128×128		256×256		512×512		
of levels	Iters	Time	Iters	Time	Iters	Time	Iters	Time	time
L=1							5264	95.25	95.25
L=2					2790	9.691	49	0.953	10.64
L=3			2906	3.138	138	0.491	43	0.841	4.470
L=4	2454	1.153	291	0.340	148	0.537	44	0.894	2.924

In practice, $L = 6$ is usually large enough to enjoy the advantage of multilevel initialization. In Table 8, we fix $L = 6$ and compare the effect of multilevel initialization on different problems sizes. The results illustrates that Algorithm 1M is much faster than Algorithm 1, the advantage is significant on large-scale problems.

Algorithm 2M. We report the results of Algorithms 2 and 2M in Tables 9 and 10. Table 9 shows that multilevel initialization could speed up Algorithm 2. With the multilevel initialization, the number of iterations on the finest grid 512×512 can be reduced from 100 to 2. This result validates the conclusions in Theorem 4.2. In Table 10, we compare the effect of the multilevel initialization on different grid sizes. The results illustrate that Algorithm 2M speeds up Algorithm 2 by $2 \sim 20$ times. For large-scale problems, the speedup effect is considerable.

Visualization of solutions. We visualize the solutions obtained by Algorithms 1M and 2M in Appendix C.

TABLE 8

Comparison of Algorithm 1 and Algorithm 1M ($L = 6$, $\varepsilon = 10^{-6}$). The term "211+337+328+346+334+261" means the algorithm takes 211, 337, 328, 346, 334, 261 iterations for level $l = 1, 2, 3, 4, 5, 6$ respectively. The term "Speedup" measures how many times Algorithm 1M speeds up Algorithm 1.

Grid size	Algorithm 1		Algorithm 1M ($L = 6$)		Speedup
	Iters	Time	Iters	Time	
$p = 1$					
128×128	1743	1.867	211+337+328+346+334+261	0.621	3.0
256×256	3034	10.63	657+482+501+383+287+110	1.110	9.6
512×512	5264	95.25	1661+479+351+268+153+49	2.228	42.7
1024×1024	8843	954.9	2889+380+310+182+58+14	4.838	197.4
$p = 2$					
128×128	1102	2.348	416+400+330+311+235+178	0.734	3.2
256×256	1960	13.06	901+615+361+282+239+118	1.761	7.4
512×512	3319	107.9	1617+520+328+268+150+60	4.032	26.8
1024×1024	5892	985.2	2856+417+314+189+71+15	8.100	121.6
$p = \infty$					
128×128	1311	3.171	356+472+288+336+440+436	1.845	1.7
256×256	2331	18.29	558+496+312+396+439+468	5.354	3.4
512×512	4068	138.4	1041+461+467+461+504+186	12.26	11.3
1024×1024	7150	1165	2147+466+524+514+275+65	26.96	43.2

TABLE 9

The effect of level number L in Algorithm 2M: 512×512, $p = 1$ tolerance $\varepsilon = 10^{-5}$. "Iters" is the number of iterations, and "time" is in second. The results support Theorem 4.2.

Number of levels	Calculation cost on each level								Total time
	64×64		128×128		256×256		512×512		
	Iters	Time	Iters	Time	Iters	Time	Iters	Time	
L=1							100	2.375	2.375
L=2					99	1.224	2	0.113	1.336
L=3			99	0.157	4	0.080	2	0.112	0.349
L=4	100	0.024	8	0.016	4	0.089	2	0.111	0.240

6.2. Comparison with other methods. In this subsection, we compare our method with other EMD algorithms [30, 2, 29, 28]. There are some other 2D EMD solvers [36, 31, 43, 6, 14, 24] we do not compare with. [36] solves EMD with a thresholded metric; [31, 24] are designed for Wasserstein-p ($p > 1$) distance; [14, 43, 6] solve EMD with the entropy regularizer, the objective function of which is not the same with us. Thus, we are not able to compare these algorithms with ours fairly in our settings.

All the results are obtained on on the DOTmark [42] dataset and reported in Table 11. We used 10 images provided in the "classic images" of DOTmark. Totally we calculated 45 Wasserstein distances for all the 45 image pairs. The time consumptions are averages taken on these image pairs. Figure 9 in Appendix D visualizes two such images and the optimal transport between them. Tree-EMD [30] and Min-cost flow [2] are exact algorithms stopping within finite steps. Other algorithms are iterative algorithms stopping by a tolerance. For Algorithms 1 and 1M, we take

TABLE 10

Comparison of Algorithm 2 and Algorithm 2M ($L = 6$). "77+69+69+24+16+11" means the algorithm takes 77,69,69,24,16,11 iterations for level $l = 1, 2, 3, 4, 5, 6$ respectively. The term "Speedup" measures how many times Algorithm 2M speeds up Algorithm 2.

Grid size	Algorithm 2		Algorithm 2M ($L = 6$)		Speedup
	Iterations	Time	Iterations	Time	
$p = 1$					
128×128	99	0.148	77+69+69+24+16+11	0.037	4.0
256×256	99	1.217	93+74+22+18+11+5	0.135	9.0
512×512	100	2.375	106+26+20+9+4+2	0.233	10.2
1024×1024	99	11.47	93+22+9+5+2+1	0.634	18.1
$p = 2$					
128×128	51	0.085	55+60+40+24+15+9	0.035	2.4
256×256	50	0.675	58+31+23+15+9+4	0.116	5.8
512×512	50	1.368	53+22+16+9+4+2	0.252	5.4
1024×1024	50	6.530	53+20+9+5+2+1	0.792	8.2
$p = \infty$					
128×128	53	0.088	49+68+53+30+15+10	0.035	2.5
256×256	54	0.714	59+41+27+14+10+6	0.147	4.9
512×512	55	1.460	67+30+14+10+6+5	0.354	4.1
1024×1024	52	6.651	65+18+11+7+6+2	1.070	6.2

$\varepsilon = 10^{-6}$; for Algorithms 2 and 2M, we take $\varepsilon = 10^{-5}$. Practically, these fixed-point-residual tolerances are small enough to guarantee the relative error of the distance value no larger than 5%.

In most of the cases, Algorithm 2M is the best. Algorithm 1M and Algorithm 2 are also competitive for large-size problems. All the first-order methods (Algorithms 1, 2, 1M, 2M) are robust to the parameter p in the ground metric $\|\cdot\|_p$. Tree-EMD [30] only works for $p = 1$; the algorithm in [2] works well when $p = 1, \infty$ and the grid size is not very large. As $p = 2$, the algorithm in [2] requires large amount of memory and calculation time.

7. Conclusion. In this paper, we have proposed two multilevel algorithms for the computation of the Wasserstein-1 metric. The algorithms leverage the L_1 type primal-dual structure in minimal flux formulation of optimal transport. The multilevel setting provides very good initializations for the minimization problems on the fine grids. So it can significantly reduce the number of iterations on the finest grid. This consideration allows us to compute the metric between two 1024×1024 images in about one second on a single CPU. It is worth mentioning that the proposed algorithm also provides the Kantorovich potential and the optimal flux function between two densities. They are useful for the related Wasserstein variation problems [33].

In future work, we will apply the multilevel method to optimal transport related minimization in mean field games and machine learning.

TABLE 11
Time consumption (seconds) on DOTmark [42]. Tree-EMD and Min-cost flow stop with finite steps and give exact solutions, first-order algorithms (Algorithms 1, 2, 1M and 2M) are iterative and give estimated solutions (5% relative error)

Grid size	32×32	64×64	128×128	256×256	512×512
$p = 1$					
Tree-EMD [30]	0.006	0.127	2.433	121.2	N/A
Min-cost flow [2]	**0.002**	0.024	0.342	7.164	157.7
Algorithm 1 [29]	0.134	0.780	3.018	18.23	162.7
Algorithm 2 [28]	0.008	0.029	0.171	1.433	2.710
Algorithm 1M	0.117	0.265	0.537	0.962	1.877
Algorithm 2M	0.008	**0.013**	**0.030**	**0.104**	**0.201**
$p = 2$					
Tree-EMD [30]	N/A	N/A	N/A	N/A	N/A
Min-cost flow [2]	0.082	1.863	N/A	N/A	N/A
Algorithm 1 [29]	0.078	0.429	2.131	13.55	106.3
Algorithm 2 [28]	**0.007**	0.021	0.116	0.908	1.807
Algorithm 1M	0.066	0.149	0.335	0.838	1.996
Algorithm 2M	**0.007**	**0.015**	**0.046**	**0.148**	**0.285**
$p = \infty$					
Tree-EMD [30]	N/A	N/A	N/A	N/A	N/A
Min-cost flow [2]	**0.002**	0.025	0.300	5.380	118.1
Algorithm 1 [29]	0.270	1.202	5.487	29.28	197.9
Algorithm 2 [28]	0.006	0.023	0.130	1.026	2.034
Algorithm 1M	0.255	0.511	1.124	2.471	5.124
Algorithm 2M	0.007	**0.015**	**0.051**	**0.217**	**0.411**

Appendix A. A lemma for interpolation operators.

LEMMA A.1. *If $h_{l-1} = 2h_l$, then we have*

$$\begin{aligned} \|\text{Interpolate}(\phi_{h_{l-1}})\|_{L^2}^2 \leq & \|\phi_{h_{l-1}}\|_{L^2}^2, \quad \forall \phi_{h_{l-1}} : \Omega^{h_{l-1}} \to \Re \\ \|\text{Interpolate}(m_{h_{l-1}})\|_{L^2}^2 \leq & \|m_{h_{l-1}}\|_{L^2}^2, \quad \forall m_{h_{l-1}} : \Omega^{h_{l-1}} \to \Re^d \\ \|\text{Interpolate}(\varphi_{h_{l-1}})\|_{L^2}^2 \leq & \|\varphi_{h_{l-1}}\|_{L^2}^2, \quad \forall \varphi_{h_{l-1}} : \Omega^{h_{l-1}} \to \Re^d \end{aligned} \tag{A.1}$$

Proof. First, we consider the interpolation of potential $\phi_{h_{l-1}}$. With $\phi_{h_l} = \text{Interpolate}\ (\phi_{h_{l-1}})$, we have

$$\begin{aligned} \|\phi_{h_l}\|_{L^2}^2 &= \sum_{x \in \Omega^{h_l}} \phi_{h_l}^2(x)(h_l)^d \\ &= \sum_{x \in \Omega^{h_l}} \left(\frac{1}{2^{|\bar{J}|}} \sum_{|y_{j_1} - x_{j_1}| \leq h_l} \cdots \sum_{|y_{j_{|\bar{J}|}} - x_{j_{|\bar{J}|}}| \leq h_l} \phi_{h_{l-1}}(y_J, y_{j_1}, y_{j_2}, \cdots, y_{j_{|\bar{J}|}}) \right)^2 (h_l)^d \\ &\leq \sum_{x \in \Omega^{h_l}} \left(\frac{1}{2^{|\bar{J}|}} \sum_{|y_{j_1} - x_{j_1}| \leq h_l} \cdots \sum_{|y_{j_{|\bar{J}|}} - x_{j_{|\bar{J}|}}| \leq h_l} \phi_{h_{l-1}}^2(y_J, y_{j_1}, y_{j_2}, \cdots, y_{j_{|\bar{J}|}}) \right) (h_l)^d \\ &= \sum_{y \in \Omega^{h_{l-1}}} c(y) \phi_{h_{l-1}}^2(y)(h_l)^d. \end{aligned}$$

The inequality in the third line above follows from Jensen's Inequality [8], and $c(y)$ is a constant which can be bounded in the following way.

$\phi_{h_{l-1}}(y)$ contributes to the nodes within a d dimensional hypercube $H(y) = \{x \in \Omega : \max_j |x_j - y_j| \le h_l\}$. First, we consider y as an interior point in Ω. There are 2^d vertices in $H(y)$, for each vertice, the weight is $1/(2^d)$. There are $2^{d-1}d$ edges in $H(y)$, each edge contains a single point $x \in \Omega^{h_l}$, $\phi_{h_{l-1}}(y)$ contributes to $\phi_{h_l}(x)$ with weight $1/(2^{d-1})$. Generally speaking, there are $2^{d-n}\binom{d}{n}$ n-dimensional hypercubes on the boundary of $H(y)$ [13], each m-dimensional hypercube contains a single point $x \in \Omega^{h_l}$, $\phi_{h_{l-1}}(y)$ contributes to $\phi_{h_l}(x)$ with weight $1/(2^{d-n})$. Moreover, the center of $H(y)$ is y, which is also in Ω^{h_l}, $\phi_{h_{l-1}}(y)$ contributes to $\phi_{h_l}(y)$ with weight 1. Thus, for interior point y, we have

$$\begin{aligned} c(y) =& \frac{1}{2^d} \times 2^d + \frac{1}{2^{d-1}} \times d2^{d-1} + \cdots + \frac{1}{2^{d-1}} \times 2^{d-1}\binom{d}{n} + \cdots + 1 \times 1 \\ =& \sum_{n=0}^{d} \binom{d}{n} = (1+1)^d = 2^d. \end{aligned}$$

For y on the boundary of Ω, there are less points in $H(y) \cap \Omega^{h_l}$, and the weight for each node is the same as above, thus, $c(y) < 2^d$. In one word, $c(y) \le 2^d$ for all $y \in \Omega^{h_{l-1}}$. Then, we obtain

$$\begin{aligned} \|\phi_{h_l}\|_{L^2}^2 \le & \sum_{y \in \Omega^{h_{l-1}}} c(y)\phi_{h_{l-1}}^2(y)(h_l)^d \le \sum_{y \in \Omega^{h_{l-1}}} \phi_{h_{l-1}}^2(y)(2h_l)^d \\ = & \sum_{y \in \Omega^{h_{l-1}}} \phi_{h_{l-1}}^2(y)(h_{l-1})^d = \|\phi_{h_{l-1}}\|_{L^2}^2. \end{aligned}$$

With the same proof line, the interpolation of $m_{h_{l-1}}$ and $\varphi_{h_{l-1}}$ can also be proved. Inequalities (A.1) are proved. ∎

Appendix B. Kantorovich potential.

The Kantorovich potential can be obtained directly from the dual solution of (3.1). Thus, Algorithm 1 directly gives the potential $\phi_h^* : \Omega^h \to \Re$. While Algorithm 2 solves (3.4) and gives $\varphi_h^* : \Omega^h \to \Re^d$, the gradient of ϕ_h^*: $\varphi_h^* = A_h^* \phi_h^*$. We obtain ϕ_h^* given φ_h^* by solving

$$A_h A_h^* \phi_h = A_h \varphi_h^*.$$

The boundary condition is given. Thus, the Laplacian operator $A_h A_h^*$ is invertible, where the solution is unique up to a constant shrift. And ϕ_h^* is given by

$$\phi_h^* = (A_h A_h^*)^{-1} A_h \varphi_h^*. \tag{B.1}$$

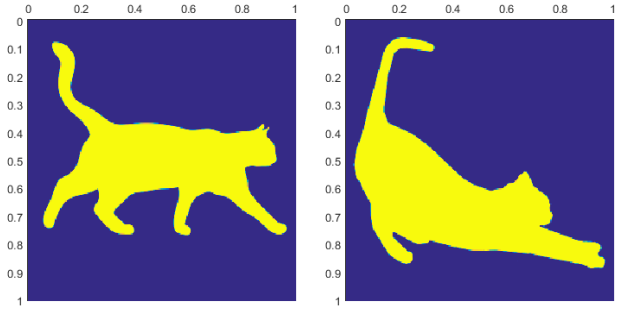
The invert Laplacian operator can be calculated efficiently by the FFT [28].

Appendix C. Visualization of the cat example.

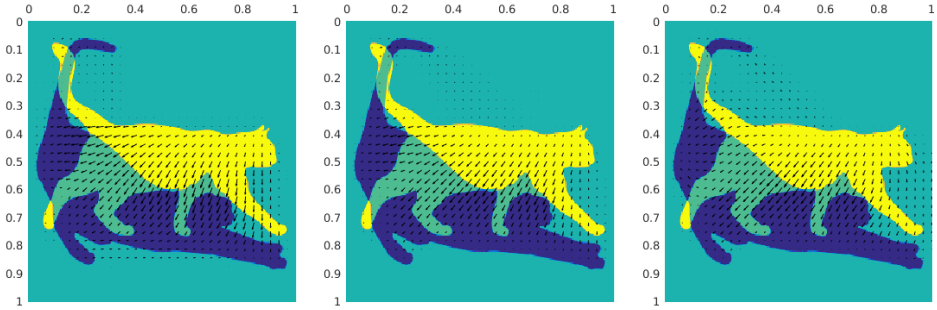
The cat example is used in Section 6.1. We visualize the two distributions ρ^0, ρ^1 and the optimal transport between them in this section. Figure 7 visualizes the primal-dual pair (m^*, ϕ^*) obtained by Algorithm 1M, Figure 8 visualizes the primal-dual pair (m^*, φ^*) obtained by Algorithm 2M. For both the two algorithms, we take $h = 1/256$, $L = 6$ and $\varepsilon = 10^{-6}$.

Appendix D. Visualization of DOTmark.

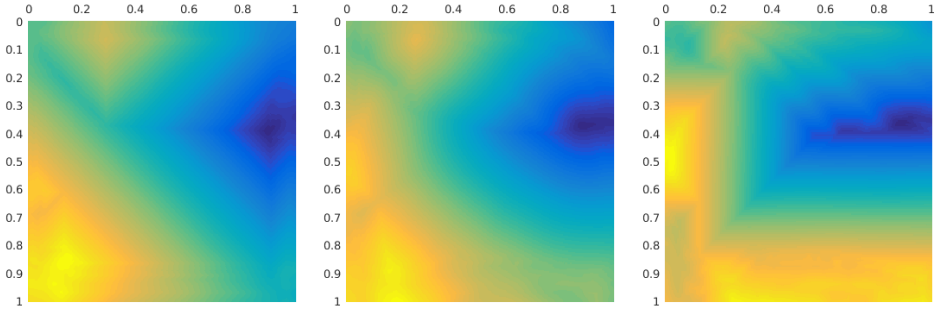
The DOTmark dataset is used in Section 6.2. We visualize two of the distributions ρ^0, ρ^1 and the optimal transport between them in Figure 9.



(a) ρ^0 (b) ρ^1



(c) $m^*(p=1)$ (d) $m^*(p=2)$ (e) $m^*(p=\infty)$



(f) $\phi^*(p=1)$ (g) $\phi^*(p=2)$ (h) $\phi^*(p=\infty)$

FIGURE 7. *Visualization of* (m^*, ϕ^*) *obtained by Algorithm 1M.* m^* *is the optimal flux;* ϕ^* *is the Kantorovich potential. The backgrounds of Fig. 7(c), 7(d) and 7(e) are all* $\rho^0 - \rho^1$.

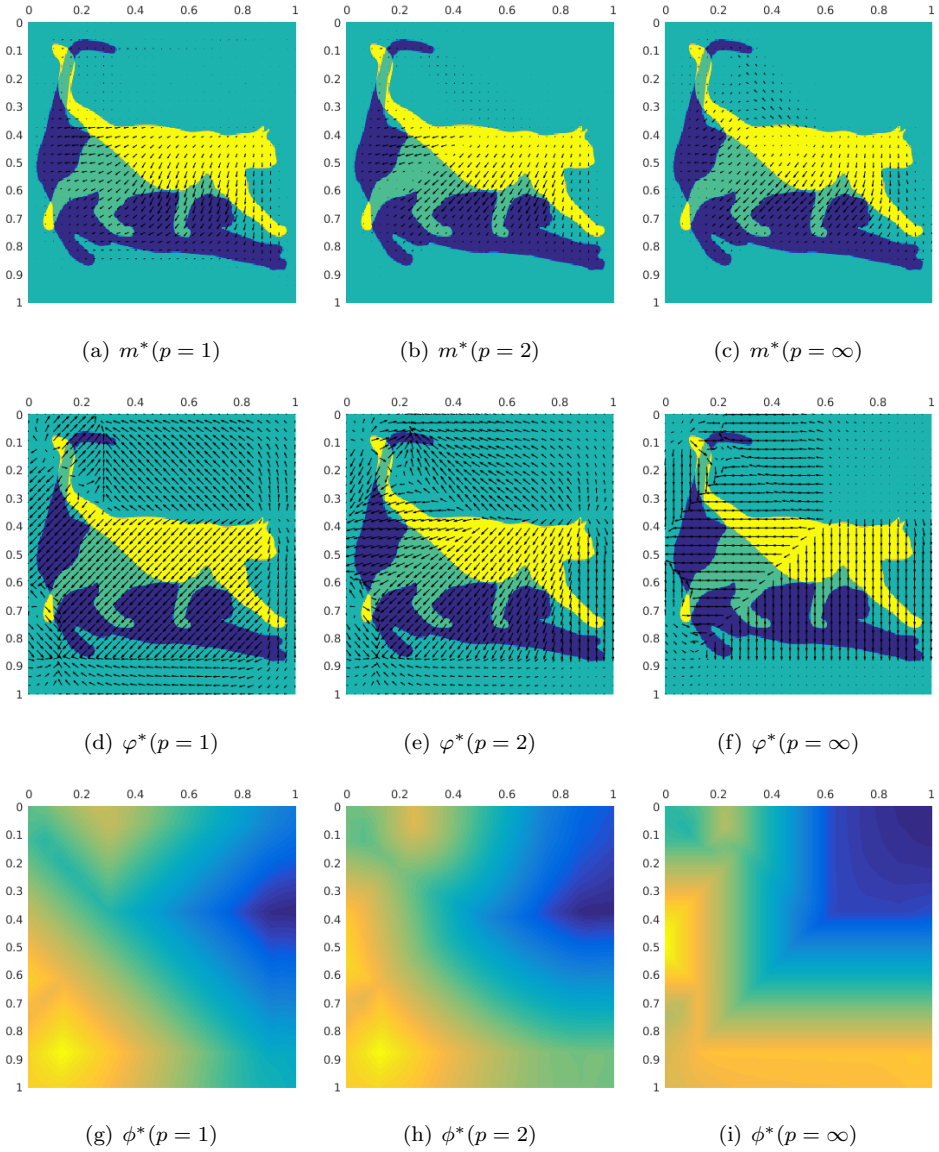
REFERENCES

[1] M. ARJOVSKY, S. CHINTALA, AND L. BOTTOU, *Wasserstein generative adversarial networks*, in Proceedings of the 34th International Conference on Machine Learning (ICML), 2017.

[2] F. BASSETTI, S. GUALANDI, AND M. VENERONI, *On the computation of kantorovich-wasserstein distances between 2d-histograms by uncapacitated minimum cost flows*, arXiv preprint arXiv:1804.00445, (2018).

[3] M. BECKMANN, *A continuous model of transportation*, Econometrica: Journal of the Econometric Society, (1952), pp. 643–660.

[4] J.-D. BENAMOU AND Y. BRENIER, *A computational fluid mechanics solution to the monge-*



(a) $m^*(p=1)$ (b) $m^*(p=2)$ (c) $m^*(p=\infty)$

(d) $\varphi^*(p=1)$ (e) $\varphi^*(p=2)$ (f) $\varphi^*(p=\infty)$

(g) $\phi^*(p=1)$ (h) $\phi^*(p=2)$ (i) $\phi^*(p=\infty)$

FIGURE 8. *Visualization of* (m^*, φ^*) *and the potential* ϕ^* *obtained by Algorithm 2M.* ρ^0, ρ^1 *are the same with those in Figure 7.* m^* *is the optimal flux;* ϕ^* *is the Kantorovich potential, that is obtained from* φ^* *by the method in Appendix B.*

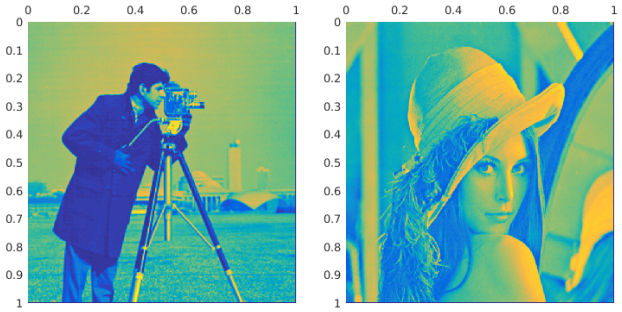
kantorovich mass transfer problem, Numerische Mathematik, 84 (2000), pp. 375–393.

[5] J.-D. BENAMOU AND G. CARLIER, *Augmented Lagrangian Methods for Transport Optimization, Mean Field Games and Degenerate Elliptic Equations*, Journal of Optimization Theory and Applications, 167 (2015), pp. 1–26.

[6] J.-D. BENAMOU, G. CARLIER, M. CUTURI, L. NENNA, AND G. PEYRÉ, *Iterative bregman projections for regularized transportation problems*, SIAM Journal on Scientific Computing, 37 (2015), pp. A1111–A1138.

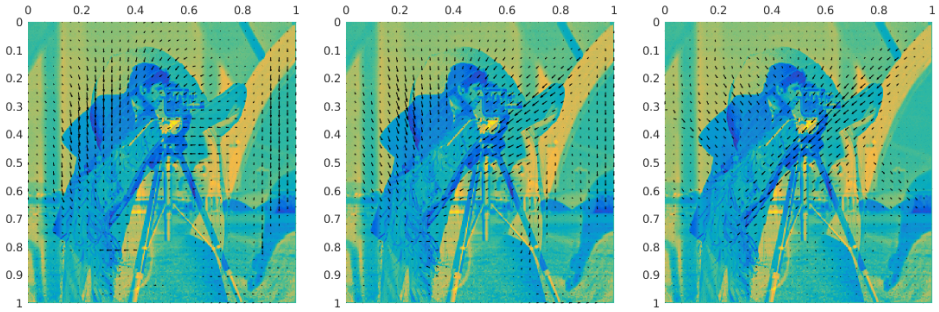
[7] F. A. BORNEMANN AND P. DEUFLHARD, *The cascadic multigrid method for elliptic problems*, Numerische Mathematik, 75 (1996), pp. 135–152.

[8] S. BOYD AND L. VANDENBERGHE, *Convex optimization*, Cambridge university press, 2004.



(a) ρ^0: Cameraman (512 × 512), one image of DOTmark

(b) ρ^1: Lena (512×512), one image of DOTmark



(c) $p = 1$ (d) $p = 2$ (e) $p = \infty$

FIGURE 9. *Visualization of the optimal transport m^* between the two images ρ^0 and ρ^1. The background is the difference between ρ^0, ρ^1: $\rho = \rho^0 - \rho^1$.*

[9] G. CARLIER, V. DUVAL, G. PEYRÉ, AND B. SCHMITZER, *Convergence of entropic schemes for optimal transport and gradient flows*, SIAM Journal on Mathematical Analysis, 49 (2017), pp. 1385–1418.

[10] A. CHAMBOLLE AND T. POCK, *A first-order primal-dual algorithm for convex problems with applications to imaging*, Journal of mathematical imaging and vision, 40 (2011), pp. 120–145.

[11] Y. CHEN, T. T. GEORGIOU, AND M. PAVON, *On the relation between optimal transport and schrödinger bridges: A stochastic control viewpoint*, Journal of Optimization Theory and Applications, 169 (2016), pp. 671–691.

[12] Y. CHEN, T. T. GEORGIOU, AND M. PAVON, *Optimal transport over a linear dynamical system*, IEEE Transactions on Automatic Control, 62 (2017), pp. 2137–2152.

[13] H. S. M. COXETER, *Regular polytopes*, Courier Corporation, 1973.

[14] M. CUTURI, *Sinkhorn distances: Lightspeed computation of optimal transport*, in Advances in neural information processing systems (NIPS), 2013, pp. 2292–2300.

[15] M. CUTURI AND G. PEYRÉ, *A smoothed dual approach for variational wasserstein problems*, SIAM Journal on Imaging Sciences, 9 (2016), pp. 320–343.

[16] J. DUCHI, S. SHALEV-SHWARTZ, Y. SINGER, AND T. CHANDRA, *Efficient projections onto the l 1-ball for learning in high dimensions*, in Proceedings of the 25th International Conference on Machine Learning (ICML), ACM, 2008, pp. 272–279.

[17] P. DUHAMEL AND M. VETTERLI, *Fast fourier transforms: a tutorial review and a state of the art*, Signal processing, 19 (1990), pp. 259–299.

[18] M. ESSID AND J. SOLOMON, *Quadratically regularized optimal transport on graphs*, SIAM Journal on Scientific Computing, 40 (2018), pp. A1961–A1986.

[19] S. FERRADANS, N. PAPADAKIS, G. PEYRÉ, AND J.-F. AUJOL, *Regularized discrete optimal*

transport, SIAM Journal on Imaging Sciences, 7 (2014), pp. 1853–1882.
[20] C. Frogner, C. Zhang, H. Mobahi, M. Araya, and T. A. Poggio, *Learning with a Wasserstein Loss*, in Advances in Neural Information Processing Systems 28, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, eds., Curran Associates, Inc., 2015, pp. 2053–2061.
[21] W. Gangbo and R. J. McCann, *The geometry of optimal transportation*, Acta Mathematica, 177 (1996), pp. 113–161.
[22] G. H. Golub and C. F. Van Loan, *Matrix computations*, vol. 3, JHU Press, 2012.
[23] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, *Improved training of wasserstein gans*, in Advances in Neural Information Processing Systems (NIPS), 2017, pp. 5767–5777.
[24] E. Haber and R. Horesh, *A multilevel method for the solution of time dependent optimal transport*, Numerical Mathematics: Theory, Methods and Applications, 8 (2015), p. 97–111, https://doi.org/10.4208/nmtma.2015.w02si.
[25] V. Hartmann and D. Schuhmacher, *Semi-discrete optimal transport-the case p= 1*, arXiv preprint arXiv:1706.07650, (2017).
[26] B. He and X. Yuan, *Convergence analysis of primal-dual algorithms for a saddle-point problem: From contraction perspective*, SIAM J. Imaging Sciences, 5 (2012), pp. 119–149.
[27] F. S. Hillier and G. J. Lieberman, *Introduction to mathematical programming*, vol. 2, McGraw-Hill New York, 1995.
[28] M. Jacobs, F. Léger, W. Li, and S. Osher, *Solving large-scale optimization problems with a convergence rate independent of grid size*, arXiv preprint arXiv:1805.09453, (2018).
[29] W. Li, E. K. Ryu, S. Osher, W. Yin, and W. Gangbo, *A parallel method for earth mover's distance*, Journal of Scientific Computing, 75 (2018), pp. 182–197.
[30] H. Ling and K. Okada, *An Efficient Earth Mover's Distance Algorithm for Robust Histogram Comparison*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 29 (2007), pp. 840–853.
[31] A. M. Oberman and Y. Ruan, *An efficient linear programming method for optimal transportation*, arXiv preprint arXiv:1509.03668, (2015).
[32] F. Otto and C. Villani, *Generalization of an inequality by talagrand and links with the logarithmic sobolev inequality*, Journal of Functional Analysis, 173 (2000), pp. 361–400.
[33] N. Papadakis, G. Peyré, and E. Oudet, *Optimal Transport with Proximal Splitting*, SIAM Journal on Imaging Sciences, 7 (2014), pp. 212–238.
[34] N. Parikh, S. Boyd, et al., *Proximal algorithms*, Foundations and Trends® in Optimization, 1 (2014), pp. 127–239.
[35] O. Pele and M. Werman, *A linear time histogram metric for improved sift matching*, in Proceedings of the 10th European Conference on Computer Vision (ECCV), Springer, 2008, pp. 495–508.
[36] O. Pele and M. Werman, *Fast and robust Earth Mover's Distances*, in 2009 IEEE 12th International Conference on Computer Vision (ICCV), 2009, pp. 460–467.
[37] H. Petzka, A. Fischer, and D. Lukovnicov, *On the regularization of wasserstein gans*, arXiv preprint arXiv:1709.08894, (2017).
[38] G. Peyré and M. Cuturi, *Computational Optimal Transport*, arXiv:1803.00567 [stat], (2018), https://arxiv.org/abs/1803.00567.
[39] Y. Rubner, C. Tomasi, and L. J. Guibas, *The earth mover's distance as a metric for image retrieval*, International journal of computer vision, 40 (2000), pp. 99–121.
[40] E. K. Ryu, Y. Chen, W. Li, and S. Osher, *Vector and Matrix Optimal Mass Transport: Theory, Algorithm, and Applications*, arXiv:1712.10279 [cs, math], (2017), https://arxiv.org/abs/1712.10279.
[41] E. K. Ryu, W. Li, P. Yin, and S. Osher, *Unbalanced and Partial L_1 Monge-Kantorovich Problem: a Scalable Parallel First-Order Method*, Journal of Scientific Computing, 75 (2018), pp. 1596–1613.
[42] J. Schrieber, D. Schuhmacher, and C. Gottschlich, *Dotmark–a benchmark for discrete optimal transport*, IEEE Access, 5 (2017), pp. 271–282.
[43] J. Solomon, F. De Goes, G. Peyré, M. Cuturi, A. Butscher, A. Nguyen, T. Du, and L. Guibas, *Convolutional wasserstein distances: Efficient optimal transportation on geometric domains*, ACM Transactions on Graphics (TOG), 34 (2015), p. 66.
[44] J. Solomon, R. Rustamov, L. Guibas, and A. Butscher, *Earth Mover's Distances on Discrete Surfaces*, ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2014, 33 (2014), pp. 1–12.
[45] C. Villani, *Optimal transport: old and new*, vol. 338, Springer Science & Business Media, 2008.

661 [46] P. Wesseling, *Introduction To Multigrid Methods*, R.T. Edwards, Inc., 2004.