



# Solving semi-discrete optimal transport problems: star shapedness and Newton's method

Luca Dieci<sup>1</sup> · Daniyar Omarov<sup>1</sup>

Received: 22 January 2024 / Accepted: 24 July 2024  
© Crown 2024

## Abstract

In this work, we propose a novel implementation of Newton's method for solving semi-discrete optimal transport (OT) problems for cost functions which are a positive combination of  $p$ -norms,  $1 < p < \infty$ . It is well understood that the solution of a semi-discrete OT problem is equivalent to finding a partition of a bounded region in Laguerre cells, and we prove that the Laguerre cells are star-shaped with respect to the target points. By exploiting the geometry of the Laguerre cells, we obtain an efficient and reliable implementation of Newton's method to find the sought network structure. We provide implementation details and extensive results in support of our technique in 2-d problems, as well as comparison with other approaches used in the literature.

**Keywords** Semi-discrete optimal transport · Laguerre tessellation · star-shaped · Newton's method

**Mathematics Subject Classification (2010)** 65D99 · 65K99 · 49M15

## 1 Introduction

In this work we provide an implementation of Newton's method to solve semi-discrete optimal transport (OT) problems, for several cost functions given by combination of  $p$ -norms ( $p \neq 1, \infty$ ), several source and target probabilities, and several different domains. Our main goals are: (i) to justify and exploit the geometry of the decomposition in Laguerre cells of the underlying solution of the problem, (ii) to provide robust algorithmic development, (iii) to give extensive numerical testing of our implementation on several examples, and (iv) to give quantitative measure of the goodness of our results, also in comparison with existing approaches.

---

✉ Daniyar Omarov  
domarov3@gatech.edu

Luca Dieci  
dieci@math.gatech.edu

<sup>1</sup> School of Mathematics, Georgia Institute of Technology, GA 30332 Atlanta, USA

Optimal transport problems have been receiving considerable attention from the analytical community for many years. The history of the field dates back 200+ years with the work of Monge in 1781, [34], though it was only in the 1940's with the works of Kantorovich, [26, 27], that the very fruitful relation to optimization, duality, and linear programming ideas made it possible to settle important theoretical questions as well as open the door to development of numerical methods. Because of this history, it is common to use the naming of Monge-Kantorovich (MK) problem for the classical OT problem. Numerical techniques for approximating the solution of OT problems have followed suite in the past 30-35 years, and new methods continue being developed and analyzed, in no small part due to the many applications (e.g., in imaging, economics, machine learning) where OT problems arise.

As we said, the scope of this work involves the study of numerical methods for semi-discrete OT problems with cost functions given by a positive combination of  $p$ -norms,  $c(x, y) = \sum_{l=1}^m \alpha_l \|x - y\|_{p_l}$ ,  $\alpha_l \geq 0$ ,  $p_l \in (1, \infty)$ . This specific case of cost function has received less attention in the literature compared to the Euclidean and the squared Euclidean cost functions, despite its numerous important applications. First, we can approach, from above/below, a solution for the ill-posed cases of the 1-norm (the absolute deviation problem) and the sup-norm (the max-deviation problem); see Example 5.3. Secondly, the case of a single  $p$ -norm corresponds to generalized hyperbola in Minkowski geometry (e.g. see [3, 23, 25]). Finally, a very significant application involves using a combination of  $p$ -norms to approximate an unknown cost function from data samples. This is particularly valuable, e.g. in city delivery services, when the cost function is unknown but it satisfies distance properties. For example, consider a set of data samples:  $\{y_i, x_j, c_{ij}\}$ ,  $i = 1, 2, \dots, K_1$ ,  $j = 1, 2, \dots, K_2$ , where  $c_{ij}$  represents the observed cost value between points  $y_i$  and  $x_j$ . Here, the points  $y_i$  can be regarded as distribution centers and the points  $x_j$  as delivery addresses. By using a model cost function,  $c(x, y) = \sum_{l=1}^m \alpha_l \|x - y\|_{p_l}$ , with positive coefficients  $\alpha_l$ 's, we can extract an approximation to the underlying (but unknown) cost function, e.g., in the least-squares sense. See Example 5.7.

Excellent expository books of both theory and numerics of OT problems have recently appeared and we refer to these as general introduction to the topic; in particular, see the book of Santambrogio, [38], and also the more computationally oriented monograph of Peyré and Cuturi, [35].

A fairly general formulation is the following. We are given two compactly supported measures  $\mu$  and  $\nu$  on  $\mathbb{R}^n$ ,  $n > 1$ , absolutely continuous with respect to Lebesgue measure, and with respective supports  $X$  and  $Y$ . Further, let  $c : X \times Y \rightarrow \mathbb{R}^+$  be a cost function, whereby  $c(x, y)$  describes the cost of transporting one unit of mass  $x$  to  $y$ . The original Monge problem is to find

$$\inf_{T: T_{\#}\mu=\nu} \int_X c(x, T(x)) d\mu(x),$$

where  $T_{\#}$  is the push-forward of  $\mu$  through  $T$ , that is  $(T_{\#}\mu)(A) = \mu(T^{-1}(A))$ . As stated, it is far from obvious that a map (Monge map)  $T$  can be found, and in fact the theoretical problem stayed open until the relaxation introduced by Kantorovich in the

1940's. After the work of Kantorovich, the MK problem was reformulated as finding a transport plan  $\pi$  of probability densities while minimizing a cost functional.

That is, the MK problem consists in determining the *optimal transport plan*, that is the joint pdf  $\pi \in \mathcal{M}(X \times Y)$ , with marginals  $\mu$  and  $\nu$ , which realizes the inf of the functional  $J(\pi)$  below:

$$\inf_{\pi} J := \int_{X \times Y} c(x, y) d\pi. \quad (1)$$

Under suitable assumptions on  $\mu, \nu, c$ , as well explained in the book of Santambrogio [38], thanks to works by Ambrosio, Benamou, Brenier, Gangbo, McCann, Villani, and others, (see [1, 4, 19, 42]), it was established that an optimal plan exists and can be chosen as a measure concentrated in  $\mathbb{R}^n \times \mathbb{R}^n$  on the graph of a map  $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$ , the optimal transport map. Oversimplifying the theory, a key point was to restrict to a strictly convex cost like  $c(x, y) = \|x - y\|_2^2$ . In this case, the map is obtained as gradient of a smooth potential  $\phi$  that satisfies the Monge-Ampere partial differential equation; numerical methods in this case have also been studied, e.g., see [5, 18] as well as our recent comparison [13] of different methods for solving the continuous OT problem with cost given by the 2-norm square.

However, here our own interest is when  $\nu$  is discrete, that is it is supported at finitely many distinct points  $y_i$ 's:  $\nu(y) = \nu_i \delta(y_i)$ ,  $i = 1, \dots, N$ ,  $\nu_i > 0$  and  $\sum_{i=1}^N \nu_i = 1$ . These are called *semi-discrete* optimal transport problems, and one has the restriction that the  $y_i$ 's are in  $X$ . In this case, a remarkable consequence of duality affords a major simplification of the structure of the optimal map, under fairly general conditions on the cost function. The dual formulation of the MK problem Eq. 1 is

$$\begin{aligned} \sup_{\phi, \psi} D &:= \int_X \phi d\mu + \int_Y \psi d\nu, \\ \text{s. t. } \phi(x) + \psi(y) &\leq c(x, y), \quad \forall (x, y) \in X \times Y, \end{aligned} \quad (2)$$

where  $\phi(x)$  and  $\psi(y)$  are absolutely integrable functions.

**Lemma 1.1** (Observation 4.3.2 in [31]) *Let  $\psi^c(x) = \inf_{y \in Y} [c(x, y) - \psi(y)]$ . If the pair  $(\phi, \psi)$  is admissible in Eq. (2), then the pair  $(\psi^c, \psi)$  is admissible as well.*

**Proof**

$$\psi^c(x) + \psi(y) = \inf_{y' \in Y} [c(x, y') - \psi(y')] + \psi(y) \leq c(x, y) - \psi(y) + \psi(y) = c(x, y).$$

□

**Lemma 1.2** (Observation 4.3.3 in [31]) *If the pair  $(\phi, \psi)$  is admissible for Eq. (2), then one obtains a better pair in terms of the value of the objective function value by replacing  $\phi$  with  $\psi^c$ .*

**Proof**

$$\left\{ \begin{array}{l} \psi^c(x) = \inf_{y \in Y} [c(x, y) - \psi(y)] \\ \phi(x) \leq c(x, y) - \psi(y), \quad \forall y \in Y \end{array} \right\} \implies \phi(x) \leq \psi^c(x).$$

□

Thus, using the above two lemmas, we can rewrite the dual formulation Eq. 2 as follows:

$$\sup_{\psi} D := \int_X \inf_{y \in Y} [c(x, y) - \psi(y)] d\mu + \int_Y \psi dv. \quad (3)$$

Then, when  $\nu$  is discrete,  $Y = \{y_i\}_{i=1}^N$ , and  $y_i \in X$ , equation Eq. 3 becomes:

$$\begin{aligned} D &= \int_X \inf_{y \in Y} [c(x, y) - \psi(y)] d\mu + \int_Y \psi dv, \\ &= \int_X \inf_{y_i \in Y} [c(x, y_i) - \psi_i] d\mu + \sum_{i=1}^N \psi_i v_i, \\ &= \sum_{i=1}^N \int_{A(y_i)} [c(x, y_i) - \psi_i] d\mu + \sum_{i=1}^N \psi_i v_i, \end{aligned}$$

where

$$A(y_i) = \{x \in X \mid c(x, y_i) - \psi_i \leq c(x, y_j) - \psi_j, \quad j \neq i\}, \quad i = 1, 2, \dots, N.$$

To be consistent with the standard semi-discrete OT problem notation, let  $X = \Omega$  and  $w_i = \psi_i$ . Then the minimization problem for the semi-discrete OT problem has the following form:

$$\min_{w \in \mathbb{R}^N} \Phi(w) := - \sum_{i=1}^N \int_{A(y_i)} [c(x, y_i) - w_i] d\mu - \sum_{i=1}^N w_i v_i, \quad (4)$$

where

$$A(y_i) = \{x \in \Omega \mid c(x, y_i) - w_i \leq c(x, y_j) - w_j, \quad j \neq i\}, \quad i = 1, 2, \dots, N.$$

In the literature, the vector  $w$  is called a shift (or weight) vector and the partition  $\{A(y_i)\}_{i=1}^N$  of the domain  $\Omega$  are known as Laguerre cells of  $\Omega$ . Once an optimal vector of weights is determined, and for as long as the cells' boundaries have  $\mu$ -measure 0, it is a simple observation that there is an optimal map  $T(x)$  and that it is piecewise constant, in that it maps every  $x \in A(y_i)$  to the same  $y_i$ , up to sets of  $\mu$ -measure 0.

Given the relevance in applications, there are several works concerned with numerical solution of semi-discrete OT problems, the chief differences between them being the cost functions considered. By far, the most widely studied case is for the cost function given by the 2-norm square:  $c(x, y) = \|x - y\|_2^2$ . In this case, the Laguerre tessellation is also known as “Power Diagram”, see [2]. The beauty of this case is in the simplicity of the cells' structure: they are polytopes, their boundaries are given by (hyper) planes. Unsurprisingly, very elegant and efficient tools of computational geometry are used to compute power diagrams, with remarkable success and stunning

display of the resulting subdivision. A likely incomplete list of computational works which tackle semi-discrete optimal transport in the case of the 2-norm squared includes the works [2], [41], [11], [40], [35, Chapter 5], [30], [32], [37], [12], [29], [8]. Conceivably, the reason for the algorithmic successes in this case of the 2-norm squared as cost is given by the intimate relation of power diagrams and Voronoi tessellation. In fact, a most useful algorithmic result in this case allows to lift a power diagram to be the same as a Voronoi tessellation in one higher space dimension, see [30], and computation of Voronoi tessellation has reached remarkable maturity in computational geometry: in the plane, the classical sweeping algorithm of Fortune [17] is still very popular, though there are of course several more modern presentations, e.g. [16], and there are also widely available codes for commonly used computational environments, like the Matlab functions `voronoi` and `voronoin`.

The situation for other cost functions is much less developed, and it is our present concern. To be sure, there are important theoretical results for more general costs than the 2-norm squared, for example see [21]. However, computational techniques do not appear to be very well developed, probably because of the lack of simple geometrical shapes of the Laguerre cells. Notable exceptions are the work [14], that considered general  $p$ -norm costs ( $p \neq 1, \infty$ ), and the work [24] that considered the case of  $p = 2$ , the standard Euclidean norm. As in [14], we will also restrict to  $p$ -norms as costs ( $1 < p < \infty$ ), but we will follow a very different algorithmic path from [14]. (Comparison with the approach of [24] is done in the main body of this paper; in particular, see Section 3.2). The key ideas of the “boundary method” of [14] were to track/refine only the boundary-zone of the Laguerre cells, and not their interiors, by a grid-subdivision/refinement algorithm, and to improve upon the weights  $w$  through an original adaptation of the auction algorithm. In the present work, our key insight is that –although the shape of the Laguerre cells is not as simple as having boundary given by intersection of hyper-planes– the Laguerre cells are star-shaped. This is the most we can say for general cost functions of the type we consider. Moreover, we will adopt a Newton’s method approach to refine the weights, and very accurate manifold techniques to track the boundaries between cells. As a result, in the end our method appears to be much more accurate than any other method we know of, but the price we pay is that we need the density  $\rho$  associated to the measure  $\mu$  (i.e.,  $d\mu = \rho(x)dx$ ) to be a sufficiently smooth density. That being the case, we develop a fully adaptive algorithm with error control, and we show that it performs extremely well both in terms of speed and accuracy on a variety of problems. Our main contributions are the following: (i) new algorithms for Newton’s method, exploiting star-shapedness of the Laguerre cells, (ii) unified treatment of the problem, (iii) comparison with other techniques, (iv) a full set of replicable results, with accuracy assessment well beyond the graphical display of results witnessed in the literature (how can one distinguish two close figures with the naked eye?), and (v) a rigorous numerical analysis, including the introduction of a proposal for a condition number of semi-discrete OT problems.

A plan of the paper is as follows. In Chapter 2, we first present results on Laguerre cells, under generous conditions on the cost function, see Section 2.1. Then, in Section 2.2 we further restrict to cost functions that allow us to obtain smooth boundaries between the cells. These first two set of results hold regardless of the measure  $\mu$  (and  $\nu$ ). Then, in Section 2.3, we finally recall results identifying the solution of the

semi-discrete problem as the minimum of a convex functional. Chapter 3 details the algorithms we implemented in 2-d, Chapter 4 discusses Newton's method in more details, and Chapter 5 gives computational examples. Some theoretical results are given in an Appendix, and pseudo-codes of all of our algorithms are given at the end so that our results can be replicated by others.

## 2 Theoretical Results

In this Chapter, we give results about the geometrical and analytical (measure theoretical) properties of the partition in Laguerre cells we are after. Some of these results have appeared before, often for more specialized cost functions than those we consider, and appropriate references will be given.

Although Laguerre tessellations, and optimal transport maps, can be –and are– defined in the entire ambient space  $\mathbb{R}^n$ , in practice we will restrict to working on a bounded set  $\Omega$ , that is henceforth chosen to satisfy the following characterization.

**Assumption 2.1** (*Bounded domain  $\Omega$* )  $\Omega$  will always refer to a compact convex domain of  $\mathbb{R}^n$ , of nonzero  $n$ -dimensional Lebesgue measure, and with  $\mathcal{C}^k$  piecewise smooth boundary,  $k \geq 1$ .

For completeness, recall that a  $\mathcal{C}^k$  piecewise smooth boundary of  $\Omega$  means that

$$\partial\Omega = \bigcup_{i=1}^N \bar{\Gamma}_i,$$

and each  $\Gamma_i$  is a  $\mathcal{C}^k$   $(n - 1)$ -dimensional surface. Moreover, two of the  $\bar{\Gamma}_i$ 's may intersect each other along a  $\mathcal{C}^k$   $(n - 2)$ -dimensional surface, etc..

**Remark 2.2** (a) The value of  $k$  in Assumption 2.1 should be at least the same value of  $k$  as we have for the function  $F$  in Lemma 2.22.

(b) In practice,  $\Omega$  is often given by a (hyper)-cube, which surely satisfies Assumption 2.1 with piecewise  $\mathcal{C}^\infty$  boundary. To witness, the standard unit square  $S$  will be our prototypical choice in 2-d, and its boundary is clearly piecewise smooth, being the union of four closed intervals, each intersecting two other intervals at a vertex.

Although the results of Section 2.1 hold regardless of the measures  $\mu$  and  $\nu$ , we will need to restrict to appropriate classes of cost functions. At first, we will consider cost functions that satisfy the following assumption, which is sufficient to prove geometrical properties of the Laguerre cells. Later, see Restriction 2.15, we will further restrict the class of cost functions considered, in order to obtain analytical results on the cells themselves.

**Assumption 2.3** (Cost function) *The cost function  $c : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^+$  satisfies the requirements of a distance function:*

$$\begin{aligned} \text{positivity: } & \begin{cases} c(x, y) \geq 0, & \forall x, y \in \mathbb{R}^n, \\ c(x, y) = 0 \iff x = y, & \forall x, y \in \mathbb{R}^n, \end{cases} \\ \text{symmetry: } & c(x, y) = c(y, x), \quad \forall x, y \in \mathbb{R}^n, \\ \text{triangular inequality: } & c(x, y) \leq c(x, z) + c(z, y), \quad \forall x, y, z \in \mathbb{R}^n, \end{aligned}$$

and additionally we also assume that the cost function satisfies these two properties:

$$\begin{aligned} \text{homogeneity: } & c(tx, ty) = |t|c(x, y), \quad \forall t \in \mathbb{R}, \forall x, y \in \mathbb{R}^n, \\ \text{shift invariance: } & c(x + z, y + z) = c(x, y), \quad \forall x, y, z \in \mathbb{R}^n. \end{aligned}$$

**Remark 2.4** (a) *It is an interesting theoretical consequence of the last two properties above that (e.g., see [6, p.21]) there exists a vector norm  $\|\cdot\|$  such that  $c(x, 0) = \|x\|$ ,  $\forall x \in \mathbb{R}^n$ , and therefore our cost is a continuous function.*  
 (b) *Below, see Restriction 2.15, we will give very concrete examples of the cost functions we consider in practice. Presently, we remark that commonly adopted choices, like the Euclidean norm squared:  $c(x, y) = \|x - y\|_2^2$ , do not satisfy Assumption 2.3, and hence some of our results below (in particular, Lemma 2.10 and Theorem 2.12) do not apply to this case.*

## 2.1 Star-Shapedness Results

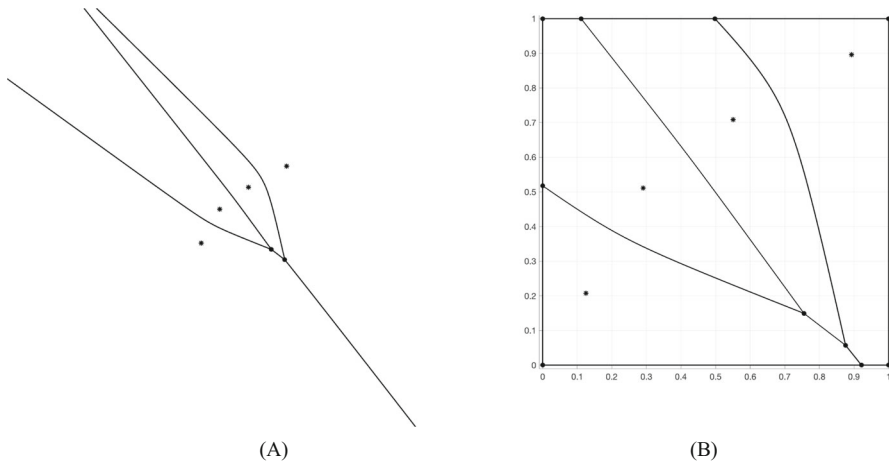
In this Section, we define Laguerre cells and tessellation and show some important geometrical properties of Laguerre cells, that are relevant for our later algorithmic development to solve semi-discrete optimal transport problems. Our main result is Theorem 2.12, that appears to be new, at least in the given generality. The results in this section are derived irrespective of the measure theoretic setting of optimal transport and only require that the cost function satisfy Assumption 2.3.

**Definition 2.5** (Laguerre tessellation) *Given a cost function  $c$  satisfying Assumption 2.3, given a set of  $N$  distinct points  $y_i \in \mathbb{R}^n$ ,  $i = 1, 2, \dots, N$ ,  $N \geq 2$ , and given a shift vector  $w \in \mathbb{R}^N$ , the Laguerre tessellation of  $\mathbb{R}^n$  associated to  $c(\cdot, \cdot)$ , to the  $y_i$ 's, and to  $w$ , is given by the set of  $N$  regions (Laguerre cells)*

$$L(y_i) = \{x \in \mathbb{R}^n \mid c(x, y_i) - w_i \leq c(x, y_j) - w_j, \quad \forall j = 1, \dots, N, j \neq i\}, \quad i = 1, 2, \dots, N. \quad (5)$$

*The points  $y_i$ ,  $i = 1, 2, \dots, N$ , are called target points. See Fig. 1. Moreover, let  $\Omega$  be as in Assumption 2.1 and let  $y_i \in \Omega^0$ ,  $i = 1, 2, \dots, N$ . Then, the Laguerre tessellation of  $\Omega$  is the set of Laguerre cells (see Fig. 1)*

$$A(y_i) = L(y_i) \cap \Omega = \{x \in \Omega \mid c(x, y_i) - w_i \leq c(x, y_j) - w_j, \quad j \neq i\}, \quad i = 1, 2, \dots, N. \quad (6)$$



**Fig. 1** For the same target points,  $w$ , and cost: (A) Laguerre cells in  $\mathbb{R}^2$  and (B) Restricted to the unit square

**Remark 2.6** The special case of  $w = 0$  gives the tessellation in terms of Voronoi cells. This is the standard proximity problem.

**Lemma 2.7** (Every  $x \in \mathbb{R}^n$  is in some  $L(y_i)$ ) For any given  $x \in \mathbb{R}^n$ , there exist an index  $i$  such that  $x \in L(y_i)$ .

**Proof** This result follows from the fact that  $\mathbb{R}^n = \bigcup_{i=1}^N L(y_i)$ . □

**Definition 2.8** (Active boundary and interior) The active boundary, or simply boundary, between two Laguerre cells  $L(y_i)$  and  $L(y_j)$ ,  $i \neq j$ , such that  $L(y_i) \cap L(y_j) \neq \emptyset$ , is indicated with  $L_{ij}$ , and it is given by the set of all  $x \in \mathbb{R}^n$  satisfying the relation

$$c(x, y_i) - w_i = c(x, y_j) - w_j \implies w_{ij} = c(x, y_i) - c(x, y_j), \quad w_{ij} = w_i - w_j,$$

as well as the  $(N - 2)$  inequalities

$$c(x, y_i) - w_i \leq c(x, y_k) - w_k, \quad k \neq i, j.$$

The boundary of a Laguerre cell  $L(y_i)$ , indicated with  $\partial L(y_i)$ , is given by

$$\partial L(y_i) = \bigcup_{\substack{j=1, j \neq i: \\ L(y_i) \cap L(y_j) \neq \emptyset}}^N L_{ij}.$$

Accordingly, we define the interior of  $L(y_i)$ , and indicate it with  $L(y_i)^\circ$ , to be the set

$$L(y_i)^\circ = L(y_i) \setminus \partial L(y_i).$$



**Lemma 2.9** (*Existence of boundary between cells: necessary condition*) Let the cost function  $c$  satisfy Assumption 2.3. The necessary condition for having an active boundary  $L_{ij}$  in  $\mathbb{R}^n$  between  $L(y_i)$  and  $L(y_j)$  is that

$$|w_i - w_j| \leq c(y_i, y_j) = c(y_j, y_i).$$

**Proof** Let  $w_{ij} = w_i - w_j$ . By Definition 2.8, the values  $x \in L_{ij}$  must satisfy the following relation:

$$c(x, y_i) - w_i = c(x, y_j) - w_j \implies w_{ij} = c(x, y_i) - c(x, y_j).$$

From the triangular inequality:

$$c(x, y_i) \leq c(x, y_j) + c(y_j, y_i) \implies w_{ij} \leq c(x, y_j) + c(y_j, y_i) - c(x, y_j) = c(y_j, y_i).$$

Similarly, with  $w_{ji} = w_j - w_i = -w_{ij}$ , one has

$$w_{ji} = c(x, y_j) - c(x, y_i) \leq c(x, y_i) + c(y_i, y_j) - c(x, y_i) = c(y_i, y_j)$$

and since  $c(y_i, y_j) = c(y_j, y_i)$ , we obtain

$$\left\{ \begin{array}{l} w_{ij} \leq c(y_j, y_i) \\ -w_{ij} \leq c(y_j, y_i) \end{array} \right\} \implies |w_{ij}| \leq c(y_j, y_i)$$

and the result follows.  $\square$

**Lemma 2.10** ( $y_i \in L(y_i)$ ) Let the cost function  $c$  satisfy Assumption 2.3. If  $L(y_i) \neq \emptyset$ , each target point  $y_i$  is contained in its own Laguerre cell  $L(y_i)$ .

**Proof** Assume  $y_i \notin L(y_i)$ . Then, because of Lemma 2.7,  $y_i \in L(y_j) \setminus \partial L(y_i)$ , for some  $j \neq i$ .

*Case 1.* Let  $L(y_i) \cap L(y_j) \neq \emptyset$ . Then, by construction of Laguerre cells, it follows that:

$$y_i \notin L(y_i), y_i \in L(y_j) \implies c(y_i, y_i) - w_i > c(y_i, y_j) - w_j \implies -w_i + w_j > c(y_i, y_j).$$

It follows that  $-w_{ij} > c(y_i, y_j)$ . But this contradicts Lemma 2.9.

*Case 2.* Let  $L(y_i) \cap L(y_j) = \emptyset$ . Then, since  $y_i \in L(y_j) \setminus \partial L(y_i)$ :

$$c(y_i, y_j) - w_j < c(y_i, y_i) - w_i = -w_i \implies w_j - w_i > c(y_i, y_j).$$

Next, let  $x_0 \in L(y_i) \setminus \partial L(y_j)$ . Such point  $x_0$  exists since  $L(y_i) \neq \emptyset$ . Then:

$$c(x_0, y_i) - w_i < c(x_0, y_j) - w_j \implies w_j - w_i < c(x_0, y_j) - c(x_0, y_i).$$

Combining the above two equations:

$$\left\{ \begin{array}{l} w_j - w_i > c(y_i, y_j) \\ w_j - w_i < c(x_0, y_j) - c(x_0, y_i) \end{array} \right\} \implies c(y_i, y_j) < c(x_0, y_j) - c(x_0, y_i).$$

Using the triangular inequality:

$$c(x_0, y_j) \leq c(x_0, y_i) + c(y_i, y_j) \implies c(y_i, y_j) < c(x_0, y_i) + c(y_i, y_j) - c(x_0, y_i) = c(y_i, y_j),$$

which is the sought contradiction.  $\square$

**Lemma 2.11** *Let the cost function  $c$  satisfy Assumption 2.3. If  $y_i \in \partial L(y_i)$ , then  $L(y_i)$  is degenerate, i.e.  $L(y_i)^0 = \emptyset$ .*

**Proof** W.l.o.g., we can assume that  $y_i \in L_{ij}$ , for some  $j \neq i$ . Then, it follows that:

$$c(y_i, y_i) - w_i = c(y_i, y_j) - w_j \implies w_j - w_i = c(y_i, y_j).$$

Next, assume by contradiction that there exist some point  $x_0 \in L(y_i)^0$ . Then  $x_0$  needs to satisfy the following relation:

$$\begin{aligned} c(x_0, y_i) - w_i &< c(x_0, y_j) - w_j \implies w_j - w_i < c(x_0, y_j) - c(x_0, y_i) \\ &\implies c(y_i, y_j) < c(x_0, y_j) - c(x_0, y_i). \end{aligned}$$

But this contradicts the triangular inequality.  $\square$

**Theorem 2.12** ( $L(y_i)$  is star shaped) *Let the cost function  $c$  satisfy Assumption 2.3. Let  $x \in L(y_i)$  and  $S(t) = \{t(x - y_i) + y_i, 0 \leq t \leq 1\}$ . Then, for all  $t \in [0, 1]$ ,  $S(t) \in L(y_i)$ .*

*As a consequence, each Laguerre cell  $L(y_i)$  is star-shaped in  $\mathbb{R}^n$  with respect to its own target point  $y_i$ .*

**Proof** The case  $t = 0$  follows from Lemma 2.10 and the case  $t = 1$  holds since  $x \in L(y_i)$ . Next, by contradiction, assume that there exists  $t_0 \in (0, 1)$  such that the point  $x_0 = t_0(x - y_i) + y_i$  is not in  $L(y_i)$ . W.l.o.g., let  $x_0 \in L(y_j) \setminus \partial L(y_i)$ , for some  $j \neq i$ . Then, it follows that, since

$$\begin{aligned} x \in L(y_i) &\implies c(x, y_i) - w_i \leq c(x, y_j) - w_j, \\ x_0 = S(t_0) &\implies c(x, y_i) = c(x, x_0) + c(x_0, y_i), \end{aligned}$$

where the second equality comes about from the following

$$\begin{aligned} c(x, x_0) + c(x_0, y_i) &= c(x, y_i + t_0(x - y_i)) + c(y_i + t_0(x - y_i), y_i) = \\ &= c((1 - t_0)(x - y_i), 0) + c(t_0(x - y_i), 0) = (1 - t_0)c(x - y_i, 0) + t_0c(x - y_i, 0) \\ &= c(x - y_i, 0) = c(x, y_i). \end{aligned}$$

As a consequence, we get

$$c(x, x_0) + c(x_0, y_i) - w_i \leq c(x, y_j) - w_j \implies c(x_0, y_i) - w_i \leq c(x, y_j) - c(x, x_0) - w_j.$$

From the triangular inequality, it follows that:

$$\begin{aligned} c(x_0, y_i) - w_i &\leq c(x, y_j) - c(x, x_0) - w_j \leq c(x, x_0) + c(x_0, y_j) - c(x, x_0) - w_j \\ &= c(x_0, y_j) - w_j, \implies c(x_0, y_i) - w_i \leq c(x_0, y_j) - w_j. \end{aligned} \quad (7)$$

In addition, since  $x_0 \in L(y_j) \setminus \partial L(y_i)$ , it holds that

$$x_0 \in L(y_j) \implies c(x_0, y_j) - w_j < c(x_0, y_i) - w_i \implies c(x_0, y_i) - w_i > c(x_0, y_j) - w_j.$$

But this inequality contradicts Eq. 7, and hence the entire segment  $S(t) \in L(y_i)$ . The fact that  $L(y_i)$  is star-shaped with respect to  $y_i$  follows at once by the fact that every point in  $L(y_i)$  can be reached via a line segment from  $y_i$  entirely contained in  $L(y_i)$ .  $\square$

**Corollary 2.13** *Let  $\Omega$  be as in Assumption 2.1, let  $y_i \in \Omega^0$ ,  $i = 1, 2, \dots, N$ , and let the cost function  $c$  satisfy Assumption 2.3. Then each Laguerre cell  $A(y_i)$  is star-shaped in  $\Omega$  with respect to the point  $y_i$ .*

**Proof** This result follows from Theorem 2.12 and the fact that the intersection of a convex set with a star-shaped set is a star-shaped set (e.g., see [7]).  $\square$

**Remark 2.14** *For Voronoi diagrams, star-shapedness is simple to infer; e.g., see [20] for results about even more general Voronoi diagrams. But, for Laguerre cells, we have found much fewer conclusive results, except for the standard Euclidean distance. Indeed, that  $L(y_i)$  are star-shaped when the cost function is the 2-norm has been observed before, for example see [9, Example 2.12], [15, p.21]. In this case of  $c(x, y) = \|x - y\|_2$ , Laguerre tessellation are often called Apollonius diagrams or also additively weighted Voronoi diagrams, and in 2-d it has been observed many times that the boundaries between cells are arcs of hyperbolas (e.g., see [9], [15], [24], or [39]), although we are not aware of a similar result for cost functions being a  $p$ -norm, with  $p \neq 2$ .*

## 2.2 Smoothness results: what cost function?

In this section we give analytical results on smoothness of the boundaries  $L_{ij}$ . To get these results, we further restrict the class of appropriate cost functions, as well as an appropriate measure  $\mu$ . As said in the introduction,  $\mu$  will always be assumed to be equivalent to Lebesgue measure and with bounded density  $\rho$  in  $\Omega$ :  $d\mu = \rho(x)dx$ . These requests on  $\mu$  allow to infer, under mild conditions on the cost function  $c(x, y)$  (namely continuity, or even lower semi-continuity), that in Eq. (1) the “inf” is realized as a “min”, a fact noted several times (e.g., see [1, 10, 36]). However, we need that there is a unique partition (map), and to obtain this result we need to place further

restrictions on the cost function. Indeed, a fundamental result of Cuesta-Albertos and Tuero-Diaz [10, Corollary 4] states that there exists a unique optimal partition (map) if the cost function  $c(x, y)$  gives  $\mu$ -measure 0 to the cells boundaries. Because of this, the key concern will be to determine that the boundaries have indeed 0-measure, in particular under what conditions on the cost function this is possible.

Several authors have addressed the issue of selecting an appropriate cost, and proposed restrictions on the cost function in order to obtain a desired end result. For example, in [21] the authors restricted to what they called *admissible costs* (see below), in [9] they restricted to so-called *radial cost* functions, whereas in [29] the authors required the cost to satisfy conditions that they called (Twist)-(Reg)-(QC), and in [12] the authors required the cost to satisfy their conditions (Diff-2) and (Cont-2); more on all these cost functions is below. We should note that the restrictions placed on the cost in the works [9], [29], and [12], appear to have been motivated by the cost given by the 2-norm square, a case that violates our original Assumption 2.3 and that therefore is not of specific interest in the present work, since we may fail to have star-shaped Laguerre cells, a fact which plays a very important role in our algorithmic development. A different approach was taken in [14] and in [24]. Namely, in [14], Dieci and Walsh considered cost given by  $p$ -norms ( $1 < p < \infty$ ) and in [24] Hartmann and Schuhmacher restricted to the 2-norm cost,  $\|x - y\|_2$ ; these choices surely satisfies Assumption 2.3 and thus lead to star-shaped Laguerre cells, although this fact was not noticed/used in [14] or [24]. The choice we will make is just slightly more general than the choice made in [14], though more general than that in [24]: we will henceforth restrict to cost functions satisfying the following Restriction 2.15, which satisfy of course our previous Assumption 2.3.

**Restriction 2.15** *Hereafter, the cost functions considered are given by positive, finite, combinations of  $p$ -norms, with  $1 < p < \infty$ . That is,*

$$c(x, y) = \sum_{l=1}^m \alpha_l \|x - y\|_{p_l}, \quad \text{where } \alpha_l > 0, \quad p_l \in (1, \infty). \quad (8)$$

The relation between our class of cost functions, and the choices made in previous works is elaborated on in Remark 2.16 below.

**Remark 2.16** *The radial cost functions of [9] are functions of the form  $c(x, y) = g(\|x - y\|_2)$ , where  $g$  is an increasing function, e.g.,  $\|x - y\|_2^2$ ; our cost functions do not fit in this class. Conditions (Reg) of [29] can be generally satisfied by our cost functions, in light of Lemma 2.21; however, conditions (Twist) and (QC) of [29] are more difficult to verify, and it may possibly not hold for our cost functions. Finally, the authors of [12] require sufficiently regular cost functions (which we can require, see Lemma 2.21), satisfying (in their notation) conditions (Diff-2-a), (Diff-2-b), (Diff-2-c) and (Cont-2); it is not difficult to verify that, with our cost functions, conditions (Diff-2-b)-(Diff-2-c)-(Cont-2) can be enforced, but condition (Diff-2-a) is more problematic. This condition asks for the existence of  $\epsilon > 0$  such that  $\|\nabla_x c(x, y_i) - \nabla_x c(x, y_j)\| > \epsilon$  for all  $x$  in  $L_{ij}$ , and thus the authors of [12] ask for this condition to hold in  $\mathbb{R}^n$ , and not just in a bounded set  $\Omega$ . For completeness, in Lemma 6.2 we show that this condition*

holds for us for even values of  $p$ , with  $c(x, y) = \|x - y\|_p$ , when we look at it on a bounded convex set  $\Omega$ , which is what we do in practice in our numerical computations. Finally, our choice of cost function always gives an admissible cost according to the definition of [21], as we prove below.

**Definition 2.17** (Admissible cost; [21]) Let  $\mathcal{L}$  be Lebesgue measure, let  $\Omega$  be as in Assumption 2.1, and let  $y_1, y_2$  be two distinct target points in  $\Omega^0$ . A cost function satisfying Assumption 2.3 is called admissible if there exist values  $w_a \leq w_b$  such that:

- $\mathcal{L}(A(y_1))$  is continuously increasing from 0 to  $\mathcal{L}(\Omega)$  as  $w_{12} = w_1 - w_2$  increases from  $w_a$  to  $w_b$ , where  $A(y_1) = \{x \in \Omega \mid c(x, y_1) - w_1 \leq c(x, y_2) - w_2\}$ ;
- likewise,  $\mathcal{L}(A(y_2))$  is continuously decreasing from  $\mathcal{L}(\Omega)$  to 0 as  $w_{12} = w_1 - w_2$  increases from  $w_a$  to  $w_b$ , where  $A(y_2) = \{x \in \Omega \mid c(x, y_2) - w_2 \leq c(x, y_1) - w_1\}$ .

**Lemma 2.18** All  $p$ -norms,  $1 < p < \infty$ , are admissible cost functions as in Definition 2.17, that is  $c(x, y) = \|x - y\|_p$ , for  $1 < p < \infty$ , is an admissible cost function. Moreover, a positive combination of admissible cost functions is an admissible cost function, and in particular a cost function satisfying Restriction 2.15 is admissible according to Definition 2.17.

**Proof** For  $1 < p < \infty$ , it is enough to observe that when  $c(x, y) = \|x - y\|_p$ , choosing  $w_a = -c(y_1, y_2) = -\|y_1 - y_2\|_p$  and  $w_b = c(y_1, y_2) = \|y_1 - y_2\|_p$  will work in Definition 2.17.

Next, take a cost function  $c(x, y) = \sum_{k=1}^m \alpha_k c_k(x, y)$  where  $\alpha_k > 0$  and  $c_k(\cdot, \cdot)$  are admissible cost functions. Let  $w_a^{(k)}$  and  $w_b^{(k)}$  be values satisfying the requirements for admissibility of the  $c_k(\cdot, \cdot)$  functions in Definition 2.17. Then, in Definition 2.17, we can take  $w_a = \sum_{k=1}^m \alpha_k w_a^{(k)}$  and  $w_b = \sum_{k=1}^m \alpha_k w_b^{(k)}$  to obtain that  $c(\cdot, \cdot)$  is admissible.  $\square$

An interesting and useful result for our cost functions is the following.

**Lemma 2.19** Let  $\Omega$  be as in Assumption 2.1 and let  $y_1, y_2 \in \Omega^0$  be two distinct target points. Let  $w_{12} = w_1 - w_2$  satisfy  $|w_{12}| \leq c(y_1, y_2)$  (see Lemma 2.9). If  $c(\cdot, \cdot)$  is an admissible cost function as in Restriction 2.15, then  $A(y_1) \cap A(y_2) \neq \emptyset$ , where

$$\begin{aligned} A(y_1) &= \{x \in \Omega \mid c(x, y_1) - w_1 \leq c(x, y_2) - w_2\}, \\ A(y_2) &= \{x \in \Omega \mid c(x, y_2) - w_2 \leq c(x, y_1) - w_1\}. \end{aligned}$$

**Proof** By direct computation, it is straightforward to show that the point  $x_0 = y_1 + \frac{k+1}{2}(y_2 - y_1)$ ,  $k = \frac{w_{12}}{c(y_1, y_2)} \in [-1, 1]$ , is contained in the boundary set  $A(y_1) \cap A(y_2)$ , where

$$A(y_1) \cap A(y_2) = \{x \in \Omega \mid c(x, y_1) - w_1 = c(x, y_2) - w_2\}.$$

Hence the result follows.  $\square$

Next, in order to obtain  $\mu$ -measure 0 for the boundaries between Laguerre cells, we will use some results of differential geometric flavor, and for these we need some preliminary results.

**Fact 2.20** Let  $a \in \mathbb{R}$ . For  $s < 1$  and  $r \geq 1$ :

$$\lim_{a \rightarrow 0} \frac{a^r}{|a|^s} = 0.$$

**Proof** This follows from the fact that  $\frac{a}{|a|^s} = |a|^{1-s} \text{sign}(a) \rightarrow 0$  since  $\text{sign}(a)$  is bounded and  $s < 1$ .  $\square$

**Lemma 2.21** Consider the function  $f(x) = \|x\|_p$ ,  $1 < p < \infty$ , for  $x$  in an open subset  $B$  of  $\mathbb{R}^n$  not containing the origin. Then,  $f$  is a  $C^k$  function in  $B$ , where

$$k = \begin{cases} \infty, & \text{if } p \text{ even} \\ m, & \text{if } m < p \leq m+1, \ m \in \mathbb{Z}^+. \end{cases}$$

**Proof** For all  $x$ , the function  $f(x)$  can be rewritten as:

$$f(x) = [|x_1|^p + |x_2|^p + \cdots + |x_d|^p]^{1/p}.$$

Then, it is enough to check smoothness of the single term  $|x_j|^p$ ,  $j \in [1, d]$ , since  $x \neq 0$ . Note that  $|x_j|^p = x_j^p$  when  $p$  is an even integer, and so the result follows for  $p$  even. Otherwise, using Fact 2.20 it is simple to show that the function  $|x_j|^p$  is  $t$  times continuously differentiable where  $t < p \leq t+1$ .  $\square$

Let  $F(x)$  be the function corresponding to the boundary set between two Laguerre cells  $L(y_1)$  and  $L(y_2)$  in  $\mathbb{R}^n$ :

$$F(x) = c(x, y_1) - c(x, y_2) \implies L(y_1) \cap L(y_2) = \{x \in \mathbb{R}^n \mid F(x) = w_{12}\}. \quad (9)$$

**Lemma 2.22** Let  $c(\cdot, \cdot)$  be an admissible cost function as in Restriction 2.15,  $y_1, y_2 \in \mathbb{R}^n$  be two distinct target points, and  $w_{12} = w_1 - w_2$  satisfy  $|w_{12}| < c(y_1, y_2)$  (see Lemma 2.9). Assume that  $w_{12}$  is a regular value for the function  $F(x)$ , that is that  $\nabla F(x_0) \neq 0$ ,  $\forall x_0 \in L(y_1) \cap L(y_2)$ . Then, the boundary set  $\{x : F(x) = w_{12}\}$  is a  $C^k$  submanifold of dimension  $(n-1)$  in  $\mathbb{R}^n$ .

**Proof** Lemma 2.19 gives the existence of at least one point  $x_0 : F(x_0) = w_{12}$ , where  $F$  is given in Eq. (9). By our Restriction 2.15, the function  $F(x)$  is a  $C^k$  function, with  $k \geq 1$ , in an open neighborhood of  $L_{ij}$ . Since  $\nabla F(x_0) \neq 0$  is assumed, then the smooth manifold theorem tells us that  $S = F^{-1}(w_{12}) \subset \mathbb{R}^n$  is a  $(n-1)$ -dimensional  $C^k$  manifold embedded in  $\mathbb{R}^n$ .  $\square$

**Remark 2.23** We notice that the request that  $w_{12}$  is a regular value for  $F$  is not a severe restriction, since the Morse-Sard Theorem guarantees that almost all values  $w_{12}$  are regular values for  $F$ . Furthermore, in the Appendix, we show in Claim 6.1 that  $\nabla F(x_0) \neq 0$ ,  $\forall x_0 \in L(y_1) \cap L(y_2)$ , whenever the cost function is a  $p$ -norm with  $p$  an even number. As a consequence, for cost functions given by positive combination of  $p$ -norms with  $p$  even, there is no need to assume that  $\nabla F(x_0) \neq 0$ , since it is always satisfied.

**Remark 2.24** In 2-d, Lemma 2.22 says that the set  $F^{-1}(w_{12})$  is a smooth curve. Cfr. with Remark 2.14 for  $c(x, y) = \|x - y\|_2$ .

**Theorem 2.25** Let  $y_i \in \mathbb{R}^n$ ,  $i = 1, 2, \dots, N$ , be a set of  $N$  distinct target points, and  $c(\cdot, \cdot)$  be an admissible cost function as in Restriction 2.15. In addition, assume that  $y_i \notin \partial L(y_i)$ ,  $i = 1, 2, \dots, N$ . Then, the boundary of each Laguerre cell  $L(y_i)$  consists of at most  $(N - 1)$  sections of  $\mathcal{C}^k$   $(n - 1)$ -dimensional manifolds of  $\mathbb{R}^n$ .

Let  $\Omega$  be as in Assumption 2.1, and let  $y_i \in \Omega^0$ ,  $i = 1, 2, \dots, N$ , be a set of  $N$  distinct target points, such that  $y_i \notin \partial A(y_i)$ ,  $i = 1, 2, \dots, N$ . Then, the boundary of each Laguerre cell  $A(y_i)$  consists of sections of  $\mathcal{C}^k$   $(n - 1)$ -dimensional manifolds of  $\mathbb{R}^n$ .

**Proof** The first statement follows from Lemma 2.22 and the form of  $\partial L(y_i)$ :

$$\partial L(y_i) = \bigcup_{\substack{j=1, j \neq i: \\ L(y_i) \cap L(y_j) \neq \emptyset}}^N L_{ij}.$$

The second statement follows from the fact that

$$\partial A(y_i) = \partial(L(y_i) \cap \Omega), \quad i = 1, 2, \dots, N,$$

from Lemma 2.22, and from the assumed piecewise smoothness of  $\partial \Omega$ .  $\square$

**Remark 2.26** In general, the union of several pieces of the same smooth  $(n - 1)$ -dimensional manifold can be part of  $A(y_i) \cap A(y_j)$ . See Fig. 2. It is guaranteed to be a single smooth section when there are exactly two target points.

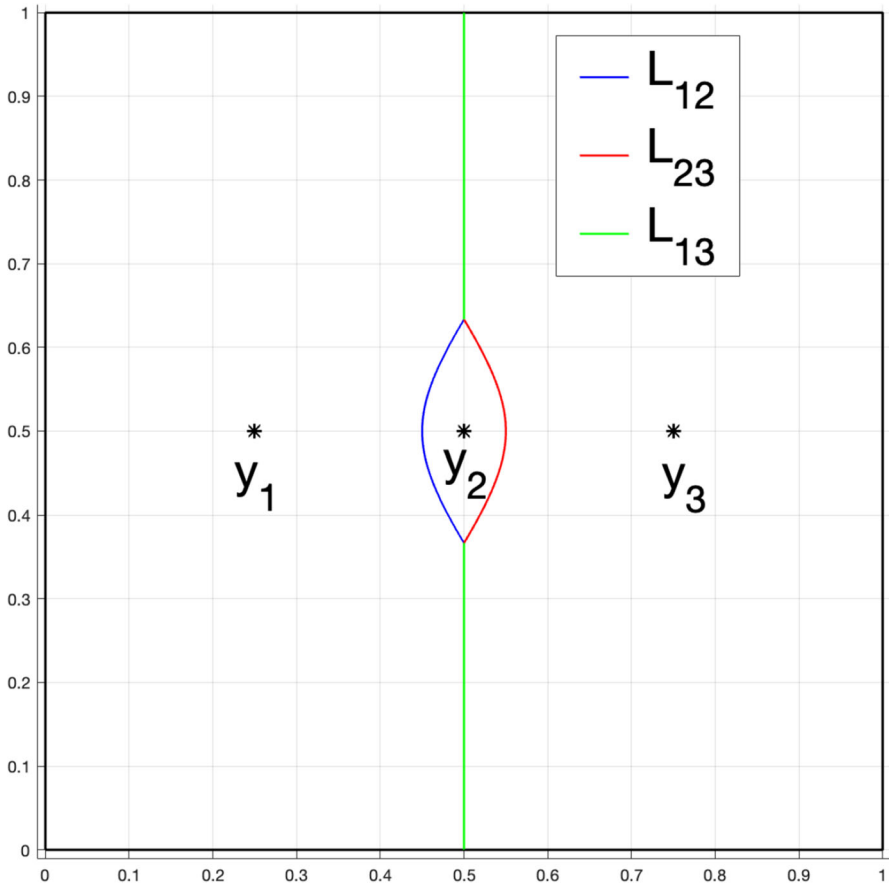
**Lemma 2.27** Let  $\mathcal{L}$  be Lebesgue measure. Let  $\Omega \subset \mathbb{R}^n$  be as in Assumption 2.1, and let  $y_1, y_2 \in \Omega^0$  be two distinct target points. Let  $c(\cdot, \cdot)$  be a cost function satisfying Restriction 2.15. Then,  $\mathcal{L}(A(y_1) \cap A(y_2)) = 0$  holds, where

$$\begin{aligned} A(y_1) &= \{x \in \Omega \mid c(x, y_1) - w_1 \leq c(x, y_2) - w_2\}, \\ A(y_2) &= \{x \in \Omega \mid c(x, y_2) - w_2 \leq c(x, y_1) - w_1\}. \end{aligned}$$

**Proof** From Lemma 2.22,  $\mathcal{L}(L_{ij}) = 0$ , and thus the result follows.  $\square$

**Corollary 2.28** Let  $\mathcal{L}$  be Lebesgue measure. Let  $\Omega \subset \mathbb{R}^n$  be as in Assumption 2.1, and let  $y_i \in \Omega^0$ ,  $i = 1, 2, \dots, N$ , be  $N$  distinct target points. If  $c(\cdot, \cdot)$  is an admissible cost function, then  $\mathcal{L}(\partial A(y_i)) = 0$ ,  $i = 1, 2, \dots, N$ .

**Proof** This result follows from Lemma 2.27, and the facts that  $\partial L(y_i) = \bigcup_{\substack{j=1, j \neq i: \\ L(y_i) \cap L(y_j) \neq \emptyset}}^N L_{ij}$ , and that  $A(y_i) = L(y_i) \cap \Omega$ .  $\square$



**Fig. 2** The boundary between  $A_1$  and  $A_3$  is made up by two disjoint pieces of the same smooth curve

**Example 2.29** We show next why the 1-norm and the  $\infty$ -norm must be excluded from consideration in our restriction on allowed costs. The reason is that they violate Lemma 2.27 and lead to lack of uniqueness, in general. This fact is actually well understood, and already true for Voronoi diagrams, but we give the following counterexamples because they are simple and can be worked out by hand.

- Counterexample for 1-norm. Let  $\Omega = [0, 3]^2$ ,  $y_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ ,  $y_2 = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$ ,  $w_1 = w_2 = 0$ . The boundary between the two Laguerre cells with respect to the 1-norm will satisfy the following equation:

$$\begin{aligned} x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in A(y_1) \cap A(y_2) &\implies \|x - y_1\|_1 = \|x - y_2\|_1 \\ &\implies |x_1 - 1| + |x_2 - 2| = |x_1 - 2| + |x_2 - 1|. \end{aligned}$$



But, to illustrate, any point in the square  $0 \leq x_1 \leq 1$  and  $0 \leq x_2 \leq 1$  will satisfy the above relation:  $(1 - x_1) + (2 - x_2) = (2 - x_1) + (1 - x_2)$ . As a result,  $\mu(A(y_1) \cap A(y_2)) \neq 0$ . See Fig. 3: any point in the shaded regions is both in  $A(y_1)$  and  $A(y_2)$ . In particular, we cannot satisfy  $\mu(A(y_1)) + \mu(A(y_2)) = 1$ .

- Counterexample for  $\infty$ -norm. Let  $\Omega = [0, 3]^2$ ,  $y_1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ ,  $y_2 = \begin{pmatrix} 3 \\ 2 \end{pmatrix}$ ,  $w_1 = w_2 = 0$ . The boundary between the two Laguerre cells with respect to  $\infty$ -norm will satisfy the following equation:

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in A(y_1) \cap A(y_2) \implies \|x - y_1\|_\infty = \|x - y_2\|_\infty, \\ \max\{|x_1 - 1|, |x_2 - 2|\} = \max\{|x_1 - 3|, |x_2 - 2|\}.$$

Now, take a point  $x$  in the square  $1 \leq x_1 \leq 2$  and  $0 \leq x_2 \leq 1$ . Then:

$$\max\{(x_1 - 1), (2 - x_2)\} = \max\{(3 - x_1), (2 - x_2)\} \implies (2 - x_2) \\ = \max\{(3 - x_1), (2 - x_2)\}.$$

But, to illustrate, any point in the region  $x_1 - x_2 > 1$  will satisfy this relation and thus  $\mu(A(y_1) \cap A(y_2)) \neq 0$ . See Fig. 3: any point in the shaded regions is both in  $A(y_1)$  and  $A(y_2)$ . In particular, we cannot satisfy  $\mu(A(y_1)) + \mu(A(y_2)) = 1$ .

Finally, the following elementary result is useful to justify some steps of our algorithm (see Section 3.1.1).

**Corollary 2.30** Let  $\Omega \subset \mathbb{R}^n$  be as in Assumption 2.1, let  $c$  satisfy Restriction 2.15, and let  $y_i \in \Omega^0$ ,  $i = 1, 2, \dots, N$ , be  $N$  distinct target points. Then, any open line segment from a point on  $\partial A(y_i)$  to the point  $y_i$  is contained in the interior of  $A(y_i)$ .

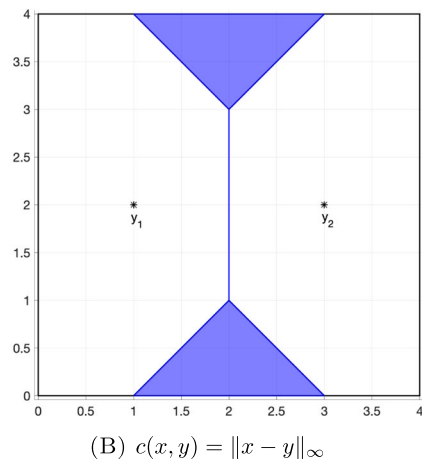
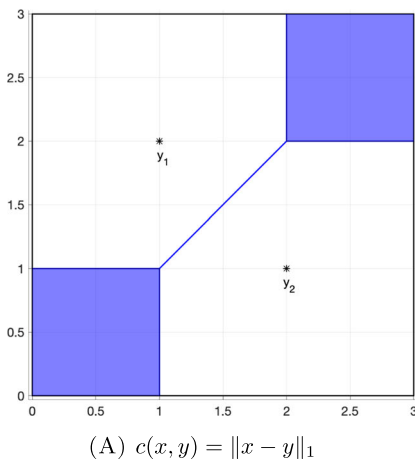


Fig. 3 Counterexamples to uniqueness:  $w = 0$

**Proof** This result follows from Corollary 2.13, piecewise smoothness of  $\partial A(y_i)$  and the fact that  $\mathcal{L}(\partial A(y_i)) = 0$ .  $\square$

## 2.3 Minimization Problem Results

The results in this section are needed to justify use of our numerical method for finding the OT partition. With the exception of Theorem 2.32, and possibly point (2) of Theorem 2.36, the results below are known in the literature on semi-discrete optimal transport.

**Theorem 2.31** *Let  $\Omega \subset \mathbb{R}^n$  be as in Assumption 2.1,  $\mu$  be a probability measure with support on  $\Omega$ , equivalent to Lebesgue measure, let  $y_i \in \Omega^0$ ,  $i = 1, 2, \dots, N$ , be a set of  $N$  distinct target points, and  $c(\cdot, \cdot)$  be an admissible cost function as in Restriction 2.15. Given a discrete measure  $\nu$  supported at the  $y_i$ 's, that is  $\nu(x) = \nu_i \delta(y_i)$ ,  $i = 1, \dots, N$ ,  $\nu_i > 0$  and  $\sum_{i=1}^N \nu_i = 1$ , then there exists a shift vector  $w$  (unique up to adding the same constant to all its entries), and a partition of  $\Omega$ :  $\{A(y_i)\}$ , such that the partition is unique (up to sets of  $\mu$ -measure 0), and such that  $\mu(A(y_i)) = \nu_i$  is satisfied for all  $i = 1, 2, \dots, N$ .*

**Proof** Since our cost function is an admissible cost (Lemma 2.18), this result is in [21, Theorem 1].  $\square$

A very important fact, relative to the solution  $w$  of Theorem 2.31, is the following *feasibility result*, which strengthens Lemma 2.9 and also has crucial algorithmic implications.

**Theorem 2.32** (*Feasibility of  $w$* ) *With the notation of Theorem 2.31, the vector  $w$  must satisfy the following relation*

$$|w_i - w_j| < c(y_i, y_j), \quad i \neq j. \quad (10)$$

**Proof** From Lemma 2.11, we have that if

$$w_j - w_i = c(y_i, y_j) \Rightarrow L^0(y_i) = \emptyset.$$

But then,  $A^0(y_i) = \emptyset$  and  $\mu(A(y_i)) = 0$  and we cannot satisfy  $\mu(A(y_i)) = \nu_i$  with  $\nu_i > 0$ .  $\square$

**Theorem 2.33** *Let  $\mu$  be a bounded probability measure on  $\Omega \subset \mathbb{R}^n$ , with  $\Omega$  satisfying Assumption 2.1, let  $y_i \in \Omega^0$ ,  $i = 1, 2, \dots, N$ , be a set of  $N$  distinct target points, let  $c(\cdot, \cdot)$  be an admissible cost function satisfying Restriction 2.15, and let  $\nu$  be a discrete measure supported at the  $y_i$ 's, that is  $\nu(x) = \nu_i \delta(y_i)$ ,  $i = 1, \dots, N$ ,  $\nu_i > 0$  and  $\sum_{i=1}^N \nu_i = 1$ . Then, the function  $\Phi(w)$*

$$\Phi(w) = - \sum_i^N \int_{A(y_i)} [c(x, y_i) - w_i] d\mu - \sum_i^N w_i \nu_i, \quad (11)$$

is convex,  $C^1$ -smooth, and has gradient

$$\nabla \Phi(w) = \{\mu(A(y_i)) - v_i\}_{i=1:N}. \quad (12)$$

Moreover, a minimizer of  $\Phi$  gives the optimal partition (piecewise constant map), up to sets of  $\mu$ -measure 0.

**Proof** With our Restriction 2.15, the result follows from [12, Proposition 1]. (See also [29, Theorem 1.1 and Corollary 1.2].)  $\square$

**Remark 2.34** (1) In agreement with the uniqueness statement of Theorem 2.31, it is plainly apparent that in Theorem 2.33 neither  $\Phi(w)$  nor  $\nabla \Phi(w)$  change if a constant value is added to all entries of  $w$  (the same constant, of course). For our problem, this is the only lack of uniqueness we have.

(2) It is worth emphasizing that the dependence on  $w$  in the gradient  $\nabla \Phi$  is through the regions  $A(y_i)$ , recall Eq. 6.

Theorem 2.33 is the basis of minimization approaches to find the OT partition; e.g., see [24] for the 2-norm cost. Our numerical method is based on seeking a root of the gradient in Eq. (12), and we will use Newton's method to this end. Of course, use of Newton's method requires the Jacobian of  $\nabla \Phi$  (the Hessian of  $\Phi$ ), and we will use the formulas for the Hessian given in [12], [29], [35]. Indeed, in spite of slightly different notations, the expressions of the Hessian given in these works are actually the same. One of the merits of our work will be to show that our implementation of Newton's method, whereby we exploit the geometrical features of Laguerre's cells and their star-shapedness, is very efficient and—in principle— it leads to arbitrarily accurate results, at least in  $\mathbb{R}^2$  and for sufficiently smooth density function  $\rho$ .

**Theorem 2.35** The Hessian matrix of  $\Phi(w)$  is given by:

$$\begin{aligned} \frac{\partial^2 \Phi}{\partial w_i \partial w_j} &= \int_{A_{ij}=A(y_i) \cap A(y_j)} \frac{-\rho(x)}{\|\nabla_x c(x, y_i) - \nabla_x c(x, y_j)\|} d\sigma, \quad j \neq i; i, j = 1, 2, \dots, N, \\ \frac{\partial^2 \Phi}{\partial w_i^2} &= - \sum_{j \neq i} \frac{\partial^2 \Phi}{\partial w_i \partial w_j}, \quad i = 1, 2, \dots, N, \end{aligned} \quad (13)$$

where  $d\sigma$  is the  $(n - 1)$ -dimensional infinitesimal surface element.

**Proof** See [12, Theorem 1].  $\square$

**Theorem 2.36** Let  $H$  be the Hessian matrix of  $\Phi(w)$  from Theorem 2.35 associated to a feasible  $w$ . Then,  $H$  has the following properties:

- (1)  $H$  is symmetric positive semi-definite, and the vector of all ones is in the nullspace of  $H$ ;
- (2)  $H$  has rank  $(N - 1)$ . In particular, 0 is a simple eigenvalue.

**Proof** The fact that  $H$  is symmetric positive semi-definite follows immediately from the structure of the Hessian matrix in Eq. (13) and the fact that the function  $\Phi(w)$  is convex. Likewise, the fact that

$$H \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = 0 \text{ is a simple verification.}$$

Next, let  $\lambda_1(H) = 0 \leq \lambda_2(H) \leq \dots \leq \lambda_N(H)$  be the eigenvalues of  $H$ . To show statement (2), we need to argue that  $\lambda_2(H) > 0$ . Consider the undirected graph  $\Gamma$  whose vertices are the target points  $y_i$ , and where there is an edge between vertices  $y_i$  and  $y_j$  (because of our assumptions on  $\mu$ ) if  $A(y_i) \cap A(y_j) \neq \emptyset$ . Then, this graph is strongly connected by construction (there is a path from any vertex to any other vertex moving along the graph edges), since  $w$  is feasible. By construction of  $H$ ,  $H$  is a *generalized Laplacian* associated to this graph (see [22, Section 13.9] for the concept of generalized Laplacian). Then, we can argue like in [22, Lemma 13.9.1] to infer that  $\lambda_1(H)$  is a simple eigenvalue. Indeed, take the matrix  $H - \eta I$  for some value of  $\eta$  so that the diagonal entries of  $H - \eta I$  are non-positive and consider the matrix  $-(H - \eta I)$  which has nonnegative entries and it is irreducible, since the graph associated to it is strongly connected (because  $\Gamma$  was). Then, by the Perron-Frobenius theorem, the largest eigenvalue of  $-(H - \eta I)$  is simple, and it is its spectral radius and therefore it is the positive value  $\eta$ , and we can take the vector of all 1's as the associated Perron eigenvector. Therefore, 0 is a simple eigenvalue of  $H$  and the rank of the Hessian matrix is  $(N - 1)$ .  $\square$

### 3 Laguerre Cell Computation: 2D

Here we describe the numerical algorithms we implemented. Since all of the computations we will present are in  $\mathbb{R}^2$ , we will delay discussion of appropriate techniques in  $\mathbb{R}^3$  to a future work, and our algorithmic description below is done relative to 2-d only.

The fundamental task is to compute  $\nabla \Phi$  in Eq. (12) and then form the Hessian in Eq. (13). In essence, this requires being able to compute the  $\mu(A(y_i))$ 's, the boundaries  $A_{ij}$ 's, and the line integrals in Eq. (13). Below, we will describe fast and precise algorithm for these tasks, and for ease of exposition, we are going to make two simplifications in our presentation: (i) the cost function is a  $p$ -norm (for  $1 < p < \infty$ ),  $c(x, y) = \|x - y\|_p$ , and (ii) the domain  $\Omega$  is the square  $[a, b] \times [a, b]$ . It is fairly straightforward to adapt the description of our algorithms to the case of cost given by a positive combination of  $p$ -norms, and for several choices of  $\Omega$ . Our computations in Chapter 5 will exemplify these two points; see Sections 5.3 and 5.6.

First of all, we note a main advantage of our technique: it is fully parallelizable. To witness, the computation of  $\mu(A(y_i))$  is independent of the computation of  $\mu(A(y_j))$ ,  $\forall j \neq i$ . Hence, we can formulate the task as follows. Let  $Y = y_i$ , for some  $i$ , then:

$$\text{Compute } \mu(A(Y)) = \int_{A(Y)} d\mu,$$

$$\text{where } A(Y) = \{x \in \Omega \mid \|x - Y\|_p - W \leq \|x - y_j\|_p - w_j, \forall j : y_j \neq Y\}.$$

The second feature of our algorithm is that it exploits star-shapedness result from Section 2.1. Thus:

$$A(Y) = \{x \in \Omega \mid \|x - Y\|_p - W \leq \|x - y_j\|_p - w_j, \forall j : y_j \neq Y\}, \text{ or } \\ A(Y) = \{x = Y + z \in \Omega \mid \|z\|_p - W \leq \|z + Y - y_j\|_p - w_j, \forall j : y_j \neq Y\}.$$

Since  $A(Y)$  is star shaped with respect to  $Y$ , we can parametrize in a unique way the values on a ray from  $Y$  in terms of distance from  $Y$  along the direction determined by an angle  $\theta$ ; in other words, in the above we can write  $z = D(\theta) \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$ , and thus write for  $A(Y)$ :

$$A(Y) = \{(\theta, D(\theta)), \theta \in [0, 2\pi] \mid \|D(\theta) \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}\|_p - W \leq \|D(\theta) \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} + Y - y_j\|_p - w_j, \forall j : y_j \neq Y\}. \quad (14)$$

Clearly, the boundary of the Laguerre cell,  $\partial A(Y)$ , is also parametrizable in terms of the angle  $\theta$ . Namely, reserving the notation  $r(\theta)$  for the points on the boundary, we have

$$x \in \partial A(Y) \iff x = Y + r(\theta) \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \text{ for some unique } \theta \in [0, 2\pi). \quad (15)$$

When determining  $\partial A(Y)$ , in addition to the value of  $r$  corresponding to each angle  $\theta$ , we will monitor also an index  $k$ , which indicates the neighboring cell or the portion of the physical boundary for that value of the angle  $\theta$ :

$$x = Y + r(\theta) \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \in A(Y) \cap A(y_j) \implies k = j \\ x = Y + r(\theta) \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \in A(Y) \cap \partial \Omega \implies k = \begin{cases} -1, & \text{when } x \text{ is on the bottom side of } \Omega \\ -2, & \text{when } x \text{ is on the right side of } \Omega \\ -3, & \text{when } x \text{ is on the top side of } \Omega \\ -4, & \text{when } x \text{ is on the left side of } \Omega. \end{cases} \quad (16)$$

Below, we give a high level description of the algorithms we used to compute  $\partial A(Y)$ ,  $\mu(A(Y))$ , and so forth, and then at the end we give a descriptions of the algorithms we used in the form of easily replicable pseudo-codes. Although we implemented our algorithms in `Matlab`, by providing these pseudo-codes in a language-independent manner, it will be possible for others to implement the techniques in the environment of their choosing.

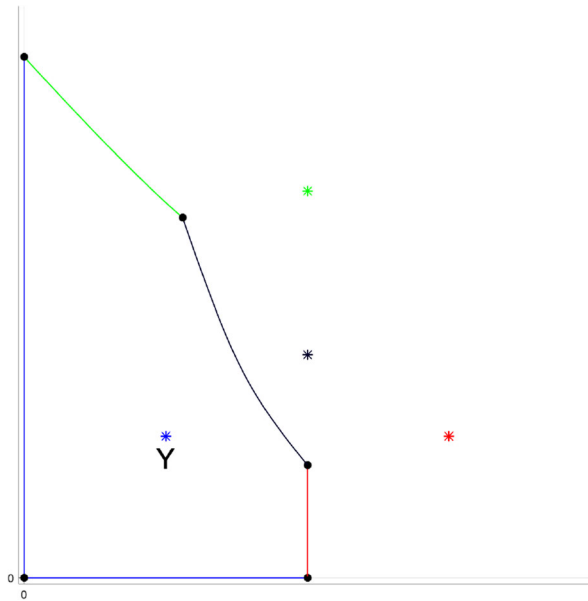


Fig. 4 Smooth arcs and breakpoints

### 3.1 Boundary Tracing Technique

Here we describe how we find the boundary of  $A(Y)$ . Our approach below is very different from that used in [14]. Namely, in [14], for a given set of weights, the boundary of  $A(Y)$  was found by a subdivision algorithm, whereby the boundary itself was covered by (smaller and smaller) squares up to a desired resolution. As explained in [14], this approach was inherently limited to a resolution of the boundary of  $\mathcal{O}(\sqrt{\text{eps}})$ <sup>1</sup> accuracy, but did not need to distinguish a-priori the different smooth arcs making up the boundary, which we instead need to do, see below. At the same time, our present technique is able to find the boundary to arbitrary accuracy (in principle).

**Remark 3.1** *We have to track the boundary of  $A(Y)$ , by making sure we also identify where changes can occur. In essence, the basic task is encoded in Fig. 4, where we identify the smooth arcs and the breakpoints making up  $\partial A(Y)$ .*

#### 3.1.1 Shooting Step

The goal of this step is to find a point on  $\partial A(Y)$ , for a given value  $\theta_0$ . That is, we will identify the value  $r_0$ , and the index  $k$  in Eq. (16), such that

$$x = Y + r_0 \begin{pmatrix} \cos(\theta_0) \\ \sin(\theta_0) \end{pmatrix} \in \partial A(Y).$$

<sup>1</sup>  $\text{eps}$  is the machine-precision, which on a double precision environment is  $\approx 2.2 * 10^{-16}$

The idea is simple: we move along the ray in the direction of the angle  $\theta_0$  until we hit the boundary. Algorithm 1, *Shooting*, shows implementation details of this technique. Algorithm 2, *BisectR*, finds the root of the boundary function  $F(r, \theta_0)$  given in Eq. (17) below using the bisection method on an interval  $[r_a, r_b]$  until  $\frac{r_b - r_a}{2}$  is sufficiently small. It is assumed/enforced that  $F(r_a, \theta_0) < 0$  and  $F(r_b, \theta_0) > 0$ . Finally, Algorithm 3, *Bound*, identifies the value on  $\partial\Omega$  intersected by the ray from  $Y$  in the direction of the angle  $\theta_0$ .

Although with the shooting technique of Section 3.1.1 we can find, for any value  $\theta$ , the value of  $r$  on  $\partial A(Y)$ , and the index  $k$ , it is more efficient to use a curve continuation technique when the index  $k$  is not changing (see Section 3.1.3 on this last issue). Our technique is a classical predictor-corrector approach, with tangent predictor and Newton corrector, described next.

### 3.1.2 Predictor-Corrector Method

Here, we exploit the fact that the value of  $r$  on the boundary of  $\partial A(Y)$  is a smooth function of  $\theta$ , and a solution of  $F(r(\theta), \theta) = 0$  for the function  $F(r, \theta)$  below:

$$F(r, \theta) = \|r \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}\|_p - \|r \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} + Y - y_j\|_p + w_j - W. \quad (17)$$

Therefore, a simple application of the implicit function theorem gives that

$$r_\theta = -\frac{F_\theta}{F_r},$$

and this allows for an immediate computation of a tangent predictor, followed by a Newton corrector.

Namely, given  $\theta_0$ ,  $r(\theta_0)$ , and an angular increment  $\Delta$ , we compute  $r(\theta + \Delta)$  by tangent predictor and Newton corrector using  $F_r$  and  $F_\theta$  below:

$$\begin{aligned} F_r(r, \theta) &= \frac{1}{p} \|r \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}\|_p^{1-p} \left[ \frac{\partial}{\partial r} |r \cos(\theta)|^p + \frac{\partial}{\partial r} |r \sin(\theta)|^p \right] - \\ &\quad - \frac{1}{p} \|Y - y_j + r \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}\|_p^{1-p} \left[ \frac{\partial}{\partial r} |Y_1 - y_{j,1} + r \cos(\theta)|^p + \frac{\partial}{\partial r} |Y_2 - y_{j,2} + r \sin(\theta)|^p \right], \\ F_\theta(r, \theta) &= \frac{1}{p} \|r \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}\|_p^{1-p} \left[ \frac{\partial}{\partial \theta} |r \cos(\theta)|^p + \frac{\partial}{\partial \theta} |r \sin(\theta)|^p \right] - \\ &\quad - \frac{1}{p} \|Y - y_j + r \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}\|_p^{1-p} \left[ \frac{\partial}{\partial \theta} |Y_1 - y_{j,1} + r \cos(\theta)|^p + \frac{\partial}{\partial \theta} |Y_2 - y_{j,2} + r \sin(\theta)|^p \right]. \end{aligned}$$

Algorithm 4, *PredCorr*, gives implementation details of the PC method. In addition to the value of  $r_1 = r(\theta_0 + \Delta)$ , this algorithm outputs also a new value for the angular increment  $\Delta$ , which reflects how close the approximation given by the tangent predictor was to the final value  $r_1$ .

**Remark 3.2** We performed extensive experiments also with the trivial, the secant, and the parabolic predictor. But, in our experiments, the tangent predictor outperformed these alternatives, not only in accuracy, but also in overall computational time.

Through use of the techniques of Sections 3.1.1 and 3.1.2, we compute smooth arcs of  $\partial A(Y)$ . To guarantee that these are indeed smooth arcs, we need to locate points where there is a change.

### 3.1.3 Breakpoint Location

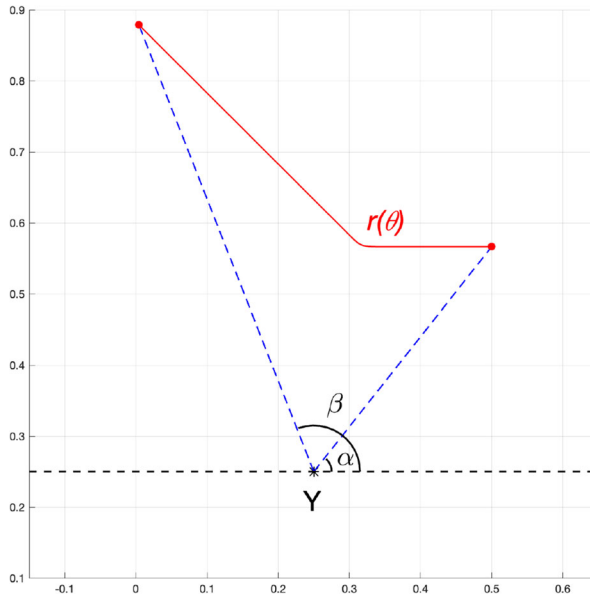
During the curve continuation process, we always monitor the index  $k$  in Eq. (16). When this index changes in the interval  $[\theta_0, \theta_0 + \Delta]$ , this means there is a *breakpoint*, i.e. there is a value  $\theta \in [\theta_0, \theta_0 + \Delta]$  such that boundary is changing from one branch to another. There are three things that can happen: (i) the boundary changes from being portion of a curve to a portion of another curve (i.e., the neighboring cell is changing from  $A(y_j)$  to another cell  $A(y_i)$ ); (ii) the boundary is changing from being a portion of the curve between two/more cells to being a portion of the boundary of  $\Omega$ , or vice versa; and (iii) the boundary changes from one smooth branch of  $\partial\Omega$  to another branch of  $\partial\Omega$  (presently, this means that two sides of  $\partial\Omega$  are meeting at a corner). The goal is to identify the breakpoint in these three cases.

In the first situation, we use bisection method on the interval  $[\theta_0, \theta_0 + \Delta]$  (see Algorithm 5, *BisectTheta*). In the second situation, we use a standard root finding technique. In fact, since we can express  $r$  in terms of  $\theta$ , we used the MATLAB function `fzero` to find the root, in the interval  $[\theta_0, \theta_0 + \Delta]$ , of the function  $F(r(\theta), \theta)$  in Eq. (17); see Algorithm 6, *CurveBound*. Finally, for the last situation, when the breakpoint is at the intersection of two branches of  $\partial\Omega$ , using the relation between indices it is easy to identify exactly the intersection point of the two branches (in our case, the corner of the square). See Algorithm 7, *BoundBound*.

## 3.2 2-d Integral Computation

The other main task is to compute the integral,  $\mu(A(Y))$ . This step is common to all method that want to minimize the functional  $\Phi(w)$  in Eq. 11 (see also Eq. (12)), whether or not by using Newton's method. The approach taken in [29] and [12], where Newton's method is used for  $c(x, y) = \|x - y\|_2^2$ , is not detailed in their works, but it is most likely based on triangulation of the cell  $A(Y)$  in order to take advantage of the polygonal shape of the latter. In [24], instead, where a quasi-Newton approach is used for  $c(x, y) = \|x - y\|_2$ , the authors adopt a quadrature based on subdivision of the given domain (the square  $\Omega$ ) in a uniform way and approximate all integrals on this uniform grid. No adaptivity, error estimation, nor error control, is given in [24]. In our algorithm, instead, we will exploit the parametrization in  $\theta$  expressed in Eq. (14), in particular, points inside  $A(Y)$  along a ray from  $Y$  are parametrized by the distance from  $Y$  and the angular direction  $\theta$  of the ray. The main advantages of our method are that it will provide error estimates, work in a fully adaptive fashion, and be much more efficient and accurate than competing techniques. The main drawback is the need to have a sufficiently smooth density  $\rho$ ,  $C^4$  to be precise, since we use a





**Fig. 5** Integration between breakpoints

composite Simpson's rule. Conceivably, there may be situation when this is a stringent requirement and alternatives will need to be adopted, either by using a quadrature rule requiring less smoothness<sup>2</sup>, or by accepting answers which are accurate to just a few digits.

Consider a smooth arc of the boundary  $\partial A(Y)$ , comprised between two breakpoints, say at the angles  $\alpha$  and  $\beta$ , and let  $r(\theta)$  be the smooth function expressing the boundary curve. Then, we have that the integral giving  $\mu(A(Y))$  restricted to the values of  $x$  falling in the sector determined by  $(\alpha, r(\alpha))$  and  $(\beta, r(\beta))$ , and center  $Y$ , is given by (see Fig. 5)

$$\int_{\alpha}^{\beta} \int_0^{r(\theta)} d\mu = \int_{\alpha}^{\beta} \int_0^{r(\theta)} \rho(x) dx = \int_{\alpha}^{\beta} \int_0^{r(\theta)} \rho(D, \theta) D dD d\theta .$$

As a consequence, for  $\mu(A(Y))$ , we have

$$\begin{aligned} \mu(A(Y)) &= \int_{A(Y)} d\mu = \int_{A(Y)} \rho(x) dx = \sum_{l=1}^{N_l} \int_{\theta_{l,0}}^{\theta_{l,1}} \int_0^{r(\theta)} \rho(D, \theta) D dD d\theta \\ &= \sum_{l=1}^{N_l} \int_{\theta_{l,0}}^{\theta_{l,1}} F(\theta) d\theta , \quad \text{with} \quad F(\theta) = \int_0^{r(\theta)} \rho(D, \theta) D dD , \end{aligned} \quad (18)$$

<sup>2</sup> We also implemented a version of composite trapezoidal rule, but for smooth  $\rho$  it was much less efficient than Simpson's.

where  $N_l$  is the number of breakpoints and  $\theta_{l,0}$  and  $\theta_{l,1}$  are the angular values corresponding to consecutive (in  $\theta$ ) breakpoints. As a result, the integration is always performed along smooth sections of the boundary  $\partial A(Y)$  and it involves 1-d integrals (in nested fashion). Because of this fact, to evaluate each integral we implemented an “adaptive composite Simpson’s rule” with error control, see Algorithm 8, *AdaptSimpson*, and Algorithm 9, *CompSimpson*. Recall that, on an interval  $\theta \in [\alpha, \beta]$ , where  $r(\theta)$  is smooth, the basic composite Simpson’s rule with  $n$  subdivisions reads

$$\int_{\alpha}^{\beta} F(\theta) d\theta = \frac{h}{3} \left[ F(t_0) + 4 \sum_{i=1}^{n/2} F(t_{2i-1}) + 2 \sum_{i=1}^{n/2-1} F(t_{2i}) + F(t_n) \right] + E(F),$$

where  $h = \frac{\beta - \alpha}{n}$  and  $t_i = \alpha + ih$ . When  $n = 2k$  for some  $k \in \mathbb{N}$ , the error term  $E(F)$  has the form:

$$E(F) = -\frac{h^4}{180} (\beta - \alpha) F^{(4)}(\xi),$$

justifying why we are requiring  $\rho$  to be  $\mathcal{C}^4$ . In the nested form of Eq. (18), the function  $F(\theta)$  is of course approximated by adaptive composite Simpson’s rule as well.

**Remark 3.3** *Adaptivity with error control turned out to be essential in order to resolve a lot of problems, since the boundary of the Laguerre cells often have large, localized, curvature values, and local refinement turned out to be appropriate. Of course, the caveat to use an accurate integral approximation, as we did, is that we are asking that the density  $\rho$  is  $\mathcal{C}^4$ . See Example 5.2-(D) for a case where it is needed to smooth out a given non-smooth density.*

## 4 Implementation of Newton’s Method and Conditioning of the OT problem

In the context of semi-discrete optimal transport problem, both [12] and [29] give implementation and testing of Newton’s method when the cost function is the 2-norm square; of course, in this case the effort to compute Laguerre cells is greatly simplified since they happen to be polygons. We are not aware of any effort for the cost functions we considered in this work, in particular for cost given by a  $p$ -norm,  $1 < p < \infty$ . Indeed, even in the important case of the 2-norm, Hartmann and Schumacher in [24] used a quasi-Newton minimization technique (LBFGMS) and, referring to the expression of the Hessian in Eq. (13), they remarked “It remains to be seen, however, if the advantage from using the Hessian rather than performing a quasi Newton method outweighs the considerably higher computational cost due to computing the above integrals numerically ...”. The main computational goal of ours in this work is to show that Newton’s method can be implemented very efficiently, and we will see in Section 5 that is quite a bit more effective than quasi-Newton approaches. In fact, computation of the Hessian itself using Eq. 13 turns out to be inexpensive (often much less expensive than computing the gradient), ultimately because it requires 1-d integrals to be approximated, rather than 2-d integrals; this is particularly apparent

in the increase in cost when approximating the Hessian by divided differences rather than by using the analytic expression Eq. 13, though of course the divided difference approximations lead to just as accurate end results. To witness, see Table 1 below, which refers to Examples (E1)-(E4) of Section 5; in Table 1, “Forward Difference” refers to the approximate Hessian whose columns are obtained as standard difference quotient:  $\frac{\nabla\Phi(w+e_i h)-\nabla\Phi(w)}{h}$ ,  $h = \sqrt{\text{eps}}$ . “Centered Difference” refers to the approximate Hessian where each column is obtained as centered difference quotient:  $\frac{\nabla\Phi(w+e_i h)-\nabla\Phi(w-e_i h)}{2h}$ ,  $h = (\text{eps})^{1/3}$ . When using these divided difference approximation, we explicitly enforce symmetry and the row-sum condition in Eq. (13).

As usual, an implementation of Newton’s method requires three main ingredients: forming the Hessian, solving for the updates and iterate, and of course providing a good initial guess. We look at these issues next.

### 4.1 Hessian Computation

To compute the Hessian, we exploit the parametrization in terms of the angular variable  $\theta$ . Namely, looking at the expression of  $\frac{\partial^2\Phi}{\partial w_i \partial w_j}$  in Eq. (13), we note that  $x$  is restricted to the boundary  $A_{ij}$  between  $A(y_i)$  and  $A(y_j)$ , and thus can be given as

$$x(r(\theta), \theta) = y_i + r(\theta) \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$$

for an appropriate range of  $\theta$ -values, and where  $r(\theta)$  refers to the associated part of the boundary curve, that is the distance from  $y_i$  along the boundary curve. Thus, the term  $\frac{\partial^2\Phi}{\partial w_i \partial w_j}$  can be rewritten as an arc-length integral:

$$\frac{\partial^2\Phi}{\partial w_i \partial w_j} = \sum_{\text{smooth arcs}} \int_{\theta_0}^{\theta_1} \frac{-\rho(r(\theta), \theta)}{\|\nabla_x c(x(r(\theta), \theta), y_i) - \nabla_x c(x(r(\theta), \theta), y_j)\|} \sqrt{r^2(\theta) + (r'(\theta))^2} d\theta$$

(19)

where  $[\theta_0, \theta_1]$  is an interval corresponding to a smooth portion of the boundary  $A(y_i) \cap A(y_j)$  and the summation is done along all smooth arcs making up  $A_{ij}$  (recall

**Table 1** Iterations refer to Newton’s steps, Difference is the  $\infty$ -norm of the difference between analytic and divided differences Hessians

Example	Analytic Eq. 13		Forward Difference			Centered Difference		
	Iterations	Time	Iterations	Time	Difference	Iterations	Time	Difference
(E1)	2	6.2872	2	9.5086	$2.5724 * 10^{-9}$	2	13.268	$1.1513 * 10^{-10}$
(E2)	6	38.559	6	66.805	$5.3836 * 10^{-8}$	6	96.908	$2.5479 * 10^{-8}$
(E3)	2	8.8570	2	17.115	$2.6139 * 10^{-8}$	2	26.294	$2.4851 * 10^{-10}$
(E4)	3	23.908	3	61.79	$1.2420 * 10^{-7}$	3	109.24	$1.4503 * 10^{-7}$

Fig. 2 and Fig. 4). Note that, just like in Section 3.1.2,  $r'(\theta)$  can be obtained using the Implicit Function theorem on the boundary equation  $F(r(\theta), \theta) = 0$ , where  $F$  is given in Eq. (17), that is:  $r_\theta = -\frac{F_\theta}{F_r}$ . The 1-d integrals in Eq. (19) are again computed by the composite adaptive Simpson's rule.

Of course, due to the symmetric structure of the Hessian matrix, and the special form of its diagonal (see Eq. (13)), only the strictly upper-triangular part of  $H$  needs to be computed. Finally, the Hessian will generally be a very sparse matrix (which is especially relevant for large number of target points), since if  $A(y_i) \cap A(y_j) = \emptyset$ , then  $\frac{\partial^2 \Phi}{\partial w_i \partial w_j} = 0$ .

## 4.2 Taking a Newton step: solving a system with $H$

Assume we have a (guess)  $w^0$ , satisfying the feasibility condition Eq. 2.32 and also such that its component add up to 0.

Now, a typical Newton step would read

Given  $w^0$ , form  $H(w^0)$  solve for  $s$   $H(w^0)s = -\nabla \Phi(w^0)$ , update  $w^1 = w^0 + s$ .

However, there are at least two obvious issues that prevent implementing this step as naively as above.

For one thing,  $H(w^0)$  is singular, see Theorem 2.36. As said before, this is a reflection of the fact that every feasible vector  $w$  can be shifted by a multiple of  $e$  without changing the associated Laguerre cells  $A(y_i)$ 's. Singularity of the Hessian has of course been recognized by those working with Newton's method, though dealt with in different ways; for example, in [29] the authors use the pseudo-inverse of  $H$ , and in [38] an approach is adopted by holding at 0 one component of  $w$  and eliminating the row and column of that index from  $H$ . However, we favor an approach that exploits

that  $H$  has a 1-dimensional nullspace spanned by the vector  $e = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in \mathbb{R}^N$ . So, we select  $w^0$  so that its components add up to 0, and we will keep this property during Newton iteration.

To achieve the above, our approach is to seek the update  $s \in \mathbb{R}^N$  by "solving"  $H(w^0)s = \nabla \Phi(w^0)$  restricting the solution for  $s$  in the (orthogonal) complement to  $e$ . In formulas, we form once and for all the orthogonal matrix  $U$  below (which is independent of  $w$ )

$$U = [q \ Q] \text{ with } q = \frac{1}{\sqrt{N}}e, \ Q = \{I_N - uu^T\}_{\{2,3,\dots,N\}} \in \mathbb{R}^{N \times N-1},$$

$$\text{and where } u = \frac{1}{\sqrt{\sqrt{N}-1}} \left( \frac{1}{\sqrt{N}}e - e_1 \right) \in \mathbb{R}^N. \quad (20)$$

Then, letting  $H^0 := H(w^0)$ , and  $b^0 = -\nabla\Phi(w^0)$ , we rewrite the problem as

$$H^0 s = b^0 \rightarrow (U^T H^0 U)(U^T s) = U^T b^0 \rightarrow \begin{bmatrix} 0 & 0 \\ 0 & Q^T H^0 Q \end{bmatrix} U^T s = \begin{bmatrix} q^T b \\ Q^T b \end{bmatrix},$$

since  $U^T H(w^0) = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}$  and  $H^0 = (H^0)^T$ . Therefore, we must have  $q^T b = 0$  and thus

$$\begin{bmatrix} 0 & 0 \\ 0 & Q^T H^0 Q \end{bmatrix} \begin{bmatrix} q^T s \\ Q^T s \end{bmatrix} = \begin{bmatrix} 0 \\ -Q^T \nabla\Phi(w^0) \end{bmatrix}$$

from which multiplying both sides by the invertible matrix  $\begin{bmatrix} 1 & 0 \\ 0 & (Q^T H^0 Q)^{-1} \end{bmatrix}$ , we obtain  $\begin{bmatrix} q^T \\ Q \end{bmatrix} s = \begin{bmatrix} 0 \\ -[Q^T H^0 Q]^{-1} Q^T \nabla\Phi(w^0) \end{bmatrix}$  and eventually our update  $s$  is given by

$$s = -(H^0)^I \nabla\Phi(w^0), \text{ where } (H^0)^I = Q[Q^T H^0 Q]^{-1} Q^T. \quad (21)$$

Since  $q^T s = 0$ , then  $e^T s = 0$ , and so the components of  $w^1 = w^0 + s$  indeed will add up to 0.

#### 4.2.1 Conditioning of semi-discrete OT problems

The second concern is that the updated shift values needs to be feasible, in other words, see Theorem 2.32, they must satisfy the condition

$$\min_{i,j} c(y_i, y_j) - |w_i^1 - w_j^1| > 0, \quad \forall i, j = [1, N], i \neq j. \quad (22)$$

If this is satisfied, then the step is accepted, otherwise we reduce by a factor of  $1/2$  the update  $s$  until the criterion Eq. 22 is satisfied (similar to damped Newton method, see below)

$$s \leftarrow s/2, \quad w^1 \leftarrow w^0 + s.$$

**Lemma 4.1** *With feasible  $w^0$ , the above updating strategy will eventually lead to a feasible  $w^1$ .*

**Proof** We rewrite the feasibility requirement in matrix form as the component-wise inequality  $|Aw| < b$ , where the matrix  $A \in \mathbb{R}^{N(N-1)/2 \times N}$  and the vector

$b \in \mathbb{R}^{N(N-1)/2}$  are given by

$$A = \begin{bmatrix} 1 & -1 & 0 & 0 & \cdots & 0 \\ 1 & 0 & -1 & 0 & \cdots & 0 \\ 1 & 0 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -1 \\ 0 & 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \end{bmatrix}, \quad b = \begin{pmatrix} c(y_1, y_2) \\ c(y_1, y_3) \\ c(y_1, y_4) \\ \vdots \\ c(y_1, y_N) \\ c(y_2, y_3) \\ c(y_2, y_4) \\ \vdots \end{pmatrix}$$

According to the strategy above, consider  $w^1 = w^0 + \frac{1}{2^k}s$ , where  $s = -H(w^0)^T \nabla \Phi(w^0)$ , and recall that  $w^0$  is feasible (hence, it satisfies Eq. 22). We claim that  $w^1$  is guaranteed to be feasible for  $k = \max \{0, \lceil \max(\log_2 |As| - \log_2 |\epsilon|) \rceil\}$ , where  $\epsilon = b - |Aw^0|$ . In fact:

$$|A(w^0 + \frac{1}{2^k}s)| = |Aw^0 + \frac{1}{2^k}As| \leq |Aw^0| + \frac{1}{2^k}|As| = b - \epsilon + \frac{1}{2^k}|As| \quad \text{and so}$$

$$\frac{1}{2^k}|As| - \epsilon < 0 \iff |As| < 2^k\epsilon \iff \log_2 |As| < k + \log_2 \epsilon \iff k > \max(\log_2 |As| - \log_2 |\epsilon|)$$

as claimed.  $\square$

**Remark 4.2** *Of course, it is possible that  $w^0$ , albeit feasible, is far from the solution of the nonlinear system; in this case, it is possible that the neighborhood of feasibility of  $w^0$  is small and that to obtain  $w^1$  by the above updating strategy will lead to very tiny steps, and ultimately numerical failure. If this is the case, there are effectively two possibilities: (i) we should give an improved initial guess  $w^0$ , or (ii) the problem is inherently ill-conditioned, in that the neighborhood of feasibility of the exact solution  $w$  is just too small, and in finite precision one cannot obtain arbitrary accuracy. Below, we elaborate on both of these aspects.*

Motivated by 22 and Remark 4.2, we introduce the *feasibility coefficient* relative to a given feasible  $w$ . Namely, we let it be the quantity:

$$\kappa(w) = \min_{i,j} \left[ 1 - \frac{|w_{ij}|}{c(y_i, y_j)} \right], \quad (23)$$

and of course  $\kappa(w)$  has to be positive. In our computations, we always monitor this quantity  $\kappa(w)$  (during and at the end of the Newton iterations), and we have consistently observed that smaller values of  $\kappa(w)$  correspond to more difficult problems; e.g., see Example 5.4. We also note that  $\kappa(w)$  is a scaling-invariant quantity (i.e., it is not changed by uniformly dilating or restricting the domain  $\Omega$ , since  $w$  rescales in the same way as the cost), and it only depends on the problem itself, that is the location of the target points, the cost function, and the value of  $w$ , which ties together the continuous and discrete probability densities. Thus, our proposal is:

**Regard  $1/\kappa$  as condition number of semi-discrete optimal transport problems.**

### 4.3 Getting an initial guess

We have implemented and experimented with several different strategies to obtain a good and feasible initial guesses for solving  $\nabla \Phi(w) = 0$ .

- (i) *Trivial initial guess.* This is the simplest strategy, we take  $w^0 = 0$  as an initial guess. This corresponds to a classical Voronoi diagram (proximity problem), and we call  $V_i$  the resulting value of  $\mu(A(y_i))$  obtained when using  $w = 0$ . In spite of its simplicity, this approach often worked remarkably well, especially in conjunction with damped Newton's method (see below).
- (ii) *Grid Based.* This technique is based on the work [32], and it is quite similar to the well known Lloyd algorithm. First, consider a grid of meshsize  $h = (b - a)/N_h$  to discretize  $\Omega$ , so that there are  $N_h^2 = \frac{(b-a)^2}{h^2}$  squares on the grid. Compute the coordinates of the  $N_h^2$  centers of these squares. For given shift values  $w$ , approximate  $\mu(A(y_i))$  as

$$\mu(A(y_i)) \approx \sum_{l=1}^{M_i} \rho(x_l) h^2,$$

where  $x_l, l = 1, 2, \dots, M_i$ , are the square centers which are contained in the Laguerre cell  $A(y_i)$ . Then, using this approximation we can adjust the values  $\mu(A(y_i))$  to satisfy  $\mu(A(y_i)) = v_i$  up to some tolerance by changing shift values. For example, if the approximated  $\mu(A(y_i))$  value is greater (smaller) than the required  $v_i$  value, then we force a decrease (increase) in the value of  $\mu(A(y_i))$  by increasing (decreasing)  $w_k, \forall k \neq i$ ; we do this by the same increment for all  $k \neq i$ . Note that the adjustment of  $\mu(A(y_i))$  is done one index at a time. In practice, we choose the tolerance to be the area of a square,  $|\mu(A(y_i)) - v_i| < h^2$ , and an initial increment to be  $\sqrt{h}$ . Depending on the problem, the desired tolerance might not be reachable for all  $\mu(A(y_i))$  values. Then, one can either increase the maximum number of allowed iterations, or decrease the mesh size  $h$ . See Algorithm 11, *GridInitialGuess*, and Algorithm 12, *GridMeasure*, for implementation details. The next two approaches are classical homotopy techniques, which we implemented as commonly done when solving nonlinear systems (e.g., see [28]).

- (iii) *Trivial Homotopy.* We want to solve  $g_1(w(\alpha)) = 0$ , where the components of  $g_1$  are given by

$$(g_1(w))_i = \mu(A(y_i)) - [(1 - \alpha)\mu(V_i) + \alpha v_i], \quad i = 1, \dots, N, \quad 0 \leq \alpha \leq 1.$$

At  $\alpha = 0, w = 0$  is a solution, which corresponds to the Voronoi diagram  $\{V_i\}$  (see (i)), and at  $\alpha = 1$  we obtain the problem we are trying to solve. We observe that the problem can be viewed, for each  $\alpha$ , as finding  $w$  such that

$$\mu(A(y_i)) = \hat{v}_i \equiv \alpha v_i + (1 - \alpha)\mu(V_i), \quad i = 1, \dots, N,$$

that is we have a semi-discrete problem where the target density is given by  $\sum_i \delta(y_i) \hat{v}_i$ , which –by virtue of Theorem 2.31– is solvable giving a solution

**Table 2** Different strategies to give an initial guess for Newton's method

Example	Grid Based $h = 0.05$		Trivial Homotopy		Non-Singular Homotopy	
	Iterations	Time	Nsteps, Iterations	Time	Nsteps, Iterations	Time
(E1)	2	7.6242	6, 16	50.767	7, 22	66.755
(E2)	7	40.951	7, 24	126.39	8, 31	148.63
(E3)	2	9.0191	5, 11	44.867	6, 18	68.335
(E4)	3	22.317	6, 19	154.22	7, 25	186.74

$w(\alpha)$  (unique up to adding a constant to all entries)<sup>3</sup>. We use Newton's method to solve from  $\alpha = 0$  to 1 choosing the steps adaptively beginning with  $\Delta^0 = 1/10$  (or  $1/100$ ) and updating it based on the number of Newton iterations  $k$  required for convergence at the previous step:

$$\Delta^i = 2^{\frac{4-k}{3}} \Delta^{i-1}, \quad \alpha^i = \alpha^{i-1} + \Delta^i.$$

We note that the Hessian resulting from using Newton's method is always the one in Eq. (13), hence it is always singular along the homotopy path, with null vector given by  $e$ .

- (iv) *Non-Singular Homotopy*. Here, we solve  $g_2(w(\alpha)) = 0$ , where the components of  $g_2$  are

$$(g_2(w(\alpha)))_i = (1 - \alpha)w + \alpha(\mu(A(y_i)) - v_i), \quad i = 1, \dots, N, \quad 0 \leq \alpha \leq 1.$$

At  $\alpha = 0$ , we have the solution  $w = 0$ , and for  $\alpha > 0$ , we can look at the problem as finding  $w$  such that  $\sum w_i = 0$  and

$$\mu(A(y_i)) = \hat{v}_i \equiv v_i - \frac{1 - \alpha}{\alpha} w_i, \quad i = 1, \dots, N, \quad 0 < \alpha \leq 1;$$

however, although  $\sum_i \hat{v}_i = 1$ , we now have no guarantee that  $\hat{v}_i > 0$ , and in fact we occasionally had difficulties with this homotopy choice precisely because of this fact. Otherwise, for as long as  $\hat{v}_i > 0$ , we can use Newton's method to solve for  $w(\alpha)$  from 0 to 1, and a main advantage now is that the Hessian (Jacobian of  $g_2$  with respect to  $w$ ) is non-singular for  $\alpha \in [0, 1)$ , since it is  $(1 - \alpha)I + \alpha H(w(\alpha))$  and  $H(\alpha)$  is positive semi-definite from Theorem 13.

On the same four problems (E1)-(E4) of Section 5, in Table 2 we show relative performance of the above strategies to provide an initial guess  $w^0$ . For the homotopy techniques, "Nsteps" refer to the number of homotopy steps, and "Iterations" to the total number of Newton's iterations. Clearly, the "Grid Based" approach worked much better, and for this reason all results we report on in Section 5 were obtained by using the "Grid Based" approach to provide the initial guess  $w^0$ .

<sup>3</sup> This homotopy path is the counterpart to what was used in [33], where the homotopy was done to move from an "easy" source density to the desired source density.



A standard modification of Newton's method is the well known "Damped Newton's Approach". This can be viewed as a mean by which we improve on a feasible initial guess  $w^0$ , e.g.  $w^0 = 0$ , to bring it closer to the solution of  $\nabla \Phi(w) = 0$ . As a general technique, damped Newton's is routinely used in solving nonlinear systems, and was also used in [29] for solving semi-discrete optimal transport problems for the 2-norm square cost. Below, we simply give its justification for our case of singular Hessian, particularly given the way we find the Newton's update.

### 4.3.1 Damped Newton Method

When one has to solve a nonlinear system  $F(w) = 0$ , and the Jacobian at  $w^0$  is invertible, then it is well known that the Newton direction is a direction of descent for the function  $g(w) := \|F(w)\|_2^2 = F(w)^T F(w)$ , in the sense that  $g(w^0 + \alpha s) < g(w^0)$  for some  $0 < \alpha \leq 1$  and  $s$  is the solution of  $DF(w^0)s = -F(w^0)$ . Below, we show that also in our case the Newton direction from Eq. (21) is an appropriate direction of descent in the subspace complementary to the null-space of  $H$ .

**Lemma 4.3** *If  $\nabla \Phi(w) \neq 0$ , then the direction  $s$  given by Eq. (21) is a direction of descent for the functional given by*

$$g(w) = (Q^T \nabla \Phi(w))^T (Q^T \nabla \Phi(w)), \quad (24)$$

where  $U = [q \ Q]$  is the orthogonal matrix defined in Eq. (20). That is, there exists  $0 < \alpha \leq 1$  such that  $g(w^0 + \alpha s) < g(w^0)$ .

**Proof** Expand:

$$\begin{aligned} g(w + \alpha s) &= g(w) + \alpha (\nabla g(w))^T s + \mathcal{O}(\alpha^2) \\ &= g(w) + 2\alpha (\nabla \Phi(w))^T Q Q^T H(w) s + \mathcal{O}(\alpha^2). \end{aligned}$$

Then, using Eq. (21), we have

$$\begin{aligned} g(w + \alpha s) - g(w) &= -2\alpha (\nabla \Phi(w))^T Q (Q^T H(w) Q) (Q^T H(w) Q)^{-1} Q^T \nabla \Phi(w) \\ &\quad + \mathcal{O}(\alpha^2) = -2\alpha g(w) + \mathcal{O}(\alpha^2). \end{aligned}$$

Therefore, the claim follows if  $g(w) > 0$ . Next, we show that  $g(w) = 0$  only if  $\nabla \Phi(w) = 0$  and the result will follow. In fact:

$$g(w) = 0 \iff Q^T \nabla \Phi(w) = 0 \iff \nabla \Phi(w) = ce, \quad c \in \mathbb{R},$$

since the kernel of  $Q^T$  is 1-dimensional and spanned by  $e$ . Now, since  $\nabla \Phi(w) = (\mu(A(y_i)) - v_i)_{i=1, \dots, N}$ , then  $\sum_{i=1}^N (\nabla \Phi(w))_i = 0$  and therefore we must have  $c = 0$ .  $\square$

Lemma 4.3 justifies use of a full damped Newton's method to solve the problem  $\nabla \Phi(w) = 0$ . And, in the end, using damped Newton's method revealed to be the

best method (among those we implemented) to perform the full iteration in order to solve  $\nabla \Phi(w) = 0$ . Namely, we always attempt to take the full Newton step, and damp it –if needed– to force a decrease in the functional  $g(w)$  in Eq. (24). Details of our implementation are provided in Algorithm 13, *DampedNewton*. That said, most of the time we found no need to damp the Newton step, see Section 5.

## 5 Computational Examples

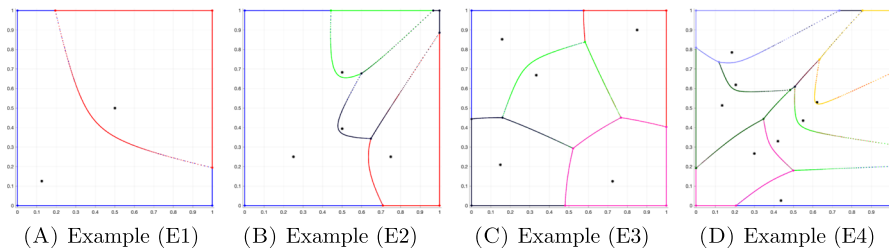
Here we present results of our algorithms. We pay special attention to give Examples that can be replicated by other approaches, and give quantitative results, rather than just pictures.

All of our computations were performed with `Matlab`. The key quantities we monitor are the **Error**, which is  $\|\nabla \Phi(w)\|_\infty$  at convergence, the number of Newton **Iterations** and further the number of **Damped** steps if required, the execution **Time**, and the **Feasibility Coefficient**  $\kappa$  as needed. The key quantities by which we control accuracy are two tolerance values, one to control accuracy of the computation of  $\mu(A(y_i))$  (and, therefore of the gradient  $\nabla \Phi$ ), the other to measure convergence of the Newton's process, as assessed by either  $\|\nabla \Phi\|_\infty$  or the  $\infty$ -norm of the Newton's updates. The default values are  $10^{-12}$  for the approximation of  $\mu(A(y_i))$  and  $10^{-8}$  for Newton's convergence.

The next four examples have been used to produce Tables 1 and 2 as well as Table 3 below. They are fairly easy problems, mostly used for testing purposes. For all of them, the domain is  $\Omega = [0, 1] \times [0, 1]$ , the cost is the 2-norm,  $c(x, y) = \|x - y\|_2$ , and  $\rho$ ,  $v$ , and the locations  $y_i$ 's are given below. See Fig. 6 below for their solution.

$$\rho(x) = 1, \quad v = \frac{1}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad y = \left\{ \begin{pmatrix} 0.125 \\ 0.125 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} \right\}, \quad (E1)$$

$$\rho(x) = 4x_1x_2, \quad v = \frac{1}{4} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \quad y = \left\{ \begin{pmatrix} 0.25 \\ 0.25 \end{pmatrix}, \begin{pmatrix} 0.75 \\ 0.25 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.25(1 + \sqrt{3}) \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.25(1 + \frac{\sqrt{3}}{3}) \end{pmatrix} \right\}. \quad (E2)$$



**Fig. 6** Solution for Examples (E1)–(E4)

Table 3 Different Methods

Example	Our method			Boundary method of [14]			Quasi-Newton (BFGS) using fminunc		
	Error	Iterations	Time	Error	Grid size	Time	Error	Iterations	Time
(E1)	$4.1959 * 10^{-11}$	2	7.9786	$7.0657 * 10^{-6}$	$2^{-15}$	1.01	$1.0365 * 10^{-2}$	1	257.69
(E2)	$1.5987 * 10^{-9}$	6	37.807	$7.8626 * 10^{-6}$	$2^{-18}$	34.5	$2.9096 * 10^{-2}$	7	431.56
(E3)	$7.2028 * 10^{-6}$	1	6.2608	$5.3557 * 10^{-6}$	$2^{-16}$	6.06	$1.1224 * 10^{-3}$	3	227.83
(E4)	$3.3980 * 10^{-7}$	2	16.170	$8.3461 * 10^{-7}$	$2^{-20}$	772	$7.6671 * 10^{-3}$	1	191.07

$$\rho(x) = 1, \quad v = \frac{1}{5} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \quad y = \frac{1}{4096} \left\{ \begin{pmatrix} 646 \\ 3491 \end{pmatrix}, \begin{pmatrix} 3480 \\ 3686 \end{pmatrix}, \begin{pmatrix} 1364 \\ 2737 \end{pmatrix}, \begin{pmatrix} 609 \\ 857 \end{pmatrix}, \begin{pmatrix} 2967 \\ 509 \end{pmatrix} \right\}. \quad (\text{E3})$$

$$\rho(x) = 1, \quad v = \frac{1}{8} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \quad y = 8 \text{ random uniform points using seed } \text{rng}(2). \quad (\text{E4})$$

**Example 5.1** In Table 3, we compare performance of our method with two other techniques, the “boundary method” of [14] and the minimization approach via use of a quasi-Newton’s method as in [24]. In the former case, the results are those obtained by the C++-code kindly provided by J.D. Walsh and the **Time** in this case refers to that of the C++-code, neglecting the final time required to measure the **Error** since the final areas computation is done by our approach. In the case of the BFGS quasi-Newton method we used the tried and true Matlab routine `fminunc`. First of all, with the boundary method we could not reach the required accuracy of  $\text{TOL} = 10^{-8}$  and thus had to lower that to  $10^{-5}$ , and even so we had to decrease considerably the minimum allowed grid-size. Secondly, with BFGS we could never obtain answers more accurate than a couple of digits and the code `fminunc` always returned a message that was not able to achieve further reduction in the error.

**Example 5.2** (Example with Different Source Densities) Here we keep  $\Omega = [0, 1] \times [0, 1]$ , the cost is  $c(x, y) = \|x - y\|_2$ , the target points are located at  $\left\{ \begin{pmatrix} 0.25 \\ 0.25 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.75 \end{pmatrix}, \begin{pmatrix} 0.75 \\ 0.25 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.3 \end{pmatrix} \right\}$ ,  $v$  is uniform, and we take four different source densities to illustrate the impact of  $\rho$  on the overall error and execution time. The source densities are

$$\rho_1(x) = 1, \text{ uniform}, \quad \rho_2(x) = 4x_1x_2, \text{ non-uniform}, \quad \rho_3(x) = \gamma e^{-10(x_1-0.5)^2-10(x_2-0.5)^2}, \text{ Gaussian}$$

$$\rho_4(x) = \begin{cases} \frac{1}{2} & 0 \leq x_1 \leq 0.3, \forall x_2 \\ g(x) & 0.3 < x_1 < 0.7, \forall x_2 \text{ non smooth, smoothed out} \\ \frac{3}{2} & 0.7 \leq x_1 \leq 1, \forall x_2 \end{cases}$$

where  $\gamma$  is the normalization constant and

$$g(x_1, x_2) = \frac{1}{2} + \frac{(500x_1(4x_1(175(x_1-3)x_1+594)-1203)+115173)(10x_1-3)^5}{131072}.$$

**Table 4** Example with Different Source Densities

Source Density	Error	Iterations	Damping	Time	Feasibility Coefficient $\kappa$
Uniform	$3.7612 \times 10^{-9}$	2	0	12.806	0.45594
Non-Uniform	$3.4750 \times 10^{-14}$	7	2	46.448	0.13112
Gaussian	$7.3751 \times 10^{-11}$	4	0	53.691	0.66334
Non smooth	$4.9682 \times 10^{-14}$	6	0	119.99	0.34405

We notice that  $g(x)$  is obtained as a  $C^4$  interpolant on a non-smooth density given by  $1/2$  before  $x_1 = 1/2$  and by  $3/2$  past it. For all problems, the initial guess is obtained with grid size  $h = 0.05$ .

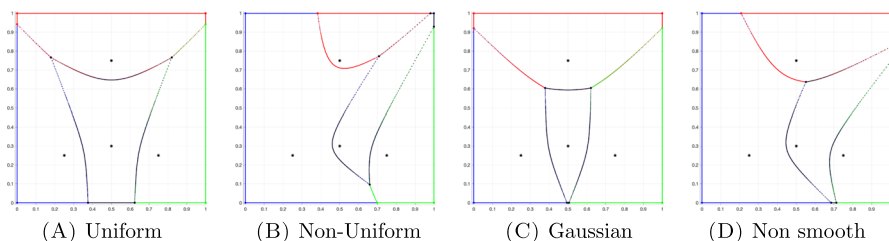
Clearly, as it can be seen in Table 4 and Fig. 7, the non-smooth case is the most computationally expensive due to the very large values of the derivatives of  $\rho$ .

**Example 5.3** (Different  $p$ -norms). Here we want to assess the impact of different  $p$ -norms for the cost, in particular we are interested in assessing the impact of using odd value of  $p$  (which gives reduced smoothness with respect to the even values of  $p$ ) and of “approaching” the  $\infty$ -norm and the 1-norm. Although with costs given by the  $\infty$ -norm and the 1-norm the problem is not well posed (see Example 2.29), we will see that as we approach these values the algorithm selects clearly defined Laguerre tessellations, and we conjecture that this is a typical scenario and deserves further study. (Incidentally, also for Example 2.29, the approach just outlined select a well defined tessellation).

We fix  $\Omega = [0, 1] \times [0, 1]$ ,  $y = \left\{ \begin{pmatrix} 0.25 \\ 0.25 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 0.75 \end{pmatrix}, \begin{pmatrix} 0.75 \\ 0.25 \end{pmatrix} \right\}$ , take  $\rho$  and  $v$  uniform, and we consider the following cost functions: Cost function:

- (1)  $c(x, y) = \|x - y\|_3$ ;  $c(x, y) = \frac{1}{2}(\|x - y\|_2 + \|x - y\|_4)$ ;  $c(x, y) = \|x - y\|_3 + \|x - y\|_5 + \|x - y\|_7$ ;
- (2) toward the  $\infty$ -norm:  $c(x, y) = \|x - y\|_{2^k}$ ,  $k = 1, 2, 3, 4, 5$ ;
- (3) toward the 1-norm:  $c(x, y) = \|x - y\|_{1+2^{-k}}$ ,  $k = 1, 2, 3, 4, 5$ .

All results obtained with initial guess from the grid based approach with  $h = 0.05$ . No appreciable difference is seen with the above three different costs, as confirmed by Table 5 and Fig. 8.


**Fig. 7** Example with Different Source Densities

**Table 5** Example with Different  $p$ -Norm Cost Functions

Cost Function	Error	Iterations	Damping	Time	Feasibility Coefficient $\kappa$
$\ \cdot\ _3$	$1.9444 \times 10^{-10}$	2	0	8.6622	0.74508
$\frac{1}{2}(\ \cdot\ _2 + \ \cdot\ _4)$	$1.0988 \times 10^{-12}$	2	0	10.756	0.74652
$\ \cdot\ _3 + \ \cdot\ _5 + \ \cdot\ _7$	$8.1481 \times 10^{-10}$	2	0	10.742	0.74023

In Table 6, we observe that for larger values of  $p$ , computation of the integrals becomes expensive because of large derivative values. Fig. 9 clearly shows the decrease in smoothness.

Also, for costs approaching 1-norm, as it can be seen in Table 7, there is a clear increase in the computation time, again due to the decreased smoothness of the boundary. Fig. 10 shows this clearly.

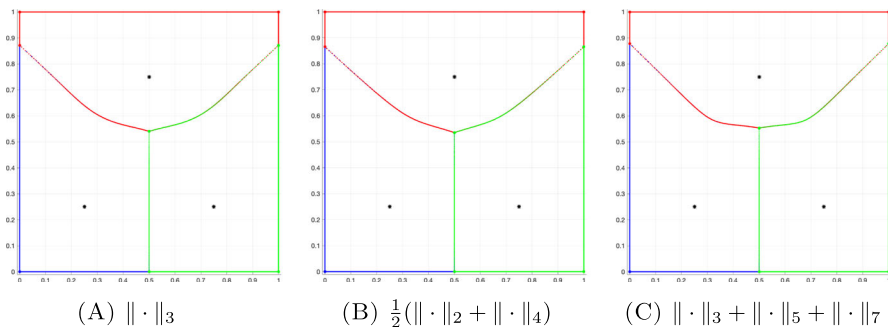
It is very interesting that the value of  $\kappa$  is effectively the same across all of the above experiments, and does not betray ill-conditioning of the problem. This is reflected in the fact that only 2 (or occasionally 3) Newton iterations are required for all of the above costs, and that the value of the Error is small. The difficulties here are entirely due to the decrease in smoothness of the boundary.

**Example 5.4 (Impact of Feasibility)** This is a difficult problem for all methods we tested, in that it becomes arbitrarily ill-conditioned. We have

$$\Omega = [0, 1] \times [0, 1], \quad c(x, y) = \|x - y\|_2, \quad y = \left\{ \begin{pmatrix} 0.25 \\ 0.5 \end{pmatrix}, \begin{pmatrix} 0.75 \\ 0.5 \end{pmatrix} \right\}, \quad \rho(x) = 1,$$

and a non-uniform target density  $v = (\frac{1}{2^k}, 1 - \frac{1}{2^k})$ ,  $k = [1, \dots, 10]$ . Results in Table 8 are all obtained with initial guess from the grid-based approach with  $h = 0.1$ . The tessellation is shown in Fig. 11.

The increase in time is entirely due to the difficulty in computing  $\mu(A(y_i))$ 's. We also note that the initial guess for  $k \geq 7$  is just not good, as reflected in the increase in Newton's iterations and the need for damped Newton for  $k = 10$ .

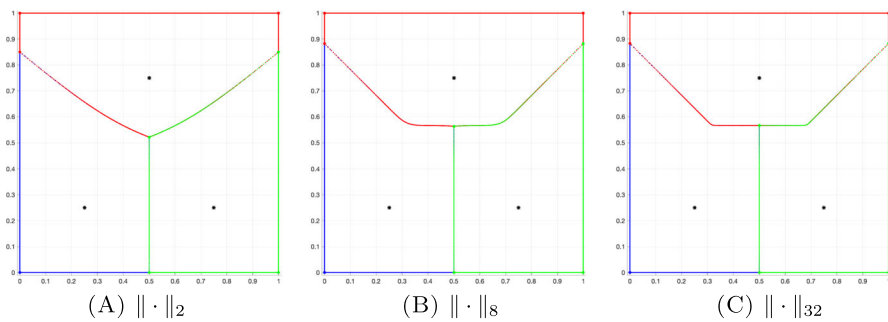
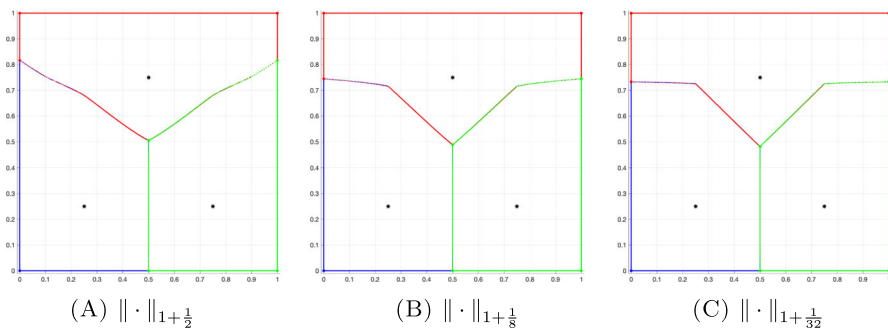

**Fig. 8** Different  $p$ -norm costs

**Table 6** Approaching the  $\infty$ -norm. Cost is  $\|x - y\|_{2^k}$ 

	Error	Iterations	Damping	Time	Feasibility Coefficient $\kappa$
$k = 1$	$1.5999 * 10^{-10}$	2	0	6.539	0.74940
$k = 2$	$7.9542 * 10^{-10}$	2	0	7.6051	0.74083
$k = 3$	$4.1729 * 10^{-9}$	2	0	7.8891	0.73576
$k = 4$	$7.0427 * 10^{-9}$	2	0	8.1167	0.73452
$k = 5$	$6.1689 * 10^{-11}$	3	0	320.97	0.73414

**Table 7** Approaching the 1-norm. Cost is  $\|x - y\|_{1+2^{-k}}$ 

	Error	Iterations	Damping	Time	Feasibility Coefficient $\kappa$
$k = 1$	$5.1627 * 10^{-9}$	2	0	46.634	0.74426
$k = 2$	$1.6436 * 10^{-11}$	3	0	214.18	0.73291
$k = 3$	$3.0620 * 10^{-9}$	2	0	165.3	0.7261
$k = 4$	$3.4457 * 10^{-10}$	2	0	249.13	0.72406
$k = 5$	$2.6122 * 10^{-11}$	2	0	283.11	0.72312


**Fig. 9**  $p$ -norm costs for growing  $p$ 

**Fig. 10**  $p$ -norm costs for  $p$  decreasing toward 1

**Table 8** Impact of  $\kappa$  on Newton iterations and time

Target Density	Error	Iterations	Damping	Time	Feasibility Coefficient $\kappa$
$(\frac{1}{2}, 1 - \frac{1}{2})$	$3.3307 * 10^{-14}$	0	0	2.2765	1
$(\frac{1}{52}, 1 - \frac{1}{52})$	$4.6629 * 10^{-14}$	3	0	11.363	$4.0243 * 10^{-1}$
$(\frac{1}{2}, 1 - \frac{1}{25})$	$5.6483 * 10^{-14}$	2	0	9.6461	$2.0029 * 10^{-1}$
$(\frac{1}{24}, 1 - \frac{1}{24})$	$2.9511 * 10^{-10}$	2	0	10.27	$7.9527 * 10^{-2}$
$(\frac{1}{25}, 1 - \frac{1}{25})$	$4.1078 * 10^{-14}$	4	0	17.766	$2.4611 * 10^{-2}$
$(\frac{1}{26}, 1 - \frac{1}{26})$	$9.0354 * 10^{-10}$	3	0	20.834	$6.6039 * 10^{-3}$
$(\frac{1}{27}, 1 - \frac{1}{27})$	$2.7367 * 10^{-11}$	9	0	59.625	$1.6834 * 10^{-3}$
$(\frac{1}{28}, 1 - \frac{1}{28})$	$2.7842 * 10^{-12}$	10	0	120.6	$4.2294 * 10^{-4}$
$(\frac{1}{29}, 1 - \frac{1}{29})$	$3.9706 * 10^{-12}$	10	0	201.91	$1.0587 * 10^{-4}$
$(\frac{1}{210}, 1 - \frac{1}{210})$	$1.2863 * 10^{-10}$	12	4	1759	$2.6475 * 10^{-5}$



**Example 5.5** *Interplay of location,  $\mu$  and  $\nu$*  Here we highlight the role played by consistency of the location of the target points with respect to the continuous and discrete densities. We fix

$$\Omega = [0, 1] \times [0, 1], \quad c(x, y) = \|x - y\|_2, \quad y = \left\{ \begin{pmatrix} 0.8 \\ 0.8 \end{pmatrix}, \begin{pmatrix} 0.8 \\ 0.9 \end{pmatrix}, \begin{pmatrix} 0.9 \\ 0.9 \end{pmatrix}, \begin{pmatrix} 0.9 \\ 0.8 \end{pmatrix} \right\},$$

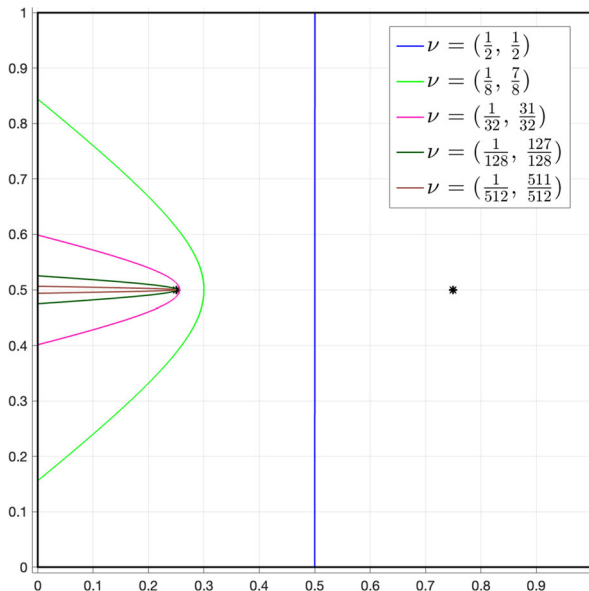
and select three different densities

- (A) Uniform source and uniform target. Here, the location of the target points is not consistent with either  $\rho$  or  $\nu$ .
- (B) Uniform source ( $\rho(x) = 1$ ) and non-uniform target ( $\nu = (0.75, 0.1, 0.05, 0.1)^T$ ). Here, the location of the target points is consistent with  $\nu$  but not with  $\mu$ .
- (C) Non-Uniform source ( $\rho(x) = 16x_1^3x_2^3$ ) and uniform target ( $\nu_i = \frac{1}{4}, \forall i$ ). Here, the location of the target points is consistent with  $\mu$  but not with  $\nu$ .

For all cases, the initial guess was obtained with the grid-based approach with  $h = 0.05$ .

The small value of  $\kappa$  in case (A) reflects the inconsistent location of the target points with respect to  $\mu$  and  $\nu$ . The high execution time for case (C), instead is entirely due to the cost of computing  $\mu(A(y_i))$ 's by our algorithm because of the change of variable we do for  $(x_1, x_2)$ , namely  $\rho(x_1, x_2) = \rho(Y_1 + D \cos(\theta), Y_2 + D \sin(\theta))$ . See Table 9 and Fig. 12.

**Example 5.6** (Different Domains) Here we show that our algorithms can be easily adapted to work with non-square domains, with no noticeable difference in performance. For  $y$  we take 10 random points in domain  $\Omega_i$ , obtained by retaining the first 10



**Fig. 11** Impact of small  $\kappa$  on the tessellation

**Table 9** Interplay between Source and Target Densities with Location of Points

Densities	Error	Iterations	Damping	Time	Feasibility Coefficient $\kappa$
Uniform to Uniform	$3.2677 * 10^{-9}$	2	0	89.095	0.02198
Uniform to Non-Uniform	$1.2056 * 10^{-12}$	3	0	45.295	0.14509
Non-Uniform to Uniform	$8.3026 * 10^{-9}$	3	0	144.78	0.86597

random points uniformly distributed in the square circumscribed to  $\Omega_i$  that fall inside  $\Omega_i$ . Further, we take  $c(x, y) = \|x - y\|_2$ ,  $\nu$  is uniform, and so is  $\mu$ :  $\rho(x) = \frac{1}{|\Omega_i|}$ ,  $\forall i$ . As domains we considered:

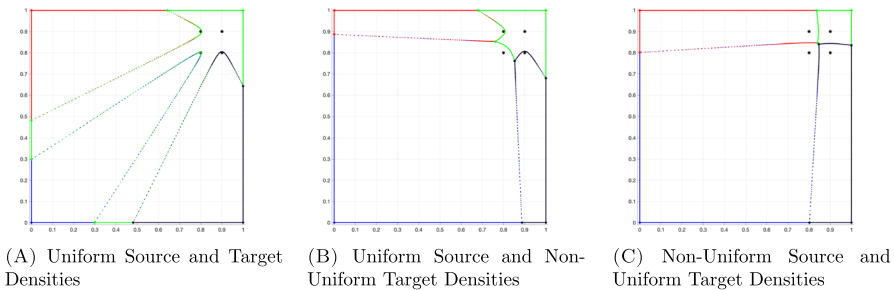
- (A)  $\Omega_1$  is the equilateral triangle centered at the origin with circumradius of 1;
- (B)  $\Omega_2$  is the regular pentagon centered at the origin with circumradius equal to 1;
- (C)  $\Omega_3$  is an irregular pentagon of centroid at (0.44239, 0.45993) and circumradius equal to 0.69813;
- (D)  $\Omega_4$  is the circle centered at the origin with radius equal to 1;

As initial guesses we take  $w^0 = 0$ .

Interestingly, the most difficult problems are those with the most regular domains, equilateral triangle and regular pentagon. However, all cases are solved fairly easily. See Table 10 and Fig. 13.

**Example 5.7** (Cost Interpolation). Here, we present an example of a problem with a (supposedly) unknown cost that will be interpolated by a model function. Consider  $\Omega = [0, 1] \times [0, 1]$  and 4 randomly generated target points  $y_i$ 's. Additionally, generate 100 random points  $x_j$ 's in  $\Omega$ , and consider the set of costs  $c_{ij}$  given by  $c_{ij} = c(x_j, y_i)$ , where

$$c(x, y) = \|x - y\|_{2.5} + 2\|x - y\|_{3.5} + 0.5\|x - y\|_{15}.$$


**Fig. 12** Interplay between Source and Target Densities with Location of Points

**Table 10** Different domains and uniform source density

Domain	Error	Iterations	Damping	Time	Feasibility Coefficient $\kappa$
Equilateral Triangle	$4.2979 * 10^{-11}$	6	0	97.79	0.07999
Regular Pentagon	$2.7436 * 10^{-10}$	6	0	130.1	0.11848
Irregular Pentagon	$5.2235 * 10^{-11}$	5	0	59.33	0.25617
Circle	$2.0969 * 10^{-11}$	4	0	22.251	0.51896

Next, define the following model for the cost:

$$k(x, y) = \alpha_1 \|x - y\|_{1.5} + \sum_{l=2}^{10} \alpha_l \|x - y\|_l.$$

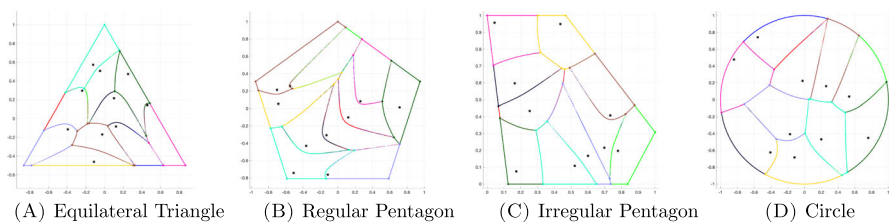
We seek  $\alpha = (\alpha_1, \dots, \alpha_{10})$  by solving the least squares problem  $\min_{\alpha} \sum_{i,j} (k(x_j, y_i) - c_{ij})^2$ , subject to the constraint  $\alpha \geq 0$ . Solving this constrained least-squares problem (we used `lsqnonneg`), we get the nonnegative least-squares solution

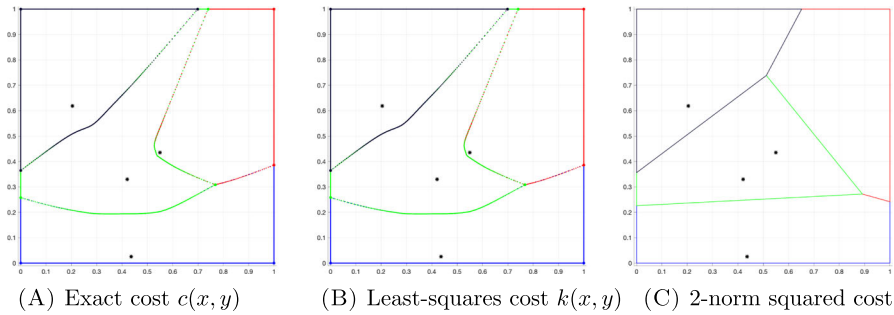
$$k(x, y) = 0.02 \|x - y\|_{1.5} + 0.03 \|x - y\|_2 + 2.72 \|x - y\|_3 + 0.73 \|x - y\|_{10}.$$

Now, let the source and target densities be uniform:  $\rho(x) = 1$  and  $v_i = \frac{1}{4}$ ,  $i = 1, 2, 3, 4$ . Then, Fig. 14A displays the partition of the domain with the exact underlying cost  $c(x, y)$ , while Fig. 14B illustrates the partition with the interpolated cost function  $k(x, y)$ . For comparison purposes, we also show the partition obtained using the cost given by  $\|x - y\|_2^2$ , see Fig. 14C. As shown in the figures, the interpolated cost produces a virtually identical partition compared to the underlying exact (but unknown) cost function, whereas the partition formed with the 2-norm squared cost function is completely wrong.

## 6 Conclusions

In this work, we presented a novel implementation of Newton's method for solving semi-discrete optimal transport problems for cost functions given by a (positive combination of)  $p$ -norm(s),  $1 < p < \infty$ , and several choices of continuous and discrete


**Fig. 13** Examples with Different domains and uniform source density



**Fig. 14** Solution of the semi-discrete problem with different cost functions

densities. To date, there appeared to have been no implementation of Newton's method for these problems, and we succeeded in making some progress by proving that the Laguerre cells providing the solution of the OT problem are star shaped with respect to the target points and by exploiting this fact in our algorithmic development. We gave a detailed description of all algorithms we implemented and provided quantitative testing on several examples, as well as comparison with competing approaches, well beyond simply showing some qualitatively faithful figures. Our algorithms proved much more robust and accurate than other methods, being based on rigorously justified adaptivity and error control, which—in principle—allows to solve the OT problem at any desired level of accuracy. The price we pay for this is the need for the continuous density to be sufficiently smooth. Future developments call for algorithms suited for 3-d, as well as for development of Newton-like methods better suited when we only have limited smoothness of the continuous density.

## Appendix

**Claim 6.1** *Let the cost function be a  $p$ -norm, where  $p$  is even:  $c(x, y) = \|x - y\|_p$ ,  $p = 2k$ ,  $k = 1, 2, \dots$ . If  $|w| < \|y_1 - y_2\|_p$ , then  $\nabla F(x_0) \neq 0$ ,  $\forall x_0$  such that  $F(x_0) = w$ , where*

$$F(x) = \|x - y_1\|_p - \|x - y_2\|_p.$$

**Proof** Assume, by contradiction, that there exists a point  $x_0$  where both  $F(x_0) = w$  and  $\nabla F(x_0) = 0$ . An explicit computation gives

$$\begin{aligned} \nabla F(x) &= \frac{1}{\|x - y_1\|_p^{p-1}} \begin{pmatrix} (x - y_1)_1^{p-1} \\ \vdots \\ (x - y_1)_d^{p-1} \end{pmatrix} - \frac{1}{\|x - y_2\|_p^{p-1}} \begin{pmatrix} (x - y_2)_1^{p-1} \\ \vdots \\ (x - y_2)_d^{p-1} \end{pmatrix} \quad \text{and so} \\ \nabla F(x_0) = 0 &\implies \frac{(x_0 - y_1)_j^{p-1}}{\|x_0 - y_1\|_p^{p-1}} = \frac{(x_0 - y_2)_j^{p-1}}{\|x_0 - y_2\|_p^{p-1}} \implies \frac{(x_0 - y_1)_j}{\|x_0 - y_1\|_p} = \frac{(x_0 - y_2)_j}{\|x_0 - y_2\|_p}, \quad j = 1, \dots, d. \end{aligned}$$

Since  $F(x_0) = 0$ , then  $\|x_0 - y_1\|_p = w + \|x_0 - y_2\|_p$  and so

$$\frac{(x_0 - y_1)_j}{w + \|x_0 - y_2\|_p} = \frac{(x_0 - y_2)_j}{\|x_0 - y_2\|_p} \implies w(x_0 - y_2)_j = \|x_0 - y_2\|_p (y_2 - y_1)_j, \quad j = 1, \dots, d.$$

Taking the  $p$ -norm of the vectors on both sides of the last relation gives

$$|w| \cdot \|x_0 - y_2\|_p = \|x_0 - y_2\|_p \|y_2 - y_1\|_p \quad \text{or} \quad |w| = \|y_2 - y_1\|_p,$$

which is a contradiction and the proof follows.  $\square$

**Lemma 6.2** *Let  $\Omega \subset \mathbb{R}^n$  satisfy Assumption 2.1,  $y_i \in \Omega^\circ$  and  $i = 1, 2, \dots, N$ , be  $N$  distinct target points, and let the shift values be feasible:  $|w_{ij}| < c(y_i, y_j)$ ,  $\forall i, j = 1, 2, \dots, N$ ,  $i \neq j$ . Consider the cost function given by a  $p$ -norm with  $p$  even. Then, there  $\exists \epsilon > 0$  such that  $\forall i, j = 1, 2, \dots, N$ ,  $i \neq j$*

$$\|\nabla_x c(x, y_i) - \nabla_x c(x, y_j)\| \geq \epsilon, \quad \forall x \in L_{ij} \cap \Omega,$$

where  $\|\cdot\|$  is the standard Euclidean distance  $\|\cdot\|_2$ .

**Proof** Firstly, we are going to show that exists a constant, which depends on  $p$ , call it  $\epsilon_{ij}(p) > 0$ , such that

$$\|\nabla_x \|x - y_i\|_p - \nabla_x \|x - y_j\|_p\|_p \geq \epsilon_{ij}(p), \quad \forall x \in L_{ij} \cap \Omega.$$

By direct computation, as in Claim 6.1, we have

$$\|\nabla_x \|x - y_i\|_p - \nabla_x \|x - y_j\|_p\|_p = \left\| \begin{pmatrix} a_1^{p-1} \\ \vdots \\ a_d^{p-1} \end{pmatrix} - \begin{pmatrix} b_1^{p-1} \\ \vdots \\ b_d^{p-1} \end{pmatrix} \right\|_p \quad \text{where}$$

$$a_j = \frac{(x - y_1)_j}{\|x - y_1\|_p}, \quad b_j = \frac{(x - y_2)_j}{\|x - y_2\|_p}, \quad j = 1, \dots, d.$$

Now, abusing of notation, we will write  $a^{p-1}$  for the Hadamard product of the vector  $a$  with itself  $(p-1)$ -times, and thus  $a^{p-1} - b^{p-1} = (a - b) \circ (a^{p-2} + a^{p-3} \circ b + \dots + b^{p-2})$ , where " $\circ$ " is the Hadamard product<sup>4</sup>. Next, using Claim 6.3 we have

$$\left\| \frac{(x - y_i)^{p-1}}{\|x - y_i\|_p^{p-1}} - \frac{(x - y_j)^{p-1}}{\|x - y_j\|_p^{p-1}} \right\|_p \geq \Delta \left\| \frac{(x - y_i)}{\|x - y_i\|_p} - \frac{(x - y_j)}{\|x - y_j\|_p} \right\|_p.$$

<sup>4</sup>  $a \circ b = \begin{pmatrix} a_1 b_1 \\ \vdots \\ a_d b_d \end{pmatrix}$

Using that  $x \in L_{ij} \cap \Omega \implies \|x - y_i\|_p = \|x - y_j\|_p + w_{ij}$ :

$$\begin{aligned} \frac{|||x - y_j||_p(x - y_i) - \|x - y_i\|_p(x - y_j)||_p}{\|x - y_i\|_p\|x - y_j\|_p} &= \frac{|||x - y_j||_p(x - y_i) - (\|x - y_j\|_p + w_{ij})(x - y_j)||_p}{\|x - y_i\|_p\|x - y_j\|_p}, \\ \implies \frac{|||x - y_j||_p(y_j - y_i) - w_{ij}(x - y_j)||_p}{\|x - y_i\|_p\|x - y_j\|_p} &\geq \frac{|||x - y_j||_p\|y_j - y_i\|_p - |w_{ij}||x - y_j\|_p|}{\|x - y_i\|_p\|x - y_j\|_p}, \\ &\implies \|\nabla_x\|x - y_i\|_p - \nabla_x\|x - y_j\|_p\|_p \\ &\geq \Delta \frac{|||y_j - y_i\|_p - |w_{ij}||}{\|x - y_i\|_p} \end{aligned}$$

Using that  $x \in \Omega \implies \|x - y_i\|_p \leq C(y_i, \Omega, p)\text{diam}(\Omega) \implies \frac{1}{\|x - y_i\|_p} \geq \frac{1}{C(y_i, \Omega, p)\text{diam}(\Omega)}$ , where  $C(y_i, \Omega, p) > 0$  is a constant depending on  $y_i, \Omega$  and  $p$ , and so:

$$\|\nabla_x\|x - y_i\|_p - \nabla_x\|x - y_j\|_p\|_p \geq \Delta \frac{|||y_j - y_i\|_p - |w_{ij}||}{C(y_i, \Omega, p)\text{diam}(\Omega)} \equiv \epsilon_{ij}(p) > 0.$$

Next, using equivalency of  $p$ -norm and 2-norm, there is a constant which depends on  $p$ , call it  $\alpha(p)$ , such that  $\|\cdot\|_2 \geq \alpha(p)\|\cdot\|_p$ ,  $\alpha(p) > 0$ , and so we get

$$\|\nabla_x\|x - y_i\|_p - \nabla_x\|x - y_j\|_p\|_p \geq \alpha(p)\|\nabla_x\|x - y_i\|_p - \nabla_x\|x - y_j\|_p\|_p = \alpha(p)\epsilon_{ij}(p) \equiv \epsilon_{ij}.$$

Finally, it follows that  $\forall i, j = 1, 2, \dots, N, i \neq j$ ,

$$\|\nabla_x\|x - y_i\|_p - \nabla_x\|x - y_j\|_p\|_p \geq \epsilon, \quad (\text{E5})$$

where  $\epsilon = \min_{i,j} \epsilon_{ij} > 0$ . □

**Claim 6.3** *With the notation from Lemma 6.2, there exists  $\Delta > 0$  such that*

$$\|a^{p-2} + a^{p-3} \circ b + \dots + b^{p-2}\|_p \geq \Delta > 0$$

where  $p$  is an even number and

$$a, b \in \mathbb{R}^n, \quad a_l = \frac{(x - y_i)_j}{\|x - y_i\|_p}, \quad b_l = \frac{(x - y_j)_j}{\|x - y_j\|_p}, \quad l = 1, \dots, d.$$

**Proof**

$$\|a^{p-2} + a^{p-3} \circ b + \dots + b^{p-2}\|_p \geq |a_k^{p-2} + a_k^{p-3}b_k + \dots + b_k^{p-2}|,$$

for an index  $k$  such that  $|a_k| = \|a\|_\infty$ . Next, using Claim 6.4 we have

$$|a_k| = \frac{(|x - y_i|)_k}{\|x - y_i\|_p} \geq \frac{\delta_a}{C(y_i, \Omega, p)\text{diam}(\Omega)} > 0.$$

Next we are going to consider cases:

- $a_k < 0$  and  $b_k \leq 0$ :

$$|a_k^{p-2} + a_k^{p-3}b_k + \dots + b_k^{p-2}| \geq |a_k^{p-2}| \geq \frac{\delta_a^{p-2}}{[C(y_i, \Omega, p)\text{diam}(\Omega)]^{p-2}}.$$

- $a_k > 0$  and  $b_k \geq 0$ :

$$|a_k^{p-2} + a_k^{p-3}b_k + \dots + b_k^{p-2}| \geq |a_k^{p-2}| \geq \frac{\delta_a^{p-2}}{[C(y_i, \Omega, p)\text{diam}(\Omega)]^{p-2}}.$$

- $a_k < 0$  and  $b_k > 0$ :

$$\begin{aligned} |a_k^{p-2} + a_k^{p-3}b_k + \dots + b_k^{p-2}| &\geq |a_k^{p-2} + a_k^{p-3}b_k + \dots + a_k b_k^{p-3}| = |a_k| |a_k^{p-3} + a_k^{p-4}b_k + \dots + b_k^{p-3}| \geq \\ &\geq |a_k| |a_k^{p-3} + a_k^{p-4}b_k + \dots + a_k b_k^{p-4}| = |a_k|^2 |a_k^{p-4} + a_k^{p-5}b_k + \dots + b_k^{p-4}| \geq \dots \geq |a_k|^{p-2} \implies \\ &\implies |a_k^{p-2} + a_k^{p-3}b_k + \dots + b_k^{p-2}| \geq |a_k|^{p-2} \geq \frac{\delta_a^{p-2}}{[C(y_i, \Omega, p)\text{diam}(\Omega)]^{p-2}}. \end{aligned}$$

- $a_k > 0$  and  $b_k < 0$ : using Claim 6.4 we have

$$|b_k| = \frac{(|x - y_j|)_k}{\|x - y_j\|_p} \geq \frac{\delta_b}{C(y_j, \Omega, p)\text{diam}(\Omega)} > 0.$$

Then it follows that

$$\begin{aligned} |b_k^{p-2} + b_k^{p-3}a_k + \dots + a_k^{p-2}| &\geq |b_k^{p-2} + b_k^{p-3}a_k + \dots + b_k a_k^{p-3}| = |b_k| |b_k^{p-3} + b_k^{p-4}a_k + \dots + a_k^{p-3}| \geq \\ &\geq |b_k| |b_k^{p-3} + b_k^{p-4}a_k + \dots + b_k a_k^{p-4}| = |b_k|^2 |b_k^{p-4} + b_k^{p-5}a_k + \dots + a_k^{p-4}| \geq \dots \geq |b_k|^{p-2} \implies \\ &\implies |a_k^{p-2} + a_k^{p-3}b_k + \dots + b_k^{p-2}| \geq |b_k|^{p-2} \geq \frac{\delta_b^{p-2}}{[C(y_j, \Omega, p)\text{diam}(\Omega)]^{p-2}}. \end{aligned}$$

As a result, it follows that

$$\|a^{p-2} + a^{p-3}b + \dots + b^{p-2}\|_p \geq \Delta$$

where

$$\Delta = \begin{cases} \frac{\delta_b^{p-2}}{[C(y_j, \Omega, p)\text{diam}(\Omega)]^{p-2}}, & \text{if } a_k > 0 \text{ and } b_k < 0, \\ \frac{\delta_a^{p-2}}{[C(y_i, \Omega, p)\text{diam}(\Omega)]^{p-2}}, & \text{otherwise.} \end{cases}$$

Thus, the result follows.  $\square$

**Claim 6.4** *With the notation from Lemma 6.2 and Claim 6.3, there exists  $\delta_a > 0$  and  $\delta_b > 0$  such that*

$$\|a\|_\infty \geq \frac{\delta_a}{C(y_i, \Omega, p)\text{diam}(\Omega)} > 0, \quad \|b\|_\infty \geq \frac{\delta_b}{C(y_j, \Omega, p)\text{diam}(\Omega)} > 0$$

where  $p$  is an even number and

$$a, b \in \mathbb{R}^n, \quad a_l = \frac{(x - y_i)_l}{\|x - y_i\|_p}, \quad b_l = \frac{(x - y_j)_l}{\|x - y_j\|_p}, \quad l = 1, \dots, d.$$

**Proof** Firstly, using assumption from Lemma 6.2, let  $w_{ij} = \xi - \|y_i - y_j\|_p$ ,  $0 < \xi < \|y_i - y_j\|_p$ . Then

$$\begin{aligned} x \in e_{ij} \cap \Omega &\implies \|x - y_i\|_p = \|x - y_j\|_p + w_{ij} = \|x - y_j\|_p + \xi - \|y_i - y_j\|_p \\ \|x - y_i\|_p &\geq \xi - \|x - y_i\|_p \implies \|x - y_i\|_p \geq \frac{\xi}{2} > 0 \end{aligned}$$

Next, using equivalency of  $p$ -norm and  $\infty$ -norm:  $\|\cdot\|_\infty \geq \gamma(p)\|\cdot\|_p$ ,  $\gamma(p) > 0$ , we get

$$\|x - y_i\|_\infty \geq \gamma(p) \frac{\xi}{2} \implies \|a\|_\infty = \frac{\|x - y_i\|_\infty}{\|x - y_i\|_p} \geq \frac{\delta_a}{C(y_i, \Omega, p) \text{diam}(\Omega)} > 0$$

where  $\delta_a = \gamma(p) \frac{\xi}{2}$ . Similarly, it follows that

$$\begin{aligned} x \in e_{ij} \cap \Omega &\implies \|x - y_j\|_p = \|x - y_i\|_p - w_{ij} = \|x - y_i\|_p - \xi + \|y_i - y_j\|_p \geq \|y_i - y_j\|_p - \xi > 0 \\ \|x - y_j\|_\infty &\geq \gamma(p)(\|y_i - y_j\|_p - \xi) \implies \|b\|_\infty = \frac{\|x - y_j\|_\infty}{\|x - y_j\|_p} \geq \frac{\delta_b}{C(y_j, \Omega, p) \text{diam}(\Omega)} > 0 \end{aligned}$$

where  $\delta_b = \gamma(p)(\|y_i - y_j\|_p - \xi)$ .



## 7 Algorithms Pseudocodes

---

### Algorithm 1 *Shooting* – Shooting Technique

---

**Input:**  $\theta_0, \{Y, W\}, \{y_j, w_j\}, \forall j, \text{TOL}$

▷ Default value  $\text{TOL} = 10^{-14}$

**Output:**  $r$  value up to a tolerance  $\text{TOL}$  and index  $k$

```

 $r_0 \leftarrow 0, d \leftarrow \frac{\min_j \|Y - y_j\|_p}{10}, \forall j \neq i$ 
while  $d \geq \text{TOL}$  do
     $r_0 \leftarrow r_0 + d$ 
     $x \leftarrow Y + r_0 \begin{pmatrix} \cos(\theta_0) \\ \sin(\theta_0) \end{pmatrix}$ 
    if  $x \in A(y_j)$  then
         $r^* \leftarrow \text{BisectR}(\theta_0, \{r_0 - d, r_0\}, \{Y, W\}, \{y_j, w_j\}, \text{TOL})$  and  $k \leftarrow j$ 
         $x \leftarrow Y + (r^* - \text{TOL}) \begin{pmatrix} \cos(\theta_0) \\ \sin(\theta_0) \end{pmatrix}$ 
        if  $x \in A(Y)$  then
            return  $r^*$  and  $k \leftarrow j$ 
        else
             $d \leftarrow \frac{d}{2}$ 
        end if
    else if  $x \notin \Omega$  then
         $\{r^*, k\} \leftarrow \text{Bound}(\theta_0, Y)$ 
         $x \leftarrow Y + (r^* - \text{TOL}) \begin{pmatrix} \cos(\theta_0) \\ \sin(\theta_0) \end{pmatrix}$ 
        if  $x \in A(Y)$  then
            return  $r^*$  and  $k$ 
        else
             $d \leftarrow \frac{d}{2}$ 
        end if
    end if
end while

```

---



---

### Algorithm 2 *BisectR* – Bisection Method on $r$ Value

---

**Input:**  $\theta_0, \{r_a, r_b\}, \{Y, W\}, \{y_j, w_j\}, \text{TOL}$

▷  $\text{TOL}$  is passed by *Shooting*

**Output:**  $r$  value up to a tolerance  $\text{TOL}$

```

while  $r_b - r_a \geq \text{TOL}$  do
     $r \leftarrow \frac{1}{2}(r_b + r_a)$ 
    if  $F(r, \theta_0) < 0$  then
         $r_a \leftarrow r$ 
    else if  $F_b(r) > 0$  then
         $r_b \leftarrow r$ 
    else
        return  $r$ 
    end if
end while

```

---

---

**Algorithm 3** *Bound* – Boundary Point Computation
 

---

**Input:**  $\theta, Y$ 
**Output:**  $r$  value and index  $k$ 

```

 $r_1 \leftarrow \frac{a-Y_1}{\cos(\theta)}, y_1 = Y_2 + r_1 \sin(\theta)$ 
 $r_2 \leftarrow \frac{b-Y_1}{\cos(\theta)}, y_2 = Y_2 + r_2 \sin(\theta)$ 
 $r_3 \leftarrow \frac{a-Y_2}{\sin(\theta)}, x_3 = Y_1 + r_3 \cos(\theta)$ 
 $r_4 \leftarrow \frac{b-Y_2}{\sin(\theta)}, x_4 = Y_1 + r_4 \cos(\theta)$ 
if  $r_1 > 0$  &  $y_1 \geq a$  &  $y_1 \leq b$  then
     $r \leftarrow r_1$  and  $k \leftarrow -4$ 
else if  $r_2 > 0$  &  $y_2 \geq a$  &  $y_2 \leq b$  then
     $r \leftarrow r_2$  and  $k \leftarrow -2$ 
else if  $r_3 > 0$  &  $x_3 \geq a$  &  $x_3 \leq b$  then
     $r \leftarrow r_3$  and  $k \leftarrow -1$ 
else if  $r_4 > 0$  &  $x_4 \geq a$  &  $x_4 \leq b$  then
     $r \leftarrow r_4$  and  $k \leftarrow -3$ 
end if
    
```

---



---

**Algorithm 4** *PredCorr* – Predictor-Corrector Step
 

---

**Input:**  $r_0, \theta_0, \Delta, \{Y, W\}, \{y_j, w_j\}, \text{TOL}$ 

 ▷ Default value  $\text{TOL} = 10^{-14}$ 
**Output:**  $r_1 = r(\theta_0 + \Delta)$  value up to a tolerance  $\text{TOL}$  and increment  $\Delta_{new}$ 

```

 $r_1^0 \leftarrow r_0 - \Delta \frac{F_\theta(r_0, \theta_0)}{F_r(r_0, \theta_0)}, r_1 \leftarrow r_1^0$ 
while  $F(r_1, \theta + \Delta) \geq \text{TOL}$  do
     $r_1 \leftarrow r_1 - \frac{F(r_1, \theta + \Delta)}{F_r(r_1, \theta + \Delta)}$ 
end while
 $\Delta_{new} \leftarrow \frac{10^{-3}}{2\sqrt{r_1 - r_1^0}} \Delta$ 
    
```

▷ Tangent Predictor Step

▷ Newton Corrector

---



---

**Algorithm 5** *BisectTheta* – Bisection Method on  $\theta$  Value
 

---

**Input:**  $r_0, \theta_0, \theta_1, \{Y, W\}, \{y_j, w_j\}, \text{TOL}$ 

 ▷ Default value  $\text{TOL} = 10^{-14}$ 
**Output:**  $r$  and  $\theta$  values up to a tolerance  $\text{TOL}$ 

```

while  $\theta_1 - \theta_0 \geq \text{TOL}$  do
     $\theta \leftarrow \frac{1}{2}(\theta_0 + \theta_1)$ 
     $r \leftarrow \text{PredCorr}(r_0, \theta_0, \theta - \theta_0, \{Y, W\}, \{y_j, w_j\}, \text{TOL})$ 
     $x \leftarrow Y + r \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$ 
    if  $x \in A(Y)$  then
         $\theta_0 \leftarrow \theta$  and  $r_0 \leftarrow r$ 
    else
         $\theta_1 \leftarrow \theta$ 
    end if
end while
    
```

---

---

**Algorithm 6** *CurveBound* – Breakpoint Between Curve and Domain Boundary

---

**Input:**  $k, \theta_0, \theta_1, \{Y, W\}, \{y_j, w_j\}, \text{TOL}$

$\triangleright k$  is the index of domain boundary

**Output:**  $r^*$  and  $\theta^*$  values up to a tolerance TOL

$\triangleright$  Default value  $\text{TOL} = 10^{-14}$

```

if  $k = -4$  then
     $r(\theta) \leftarrow \frac{a-Y_1}{\cos(\theta)}$ 
else if  $k = -2$  then
     $r(\theta) \leftarrow \frac{b-Y_1}{\cos(\theta)}$ 
else if  $k = -1$  then
     $r(\theta) \leftarrow \frac{a-Y_2}{\sin(\theta)}$ 
else if  $k = -3$  then
     $r(\theta) \leftarrow \frac{b-Y_2}{\sin(\theta)}$ 
end if
 $\theta^* \leftarrow \text{mod}(\text{fzero}(F(r(\theta), \theta), [\theta_0, \theta_1], \text{TOL}), 2\pi)$ 
 $r^* \leftarrow r(\theta^*)$ 

```

---



---

**Algorithm 7** *BoundBound* – Breakpoint Between Two Domain Boundary Branches

---

**Input:**  $k_1, k_2$

**Output:**  $r^*$  and  $\theta^*$  values

```

if  $(k_1 = -4 \ \& \ k_2 = -1) \vee (k_1 = -1 \ \& \ k_2 = -4)$  then
     $x \leftarrow \begin{pmatrix} a \\ a \end{pmatrix}$ 
else if  $(k_1 = -1 \ \& \ k_2 = -2) \vee (k_1 = -2 \ \& \ k_2 = -1)$  then
     $x \leftarrow \begin{pmatrix} b \\ a \end{pmatrix}$ 
else if  $(k_1 = -2 \ \& \ k_2 = -3) \vee (k_1 = -3 \ \& \ k_2 = -2)$  then
     $x \leftarrow \begin{pmatrix} b \\ b \end{pmatrix}$ 
else if  $(k_1 = -3 \ \& \ k_2 = -4) \vee (k_1 = -4 \ \& \ k_2 = -3)$  then
     $x \leftarrow \begin{pmatrix} a \\ b \end{pmatrix}$ 
end if
 $\theta^* \leftarrow \text{mod}(\arctan(\frac{x_2-Y_2}{x_1-Y_1}), 2\pi)$ 
 $r^* \leftarrow \frac{x_1-Y_1}{\cos(\theta^*)}$ 

```

---

---

**Algorithm 8** *AdaptSimpson* – Adaptive Integral Computation
 

---

**Input:**  $f(x)$ ,  $[a, b]$ ,  $\text{TOL}$  ▷ Default value  $\text{TOL} = 10^{-12}$   
**Output:**  $I = \int_a^b f(x)dx$  up to a tolerance  $\text{TOL}$   
 $\text{check} \leftarrow 0$ ;  $N_{\max} \leftarrow 1024$ ;  $i_{\max} \leftarrow \lfloor \log_2(\text{TOL}/\text{eps}) \rfloor$  ▷  $\text{eps}$  is machine precision  
 $I \leftarrow 0$ ;  $\text{Err} \leftarrow 0$   
**for**  $i \leftarrow 0$  to  $i_{\max}$  **do**  
 $x_i \leftarrow [a : (b-a)/(2^i) : b]$   
 $\text{TOL}_i \leftarrow \text{TOL}/(2^i)$   
**for**  $j \leftarrow 1$  to  $2^i$  **do**  
     **if**  $\text{check}(j) = 0$  **then**  
          $[I_j, \text{err}_j] \leftarrow \text{CompSimpson}(f(x), x_i(j), x_i(j+1), \text{TOL}_i, N_{\max})$   
         **if**  $\text{Err}_j < \text{TOL}_i$  **then**  
              $I \leftarrow I + I_j$ ;  $\text{Err} \leftarrow \text{Err} + \text{err}_j$ ;  $\text{check}(j) \leftarrow 1$   
         **end if**  
     **end if**  
**end for**  
**if**  $\min(\text{check}) = 1$  **then**  
     **return**  $I$   
**else**  
      $\text{check} \leftarrow \text{repelem}(\text{check}, 2)$  ▷ Repeat each of  $\text{check}$  vector's elements two times  
     **end if**  
     **if**  $\text{mod}(i, 3) = 2$  **then**  
          $N_{\max} \leftarrow 8 \times N_{\max}$   
     **end if**  
**end for**

---



---

**Algorithm 9** *CompSimpson* – Integral Computation using Composite Simpson's 1/3 Rule
 

---

**Input:**  $f(x)$ ,  $[a, b]$ ,  $\text{TOL}$ ,  $N_{\max}$  ▷  $\text{TOL}$  is passed by *AdaptSimpson*  
**Output:**  $I = \int_a^b f(x)dx$  up to a tolerance  $\text{TOL}$  and relative error value  $\text{err}$   
 $i_{\max} \leftarrow \lfloor \log_2(N_{\max}) \rfloor$ ;  $n \leftarrow 2$   
 $I_n \leftarrow \text{Simpson}(f(x), [a, b], n)$  ▷ Computes integral using Simpson's 1/3 Rule with  $n$  sub-intervals  
**for**  $i \leftarrow 1$  to  $i_{\max}$  **do**  
 $I_{2n} \leftarrow \text{Simpson}(f(x), [a, b], 2n)$   
 $\text{err} \leftarrow 16|I_{2n} - I_n|/15$ ;  $N \leftarrow n(\text{err}/\text{TOL})^{1/4}$ ;  $N \leftarrow 2\lfloor N/2 \rfloor$   
**if**  $\text{err} < \text{TOL} \vee N > N_{\max}$  **then**  
     **return**  $I_{2n}$  and  $\text{err}$   
**else**  
      $n \leftarrow 2n$ ;  $I_n \leftarrow I_{2n}$   
**end if**  
**end for**

---



---

**Algorithm 10** *Simpson* – Integral Computation using Simpson's 1/3 Rule
 

---

**Input:**  $f(x)$ ,  $[a, b]$ ,  $N$   
**Output:**  $I = \int_a^b f(x)dx$  approximation using Simpson's 1/3 Rule with  $N$  sub-intervals  
 $h \leftarrow \frac{b-a}{N}$ ,  $I \leftarrow \frac{1}{3}hf(a)$   
**for**  $i \leftarrow 2$  to  $N$  **do**  
     **if**  $\text{mod}(i, 2) = 0$  **then**  
          $I \leftarrow I + \frac{4}{3}hf(a + (i-1)h)$   
     **else if**  $\text{mod}(i, 2) = 1$  **then**  
          $I \leftarrow I + \frac{2}{3}hf(a + (i-1)h)$   
     **end if**  
**end for**  
 $I \leftarrow I + \frac{1}{3}hf(b)$

---

---

**Algorithm 11** *GridInitialGuess* – Initial Guess Computation on a Grid

---

**Input:**  $h, (a, b), \{y_i, v_i\}, \forall i, \text{MAXIT}$  ▷ Default is MAXIT = 50  
**Output:**  $w^0$  value  
 $w^0 \leftarrow 0, N_h \leftarrow \frac{b-a}{h},$  ▷  $N_h$  is the number of squares on the grid  
 $X_1 \leftarrow [a + \frac{h}{2} : h : b - \frac{h}{2}], X_2 \leftarrow [a + \frac{h}{2} : h : b - \frac{h}{2}],$  ▷ Coordinates of centers of squares  
**for**  $iter \leftarrow 1$  to MAXIT **do**  
     $M \leftarrow \text{GridMeasure}(h, N_h, \{X_1, X_2\}, \{y_i, w_i^0\}, \forall i)$   
    **for**  $k \leftarrow 1$  to  $N$  **do** ▷  $N$  is the number of target points  
         $M^0 \leftarrow \text{GridMeasure}(h, N_h, \{X_1, X_2\}, \{y_i, w_i^0\}, \forall i)$   
         $incr \leftarrow \sqrt{h}$   
        **while**  $|M^0(k) - v_k| > h^2 \wedge incr > h^4$  **do**  
             $w^0(i) \leftarrow w^0(i) + \text{sign}(M^0(k) - v_k) \frac{incr}{N-1}, \forall i \neq k$   
             $M^1 \leftarrow \text{GridMeasure}(h, N_h, \{X_1, X_2\}, \{y_i, w_i^0\}, \forall i)$   
            **if**  $|M^1(k) - v_k| > |M^0(k) - v_k|$  **then**  
                 $incr \leftarrow \sqrt{h} \times incr$   
            **end if**  
             $M^0 \leftarrow M^1$   
        **end while**  
    **end for**  
    **if**  $\max_i |M^0(i) - v_i| < h^2 \vee \max_i |M^0(i) - v_i| \geq \max_i |M(i) - v_i|$  **then**  
        **return**  $w^0$   
    **end if**  
**end for**

---



---

**Algorithm 12** *GridMeasure* – Measure Approximation on the Grid

---

**Input:**  $h, N_h, \{X_1, X_2\}, \{y_i, w_i\}, \forall i$   
**Output:**  $M$  values, which approximate  $\mu(A(y_i))$  values on the grid  
 $M \leftarrow 0$   
**for**  $l \leftarrow 1$  to  $N_h$  **do**  
    **for**  $m \leftarrow 1$  to  $N_h$  **do**  
         $x \leftarrow \begin{pmatrix} X_1^{(l)} \\ X_2^{(m)} \end{pmatrix}$   
         $ind \leftarrow \arg \min_i c(x, y_i) - w_i$   
         $M(ind) \leftarrow M(ind) + \rho(x)h^2$   
    **end for**  
**end for**

---

---

**Algorithm 13** *DampedNewton* – Damped Newton’s Method
 

---

**Input:**  $w^0, \mu, \{y_i, v_i\}, \forall i, \text{MAXIT}, \text{TOL}$  ▷ Default MAXIT = 20, TOL =  $10^{-8}$   
**Output:**  $w^k$  value such that  $\max_i |\mu(A(y_i, w^k)) - v_i| \leq \text{TOL}$   
 $err^0 \leftarrow \mu(A(y_i, w^0)) - v_i, \forall i$   
**for**  $k \leftarrow 1$  to MAXIT **do**  
 $H^0 \leftarrow \frac{\partial^2 \Phi}{\partial w_i^0 \partial w_j^0}, \forall i, j$   
 $s \leftarrow -(H^0)^{-1} err^0, w^1 \leftarrow w^0 + s$   
**while**  $\min_{i,j} \|1 - |w_i^1 - w_j^1|/c(y_i, y_j)\| \leq 0$  **do** ▷ See Eq. (22)  
 $s \leftarrow \frac{1}{2}s, w^1 \leftarrow w^0 + s$   
**end while**  
 $err^1 \leftarrow \mu(A(y_i, w^1)) - v_i, \forall i$   
**while**  $\|err^1\|_2^2 > \|err^0\|_2^2 \wedge \|s\|_\infty \geq \frac{\text{TOL}}{2^{\text{MAXIT}}}$  **do** ▷ Damped Newton  
 $s \leftarrow \frac{1}{2}s, w^1 \leftarrow w^0 + s$   
 $err^1 \leftarrow \mu(A(y_i, w^1)) - v_i, \forall i$   
**end while**  
**if**  $\|err^1\|_\infty \leq \text{TOL} \vee \|s\|_\infty \leq \frac{\text{TOL}}{2^{\text{MAXIT}}}$  **then**  
 $\text{return } w^1$   
**else**  
 $w^0 \leftarrow w^1, err^0 \leftarrow err^1$   
**end if**  
**end for**

---

**Acknowledgements** Not applicable.

**Author Contributions** L.D. and D.O. contributed equally to the manuscript. D.O. is responsible for the pseudocodes in the Appendix.

**Funding** Not applicable.

**Availability of supporting data** Not applicable.

## Declarations

**Competing interests** The authors declare no competing interests.

**Ethical approval** Not applicable.

## References

1. Ambrosio, L.: Lecture Notes on Optimal Transport Problems, pp. 1–52. Springer, Berlin Heidelberg, Berlin, Heidelberg (2003)
2. Aurenhammer, F.: Power diagrams: Properties, algorithms and applications. *SIAM J. Comput.* **16**(1), 78–96 (1987)
3. Balestro, V., Martini, H.: Minkowski geometry—some concepts and recent developments. *Surv. Geom.* **I**, 49–95 (2022)
4. Benamou, J.-D., Brenier, Y.: A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem. *Numerische Mathematik*, **84**(3), 375–393, (2000)
5. Benamou, J.-D., Froese, B.D., Oberman, A.M.: Numerical solution of the optimal transportation problem using the Monge-Ampère equation. *J. Comput. Phys.* **260**, 107–126 (2014)
6. Bollobás, B.: Linear Analysis, Cambridge Mathematical Textbooks, p. 21 Cambridge University Press, (1999)
7. Bounceur, A., Bezoui, M., Euler, R.: Boundaries and hulls of Euclidean graphs: From theory to practice, CRC Press, p. 34 (2018)
8. Bourne, D.P., Roper, S.M.: Centroidal power diagrams, Lloyd’s algorithm, and applications to optimal location problems. *SIAM J. Numer. Anal.* **53**(6), 2545–2569 (2015)
9. Bourne, D.P., Schmitzer, B., Wirth, B.: Semi-discrete unbalanced optimal transport and quantization. [arXiv: Optimization and Control](#), 2018–08
10. Cuesta-Albertos, J.A., Tüero-Díaz, A.: A characterization for the solution of the Monge-Kantorovich mass transference problem. *Stat. Probab. Lett.* **16**(2), 147–152 (1993)
11. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems*, Curran Associates, Inc., vol. 26, pp. 2292–2300 (2013)
12. De Gournay, F., Kahn, J., Lebrat, L.: Differentiation and regularity of semi-discrete optimal transport with respect to the parameters of the discrete measure. *Numer. Math.* **141**, 429–453 (2019)
13. Dieci, L., Omarov, D.: Techniques for continuous optimal transport problem. *Comput. Math. Appl.* **146**, 176–191 (2023)
14. Dieci, L., Walsh, J.D., III.: The boundary method for semi-discrete optimal transport partitions and Wasserstein distance computation. *J. Comput. Appl. Math.* **353**, 318–344 (2019)
15. Dobrin, A.: A review of properties and variations of Voronoi diagrams. Whitman College. (2005)
16. Du, Q., Faber, V., Gunzburger, M.: Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Rev.* **41**(4), 637–676 (1999)
17. Fortune, S.: A sweepline algorithm for Voronoi diagrams. *Proc. Second Ann. Symp. Comput. Geom. SCG New York, ACM.* **86**, 313–322 (1986)
18. Froese, B.D.: A numerical method for the elliptic Monge-Ampère equation with transport boundary conditions. *SIAM J. Sci. Comput.* **34**(3), A1432–A1459 (2012)
19. Gangbo, W., McCann, R.J.: The geometry of optimal transportation. *Acta Math.* **177**(2), 113–161 (1996)

20. Gao, J., Gupta, R.: Efficient proximity search for 3-D cuboids. In: Kumar, Vipin, Gavrilova, Marina L., Tan, Chih Jeng Kenneth., L'Ecuyer, Pierre (eds.) *Comput. Sci. Its Appl. – ICCSA 2003*, pp. 817–826. Berlin Heidelberg, Berlin, Heidelberg, Springer (2003)
21. Geiß, D., Klein, R., Penninger, R., Rote, G.: Optimally solving a transportation problem using Voronoi diagrams. *Comput. Geom.* **46**(8), 1009–1016 (2013)
22. Godsil, C., Royle, G.F.: *Algebraic Graph Theory*, Springer Science & Business. Media **207**, 279 (2001)
23. Groß, C. and Strempel, T.-K.: On generalizations of conics and on a generalization of the Fermat-Torricelli problem. *The American mathematical monthly*, **105**(8), 732–743 (1998)
24. Hartmann, V.N., Schuhmacher, D.: Semi-discrete optimal transport: a solution procedure for the unsquared Euclidean distance case. *Mathematical Methods of Operations Research*, pp. 1–31, (2020)
25. Jahn, T.: *An invitation to generalized Minkowski geometry*. (2019)
26. Kantorovich, L.V.: On the translocation of masses. *C.R. (Doklady) Acad. Sci. URSS (N.S.)*, **37**, 199–201 (1942)
27. Kantorovich, L.V.: On a problem of Monge. *Usp. Mat. Nauk* **3**, 225–226 (1948)
28. Keller, H.B.: *Numer. methods bifurcation probl.*, vol. Tata. Institute of Fundamental Research. Springer-Verlag, Bombay (1987)
29. Kitagawa, J., Mérigot, Q., Thibert, B.: Convergence of a Newton algorithm for semi-discrete optimal transport. *J. Eur. Math. Soc.* **21**(9), 2603–2651 (2019)
30. Lévy, B.: A numerical algorithm for  $L_2$  semi-discrete optimal transport in 3D. *ESAIM: Mathematical Modelling and Numerical Analysis*, **49**(6), 1693–1715 (2015)
31. Lévy, B., Schwindt, E.L.: Notions of optimal transport theory and how to implement them on a computer. *Comput. Graph.* **72**, 135–148 (2018)
32. Mérigot, Q.: A comparison of two dual methods for discrete optimal transport. In Frank Nielsen and Frédéric Barbaresco, editors, *GSI 2013 — Geometric Science of Information*, Aug 2013, Paris, France, of *Lecture Notes in Computer Science*, Springer, vol. 8085 pp. 389–396 (1781)
33. Meyron, J.: Initialization procedures for Discrete and semi-discrete optimal transport. *Comput. Aided. Des.* **115**, 13–22 (2019)
34. Monge, G.: *Mémoire sur la théorie des déblais et des remblais*. In *Histoire de l'Académie Royale des Sciences de Paris, avec les Mémoires de Mathématique et de Physique pour la même année*, Académie des sciences (France), In French. pp. 666–704 (1781)
35. Peyré, G., Cuturi, M.: Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, **11**(5-6), 355–607 (2019)
36. Pratelli, A.: On the equality between Monge's infimum and Kantorovich's minimum in optimal mass transportation. *Annales de l'Institut Henri Poincaré (B) Probability and Statistics*, **43**(1), 1–13 (2007)
37. Rüschendorf, L., Uckelmann, L.: Numerical and analytical results for the transportation problem of Monge-Kantorovich. *Metrika* **51**(3), 245–258 (2000)
38. Santambrogio, F.: *Optimal transport for applied mathematicians*. Springer **55**, 94 (2015)
39. Sharir, M.: Intersection and closest-pair problems for a set of planar discs. *SIAM J. Comput.* **14**(2), 448–468 (1985)
40. Sinkhorn, R.: Diagonal equivalence to matrices with prescribed row and column sums. *Am. Math. Mon.* **74**(4), 402–405 (1967)
41. Solomon, J., Goes, F., Peyré, G., Cuturi, M., Butscher, A., Nguyen, A., Du, T., Guibas, L.: Convolutional Wasserstein distances: Efficient optimal transportation on geometric domains. *ACM Transactions on Graphics*, **34**, 07 (2015)
42. Villani, C.: *Topics in Optimal Transportation*. Graduate Studies in Mathematics, vol. 58. American Mathematical Society, Providence, R.I. (2003)