Minimum Partial Matching and Hausdorff RMS-Distance Under Translation: Combinatorics and Algorithms

Rinat Ben-Avraham[1,⋆], Matthias Henze[2,⋆⋆], Rafel Jaume[2,⋆⋆⋆], Balázs Keszegh[3,†], Orit E. Raz[1,‡], Micha Sharir[1,§], and Igor Tubis[1]

[1] Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel,
rinatba@gmail.com, {oritraz,michas}@post.tau.ac.il, mrtubis@gmail.com
[2] Institut für Informatik, Freie Universität Berlin, Berlin, Germany,
matthias.henze@fu-berlin.de, jaume@mi.fu-berlin.de
[3] Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences, Budapest, Hungary,
keszegh@renyi.hu

Abstract. We consider the RMS-distance (sum of squared distances between pairs of points) under translation between two point sets in the plane. In the Hausdorff setup, each point is paired to its nearest neighbor in the other set. We develop algorithms for finding a local minimum in near-linear time in the line, and in nearly quadratic time in the plane. These improve substantially the worst-case behavior of the popular ICP heuristics for solving this problem. In the partial matching setup, each points in the smaller set is matched to a distinct point in the bigger set. Although the problem is not known to be polynomial, we establish several structural properties of the underlying subdivision of the plane and derive improved bounds on its complexity. In addition, we show how to compute a local minimum of the partial matching RMS-distance under translation, in polynomial time.

Keywords: partial matching, Hausdorff RMS-distance, polyhedral subdivision, local minimum

⋆ Supported by Grant 2012/229 from the U.S.-Israel Binational Science Foundation.
⋆⋆ Supported by ESF EUROCORES programme EuroGIGA-VORONOI, (DFG): RO 2338/5-1.
⋆⋆⋆ Supported by La-Caixa and the DAAD.
† Supported by Hungarian National Science Fund (OTKA), under grant PD 108406, NN 102029 (EUROGIGA project GraDR 10-EuroGIGA-OP-003), NK 78439, by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and by the DAAD.
‡ Supported by Grant 892/13 from the Israel Science Foundation.
§ Supported by Grant 2012/229 from the U.S.-Israel Binational Science Foundation, by Grant 892/13 from the Israel Science Foundation, by the Israeli Centers for Research Excellence (I-CORE) program (center no. 4/11), and by the Hermann Minkowski–MINERVA Center for Geometry at Tel Aviv University.

1 Introduction

Let A and B be two finite sets of points in the plane, of respective cardinalities n and m. We are interested in measuring the similarity between A and B, under a suitable proximity measure. We consider two such measures. In the first, each point is assigned to its nearest neighbor in the other set, and the proximity is the sum of the squared distances between the assigned pairs. See [1] for a similar generalization of the Hausdorff distance. On the other hand, there are situations where we want a one-to-one matching between A and B, which minimizes the sum of squares of the matched pairs [12, 18, 19]. In general, the sets A and B need not have the same size, say $|A| > |B|$, and then we want to match all the points of B (a specific pattern that we want to identify), in a one-to-one manner, to a subset of A (a larger picture that "hides" the pattern) of size $|B|$.

We refer to the measured distance between the sets, in both versions, as the *RMS distance*. In the former setup the measure is called the *Hausdorff RMS-distance*, and in the latter we call it the *(partial) matching RMS-distance*. In both variants the sets A and B are in general not aligned, so we seek a translation of one of them that will minimize the appropriate RMS-distance, Hausdorff or partial matching.

The Hausdorff RMS distance problem. Let $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_m\}$ be two sets of points in the plane, and let $N_A(x)$ (resp., $N_B(x)$) denote the nearest neighbor in A (resp., in B) of a point $x \in \mathbb{R}^2$. The *unidirectional RMS distance* between B and A is defined as

$$RMS(B, A) = \sum_{b \in B} \|b - N_A(b)\|^2.$$

We also consider *bidirectional* RMS distances, in which we also measure distances from the points of A to their nearest neighbors in B. We consider two variants of this notion. The first variant is the *L_1-bidirectional RMS distance* between A and B, which is defined as

$$RMS_1(B, A) = RMS(A, B) + RMS(B, A).$$

The second variant is the *L_∞-bidirectional RMS distance* between A and B, and is defined as

$$RMS_\infty(B, A) = \max\{RMS(A, B), RMS(B, A)\}.$$

Allowing one of the sets (say, B) to be translated, we define the *minimum unidirectional RMS distance under translation* to be

$$RMS_T(B, A) = \min_{t \in \mathbb{R}^2} RMS(B + t, A) = \min_{t \in \mathbb{R}^2} \sum_{b \in B} \|b + t - N_A(b + t))\|^2,$$

where $B + t = \{b_1 + t, \ldots, b_m + t\}$. Similarly, we define the *minimum L_1- and L_∞-bidirectional RMS distances under translation* to be

$$RMS_{T,1}(B, A) = \min_{t \in \mathbb{R}^2} RMS_1(B + t, A) \qquad \text{and}$$
$$RMS_{T,\infty}(B, A) = \min_{t \in \mathbb{R}^2} RMS_\infty(B + t, A).$$

The partial matching RMS-distance problem. Here we assume that $m = |B| < |A| = n$, and seek a minimum-weight *complete matching* of B into A. This is a subset M of edges of the complete bipartite graph with edge set $B \times A$, so that each $b \in B$ appears in exactly one edge of M, and each $a \in A$ appears in at most one edge. The weight of an edge (b, a) is $\|b - a\|^2$, and the weight of a matching is the sum of the weights of its edges.

A maximum-cardinality matching can be identified as an injective assignment π of B into A. With a slight abuse of notation, we denote by $a_{\pi(i)}$ the point a_j that π assigns to b_i. In this notation, the minimum RMS partial matching problem (for fixed locations of the sets) is to compute

$$M(B, A) = \min_{\pi: B \to A \text{ injective}} \sum_{i=1}^{m} \left\| b_i - a_{\pi(i)} \right\|^2 .$$

Allowing the pattern B to be translated, we obtain the problem of the minimum partial matching RMS-distance under translation, defined as

$$M_T(B, A) = \min_{t \in \mathbb{R}^2} M(B + t, A) = \min_{\substack{t \in \mathbb{R}^2, \pi: B \to A, \\ \pi \text{ injective}}} \sum_{i=1}^{m} \left\| b_i + t - a_{\pi(i)} \right\|^2 ;$$

The function $F(t) := M(B + t, A)$ induces a subdivision of $\mathbb{R}^2$, where two points $t_1, t_2 \in \mathbb{R}^2$ are in the same region if the minimum of F at t_1 and at t_2 is attained by the same assignment $\pi : B \to A$. We refer to this subdivision, following Rote [15], as the *partial matching subdivision* and denote it by $\mathcal{D}_{B,A}$. We say that a matching is *optimal* if it attains $F(t)$ for some $t \in \mathbb{R}^2$.

Background. The problem of Hausdorff RMS minimization under translation has been considered in the literature (see, e.g., [1] and references therein), although only scarcely so.

If A and B are sets of points on the line, the complexity of the Hausdorff RMS function, as a function of t, is $O(mn)$ (and this bound is tight in the worst case). Moreover, the function can have many local minima (up to $\Theta(mn)$ in the worst case). Hence, finding the translation that minimizes the Hausdorff RMS distance can be done in brute force, in $O(mn \log(mn))$ time, but a worst-case near linear algorithm is not known. In practice, though, there exists a popular heuristic technique, called the ICP (Iterated Closest Pairs) algorithm, proposed by Besl and McKay [6] and analyzed in Ezra et al. [10]. Although the algorithm is reported to be efficient in practice, it might perform $\Theta(mn)$ iterations in the worst case. Moreover, each iteration takes close to linear time (to find the nearest neighbors in the present location).

The situation is worse in the plane, where the complexity of the RMS function is $O(m^2 n^2)$, a bound which is worst-case tight, and the bounds for the performance of the ICP algorithm, are similarly worse. Similar degradation shows up in higher dimensions too; see, e.g., [10].

The problem of RMS minimization under translations is significantly more involved for partial matchings. A thorough initial study of the problem is given

by Rote [15]; see also [8, 16] for two follow-up studies, another study in [14], and an abstract of part of this paper [11]. The resulting subdivision $\mathcal{D}_{B,A}$, as defined above, is shown in [15] to be a convex subdivision. Rote's main contribution for the analysis of the complexity of $\mathcal{D}_{B,A}$ was to show that a line crosses only $O(mn)$ regions of the subdivision (see Theorem 1 below). However, obtaining sharp bounds for the complexity of $\mathcal{D}_{B,A}$ is still an open issue, where the best known upper bounds are exponential.

Our results. In this paper we study these two fairly different variants of the problem of minimizing the RMS distance under translation, and improve the state of the art in both of them.

For the Hausdorff variant, we provide improved algorithms for computing a local minimum of the RMS function, in one and two dimensions. Assuming $|A| = |B| = n$, in the one-dimensional case the algorithms run in time $O(n \log^2 n)$, and in the two-dimensional case they run in time $O(n^2 \log n)$. Our approach thus beats the ICP algorithm (used for decades to solve this problem) worst-case running time. The techniques are reasonably standard, although their assembly is somewhat involved. The approach is an efficient search through the (large number of) critical values of the RMS function.

In the partial matching variant, we first analyze the complexity of $\mathcal{D}_{B,A}$. We significantly improve the bound from the naive $O(n^m)$ to $O(n^2 m^{3.5}(e \ln m)^m)$. A preliminary informal exposition of this analysis by a subset of the authors is given in [11]. This paper expands the previous note, derives additional interesting structural properties of the subdivision, and significantly improves the complexity bound. The arguments that establish the bound can be generalized to bound the number of regions of the analogous subdivision in $\mathbb{R}^d$ by $O\left((mn^2)^d (e \ln m)^m)/\sqrt{m}\right)$. The derivation of the upper bound proceeds by a reduction that connects partial matchings to a combinatorial question based on a game theoretical problem, which we believe to be of independent interest.

Next we present a polynomial-time algorithm for finding a local minimum of the partial matching RMS-distance. This is significant, given that we do not have a polynomial bound on the size of the subdivision. We also fill in the details of explicitly computing the intersections of a line with $\mathcal{D}_{B,A}$. Although Rote hinted at such an algorithm in [15], by exploiting some new properties of $\mathcal{D}_{B,A}$ derived here, we manage to compute the intersections in a simple, more efficient manner.

2 Properties of $\mathcal{D}_{B,A}$

We begin by reconstructing several basic properties of $\mathcal{D}_{B,A}$ that have been noted in [15]. First, if we fix the translation $t \in \mathbb{R}^2$ and the assignment π, the cost of the matching, denoted by $f(\pi, t)$, is

$$f(\pi, t) = \sum_{i=1}^{m} \left\| b_i + t - a_{\pi(i)} \right\|^2 = c_\pi + \langle t, d_\pi \rangle + m \left\| t \right\|^2, \tag{1}$$

where $c_\pi = \sum_{i=1}^{m} \left\| b_i - a_{\pi(i)} \right\|^2$ and $d_\pi = 2 \sum_{i=1}^{m} (b_i - a_{\pi(i)})$. For t fixed, the assignment π that minimizes $f(\pi, t)$ is the same assignment that minimizes

$g(\pi, t) := c_\pi + \langle t, d_\pi \rangle$. It follows that $\mathcal{D}_{B,A}$ is the minimization diagram (the xy-projection) of the graph of the function

$$\mathcal{E}_{B,A}(t) = \min_{\pi: B \to A \text{ injective}} (c_\pi + \langle t, d_\pi \rangle), \quad t \in \mathbb{R}^2.$$

This is a lower envelope of a finite number of planes, so its graph is a convex polyhedron, and its projection $\mathcal{D}_{B,A}$ is a convex subdivision of the plane, whose faces are convex polygons. The great open question regarding minimum partial matching RMS-distance under translation, is whether the number of regions of $\mathcal{D}_{B,A}$ is polynomial in m and n. A significant, albeit small step towards settling this question is the following result of Rote [15].

Theorem 1 (Rote [15]). *A line intersects the interior of at most $m(n-m)+1$ different regions of the partial matching subdivision $\mathcal{D}_{B,A}$.*

The following property observed by Rote [15] seems to be well known [19]

Lemma 1. *For any $A' \subset A$, with $|A'| = m$, the optimal assignment that realizes the minimum $M(B + t, A')$ is independent of the translation $t \in \mathbb{R}^2$.*

Next, we derive several additional properties of $\mathcal{D}_{B,A}$ which show that the diagram has, at least locally, low-order polynomial complexity.

Lemma 2. *Every edge of $\mathcal{D}_{B,A}$ has a normal vector of the form $a_j - a_i$ for suitable $i, j \in \{1, \ldots, n\}$.*

Proof. Let E be an edge of $\mathcal{D}_{B,A}$ common to the regions associated with the injections $\pi, \sigma : B \to A$, respectively. By definition, $g(\pi, t) = g(\sigma, t)$ for any $t \in E$ and $g(\pi, t) = g(\sigma, t) \leq g(\delta, t)$ for every injection $\delta : B \to A$. By Equation (1), E is contained in the line $\ell(\pi, \sigma) = \{t \in \mathbb{R}^2 : \langle t, d_\pi - d_\sigma \rangle = c_\sigma - c_\pi\}$. Let $H = (\pi \setminus \sigma) \cup (\sigma \setminus \pi)$. It is easy to see that H consists of a vertex-disjoint union of cycles and alternating paths. Let $\gamma_1, \ldots, \gamma_p$ be these cycles and paths. It is not hard to see that every cycle and every path can be "flipped" independently while preserving the validity of the matching; that is, we can choose, within any of the γ_j's, either all the edges corresponding to π or all the ones corresponding to σ, without interfering with other cycles or paths, so that the resulting collection of edges still represents an injection from B into A. Observe now that $\ell(\pi, \sigma) = \{t \in \mathbb{R}^2 : \left\langle t, \sum_{j=1}^p d_{\gamma_j} \right\rangle = -\sum_{j=1}^p c_{\gamma_j}\}$, where d_{γ_j} is the sum of the terms in $d_\pi - d_\sigma$ that involve only the $a_i \in A$ contained in γ_j and c_{γ_j} is analogously defined for $c_\pi - c_\sigma$. Note that d_{γ_j} is 0 for every cycle γ_j and, therefore, at least one of the γ_j's is a path. Then, we must have $\langle t, d_{\gamma_j} \rangle = -c_{\gamma_j}$ for all $j = 1, \ldots, p$ and every $t \in \ell(\pi, \sigma)$. Otherwise, a flip in a path or cycle violating the equation would contradict the optimality of π or of σ along $\ell(\pi, \sigma)$. Therefore, all the vectors d_{γ_j} must be linearly dependent. In particular, the direction of $d_\pi - d_\sigma$ is the same as the one of d_{γ_j} for every *path* γ_j. If a path, say γ_1, starts at some a_j and ends at some a_i, then $d_{\gamma_1} = a_j - a_i$, which concludes the proof. □

Remark 1. It follows that if A is in general position then H has exactly one alternating path, and the pair a_i, a_j is unique.

Lemma 3. *i) $\mathcal{D}_{B,A}$ has at most $4m(n-m)$ unbounded regions.*
ii) Every region in $\mathcal{D}_{B,A}$ has at most $m(n-m)$ edges.
iii) Every vertex in $\mathcal{D}_{B,A}$ has degree at most $2m(n-m)$.
iv) Any convex path can intersect at most $m(n-m)+n(n-1)$ regions of $\mathcal{D}_{B,A}$, i.e., while translating B along any convex path, the optimal partial matching can change at most $m(n-m)+n(n-1)$ times.

Proof. i) Take a bounding box that encloses all the vertices of the diagram. By Theorem 1, every edge of the bounding box crosses at most $m(n-m)+1$ regions of $\mathcal{D}_{B,A}$. The edges of the box traverse only unbounded regions, and cross every unbounded region exactly once, except for the coincidences of the last region traversed by an edge and the first region traversed by the next edge.

ii) By Lemma 2, the normal vector of every edge of a region corresponding to the injection π is a multiple of $a_j - a_i$ for some $a_i \in \pi(B)$ and $a_j \notin \pi(B)$. There are exactly $m(n-m)$ such possibilities.

iii) Let v be a vertex of $\mathcal{D}_{B,A}$. Draw two generic parallel lines close enough to each other to enclose v and no other vertex. Each edge adjacent to v is crossed by one of the lines, and by Theorem 1 each of these lines crosses at most $m(n-m)$ edges.

iv) We use the following property that was observed in Rote's proof of Theorem 1. Suppose that we translate B along a line in some direction v. Rank the points of A by their order in the v-direction, i.e., $a < a'$ means that $\langle a, v\rangle < \langle a', v\rangle$ (for simplicity, assume that v is generic so there are no ties). Let Φ denote the sum of the ranks of the m points of A that participate in the optimal partial match. As Rote has shown, whenever the optimal assignment changes, Φ must increase. Now follow our convex path γ, which, without loss of generality, can be assumed to be polygonal. As we traverse an edge of γ, Φ obeys the above property, increasing every time we cross into a new region of $\mathcal{D}_{B,A}$. When we turn (counterclockwise) at a vertex of γ, the ranking of A may change, but each such change consists of a sequence of swaps of consecutive elements in the present ranking. At each such swap, Φ can decrease by at most 1. Since γ is convex, each pair of points of A can be swapped at most twice, so the total decrease in Φ is at most $2\binom{n}{2} = n(n-1)$. Hence, the accumulated increase in Φ, and thus also the total number of regions of $\mathcal{D}_{B,A}$ crossed by γ, is at most $\Big(n + (n-1) + \ldots + (n-m+1)\Big) - \Big(1 + 2 + \ldots + m\Big) + n(n-1)$. □

In the remainder of this section, we focus on establishing a global bound on the complexity of the diagram $\mathcal{D}_{B,A}$. We begin by deriving the following technical auxiliary results.

Lemma 4. *Let π be an optimal assignment for a fixed translation $t \in \mathbb{R}^2$.*

i) There is no cyclic sequence $(i_1, i_2, \ldots, i_k, i_1)$ satisfying $\|b_{i_j} + t - a_{\pi(i_j)}\| < \|b_{i_j} + t - a_{\pi(i_{j+1})}\|$ for all $j \in \{1, \ldots, k\}$ (modulo k).
ii) Each point of $B + t$ is matched to one of its m nearest neighbors in A.
iii) At least one point in $B + t$ is matched to its nearest neighbor in A.

iv) There exists an ordering $\langle b_1, \ldots, b_m \rangle$ of the elements of B, such that each b_k is assigned by π to one of its k nearest neighbors in A, for $k = 1, \ldots, m$.

Proof. i) For the sake of contradiction, we assume that there exists a cyclic sequence that satisfies all the prescribed inequalities. Consider the assignment σ defined by $\sigma(i_j) = i_{j-1}$ for all $j \in \{0, \ldots, k\}$ and $\sigma(\ell) = \pi(\ell)$ for all other indices ℓ. Since π is a one-to-one matching, we have that $\pi(i_j) \neq \pi(i_{j'})$ for all different $j, j' \in \{1, \ldots, k\}$ and, consequently, σ is one-to-one as well. It is easily checked that $f(\sigma, t) < f(\pi, t)$, contradicting the optimality of π.

ii) For contradiction, assume that for some point $b \in B$, $b + t$ is not matched by π to one of its m nearest neighbors in A. Then, at least one of these neighbors, say a, cannot be matched (because these m points can be claimed only by the remaining $m-1$ points of $B+t$). Thus, we can reduce the cost of π by matching $b + t$ to a, a contradiction that establishes the claim.

iii) Again we assume for contradiction that π does not match any of the points of $B + t$ to its nearest neighbor in A. We construct the following cyclic sequence in the matching π. We start at some arbitrary point $b_1 \in B$, and denote by a_1 its nearest neighbor in A (to simplify the presentation, we do not explicitly mention the translation t in what follows). By assumption, b_1 is not matched to a_1. If a_1 is also not claimed in π by any of the points of B, then b_1 could have claimed it, thereby reducing the cost of π, which is impossible. Let then b_2 denote the point that claims a_1 in π. Again, by assumption, a_1 is not the nearest neighbor a_2 of b_2, and the preceding argument then implies that a_2 must be claimed by some other point b_3 of B. We continue this process, and obtain an alternating path $(b_1, a_1, b_2, a_2, b_3, \ldots)$ such that the edges (b_i, a_i) are not in π, and the edges (b_{i+1}, a_i) belong to π, for $i = 1, 2, \ldots$. The process must terminate when we reach a point b_k that either coincides with b_1, or is such that its nearest neighbor is among the already encountered points a_i, $i < k$. We thus obtain a cyclic sequence as in part i), reaching a contradiction.

iv) Start with some point $b_1 \in B$ such that $b_1 + t$ goes to its nearest neighbor a_1 in A in the optimal partial matching π; such a point exists by part iii). Delete b_1 from B, and a_1 from A. The optimal matching of $B \setminus \{b_1\}$ into $A \setminus \{a_1\}$ (relative to t) is equal to the restriction of π to the points in $B \setminus \{b_1\}$, because otherwise we could have improved π itself. We apply part iii) to the reduced sets, and obtain a second point $b_2 \in B \setminus \{b_1\}$ whose translation $b_2 + t$ is matched to its nearest neighbor a_2 in $A \setminus \{a_1\}$, which is either its first or second nearest neighbor in the original set A. We keep iterating this process until the entire set B is exhausted. At the k-th step we obtain a point $b_k \in B \setminus \{b_1, \ldots, b_{k-1}\}$, such that the nearest neighbor a_k in $A \setminus \{a_1, \ldots, a_{k-1}\}$ is matched to b_k by π, so a_k is among the k nearest neighbors in A of $b_k + t$. □

Observe, that the geometric properties in Lemma 4 can be interpreted in purely combinatorial terms. Indeed, for t fixed, associate with each $b_i \in B$ an ordered list $L_t(b_i)$, called its *preference list*, which consists of the points of A sorted by their distances from $b_i + t$. In general, given m such ordered lists on n elements, an injective assignment from $\{1, \ldots, m\}$ to $\{1, \ldots, n\}$ such that

there is no cycle as in part i) is called *stable* or *Pareto efficient*. The problem of finding a sable matching was studied, for the case $m = n$, in the game theory literature under the name of the *House Allocation Problem* [17]. Note also that the proofs of parts ii)–iv) can be carried out in this abstract setting, and hold for any stable matching. Now assume that we let t vary, but constrain it to stay in a region in which there is no change in any of the preference lists $L_t(b_i)$, for $i = 1, \ldots, m$. Then part iii) and the proof of part iv) immediately yield an upper bound of $m!$ on the number of stable matchings. This bound is tight for the combinatorial problem, since if the ordered lists all coincide there are $m!$ different stable matchings. A recent study motivated by the extended abstract [11] prior to this work studied this combinatorial problem and derived the following.

Lemma 5 (Asinowski et al. [2]). *The number of elements that belong to some stable matching on m ordered preference lists is at most $m(\ln m + 1)$.*

The properties derived so far imply the following significantly improved upper bound on the complexity of $\mathcal{D}_{B,A}$.

Theorem 2. *The combinatorial complexity of $\mathcal{D}_{B,A}$ is $O(n^2 m^{3.5}(e \ln m)^m)$.*

Proof. The proof has two parts. First, we identify a convex subdivision K such that in each of its regions the ordered preference lists $L_t(b)$ of neighbors of each $b + t$, according to their distance from $b + t$, are fixed for all $b \in B$. We show that the complexity of K is only polynomial; specifically, it is $O(n^2 m^4)$. Second, we give an upper bound on how many regions of $\mathcal{D}_{B,A}$ can intersect a given region of K, using Theorem 5. Together, these imply an upper bound on the complexity of $\mathcal{D}_{B,A}$. Due to lack of space, the proof of the first part is given in Appendix A. We now consider all possible translations t in the interior of some fixed region τ of K and their corresponding optimal matchings. Lemma 4(i) ensures that all of them must be stable with respect to the fixed preference lists $L_t(b)$, for $b \in B$, over $t \in \tau$. In addition, Lemma 1 ensures that we only need to bound the number of different image sets of such stable matchings. Using the bound in Lemma 5, we can derive that the number of optimal matchings for translations in τ is then $O\left(\binom{m(\ln m+1)}{m}\right) = O\left(\frac{m^m \ln^m m}{m!}\right) = O\left(\frac{\ln^m(m)}{\sqrt{m}e^{-m}}\right)$, where in the second step we used Stirling approximation. Hence, by multiplying this bound by the number of regions in K, we conclude that the number of assignments corresponding to optimal matchings, and thus also the complexity of $\mathcal{D}_{B,A}$, is at most $O(n^2 m^{3.5}(e \ln m)^m$. □

The following proposition (proved in Appendix A) sets an obstruction for the combinatorial approach alone to yield a polynomial bound for $\mathcal{D}_{B,A}$.

Proposition 1. *For every $n \geq \lfloor \frac{m}{2} \rfloor + m$, there exists m preference lists of $\{1, \ldots, n\}$ with $\Omega\left(\frac{2^m}{\sqrt{m}}\right)$ different images of stable matchings.*

3 Finding a local minimum of the partial matching RMS-distance under translation

The high-level algorithm. We now concentrate on the algorithmic problem of computing, in polynomial time, a local minimum of the partial matching RMS-distance under translation.

We "home in" on a local minimum of $F(t)$ by maintaining a vertical slab I in the plane that is known to contain such a local minimum in its interior, and by repeatedly shrinking it until we obtain a slab I^* that does not contain any vertex of $\mathcal{D}_{B,A}$. That is, any (vertical) line contained in I^* intersects the same sequence of regions, and, by Theorem 1, the number of these regions is $O(mn)$. We compute these regions, find the optimal partial matching assignment in each region, and the corresponding explicit (quadratic) expression of $F(t)$, and search for a local minimum within each region.

A major component of the algorithm is a procedure, that we call $\Pi_1(\ell)$, which, for a given input line ℓ, constructs the intersection of $\mathcal{D}_{B,A}$ with ℓ, computes the global minimum t^* of F on ℓ, and determines a side of ℓ, in which F attains strictly smaller values than $F(t^*)$. If no such decrease is found in the neighborhood of t^* then it is a local minimum of F, and we stop.

We use this "decision procedure" as follows. Suppose we have a current vertical slab I, bounded on the left by a line ℓ^- and on the right by a line ℓ^+. We assume that Π_1 has been executed on ℓ^- and on ℓ^+, and that we have determined that F assumes smaller values than its global minimum on ℓ^- to the right of ℓ^-, and that it assumes smaller values than its global minimum on ℓ^+ to the left of ℓ^+. This is easily seen to imply that F must contain a local minimum in the interior of I. Note that just finding a *local* minimum of F along ℓ^+ or ℓ^- is not sufficient; see Appendix C for a discussion of a similar issue in the case of the Hausdorff RMS-distance. Let ℓ be some vertical line passing through I. We run Π_1 on ℓ. If it determines that F attains smaller values to its left (resp., to its right), we shrink I to the slab bounded by ℓ^- and ℓ (resp., the slab bounded by ℓ and ℓ^+). By what has just been argued, this ensures that the new slab also contains a local minimum of F in its interior.

To initialize the slab I, we choose an arbitrary *horizontal* line λ, and run Π_1 on λ, to find the sequence S of its intersection points with the edges of $\mathcal{D}_{B,A}$. We run a binary search through S, where at each step we execute Π_1 on the vertical line through the current point. When the search terminates, we have a vertical slab I_0 whose intersection with λ is contained in a single region σ_0 of $\mathcal{D}_{B,A}$.

After this initialization, we find the region σ_1 that lies directly above σ_0 and that the final slab I^* should cross. In general, there are possibly many such regions, but fortunately, by Lemma 3(ii), their number is only at most $m(n-m)$.

To find σ_1, we compute the boundary of σ_0; details about this procedure are given in Appendix B. Once we have explored the boundary of σ_0, we take the sequence of all vertices of σ_0, and run a Π_1-guided binary search on the vertical lines passing through them, exactly as we did with the vertices of S, to shrink I_0 into a slab I_1, so that σ_0 intersects I_1 in a trapezoid (or a triangle), with a single (portion of an) edge at the top and a single edge at the bottom. This allows us

to determine σ_1, which is the region lying on the other (higher) side of the top edge. A symmetric variant of this procedure will find the region lying directly below σ_0 in the final slab.

We repeat the previous step to find the entire stack of regions that I^* crosses, where each step shrinks the current slab and then crosses to the next region in the stack. Once this is completed, we find a local minimum within I^* as explained above. Details for this are again given in Appendix B.

In summary, we have the following main result of this section.

Theorem 3. *Given two finite point sets A, B in $\mathbb{R}^2$, with $n = |A| > |B| = m$ and no two pairs $(a_1, a_2), (a_1, a_2) \in A$ with $a_1 - a_2 = a_3 - a_4$, a local minimum of the partial matching RMS-distance under translation can be computed in $O(m^6 n^3 \log n)$ time.*

4 Finding a local minimum of the Hausdorff RMS-distance under translation

In this section, we turn to the simpler problem involving the Hausdorff RMS-distance, and present efficient algorithms for computing a local minimum of the RMS function in one and two dimensions. Due to lack of space, most of the material in this section is delegated to Appendix C, and we only provide here a high-level review of our algorithms.

The one-dimensional unidirectional case. Let $N_A(b + t)$ be the nearest neighbor in A of $b + t$, for $b \in B$, and $t \in \mathbb{R}$. The function $r(t) := RMS(B + t, A) = \sum_{b \in B}(b+t-N_A(b+t))^2$ is continuous and piecewise parabolic, with $O(mn)$ non-smooth breakpoints, which are the breakpoints of the step functions $N_A(b + t)$. For any given t_0, it is easy to compute, in $O(m \log n)$ time, the derivative $r'(t_0)$, or its left and right one-sided versions $r'(t_0)^-$, $r'(t_0)^+$ (when t_0 is a breakpoint). A simple observation is that if $I = [t_1, t_2]$ is an interval satisfying $r'(t_1)^+ < 0$ and $r'(t_2)^- > 0$ then I contains a local minimum of r. We thus start with a large interval I that contains all breakpoints of r, and keep shrinking it, halving the number of breakpoints in I in each step, until it contains only linearly many breakpoints, in which case r can be constructed explicitly over I, and searched for a local minimum, in near-linear time. See Appendix C for full details, which imply the following.

Theorem 4. *Given two finite point sets A, B on the real line, with $|A| = n$ and $|B| = m$, a local minimum of the unidirectional RMS distance under translation from B to A can be obtained in time $O(m \log^2 n + n \log n)$.*

The one-dimensional bidirectional case. Simple extensions of the procedure given above apply to the two variants of the minimum bidirectional Hausdorff RMS-distance, as defined in the introduction. Omitting the fairly routine details of these extensions, we obtain:

Theorem 5. *Given two finite point sets A, B on the real line, with $|A| = n$ and $|B| = m$, a local minimum under translation of the L_1-bidirectional or L_∞-bidirectional RMS distance between A and B, can be computed in time $O((n \log m + m \log n) \log \min\{m, n\})$.*

Minimum Hausdorff RMS-distance under translation in two dimensions. Here the function $r(t) := RMS(B+t, A) = \sum_{b \in B} \|b+t-N_A(b+t)\|^2$ induces a convex subdivision of the plane, where in each of its regions σ, all the m values $N_A(b+t)$, for $b \in B$, are fixed for $t \in \sigma$. This subdivision is simply the overlay M of the m shifted copies $\mathcal{V}(A-b)$, for $b \in B$, of the Voronoi diagram of A. These copies have a total of $O(mn)$ edges, and their overlay has thus complexity $O(m^2n^2)$ (which is tight in the worst case). Over each region of M, $r(t)$ is a quadratic function (a paraboloid), and the explicit expression for $r(t)$ can be updated in $O(1)$ time as we cross from one region to an adjacent one.

The goal is to search for a local minimum of r without explicitly constructing these many features of M. Similarly to the one-dimensional case, we maintain a vertical slab I, known to contain a local minimum, and keep shrinking it until it contains no vertices of M. In this case it overlaps only $O(mn)$ regions of M, vertically stacked above one another, and it is straightforward to enumerate all of them, get the explicit expressions of r over each of them, and search for a local minimum in each part, in a total of $O(mn)$ time.

To shrink I we use a "decision procedure" that, given a vertical line ℓ, computes the *global* minimum $\bar{t}$ of r restricted to ℓ, and tests the x-derivative of r at $\bar{t}$ to determine which side of ℓ contains smaller values than $r(\bar{t})$. For a vertical slab I and a vertical line ℓ in the interior of I, this procedure enables us to replace I by its portion to the left or to the right of ℓ, according to the output of the procedure at ℓ. The decision procedure takes $O(mn \log mn)$ time.

The shrinking of I is performed in two phases. We first enumerate all $O(mn)$ Voronoi vertices of the original diagrams, and run a binary search through them, as above. The resulting intermediate slab contains no original vertices, so the edges that cross it behave like lines. They might still intersect at $O(m^2n^2)$ points within I, but we can run a binary search through them efficiently, using the (dual version of the) *slope selection* algorithm of [7], so that each step takes only $O(mn \log mn)$ time.

The missing details are in Appendix C, and we obtain:

Theorem 6. *Given two finite point sets A, B in $\mathbb{R}^2$, with $|A| = n$ and $|B| = m$, a local minimum of the unidirectional Hausdorff RMS-distance from B to A under translation can be computed in time $O(mn \log^2 mn)$.*

The bidirectional variants can be handled in much the same way, and, omitting the details, we get:

Theorem 7. *Given two finite point sets A, B in $\mathbb{R}^2$, with $|A| = n$ and $|B| = m$, a local minimum of the L_1-bidirectional or the L_∞-bidirectional Hausdorff RMS-distance between A and B under translation can be computed in $O(mn \log^2 mn)$ time.*

References

1. P. K. Agarwal, S. Har-Peled, M. Sharir and Y. Wang, Hausdorff distance under translation for points, disks, and balls, *ACM Trans. on Algorithms* 6 (2010), 1–26.
2. A. Asinowski, B. Keszegh and T. Miltzow, Counting Houses of Pareto Optimal Matchings in the House Allocation Problem, arXiv:1401.5354v2.
3. J. Balogh, O. Regev, C. Smyth, W. Steiger and M. Szegedy, Long monotone paths in line arrangements, In *Discrete Comput. Geom.* 32 (2003), pp. 167–176
4. S. Belongie, J. Malik and J. Puzicha, Shape matching and object recognition using shape contexts, *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (2001), 509–522.
5. M. de Berg, M. van Kreveld, M. Overmars and O. Schwarzkopf, Computational Geometry, Algorithms and Applications. Springer-Verlag (1997); second edition (2000).
6. P. J. Besl and N. D. McKay, A method for registration of 3-d shapes, *IEEE Trans. Pattern Anal. Mach. Intell.* 14 (1992), 239–256.
7. R. Cole, J. Salowe, W. Steiger, and E. Szemerédi, An optimal-time algorithm for slope selection, *SIAM J. Comput.* 18 (1989), 792–810.
8. A. Dumitrescu, G. Rote and C.D. Tóth, Monotone paths in planar convex subdivisions and polytopes, in *Discrete Geometry and Optimization*, Károly Bezdek, Antoine Deza, and Yinyu Ye, editors, Fields Institute Communications 69, Springer-Verlag, 2013, pp. 79–104.
9. J. Edmonds and R. M. Karp., Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems, J. ACM 19, 2 (April 1972), pp. 248–264.
10. E. Ezra, M. Sharir and A. Efrat, On the ICP Algorithm, *Comput. Geom. Theory Appl.* 41 (2008), 77–93.
11. M. Henze, R. Jaume, and B. Keszegh, On the complexity of the partial least-squares matching Voronoi diagram, in *Proc. 29th European Workshop Comput. Geom. (EuroCG'13)*, pp. 193–196, 2013.
12. I. Jung and S. Lacroix, A robust interest points matching algorithm, in *Proc. ICCV'01*, volume 2, pages 538–543, 2001.
13. H. W. Kuhn, The Hungarian method for the assignment problem, *Naval Research Logistics Quarterly* 2(1–2) (1955), 83–97.
14. J. M. Phillips and P. K. Agarwal, On bipartite matching under the RMS distance, in *Proc. 18th Canadian Conf. Comput. Geom. (CCCG'06)*, pp. 143–146, 2006.
15. G. Rote, Partial least-squares point matching under translations, in, *Proc. 26th European Workshop Comput. Geom. (EuroCG'10)*, pp. 249–251, 2010.
16. G. Rote, Long monotone paths in convex subdivisions, in *Proc. 27th European Workshop Comput. Geom. (EuroCG'11)*, pp. 183–184, 2011.
17. L. S. Shapley and H. Scarf, On cores and indivisibility, *J. Math. Economics*, 1(1974), 23–37.
18. S. Umeyama, Least-squares estimation of transformation parameters between two point patterns, *IEEE Trans. Pattern Anal. Mach. Intell.* 13(4) (1991), 376–380.
19. K. Zikan and T. M. Silberberg, The Frobenius metric in image registration, in L. Shapiro and A. Rosenfeld, editors, *Computer Vision and Image Processing*, pp. 385–420, Elsevier, 1992.

A Proofs and remarks in Section 2

Proof (First part of Theorem 2.). Recall that the goal is to identify a convex subdivision K such that in each of its regions the ordered preference lists $L_t(b)$

of neighbors of each $b+t$, according to their distance from $b+t$, are fixed for all $b \in B$. We claim that the complexity of K is only polynomial; specifically, it is $O(n^2m^4)$.

In order to see this, fix $b \in B$, and consider the coarser subdivision $\mathcal{V}(b, A)$, in which only the single list $L_t(b)$ is required to be fixed withing each cell. A naive way of bounding the complexity of $\mathcal{V}(b, A)$ is to draw all the $O(n^2)$ bisectors between the pairs of points in $A - b$, and form their arrangement. Each cell of the arrangement has the desired property, as is easily checked. As a matter of fact, this naive analysis can be applied to the entire structure, over all $b \in B$. Altogether there are $O(mn^2)$ such bisectors, and their arrangement thus consists of $O(m^2n^4)$ regions.

To obtain the improved bound asserted above, we note that it suffices to draw only relevant portions of the bisectors. Specifically, let $b \in B$ and $a, a' \in A$. In view of Lemma 4(ii), we need to consider only the portion of the bisector $\beta_{a-b,a'-b}$ between $a - b$ and $a' - b$ that consists of those points t such that a and a' are among the m nearest neighbors of $b + t$ in A; other portions of the bisector are "transparent" and have no effect on the structure of K.

In general, the relevant portion of a bisector $\beta_{a-b,a'-b}$ need not be connected. To simplify the analysis, we will bound the number of (entire) bisectors of this form whose relevant portion is nonempty. Moreover, we will carry out this analysis for each $b \in B$ separately.

This analysis can be carried out via the Clarkson-Shor technique, albeit in a somewhat non-standard manner. Specifically, with b fixed, we have the set A of n points, and a system of bisectors $\beta_{a-b,a'-b}$, each defined by two points $a, a' \in A$. Each bisector $\beta_{a-b,a'-b}$ has a *conflict set*, which we define to be a smallest subset A' of A, such that there exists a point t on the bisector, whose two nearest neighbors in $A \setminus A'$ are a and a'. Clearly, the conflict set is not uniquely defined, but this is fine for the Clarkson-Shor technique to apply, because, if we draw a random sample R of A, it still holds that the probability that $\beta_{a-b,a'-b}$ will generate an edge of the Voronoi diagram of $R - b$ is *at least* the probability that a and a' are chosen in R and none of the points in the specific conflict set is not chosen. This lower bound suffices for the Clarkson-Shor technique to apply, and it implies that the number of bisectors that contribute a portion to $\mathcal{V}(b, A)$ (each of which has a conflict set of size at most m) is $O(m^2)$ times the complexity of the Voronoi diagram of $R - b$, for a random sample R of n/m points of A. That is, the number of such bisectors is $O(mn)$, instead of the number $O(n^2)$ of all bisectors. The claim about the complexity of K is now immediate. □

Proof (Proposition 1.). We construct a set of lists such that for every $i \in \{1, \ldots, m\}$ the $\lfloor \frac{m}{2} \rfloor$ smallest elements S are the same (and in the same order). For the position $\lfloor \frac{m}{2} \rfloor + 1$ of the lists, we use a set S' of m elements such that $S \cap S' = \emptyset$. Given a permutation λ of $\{1, \ldots, n\}$, consider the matching assigning to each $i \in \{1, \ldots, m\}$ the first element in its list, in the order λ, that was not assigned to any previous element. It is easy to see that this matching is stable and that its image consists of S and the subset of S' corresponding to the last $\lceil \frac{m}{2} \rceil$ positions of λ. Therefore, every subset of S' of size $\lceil \frac{m}{2} \rceil$ is, together

with S, the image of a stable matching. Hence, $\binom{m}{\lceil \frac{m}{2} \rceil} = \Omega\left(\frac{2^m}{\sqrt{m}}\right)$ different sets correspond to images of stable matchings. □

Remark: Convex paths. In order to gain better understanding of how the potential Φ (defined in the proof for Lemma 3iv) changes when we translate the set B along a convex path, we consider a standard dual construction [5], where we map the points $a \in A$ to lines in the following manner:

$$a = (a_x, a_y) \longmapsto y = a_y x + a_x.$$

The duality is order persevering in the sense that, given two points a_1, a_2, and a direction $u = (u_x, u_y)$ in the primal plane, then it can be easily checked that $a_2 \cdot u > a_1 \cdot u$ if and only if a_2 is above a_1 at the x-coordinate that corresponds to u_y/u_x in the dual plane, i.e., to the direction of u. Thus, we get an arrangement of n lines, in which the height of the lines in each x-coordinate in the dual plane represents the order of A along the corresponding direction in the primal plane; see Figure 1 for an illustration.

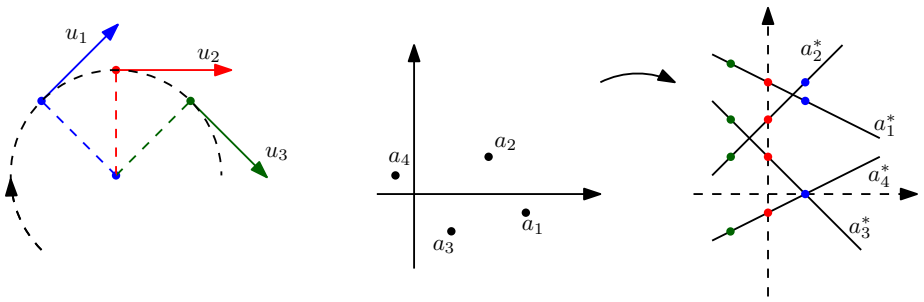

Fig. 1: An example for a convex path with three highlighted directions, a set of four given points, and the resulting dual arrangement.

Furthermore, for each point on a convex path, we can mark at the corresponding x-coordinate in the dual plane, the m dual lines that correspond to the m points which are currently (optimally) matched. By the observations in the proofs for Rote's Theorem 1 in [15], and in Lemma 3iv, if the matching changes, it must be that some point in B changes its matching to a point that is further to the right. In our dual setting, it simply means that when a matching changes, the points can only skip upwards to a line that passes above them. The sum of the indeces of the marked lines (those that participate in the matching) is exactly the rank defined in the proof of Theorem 1.

This dual setting also demonstrates how and when the potential Φ that is defined in 3iv could drop — it happens when two lines intersect in the dual plane, and thus the height of a matched point (its rank) can drop by 1. If

one can bound the amount of such drops, i.e., for m points moving along the dual n lines, from left to right, skipping from line to line only upwards, then it immediately gives a bound for the amount of intersections of a convex path with $\mathcal{D}_{B,A}$. Unfortunately, an almost quadratic lower bound for the complexity of such monotone paths was in fact presented in [3], and thus it seems hopeless to get a better upper bound for the amount of intersections of a convex path with $\mathcal{D}_{B,A}$ than the one in Lemma 3iv, without exploiting any additional geometric properties.

B The procedures used in Section 3

Minimum partial matching at a fixed translation: The Hungarian algorithm. The Hungarian method, developed by Kuhn in 1955 [13], is an efficient procedure for computing a perfect maximum weight (or, for us, minimum weight) bipartite matching between two sets A, B of equal size m, with running time $O(m^4)$, which has been improved to $O(m^3)$ by Edmond and Karp [9]. The original algorithm proceeds iteratively, starting with an empty set M_0 of matched pairs. In the i-th iteration it takes the current set M_{i-1} of $i-1$ matched pairs, and transforms it into a set M_i with i matched pairs, until it obtains the desired optimal perfect matching with m pairs.

Let us sketch the technique for minimum-weight matching, which is the one we want. The i-th iteration is implemented as follows. Define D to be the (bipartite) *directed* graph, with vertex set $A \cup B$, whose edges are the edges of M_{i-1} directed from B to A, and the edges of $(A \times B) \setminus M_{i-1}$, directed from A to B. We look for an augmenting path p that starts at B and ends at A, of minimum weight, and we set $M_i := M_{i-1} \oplus p$ (here $\oplus$ denotes symmetric difference).

To find p, we run the Bellman-Ford algorithm for shortest paths on the directed graph D, with a suitable initialization of the so-called *distance labels*. The main operation of the algorithm is the RELAX operation, which, for an edge (u, v) of D, compares $l(u)+w(u, v)$ with $l(v)$, and updates $l(v)$ if $l(u)+w(u, v)$ is smaller; here the quantities $l(u)$, $l(v)$ are the distance labels that the algorithm maintains; upon termination, $l(v)$, for $v \in A \cup B$, will be the minimum weight of a path that starts at some vertex of B and ends at v.

In our case there will be one instance where we will want to run the Hungarian algorithm with $|B| < |A|$, which can be done with a suitable padding of B to make it have the same size as A; we omit the routine details of this extension. All other invocations of this procedure will be with sets A, B of equal size; see below for details.

Hence, to recap, given a translation t, we can compute $M(B + t, A)$ by the above algorithm, where the weight of an edge $(a, b) \in A \times B$ is $\|b + t - a\|^2$. We denote this procedure as $\Pi_0(t)$; its output is the set of matched pairs, or, in our notation, the injective assignment $\pi : B \to A$.

Computing the boundary of σ_0. Let $A_0 \subset A$ be the set of the m matched points of A, for translations $t \in \sigma_0$. A_0 can be computed by running $\Pi_0(t_0)$, for finding

the optimal complete matching M_0 for the translation t_0, in time $O(n^3)$ [9]. To find σ_1, we first compute $\partial\sigma_0$. By Lemma 2, we know that there are $O(mn)$ possible directions for the bisectors forming $\partial\sigma_0$. Moreover, when we cross an edge of σ_0, the new optimal matching M_1 that replaces M_0 is obtained from a collection of alternating paths (and possibly also cycles), where in each path we replace the edges of M_0 in the path by the (same number of) edges of M_1. The subset A_1 of the m matched points of A in M_1 is obtained by replacing, for each of these paths, the starting point a_i of the path (which belongs to A_0) by the terminal point a_j (which belongs to A_1). As shown in Lemma 2, this implies that the bisector through which we have crossed from σ_0 to the neighbor region σ_1 must be perpendicular to $a_i - a_j$, for each of the pairs a_i, a_j, one for each pair. Moreover, assuming general position, and specifically that there are no two pairs of points $(a_1, a_2), (a_3, a_4) \in A \times A$ such that $\overrightarrow{a_1a_2}$ and $\overrightarrow{a_3a_4}$ are parallel, it follows that each edge of $\mathcal{D}_{B,A}$, and specifically of $\partial\sigma_0$, corresponds to a *single* such alternating path, and to a single replacement pair (a_i, a_j). In other words, under the above general position assumption, over each edge of σ_0 only one point $a_i \in A_0$ exits the optimal matching and another point $a_j \in A \setminus A_0$ replaces it. Therefore, we can construct $\partial\sigma_0$ easily and efficiently in the following manner. For each of the m points $a_i \in A_0$, we replace it by one of the $n - m$ points $a_j \in A \setminus A_0$. For each such replacement we compute the new optimal (perfect) matching M_1 between B and $A_1 = A \setminus \{a_i\} \cup \{a_j\}$ (recall that, by Lemma 1, once A_1 is fixed, the matching M_1 is independent of the translation, so it can be computed at any translation, e.g., at t_0). We then find the bisector, by comparing (1) between the new matching M_1 and the optimal matching M_0 in σ_0. This provides us with a total of $O(mn)$ potential bisectors. We now obtain σ_0 as the intersection of the $O(mn)$ halfplanes, bounded by these bisectors and containing t_0. This takes $O(mn \log mn)$ additional time.

Note that for each edge on $\partial\sigma_0$ we also know the optimal assignment on its other side. The overall cost of the procedure is $O(mn \cdot m^3) = O(m^4 n)$, since it runs the Hungarian algorithm $O(mn)$ times, each time on two sets of size m, and the cost of the other steps is dominated by this bound.

Solving $\Pi_1(\ell)$. Let ℓ be a given line in $\mathbb{R}^2$; without loss of generality assume ℓ to be vertical. We start at some arbitrary point $t_0 \in \ell$, run $\Pi_0(t_0)$, and obtain the optimal injective assignment π_0 for the partial matching between $B + t_0$ and A. We now proceed from t_0 upwards along ℓ, and seek the intersection of this ray with the boundary of the region σ_0 of $\mathcal{D}_{B,A}$ that contains t_0. Finding this intersection will also identify the next region of the subdivision that ℓ crosses into, and we will continue in this manner, finding all the regions of $\mathcal{D}_{B,A}$ that the upper ray of ℓ crosses. In a fully symmetric manner, we find the regions crossed by the lower ray, altogether $O(mn)$ regions, by Theorem 1.

To find the intersection t^* of the upper ray of ℓ with $\partial\sigma_0$, we apply a simplified variant of the procedure for computing $\partial\sigma_0$. That is, we construct the $O(mn)$ potential bisectors between σ_0 and the neighboring regions, exactly as before. The point t^* is then the lowest point of intersection of ℓ with all these bisectors lying above t_0. We repeat this process for each new region that we encounter,

and do the same in the opposite direction, along the lower ray from t_0, until we find all the regions of $\mathcal{D}_{B,A}$ crossed by ℓ.

The number of regions is $O(mn)$. We compute the explicit expression for $F(t)$ in each of them, and thereby find the global minimum $\bar{t}$ of F along ℓ. Finally, we compute $\frac{\partial F}{\partial x}(\bar{t})$ (which is a linear expression in t, readily obtained from the explicit quadratic expression for F in the neighborhood of $\bar{t}$). We note that $\bar{t}$ cannot be a breakpoint of F (that is, lie on an edge of $\mathcal{D}_{B,A}$), since a local minimum in $\bar{t}$ implies that $F(t)$ in both neighboring regions is decreasing towards $\bar{t}$, but no bisecting edge can pass through such a point. If it is negative (resp., positive), we conclude that F attains lower values than its minimum on ℓ to the right (resp., left) of ℓ, and we report this direction. If the derivative is 0, we have found a local minimum of F and we stop the whole algorithm.

The cost of $\Pi_1(\ell)$ is $O(mn \cdot mn \cdot m^3) = O(m^5n^2)$, as we encounter $O(mn)$ regions along ℓ, and for each of them we examine $O(mn)$ potential bisectors, each of which is obtained by running Π_0, in $O(m^3)$ time.

Running time of the algorithm The running time of the whole algorithm is dominated by the cost of constructing the $O(mn)$ regions that the final slab I^* crosses. Each region is constructed in $O(m^4n)$ time, after which we run a Π_1-guided binary search through its vertices, in time $O(m^5n^2 \log mn)$. Multiplying by the number of regions, we get a total running time of $O(m^6n^3 \log mn) = O(m^6n^3 \log n)$. Note that the overall course of the algorithm is incremental in nature. That is, every time we need to compute a matching, we have available the subset of A that participates in the matching, which is obtained from the preceding subset by removing an element and adding a new element. This is fine except for the initial call to the matching procedure, where we run it on $B + t_0$, for a suitable translation t_0, and the entire A. This call costs $O(n^3)$ time, which is negligible in comparison to the total running time.

C Missing details in Section 4

We begin with the simpler one-dimensional case.

The N_A function. The function N_A is a step function with the following structure. Assume that the elements of A are sorted as $a_1 < a_2 < \cdots < a_n$, and put $\mu_i = \frac{a_i + a_{i+1}}{2}$, for $i = 1, \ldots, n-1$. Then, breaking ties in favor of the larger point, we have:

$$N_A(x) = \begin{cases} a_1 & \text{for } x < \mu_1 \\ a_i & \text{for } i = 2, \ldots, n-1 \text{ and } \mu_{i-1} \le x < \mu_i \\ a_n & \text{for } x \ge \mu_{n-1} \ . \end{cases}$$

See Figure 2 for an illustration.

Moreover, each of the m functions $N_A(b_i + t)$ is just a copy of N_A, x-translated to the left by the respective b_i.

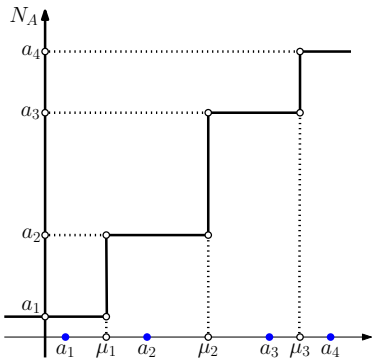

Fig. 2: The step function N_A.

The one-dimensional unidirectional case. Denote $RMS(B + t, A)$ as $r(t)$ for short, regarding A and B as fixed. We also use the shorthand notation $N_{A,i}(t)$ for $N_A(b_i + t)$, the nearest neighbor in A of $b_i + t$, for $b_i \in B$, and $t \in \mathbb{R}$. We thus want to compute a (local) minimum of the function

$$r(t) = \sum_{i=1}^{m} (b_i + t - N_{A,i}(t))^2.$$

We observe that $r(t)$ is continuous and piecewise differentiable (except at the points of discontinuity of the step functions $N_{A,i}(t)$) and each of its pieces is a parabolic arc. For any t, which is not one of these singular points, also referred to as *breakpoints*, the derivative of each of the step functions is 0. Hence we have, for any non-singular local minimum t of r,

$$r'(t) = 2 \sum_{i=1}^{m} (b_i + t - N_{A,i}(t)) = 0. \tag{2}$$

Clearly, for any given (non-singular) translation t_0, $r(t_0)$ and $r'(t_0)$ can be computed (from scratch) in $O(m \log n)$ time. This also holds for the left and right one-sided derivatives of $r(t_0)$, at a breakpoint t_0, denoted respectively as $r'(t_0)^-$ and $r'(t_0)^+$.

We note that a local minimum of $r(t)$ cannot occur at a singular point. Indeed, for the local minimum to occur at a breakpoint t_{step}, we must have $r'(t_{step})^- \leq 0$ and $r'(t_{step})^+ \geq 0$. However, referring to equation (2), one easily verifies that the value of $r'(t)$ can only decrease at t_{step}, contradicting the above inequalities.

Another simple observation is that if there is an interval $I = [t_1, t_2]$, such that $r'(t_1)^+ < 0$ and $r'(t_2)^- > 0$, then there exists a local minimum of $r(t)$ inside I. Our algorithm starts with a large interval with this property, and shrinks it repeatedly, while ensuring that it continues to contain a local minimum. At

every step of the shrinking process, the number of breakpoints of $r(t)$ over I reduces by (at least) half, and the process terminates when I contains only $O(\max\{m,n\})$ breakpoints. At this point it is straightforward to calculate a local minimum in linear time, e.g., by constructing the explicit representation of r over I and by searching each of its $O(\max\{m,n\})$ smooth parabolic subgraphs for a local minimum, recalling the property that the explicit expression for a smooth parabolic portion of $r(t)$ can be obtained in $O(1)$ time from the expression for the preceding portion.

Assuming after relabelling that $b_1 < \ldots < b_m$, set $t_1 = \mu_1 - b_m$, and $t_2 = \mu_{n-1} - b_1$. We start with $I = [t_1, t_2]$. It is easily checked that $r(t)$ has no breakpoints outside I, and it is in fact decreasing for $t < t_1$ and increasing for $t > t_2$. Thus I contains the global minimum of $r(t)$.

We next describe the procedure for shrinking I, i.e., computing a subinterval $I' \subset I$, such that the number of breakpoints of $r(t)$ over I' is (approximately) half the number of breakpoints over I, while maintaining the invariant that $r'(t)^+ < 0$ at the left endpoint of I', and $r'(t)^- > 0$ at the right endpoint. Such a "halving" of I is performed in a single iteration of the procedure, and since we start with $O(mn)$ breakpoints, and finish with $\Theta(\max\{m,n\})$ breakpoints, the algorithm executes $O(\log\min\{m,n\})$ iterations.

The shrinking process is performed as follows.
(1) Each iteration starts with an interval $I = [t_1, t_2]$ such that $r'(t_1)^+ < 0$ and $r'(t_2)^- > 0$ (where the initial values of t_1 and t_2 are given above). We calculate, for each $i = 1, \ldots, m$, the median step ξ_i of $N_{A,i}$ among its steps within I, which can be done in $O(\log n)$ time. These m median steps are thus found in total time $O(m \log n)$. We sort them into a list $L = (\xi_1, \ldots, \xi_m)$.
(2) We perform a binary search over L for finding a local minimum of $r(t)$ between two consecutive elements of L. At each step of the search, with some value $t = \xi_k$, we compute $r'(\xi_k)^-$ and $r'(\xi_k)^+$ in $O(m \log n)$ time; as noted earlier, there are only three possible cases:

(a) If $r'(\xi_k)^- > 0$ and $r'(\xi_k)^+ > 0$, we go to the left, replacing t_2 by ξ_k.
(b) If $r'(\xi_k)^- < 0$ and $r'(\xi_k)^+ < 0$, we go to the right, replacing t_1 by ξ_k.
(c) If $r'(\xi_k)^- > 0$ and $r'(\xi_k)^+ < 0$, it does not matter where to go—there are local minima on both sides; we go to the left, say, resetting t_2 as in (a).

In this way, we maintain our invariant. At the end of the binary search, we get an interval $I' = [\xi_j, \xi_{j+1}]$ with $r'(\xi_j)^+ < 0$ and $r'(\xi_{j+1})^- > 0$. The progress that we have made by passing from I to I' is that, for each step function $N_{A,i}$, we got rid of at least half of its steps within I: if $\xi_k \le \xi_j$ (resp., $\xi_k \ge \xi_j$) then the leftmost (resp., rightmost) half of the steps of $N_{A,i}$ within I is discarded.
(3) We keep shrinking I, until the number of breakpoints (from all $N_{A,i}$) within I is $O(\max\{m,n\})$. We gather all the breakpoints in the final I in time $O(m \log n)$, and sort them in time $O(\max\{m,n\} \log\max\{m,n\})$. We then explicitly construct the graph of r over I, which consists of $O(\max\{m,n\})$ parabolic arcs, and search for a local minimum of r within each of these pieces. By the invariant, at least one such minimum will be found. As already mentioned, each breakpoint indicates the point b_i that changes its neighbor at the breakpoint, and therefore

it is easy to update $r(t)$, while crossing over a breakpoint, in constant time. The overall running time of step (3) is thus $O(\max\{m,n\}\log\max\{m,n\})$.

Hence, the overall running time is $O(m\log m\log n\log\min\{n,m\}+n\log n)$. When $|A|=|B|=n$, the running time is simply $O(n\log^3 n)$.

An improved algorithm. Step (2) of the preceding algorithm, performs $\log m$ binary search steps over the list of the median breakpoints of the step functions within I, in order to eliminate (at least) half of the breakpoints inside the interval, but we can get rid of a quarter of these breakpoints by just performing the first step of the search, thereby saving a logarithmic factor in the running time bound. Omitting the rather standard details, we obtain with this improvement the following result.

Minimum Hausdorff RMS-distance under translation in two dimensions. Again, we focus on the unidirectional variant, but the analysis and results extend in a straightforward manner to the bidirectional variants. Recall that, for $A=\{a_1,\ldots,a_n\}$ and $B=\{b_1,\ldots,b_m\}$ two sets in $\mathbb{R}^2$, the *minimum unidirectional Hausdorff RMS-distance under translation* from B to A is

$$RMS_T(B,A)=\min_{t\in\mathbb{R}^2}RMS(B+t,A)=\min_{t\in\mathbb{R}^2}\sum_{i=1}^{m}\|b_i+t-N_{A,i}(t)\|^2.$$

As before, we put $r(t)=RMS(B+t,A)$, for $t\in\mathbb{R}^2$, and we seek a translation $t^*\in\mathbb{R}^2$ that brings r to a local minimum.

Here too, one can compute a local minimum by applying the two-dimensional version of the ICP algorithm, but in the worst case it might perform $O(m^2n^2)$ iterations, each taking $O(m\log n)$ time [10]. Moreover, one can calculate the *global* minimum of $r(t)$ in $O(m^2n^2\log(mn))$ time, as follows.

Let $\mathcal{V}(A)$ denote the Voronoi diagram of A, and let M denote the overlay subdivision of the m shifted Voronoi diagrams, $\mathcal{V}(A-b_i)$, for $b_i\in B$, which are just copies of $\mathcal{V}(A)$, shifted by the corresponding $b_i\in B$. M has $O(m^2n^2)$ regions, and this bound is tight in the worst case [10]. M can be constructed in $O(m^2n^2\log(mn))$ time, using, e.g., a standard line-sweep technique.

Within each region τ of M, the nearest-neighbor assignments $N_{A,i}(t)$, for $i=1,\ldots,m$, are fixed for all $t\in\tau$. Hence, the graph of $r(t)$ over τ is a portion of a single paraboloid of the form $r(t)=m\|t\|^2+\langle d_\tau,t\rangle+c_\tau$, for a suitable vector d_τ and scalar c_τ. This allows us (when c_τ and d_τ are available) to find a local minimum of $r(t)$ within the interior of τ (if one exists) in constant time. One also needs to test for a local minimum over each edge and vertex of M, and this too can be done in constant time for each such feature, provided that the explicit expression for $r(t)$ over that feature is known. This expression can be updated in constant time as we move from one feature of M to a neighboring feature. All this leads to the promised computation of the global minimum of $r(t)$ in $O(m^2n^2\log(mn))$ time, and our goal is to find a local minimum much faster.

Finding a local minimum. We now present an improved algorithm that runs in time $O(mn \log^2(mn))$. Similar to the one-dimensional case, we search for a local minimum of $r(t)$ within a vertical slab I, that we keep shrinking until it contains no vertex of M in its interior. This final slab crosses M in a sequence of only $O(mn)$ regions, stacked above one another and separated by (portions of) edges of M. It is then routine to scan these regions, construct the explicit expression for $r(t)$ over each region, updating these expressions in constant time as we go from one region to an adjacent one, and searching for the global minimum within I, all in $O(mn)$ time.

A main component in our approach is a procedure that decides whether $r(t)$ has a local minimum to the left or to the right of a vertical line λ. In analogy to the analysis in Section 3, we call this procedure $\Gamma_1(\lambda)$. This decision can be made in $O(mn \log(mn))$ time, as follows. We first calculate the *global* minimum of $r(t)$ restricted to λ, by intersecting λ with the $O(mn)$ edges of the shifted Voronoi diagrams $\mathcal{V}(A - b_i)$, by sorting these intersections along λ, and by constructing the explicit expressions for $r(t)$ over each interval between two consecutive intersections, updating these expressions in $O(1)$ time as we cross from one interval to an adjacent one. Having found the global minimum $\bar{t}$ along λ, we then inspect the sign of $\frac{\partial r}{\partial x}$ at $\bar{t}$, and go in the direction where r is locally smaller than $r(\bar{t})$. All this takes $O(mn \log(mn))$ time.

Local minima along the slab boundary do not suffice. As in the case of partial matching, computing only a local minimum along each of the lines bounding some slab I, and verifying that r decreases to the right of the left local minimum and to the left of the right local minimum, is not sufficient to guarantee that $r(t)$ has a local minimum within (the interior of) I. This is illustrated in Figure 3. However, if $r(t)$ decreases to the right of the left *global* minimum and to the left of the right *global* minimum, then $r(t)$ does have a local minimum within the interior of I, as is easily checked.

We use the decision procedure Γ_1 for shrinking the slab I, while ensuring that it continues to contain a local minimum. The shrinking is done in two stages. The first stage narrows I until it has no *original* vertices of the shifted Voronoi diagrams inside it. The second stage narrows the slab further to a slab that has no vertices of M (i.e., intersection points of Voronoi edges of different diagrams) in it.

Pruning the original Voronoi vertices. The overlay M contains $s = O(mn)$ original Voronoi vertices. We sort these vertices into a list $L = (v_1, \ldots, v_s)$, by their x-coordinates.

The slab that we start with is $I = [\lambda_1, \lambda_s]$, where λ_i denotes the y-parallel line through v_i, for $i = 1, \ldots, s$. We run $\Gamma_1(\lambda_1)$ and $\Gamma_1(\lambda_s)$, computing the global minima on λ_1 and on λ_s, and inspecting the signs of $\frac{\partial r}{\partial x}$ at these minima. If the output points to a local minimum outside I, then one of the semi-x-unbounded side slabs to the left or to the right of I contains a local minimum and has no original Voronoi vertex in its interior, and we stop the first stage with that slab. Otherwise, we perform a binary search over L, where at each step of the search,

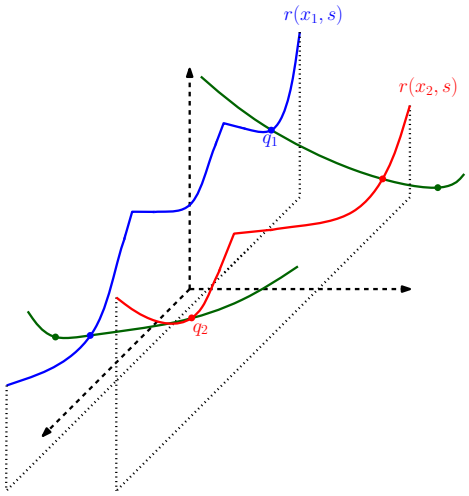

Fig. 3: Illustration of an impossible case where $r(t)$ attains a local minimum at a breakpoint.

at some vertex v_i, we run the decision procedure $\Gamma_1(\lambda_i)$ and determine which of the two sub-slabs that are split from the current slab by λ_i contains a local minimum, using the rule stated above. This stage takes $O(mn \log^2(mn))$ time.

Pruning the remaining vertices. Let I denote the final slab of the previous stage. Since I does not contain any original Voronoi vertices, every edge of any Voronoi diagram that meets I crosses it from side to side, so its intersection with I coincides with the intersection of the line supporting the edge. Let S denote the set of these lines.

The number of intersections between the lines of S within I can still be large (but at most $O(m^2n^2)$). We run a binary search over these intersections, to shrink I further to a slab between two consecutive intersections (that contains a local minimum). To guide the binary search, we use the classical *slope-selection* procedure [7] that can compute, for a given slab I and a given parameter k, the k-th leftmost intersection point of the lines in S within I, in $O(mn \log(mn))$ time. With this procedure at hand, the binary search performs $O(\log(mn))$ steps, each taking $O(mn \log(mn))$ time, both for finding the relevant intersection point, and for running Γ_1 at the corresponding vertical line. Thus, this stage takes (also) $O(mn \log^2(mn))$ time.

Computing a local minimum in the final slab. Once there are no vertices of M within I, I is crossed by at most $O(mn)$ edges of M, each crossing I from its left line to its right line. Consequently, these edges partition I into $O(mn)$ trapezoidal or triangular slices, each being a portion of a single region of M,

and is bounded by the left and right bounding lines of I, and by two consecutive edges of M (in their y-order).

We compute $r(t)$ in, say, the top slice, and its minimum in that slice. Then we traverse the slices from top to bottom, and update $r(t)$ for every slice that we encounter in constant time. In each slice, $r(t)$ attains a minimum either inside the slice or on its boundary, and we keep the smallest value of $r(t)$ that we encounter along the way. Since, by construction, the slab must contain a local minimum, we will find it.

The running time of this final computation is comprised of computing $r(t)$ once, in $O(m \log n)$ time, and afterwards updating it, in constant time, $O(mn)$ times, and computing its minimum in I. Therefore, this stage takes a total of $O(mn)$ time.

Thus, with all these components, we get the main result of Section 4.