Low Distortion Maps Between Point Sets[*]

Claire Kenyon [†] Yuval Rabani [‡] Alistair Sinclair [§]

ABSTRACT

We initiate the study of the *minimum distortion* problem: given as input two n-point metric spaces, find a bijection between them with minimum distortion. This is an abstraction of certain geometric problems in shape and image matching, and is also a natural variation and extension of the fundamental problems of graph isomorphism and bandwidth. Our focus is on algorithms that find an optimal (or near-optimal) bijection when the distortion is fairly small. We present a polynomial time algorithm that finds an optimal bijection between two line metrics, provided the distortion is less than $3 + 2\sqrt{2}$. We also give a parameterized polynomial time algorithm that finds an optimal bijection between an arbitrary unweighted graph metric and a bounded-degree tree metric.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*computations on discrete structures, geometrical problems and computations, pattern matching*

General Terms

Algorithms,Theory

Keywords

Metric spaces, low distortion embeddings, shape matching, dynamic programming

1. INTRODUCTION

Given two n-point metric spaces (U, d) and (V, d'), and a bijection $\sigma : U \to V$, the *expansion* of σ is defined as

$$\max_{x,y \in U, x \neq y} \frac{d'(\sigma(x), \sigma(y))}{d(x, y)}.$$

The *distortion* of σ is defined as the product $\text{dist}(\sigma) = \text{expansion}(\sigma) \times \text{expansion}(\sigma^{-1})$. In this paper we initiate the study of the MINIMUM DISTORTION problem: Given as input two equal-sized finite metric spaces, find a bijection between them with minimum distortion. This is a very natural variant of the bi-Lipschitz embeddings questions that were initially motivated by the study of Banach spaces, and more recently explored by theoretical computer scientists due to their extensive applications in graph theory, combinatorial optimization, learning theory, and high-dimensional computational geometry (see, e.g., [24, 8, 32, 28, 23]). From a computer science perspective, the core question of this line of work has been to bound the distortion of an *injection* from an input metric space into an implicit *infinite* host space (for example, Euclidean space). In view of the proliferation of such methods in current research, we find it surprising that our problem apparently has not been studied before.

Our research was originally inspired by shape matching and object recognition applications. For example, the goal could be to identify a hand-written number by comparing it with a set of stored prototype shapes. Matching of facial features is another obvious domain. Typical feature-based approaches to these applications first use an edge detector to extract a shape's silhouette or contours, and then represent the shape by a sample of points on the detected curve(s). Shapes are then compared by defining a distance function between pairs of point sets. (This process could be part of a bigger task of comparing large raster images involving the matching of "signatures" of local pieces; see, for example, the survey [39].) Distortion is an attractive measure of similarity between point sets which is more sensitive than many currently used approaches. For example, recent work [3] proposes matching shapes by extracting a sample of about 100 points and then attempting to match, for pairs of points from the two shapes, their histograms of distances

to other points. (See also [22] for a different similarity measure, and [6] for a bijection which approximately preserves distances under that notion of similarity.)

Closely related to our problem is recent work on matching gel electrophoresis images. Gel electrophoresis is a technique for surveying the protein contents of cells, resulting in two-dimensional images consisting of thousands of points. Given two such point sets U and V, one would like to find $U' \subset U$ and $V' \subset V$ of maximum cardinality $|U'| = |V'|$ such that U' and V' are "similar". It was proposed in [1] to use a bound on distortion as the notion of similarity. A more complicated notion of similarity involving distortion and other parameters appeared earlier in [19]. This MAXIMUM SIMILARITY problem is in a sense the dual of the minimum distortion problem. As detailed below, some of our results apply to this problem as well.

More abstractly, we find the minimum distortion problem theoretically appealing due to its close relationship to other fundamental algorithmic questions. Firstly, it is a natural optimization version of the GRAPH ISOMORPHISM problem [26, 14]. In fact, the graph isomorphism problem on input graphs G and H is trivially reduced to deciding if there exists an isometric (i.e., distortion 1) bijection between $\mathcal{M}_G$ and $\mathcal{M}_H$, where $\mathcal{M}_X$ denotes the shortest path metric of a graph X. As we shall show, the minimum distortion problem is apparently computationally harder than graph isomorphism; on the other hand, it allows one to formulate a natural notion of approximation for graph isomorphism which seems interesting in its own right. In a different direction, our problem can also be viewed as a variation and generalization of the MINIMUM BANDWIDTH problem [7, 9]. This is precisely the problem of minimizing the *expansion* of a bijection that maps $\mathcal{M}_G$ to $\mathcal{M}_P$, where G is the input graph and P is a path on the same set of vertices. We note that, typically, good solutions to the bandwidth problem incur a very large contraction, so they do not seem useful for solving the distortion problem.

Our goal in this paper is twofold: to prove some initial results about the minimum distortion problem, and to stimulate further work on it. We begin with the simple observation that the problem is NP-hard, and indeed hard to approximate within a factor better than 2, or better than 4/3 in the restricted case where one of the metrics is an unweighted tree and the other an unweighted graph. Then, in view of the potential applications mentioned above, we focus on designing polynomial time algorithms that are guaranteed to find the optimum distortion when it is small. This represents a further change in emphasis from most existing work on low distortion embeddings, which typically aims for worst case bounds for a whole class of input metrics.

In our first main result, we give an algorithm that finds the minimum distortion bijection between two *line metrics* (i.e., sets of points in one dimension), provided the distortion between them is less than $3 + 2\sqrt{2}$. In addition to being a natural (and, as it turns out, non-trivial) first step, line metrics could be used to design heuristics for higher dimensional point sets by projecting each set in several different directions and then trying to match the two collections of projected images. In fact, the main difficulty that our algorithm overcomes is that well-separated clusters of points on the line may be "flipped" without incurring large distortion. This is precisely the effect that projecting a high-dimensional point set in two slightly different directions might have. We also note that our algorithm can be adapted to solve the maximum similarity problem (see above) between two line met-

rics, thus settling a question raised in [1].

In our second main result, we present an algorithm that finds, given an arbitrary graph G and a bounded-degree tree T, a minimum distortion bijection between $\mathcal{M}_G$ and $\mathcal{M}_T$. The running time of this algorithm is polynomial in the input size, but exponential in the degree of T and doubly exponential in the distortion. This algorithm complements our hardness result mentioned earlier, which implies that this problem is NP-hard without the degree restriction on T. It can also be viewed as a parameterized complexity result [11, 10], and should be compared with the dynamic programming algorithm of [37] for recognizing small-bandwidth graphs. However, we note that both the objective function and the target space are simpler in the bandwidth case. Our algorithms are based on dynamic programming, though in both cases the dynamic programming relies on some non-trivial structure of low-distortion embeddings which we develop in the paper.

To the best of our knowledge, the minimum distortion problem has not been addressed before. Here we briefly survey previous work on related problems. The closest problem to ours is the "maximum similarity" problem, as mentioned above and formulated in [1]. In that paper, Akutsu *et al.* prove that the two-dimensional version of maximum similarity is NP-hard, where similarity is defined as having a distortion at most α, for any $1 < \alpha < 2.07$. They also present a polynomial time algorithm for the one-dimensional version under the (severe) restriction that the mapping between the maximum subsets U', V' is monotone (i.e., contains no "flips"). Graph isomorphism is one of a very short list of fundamental problems whose structural complexity remains open. The problem is clearly in NP, but unlikely to be NP-complete (such a result would collapse the polynomial hierarchy [26]). Efficient algorithms are known for several special cases, including bounded-degree graphs [30], bounded genus graphs [20, 33], circular-arc graphs [21], and graphs with bounded eigenvalue multiplicity [2]. No polynomial time algorithm is known for arbitrary graphs, even with the aid of quantum computing. Minimum bandwidth, too, is a notoriously elusive problem. It is NP-hard, even for special classes of trees [35, 15, 34]. Only recently have breakthrough results on approximating it been discovered. These give an approximation algorithm for general graphs with polylogarithmic guarantees [12, 13] and a super-constant hardness result, even for caterpillars [38] (see also [5, 4]). Better approximation bounds are known for some special cases [25, 17, 18]. There are polynomial time algorithms for recognizing small bandwidth graphs [37] and for computing the bandwidth of interval graphs [27]. Finally, as we have mentioned, most results on low distortion embeddings of finite metrics into normed or other spaces deal with worst-case bounds (see, e.g., [32, 23]), in contrast to our emphasis on optimal or near-optimal distortion for individual metrics. One notable exception is the algorithm based on semidefinite programming for finding a minimum distortion embedding of an arbitrary finite metric into Euclidean space [29].

2. PRELIMINARIES

We begin with two simple but useful observations.

PROPOSITION 2.1. *Consider a bijection σ mapping (U, d) to (V, d'). Then, $\text{dist}(\sigma)$ is invariant under scaling either d or d', is always ≥ 1, and is equal for σ and σ^{-1}. Moreover, $\text{dist}(\sigma) = 1$ if and only if d and d' are isometric.*

PROPOSITION 2.2. *The minimum distortion problem can be reduced to the following decision problem: given (U, d) and (V, d') and a real number $\alpha \geq 1$, does there exist a bijection between U and V with expansion and inverse expansion at most α?*

PROOF. Let $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$. Let σ denote an (unknown) bijection with minimum distortion. Observe that there is only a polynomial-size set of possibilities for the expansion of σ, namely, the set of $d'(v_k, v_\ell)/d(u_i, u_j)$ as v_k, v_ℓ range over V and u_i, u_j range over U; similarly there is only a polynomial-size set of possibilities for the expansion of σ^{-1}. Thus one can guess expansion(σ) and expansion(σ^{-1}). Note that scaling U by a factor of x while leaving V unchanged has the effect of multiplying the expansion of σ by $1/x$ and the expansion of σ^{-1} by x. Thus after guessing expansion(σ) and expansion(σ^{-1}), one can scale U so that the expansions of σ and of σ^{-1} are equal; call that number α. The distortion is then α^2. □

The following observation is a simple consequence of the triangle inequality, and is useful when considering metrics that are shortest-path metrics of graphs.

PROPOSITION 2.3. *Suppose (U, d) and (V, d') are the shortest-path metrics of weighted undirected graphs G and H respectively, and let σ be any bijection between U and V. Then the expansion of σ is achieved by an adjacent pair $\{u_i, u_j\}$ in G, and the expansion of σ^{-1} by an adjacent pair $\{v_k, v_\ell\}$ in H.*

Finally in this section, we briefly address the computational complexity of the minimum distortion problem. Evidently the problem is at least as hard as Graph Isomorphism (which is equivalent to the special case where both input metrics are unweighted graphs, and a distortion of 1 is sought). Not surprisingly, it is in fact apparently harder, as the following simple reductions demonstrate:

PROPOSITION 2.4. *(a) It is NP-hard to approximate the minimum distortion problem within a factor better than 2. (b) It is NP-hard to approximate the minimum distortion problem within a factor better than $4/3$ even in the restricted setting where one of the metrics is an unweighted tree metric and the other is a graph metric with edge weights $1/2$ or 1.*

PROOF. For part (a), we give a simple reduction from Hamilton Cycle. Let $G = (U, E)$ be an unweighted, undirected graph on n vertices. Construct a metric (U, d) by setting $d(u, v) = 1$ if $\{u, v\}$ is an edge of G, and $d(u, v) = 2$ otherwise. Similarly, let $C = (V, E')$ be the unweighted cycle on n vertices, and construct a metric (V, d') using the same operation. Now it is straightforward to check that, if G contains a Hamilton cycle, then an optimal bijection between (U, d) and (V, d') has distortion exactly 2. On the other hand, if there is no such cycle then any bijection must have distortion at least 4. We defer the proof of part (b) to the appendix. □

Remark: Part (b) of the above proposition further motivates our Theorem 4.1 below, in which we give a polynomial time algorithm for this same restricted setting except that we require in addition that the tree have bounded degree. (Actually in that theorem we also require the graph to be unweighted, but it is easily seen that the proof generalizes to graphs with bounded edge weights.)

In this paper, we begin to attack the minimum distortion problem for certain restricted metric spaces, with an emphasis on devising polynomial time algorithms when the distortion is small. We begin by discussing the one-dimensional problem, in which both U and V are sets of points on a line. This is a natural simplification of the important case of matching two point sets in the plane (under the Euclidean metric), which is motivated, e.g., by the shape matching application mentioned in the introduction. The one-dimensional problem turns out to exhibit some interesting structure that highlights some of the issues involved in more general cases of the problem. We will then go on to deal in Section 4 with the case where U is a bounded-degree tree metric and V is an arbitrary unweighted graph metric.

3. MINIMUM DISTORTION MAPS ON THE REAL LINE

The main result of this section is the following:

THEOREM 3.1. *Let $U = \{u_1, \ldots, u_n\}$, $V = \{v_1, \ldots, v_n\}$ be two subsets of the real numbers. Let $\alpha < \sqrt{3 + 2\sqrt{2}}$. There is a polynomial time algorithm to decide whether there exists a bijection between U and V with expansion and inverse expansion at most α.*

Applying Proposition 2.2, we immediately deduce a polynomial time algorithm for the minimum distortion problem in one dimension provided the distortion is below the threshold value $3 + 2\sqrt{2}$:

COROLLARY 3.2. *In the situation of Theorem 3.1, if the optimal bijection between U and V has distortion less than $3 + 2\sqrt{2}$, then there is a polynomial time algorithm for finding it.*

In the following subsection we develop some structural properties of low-distortion one-dimensional maps, and then in the next subsection we use these properties to devise an algorithm to find a low-distortion map.

3.1 Structural properties

We begin by introducing the key concept of a *forbidden pattern*.[1] In our context, this is a bijection between subsets of U and V whose presence implies large distortion. Let U, V be sets of n points on the real line, and let π be a fixed permutation on $\{1, \ldots, k\}$ (a "pattern"). We say that a bijection σ from U to V *contains* π if there exists a subset $u_1 < u_2 < \cdots < u_k$ of points in U such that

$$\sigma(u_i) < \sigma(u_j) \text{ if and only if } \pi(i) < \pi(j), \text{ for } 1 \leq i, j \leq k.$$

Otherwise, we say that σ *avoids* π.

Specifically, we will consider the pattern on four points shown in Figure 1. The following lemma provides a lower bound on the distortion of any bijection that contains this pattern (or its inverse).

LEMMA 3.3. *Let σ be any bijection between U and V with distortion less than $3 + 2\sqrt{2}$. Then σ must avoid the pattern in Figure 1 and its inverse.*

PROOF. Note that a bijection σ contains the pattern iff σ^{-1} contains the inverse pattern. Thus it is sufficient to

---

[1]Permutations with forbidden patterns have been intensively studied in enumerative combinatorics. See, for example, [31] and the references therein.
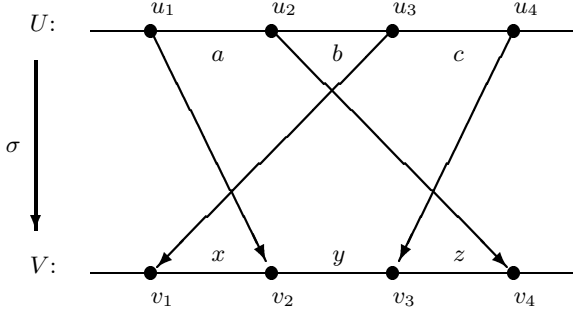
Figure 1: A forbidden pattern

show that any σ containing the pattern necessarily has distortion at least $3 + 2\sqrt{2}$. Let $u_1 < u_2 < u_3 < u_4$ be points in U occurring in the pattern, and $v_1 < v_2 < v_3 < v_4$ their images in V. Denote the inter-point distances in U, V by the parameters a, b, c, x, y, z as shown in Figure 1. Our aim is to show that the infimum, over all strictly positive values of these parameters, of the distortion of this four-point mapping is at least $3 + 2\sqrt{2}$.

First, note that we may assume without loss of generality that $a = c$ and $x = z$. For if not, we may consider the configuration in which the values of a, c and of x, z are interchanged, which by symmetry has the same distortion. Averaging these two configurations yields a configuration with $a = c$ and $x = z$, whose distortion is no larger than the orginal one.

Recall from Proposition 2.3 that the expansion of σ is achieved by an edge, i.e., in this case by an adjacent pair $\{u_i, u_{i+1}\}$: call such an edge *tight*. Now we may assume without loss of generality that all three edges are tight. For suppose, e.g., that the middle edge $\{u_2, u_3\}$ is not tight. Then we may reduce its length b (keeping $a = c$ fixed) until it becomes tight, thus keeping the expansion of σ unchanged and not increasing the expansion of σ^{-1}. A similar argument applies to the other two edges.

With the above observations, denoting the infimum distortion by α^2, by rescaling if necessary we may assume the existence of a set of distances $a = c, b, x = z, y$ such that

(i) the expansion of every edge in U under σ is exactly α;

and

(ii) the expansion of every edge in V under σ^{-1} is at most $\alpha + \epsilon$, for an arbitrarily small $\epsilon > 0$.

Now (i) implies the following equalities:

$$\begin{aligned} x + y &= \alpha a; \\ 2x + y &= \alpha b, \end{aligned} \tag{1}$$

and (ii) the following inequalities:

$$\begin{aligned} a + b &\leq (\alpha + \epsilon)x; \\ 2a + b &\leq (\alpha + \epsilon)y. \end{aligned} \tag{2}$$

Solving the equations (1) gives $x = \alpha(b - a)$ and $y = \alpha(2a - b)$, which when substituted into the inequalities (2) yields

$$\begin{aligned} a(\alpha(\alpha + \epsilon) + 1) &\leq b(\alpha(\alpha + \epsilon) - 1); \\ b(\alpha(\alpha + \epsilon) + 1) &\leq 2a(\alpha(\alpha + \epsilon) - 1). \end{aligned}$$

This in turn implies that $(\alpha(\alpha + \epsilon) + 1)^2 \leq 2(\alpha(\alpha + \epsilon) - 1)^2$ for all $\epsilon > 0$, and hence by continuity also for $\epsilon = 0$. Thus we have $(\alpha^2 + 1) \leq \sqrt{2}(\alpha^2 - 1)$, whence $\alpha^2 \geq 3 + 2\sqrt{2}$ as required. In fact this value of α is actually achieved by setting $a = 1$, $b = \frac{\alpha^2 + 1}{\alpha^2 - 1} = \sqrt{2}$, $x = 2\alpha$ and $y = 2\sqrt{2}\alpha$. $\quad\square$

We can now exploit the fact that σ does not contain the above pattern to obtain a decomposition of σ into two bijections between smaller sets:

LEMMA 3.4. *Let $U = u_1 < u_2 < \cdots < u_n$ and $V = v_1 < v_2 < \cdots < v_n$. Consider any bijection σ between U and V that avoids the pattern in Figure 1 and its inverse. Then there exists an initial subinterval $I = \{u_1, \ldots, u_m\}$ of U of cardinality $0 < m < n$ such that either $\sigma(I) = \{v_1, \ldots, v_m\}$ or $\sigma(I) = \{v_{n-m+1}, \ldots, v_n\}$. I.e., the image of I is either an initial or a final subinterval of V.*

PROOF. If $\sigma(u_1) = v_1$ or $\sigma(u_1) = v_n$, then we take $m = 1$ and we are done. Otherwise, let $v_i = \sigma(u_1)$. We now assume that $\sigma(u_n) > v_i$; the other case will be dealt with later. Let u_m be maximal such that $\sigma(u_m) < v_i$; note that $1 < m < n$. Then let v_k be maximal such that $\sigma(u_j) = v_k$ for some $j < m$, and let u_ℓ be such that $\sigma(u_\ell) = v_k$. If $k = i$ then $\sigma(\{u_1, \ldots, u_m\}) = \{v_1, \ldots, v_i\}$ and we are done, so assume $k > i$. Clearly we have $\sigma(\{u_1, \ldots, u_m\}) \subseteq \{v_1, \ldots, v_k\}$. We claim that this inclusion is actually an equality, so that σ maps $I = \{u_1, \ldots, u_m\}$ to an initial subinterval of V.

Indeed, suppose it is not. Then there must exist $p > m$ such that $\sigma(u_p) < v_k$. But this means we have four points $u_1 < u_\ell < u_m < u_p$ whose images satisfy

$$\sigma(u_m) < v_i = \sigma(u_1) < \sigma(u_p) < v_k = \sigma(u_\ell).$$

This is precisely the forbidden pattern of Figure 1, so we have a contradiction to Lemma 3.3. Thus $\sigma(\{u_1, \ldots, u_m\}) = \{v_1, \ldots, v_k\}$, which concludes the proof for the case $\sigma(u_n) > v_i$.

For the other case $(\sigma(u_n) < v_i)$, we apply the same argument as above to the reflected permutation $\sigma'(u) = n - \sigma(u) + 1$, which avoids the pattern in Figure 1 by virtue of the fact that σ avoids the inverse pattern. This establishes that σ' maps an initial subinterval I to an initial subinterval of V, and hence that σ maps I to a final subinterval of V. This concludes the proof for the second case. $\quad\square$

Lemma 3.4 suggests that, in order to solve the decision problem, one might guess the right value of m (via dynamic programming) and then solve each of the two corresponding subproblems. However, these two subproblems are not independent so this naïve approach fails. Nonetheless, Proposition 2.3 guarantees that the constraints induced by one subproblem on the other can be captured by the images of a small number of points, which again we can afford to guess. This observation allows us to formulate a dynamic programming algorithm, as described next.

3.2 The algorithm

The algorithm builds a dynamic programming table T which is boolean and indexed by the following parameters:

- a subinterval $I = \{u_m, u_{m+1}, \ldots, u_{m+c-1}\}$ of U and a subinterval $J = \{v_{m'}, v_{m'+1}, \ldots, v_{m'+c-1}\}$ of V of the same cardinality $c \geq 1$;

- two elements v and v' of J, and two elements u and u' of I.

The corresponding entry is true iff there is a bijection σ between I and J, such that $\sigma(u_m) = v$, $\sigma(u_{m+c-1}) = v'$, $\sigma^{-1}(v_{m'}) = u$ and $\sigma^{-1}(v_{m'+c-1}) = u'$, and with expansion and inverse expansion at most α.

The table is computed in order of increasing values of c. The base case $c = 1$ is trivial. For $c > 1$, to compute $T[I, J, v, v', u, u']$, we run through all partitions of I and J into two non-empty intervals, $I = I_\ell \cup I_r$ and $J = J_\ell \cup J_r$, and set our result to true if there is at least one combination such that one of the following two conditions holds:

(i) for some u_ℓ, u_r, v_ℓ, v_r, the entries $T[I_\ell, J_\ell, v, v_\ell, u, u_\ell]$ and $T[I_r, J_r, v_r, v', u_r, u']$ are both true, and also $v_r - v_\ell \leq \alpha(\min(I_r) - \max(I_\ell))$ and $u_r - u_\ell \leq \alpha(\min(J_r) - \max(J_\ell))$;

or

(ii) for some u_ℓ, u_r, v_ℓ, v_r, the entries $T[I_\ell, J_r, v, v_r, u_\ell, u']$ and $T[I_r, J_\ell, v_\ell, v', u, u_r]$ are both true, and also $v_r - v_\ell \leq \alpha(\min(I_r) - \max(I_\ell))$ and $u_r - u_\ell \leq \alpha(\min(J_r) - \max(J_\ell))$.

The two inequalities in the above conditions state that the expansion of the edge $\{\max(I_\ell), \min(I_r)\}$ and the inverse expansion of the edge $\{\max(J_\ell), \min(J_r)\}$ are each at most α. (See Figure 2.)
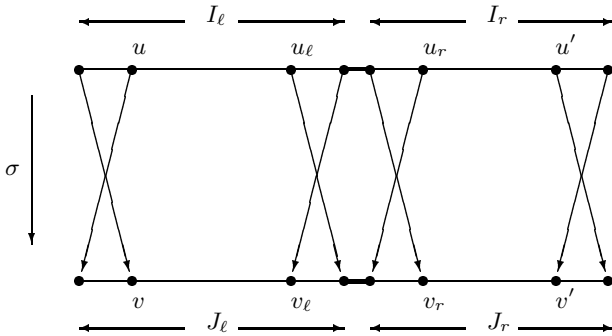

Figure 2: Filling in the table: case (i)

Once the table is filled, to decide whether there is a bijection from U to V with expansion and inverse expansion at most α, we only need to check that $T[U, V, v, v', u, u']$ is true for at least one setting of the parameters (v, v', u, u').

The correctness of the algorithm follows from Lemma 3.4 (which supplies the recursive decomposition) and Proposition 2.3 (which ensures that it is sufficient when combining subproblems to check only the expansion of the edge $\{\max(I_\ell), \min(I_r)\}$ and the inverse expansion of the edge $\{\max(J_\ell), \min(J_r)\}$). The running time can be bounded crudely by inspection. The size of the table is $O(n^7)$, and computing one entry of the table takes time $O(n^5)$, for an overall running time of $O(n^{12})$. This concludes the proof of Theorem 3.1. $\square$

3.3 Maximum similarity

Recall from the introduction the "maximum similarity" problem, which takes as input two metrics (U, d) and (V, d') and a value α, and asks for two subsets $U' \subseteq U$ and $V' \subseteq V$ of maximum cardinality such that there is a bijection between U' and V' with distortion at most α. This problem was formulated in [1], where the one-dimensional version was posed as an open question. Our algorithm above can

be easily extended to solve the maximum similarity problem for line metrics, with the same restriction on the distortion.

THEOREM 3.5. *Let $U = \{u_1, \ldots, u_n\}$, $V = \{v_1, \ldots, v_n\}$ be two subsets of the real numbers. Let $\alpha < \sqrt{3 + 2\sqrt{2}}$. There is a polynomial time algorithm to find $U' \subset U$ and $V' \subset V$ of maximum cardinality $|U'| = |V'|$, such that there exists a bijection between U' and V' with expansion and inverse expansion at most α.*

The algorithm is a simple modification of the one above, and we sketch it briefly. The dynamic programming table is indexed as before, except that now v, v', u, u' denote the images and inverse images of the minimum and maximum *matched* elements of U and V respectively. The entries of the table are integers corresponding to the maximum cardinality of a pair of subsets having a bijection within the specified distortion, and respecting the images of the extreme points. In addition to consistency conditions, when combining two entries after partitioning $I = I_\ell \cup I_r$ and $J = J_\ell \cup J_r$, the algorithm checks whether the expansion of the pair $\{\max\{u \in I_\ell, u \text{ matched}\}, \min\{u \in I_r, u \text{ matched}\}\}$ and the inverse expansion of the pair $\{\max\{v \in J_\ell, v \text{ matched}\}, \min\{v \in I_r, v \text{ matched}\}\}$ are both at most α.

4. MAPPING GRAPHS TO BOUNDED-DEGREE TREES

In this section we prove the following theorem, which gives a fixed-parameter polynomial time algorithm in similar vein to that for bandwidth given in [37].

THEOREM 4.1. *Let (U, d) be the shortest-path metric of an unweighted tree T of maximum degree b, and (V, d') the shortest-path metric of an arbitrary unweighted graph G'. Then for any fixed constants b and $\alpha \geq 1$, there is an $O(n^2)$ time algorithm that decides whether there exists a bijection between U and V with expansion and inverse expansion at most α.*

As will become evident shortly, the running time is exponential in the degree b and doubly exponential in the distortion α (actually it is about $\exp(b^{O(\alpha^3)})$).

Note that we cannot immediately appeal to Proposition 2.2 to deduce the existence of a polynomial time algorithm for finding a minimum distortion bijection in this case. This is because the reduction to the decision problem given in Proposition 2.2 uses *scaling* of the metrics, while Theorem 4.1 holds only for unweighted graph metrics. However, as will become clear below, a simple modification of our algorithm for Theorem 4.1 can be used to test for the existence of a bijection with expansion α_1 and inverse expansion α_2 (for any constants $\alpha_1, \alpha_2 \geq 1$). Given this observation, it is then easy to determine a minimum distortion bijection by trying the polynomially many candidate pairs of integer values for α_1 and α_2. Accordingly, we get:

COROLLARY 4.2. *In the situation of Theorem 4.1, for any fixed constant c, if the optimal bijection between U and V has distortion at most c then there is a polynomial time algorithm for finding it.*

Remarks: (i) It is not hard to check that the algorithm still works, with minor modifications, when the edges of G are given weights in some bounded range. (ii) We have not been able to adapt the algorithm to handle the maximum similarity problem, as we did for line metrics in Section 3.3.

4.1 Structural properties

The algorithm relies on several structural properties spelled out in the following lemma. For a vertex $u \in T$ (resp., of G'), let $\mathcal{B}(u, \ell)$ (resp., $\mathcal{B}'(u, \ell)$) denote the closed ball of radius ℓ around u in T (resp., in G'); and for a subset of vertices $A \subseteq T$ (resp., in G'), we denote by $\Gamma(A)$ (resp., $\Gamma'(A)$) the set of neighbors of A that lie outside A. We also assume throughout that T is rooted at some arbitrary vertex r_0, and for any vertex r of T we denote by T_r the subtree rooted at r. We shall abuse notation slightly by identifying subsets of vertices of T or of G' with the subgraphs that they induce, where convenient.

LEMMA 4.3. *Let* $\sigma : U \to V$ *be a bijection with expansion and inverse expansion at most* α. *Then*

1. G' *has maximum vertex degree at most* b^α.

2. *For any* $r \in T$, *each connected component of* $G \setminus \mathcal{B}'(\sigma(r), \alpha^2)$ *lies either entirely in* $\sigma(T_r)$ *or entirely in* $G' \setminus \sigma(T_r)$.

3. *For any* $r \in T$, *for any adjacent pair* (u', v') *in* G' *with* $u' \in \sigma(T_r)$ *and* $v' \notin \sigma(T_r)$, *both* $\sigma^{-1}(u')$ *and* $\sigma^{-1}(v')$ *are in* $\mathcal{B}(r, \alpha - 1)$.

PROOF. To prove the first statement, note that for any $v \in V$ the expansion of σ^{-1} implies $\sigma^{-1}(\mathcal{B}'(v, 1)) \subseteq \mathcal{B}(\sigma^{-1}(v), \alpha)$, and the cardinality of this latter ball is at most b^α by the degree bound on T.

To prove the second statement, let $v' = \sigma(v)$ be a vertex of $\Gamma'(\sigma(T_r))$. By definition, v' is adjacent to some vertex $u' = \sigma(u)$ of $\sigma(T_r)$. From the expansion of σ^{-1} we have $d(u, v) \leq \alpha$. Since $u \in T_r$ and $v \notin T_r$, and since T is a tree, the shortest path from u to v goes through r, and thus $d(r, v) \leq \alpha$. Hence, using the expansion of σ, we have $d'(\sigma(r), v') \leq \alpha^2$. Equivalently,

$$\Gamma'(\sigma(T_r)) \subseteq \mathcal{B}'(\sigma(r), \alpha^2),$$

which after a moment's thought is seen to imply the second statement.

To prove the third statement, we have $d'(u', v') = 1$ by assumption, which by the expansion of σ^{-1} implies that $d(\sigma^{-1}(u'), \sigma^{-1}(v')) \leq \alpha$. As above, this in turn implies that $d(\sigma^{-1}(u'), r) \leq \alpha - 1$ and $d(\sigma^{-1}(v'), r) \leq \alpha - 1$, as required. $\square$

We are now ready to specify the dynamic programming algorithm.

4.2 The algorithm

The algorithm begins by verifying that $|U| = |V| = n$ and that the maximum vertex degree in G' is at most b^α (as required by the first property in Lemma 4.3). If not, it rejects.

Now root T at some arbitrary vertex r_0. The algorithm builds a dynamic programming table $\mathcal{T}$ which is boolean and indexed by the following parameters:

- $r \in \{u_1, \ldots, u_n\}$, the root of a subtree T_r of T (wrt the rooting r_0 of T).

- $r' \in \{v_1, \ldots, v_n\}$, an injection τ from $\mathcal{B}(r, \alpha) \cap T_r$ into $\mathcal{B}'(r', \alpha^2)$, and a subset S of the vertices of G' with the property that each connected component of $G' \setminus \mathcal{B}'(r', \alpha^2)$ lies entirely within S or entirely outside S.

The corresponding entry is true iff there exists an *injection* $\sigma : T_r \hookrightarrow G'$ such that $\sigma(r) = r'$, σ coincides with τ on $\mathcal{B}(r, \alpha) \cap T_r$, and $\sigma(T_r) = S$, and such that the expansion of every edge of T_r, and the inverse expansion of every edge of $\sigma(T_r)$, are each at most α. Observe that inverse expansion is guaranteed only between neighbors in (and hence within connected components of) the image $\sigma(T_r)$.

Informally, the idea here is that we memorize in detail the restriction of the map σ to the ball of radius α around r. For the rest of the map, we memorize only the set of points of G' that are in the image. The second part of Lemma 4.3 ensures a compact encoding, as described below. As we shall see presently, detailed local knowledge of σ around r is sufficient to allow us to combine the maps from different subtrees and check the distortion as we go.

The table is computed by considering the subtree roots r in bottom-up order. Let $\{r_i\}$ be the children of a given root r. To compute $\mathcal{T}(r, r', \tau, S)$, we run over all combinations of entries $\{\mathcal{T}(r_i, r'_i, \tau_i, S_i)\}_i$ all of which have value true. We set our result to be true if at least one of these combinations satisfies the three conditions below, and false otherwise. We may of course assume that the indices (r, r', τ, S) are self-consistent, i.e., that $\tau(r) = r'$ and that the image of τ is contained in S.

1. The map τ is consistent with all the maps τ_i, the S_i are disjoint and do not include r', and S is the union of the S_i plus the vertex r'.

2. For each r'_i, we have $d'(r', r'_i) \leq \alpha d(r, r_i)$.

3. For each adjacent pair v', w' in G', that belong to different sets S_i (or with $v' = r'$), both v' and w' are in the image of τ and satisfy $d(\tau^{-1}(v'), \tau^{-1}(w')) \leq \alpha$.

Once the dynamic programming table is filled, to produce the final output the algorithm simply checks whether some table entry $\mathcal{T}(r_0, \cdot, \cdot, \cdot)$ is true.

4.3 Analysis

Running time

Note first that the degree bound on G' in the first part Lemma 4.3 implies that $\mathcal{B}'(v, \alpha^2)$ has size at most b^{α^3} for any v.

We claim that the size of table $\mathcal{T}$ is at most

$$n \times n \times (b^{\alpha^3})^{b^\alpha} \times 2^{2b^{\alpha^3}} = O(n^2)$$

The first two factors arise from r, r' respectively, while the third factor bounds the number of maps from $\mathcal{B}(r, \alpha)$ to $\mathcal{B}'(r', \alpha^2)$. The fourth factor bounds the number of possibilities for the set S: from the second part of Lemma 4.3, note that S can be specified by a subset of points in the ball $\mathcal{B}'(r', \alpha^2)$ together with a subset of the connected components of $G' \setminus \mathcal{B}'(r', \alpha^2)$ (of which there are at most b^{α^3}). We observe that the constant hidden in the $O(\cdot)$ is exponential in b and doubly exponential in α, as mentioned earlier.

How long does it take to fill in one entry of table $\mathcal{T}$? Given r and r', we only have to consider combinations such that $r'_i \in \mathcal{B}'(r', \alpha)$, so the number of combinations is $O(1)$. For any given combination $\{\mathcal{T}(r_i, r'_i, \tau_i, S_i)\}_i$, the three conditions can be checked in constant time. Condition 1 involves checking consistency of a fixed number $(b + 1)$ of functions on a fixed-size domain, and of $b+1$ sets each of which is compactly represented as described earlier. Condition 2 requires

just b simple comparisons. And condition 3 requires one comparison for each of at most $(b^\alpha)^2$ pairs $\tau^{-1}(v'), \tau^{-1}(w')$.

The overall running time is thus $O(n^2)$.

Correctness

We prove by induction (bottom-up in T) that, as claimed in the specification of the algorithm, the following property holds: a table entry of $\mathcal{T}$ is true iff there exists an injection $\sigma : T_r \hookrightarrow G'$ such that $\sigma(r) = r'$, σ coincides with τ on $\mathcal{B}(r, \alpha) \cap T_r$, and $\sigma(T_r) = S$, and such that the expansion of every edge of T_r under σ, and the inverse expansion of every edge of $\sigma(T_r)$, are each at most α.

First, if there exists an injection σ from T_r satisfying the above property, then σ induces injections σ_i from T_{r_i} also satisfying the property, by induction the corresponding table entries will be true, and all the conditions checked by the algorithm will obviously be satisfied except for the one stating that v', w' are in the image of τ; but one can readily verify that this condition follows from the third part of Lemma 4.3 applied to r_i. Thus the corresponding table entry for r will be set to true, as claimed.

Conversely, suppose that we find entries for the r_i which are all true and whose combination satisfies the three conditions checked by the algorithm. By induction, each of those entries has a corresponding injection $\sigma_i : T_{r_i} \hookrightarrow G'$ satisfying the above property. Since these injections are all consistent (by condition 1), we can combine them and extend them to an injection $\sigma : T_r \hookrightarrow G'$ by adding the image $\sigma(r) = r'$. This σ will be consistent with τ and S, so it remains only to verify its expansion and inverse expansion properties. The expansion of edges $\{r, r_i\}$ is checked explicitly in condition 2, and all other edges lie within one of the T_{r_i} and hence have bounded expansion by induction. For the inverse expansion, consider any edge $\{v, w\}$ of $\sigma(T_r)$. If $\sigma^{-1}(v)$ and $\sigma^{-1}(w)$ both lie in the same subtree T_{r_i}, then the inverse expansion is bounded by induction. Otherwise, condition 3 ensures that their inverse expansion does not exceed α.

Finally, note that a bijection between T and G' of the desired form exists iff some table entry $\mathcal{T}(r_0, \cdot, \cdot, \cdot)$ is true, where r_0 is the root of the entire tree T. This follows from the fact that the corresponding injection σ must in fact be a bijection since $|U| = |V|$, and must have expansion at most α on all edges of T and inverse expansion at most α on all edges of G', hence by Proposition 2.3 overall expansion and inverse expansion at most α.

This completes the verification of the algorithm, and hence the proof of Theorem 4.1. $\quad\square$

5. OPEN QUESTIONS

In this paper we have examined a new set of issues related to finite metrics, and have obviously raised many more questions than we have resolved. Below we mention some of the most pertinent open problems that arise from our investigations.

1. As mentioned earlier, it would be of great interest in geometric applications to extend our one-dimensional result of Section 3 to two dimensions, for example by proving a two-dimensional analog of Theorem 3.1, with some constant $c > 1$ replacing $3 + 2\sqrt{2}$. As noted in Section 3.3, our one-dimensional algorithm can be modified to handle the maximum similarity problem. On the other hand, Akutsu et al. [1] prove that the two-dimensional maximum similarity problem is NP-

hard when similarity means distortion at most α, for any $1 < \alpha < 2.07$. This may indicate that the two-dimensional minimum distortion problem is hard even for small distortion, or at least that it requires a different approach. In this connection, we note also that very recently Papadimitriou and Safra [36] have proved that the three-dimensional version of the problem (i.e., deciding whether there is a bijection of distortion at most α between two point sets in three-dimensional Euclidean space) is NP-hard for any fixed α. (They also show that it is NP-hard to approximate the minimum distortion in this setting within a factor of 3.) Thus we cannot hope for a three-dimensional result of similar strength to our one-dimensional one.

2. In Section 4 we gave an algorithm that finds an optimal bijection between an arbitrary graph metric and a bounded-degree tree metric. Can one extend this result to a wider class of graph metrics than bounded-degree trees, possibly at the cost of some approximation in the optimal distortion?

3. Our algorithm for the one-dimensional problem in Section 3 was based on identifying a specific forbidden pattern whose presence guarantees large distortion (at least $3 + 2\sqrt{2}$). It is tempting to suggest that this approach can be extended to construct a parameterized polynomial time algorithm for all values of the distortion. Indeed, one can construct a family of forbidden patterns, of increasing sizes, which guarantee arbitrarily large distortion. However, it does not seem immediately clear how to use these larger patterns algorithmically as we did for the four-point pattern in Lemma 3.4.

4. Our emphasis has been on algorithms that find optimal solutions in cases where the optimal distortion is fairly small, which we believe is the most appealing scenario for applications. From a theoretical perspective, it would be interesting to determine the approximability of the minimum distortion problem. Is there a constant factor approximation algorithm for the general case? We suspect that the problem is hard unless the input metrics are severely restricted.

Acknowledgments

6. REFERENCES

[1] T. Akutsu, K. Kanaya, A. Ohyama, and A. Fujiyama. Point matching under non-uniform distortions. *Discrete Applied Mathematics*, 127:5–21, 2003.

[2] L. Babai, D.Y. Grigoryev, and D.M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proc. of the 14th Ann. ACM Symp. on Theory of Computing*, 1982, pages 310–324.

[3] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(4):509–522, 2002.

[4] G. Blache, M. Karpinski, and J. Wirtgen. On approximation intractability of the bandwidth problem. *ECCC* technical report number TR98-014, 1998.

[5] A. Blum, G. Konjevod, R. Ravi, and S. Vempala. Semi-definite relaxations for minimum bandwidth and other vertex-ordering problems. *Theoretical Computer Science*, 235(1):25–42, 2000.

[6] B. Chazelle, D. Liu, and A. Magen. Sublinear Geometric Algorithms. In *Proc. of the 35th Ann. ACM Symp. on Theory of Computing*, 2003, pages 531–540.

[7] P.Z. Chinn, J. Chvatalova, A.K. Dewdney, and N.E. Gibbs. The bandwidth problem for graphs and matrices — A survey. *Journal of Graph Theory*, 6:223–254, 1982.

[8] M. Deza and M. Laurent. *Geometry of Cuts and Metrics*. Springer, Algorithms and Combinatorics, Vol. 15, 1997.

[9] J. Díaz, J. Petit, and M. Serna. A survey on graph layout problems. *ACM Computing Surveys*, to appear.

[10] R. Downey. Parameterized complexity for the skeptic. In *Proc. of the 18th IEEE Conference on Computational Complexity*, 2003, pages 147–169.

[11] R. Downey, M. Fellows, and U. Stege. Parameterized complexity: a framework for systematically confronting computational intractability. In: R. Graham, J. Kratochvil, J. Nesetril, and F. Roberts, eds. *Contemporary Trends in Discrete Mathematics*, AMS-DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 49, 1999, pages 49–99.

[12] U. Feige. Approximating the bandwidth via volume respecting embeddings. *Journal of Computer and Systems Sciences*, 60(3):510–539, 2000.

[13] U. Feige and R. Krauthgamer. Improved performance guarantees for bandwidth minimization heuristics. Unpublished manuscript, November 1998.

[14] S. Fortin. The graph isomorphism problem. Technical Report 96-20, University of Alberta, Edomonton, Alberta, Canada, 1996.

[15] M.R. Garey, R.L. Graham, D.S. Johnson, and D.E. Knuth. Complexity results for bandwidth minimization. *SIAM J. Appl. Math.*, 34(3):477–495, 1978.

[16] M.R. Garey and D.S. Johnson. Strong NP-completeness results: motivation, examples and implications. *J. Assoc. Comput. Mach.*, 25:499–508, 1978.

[17] A. Gupta. Improved bandwidth approximation for trees and chordal graphs. *J. Algorithms*, 40(1):24–36, 2001.

[18] J. Haralambides, F. Makedon, and B. Monien. Bandwidth minimization: an approximation algorithm for caterpillars. *Math. Systems Theory*, 24(3):169–177, 1991.

[19] F. Hoffmann, K. Kriegel, and C. Wenk. Matching 2D patterns of protein spots. In *Proc. of the 14th Ann. ACM Symp. on Computational Geometry*, 1998, pages 231–239.

[20] J. Hopcroft and J. Wonk. A linear time algorithm for isomorphism of planar graphs. In *Proc. of the 6th Ann. ACM Symp. on Theory of Computing*, 1974, pages 172–184.

[21] W-L. Hsu. $O(M * N)$ algorithms for the recognition and isomorphism problems on circular-arc graphs. *SIAM J. Comput.* 24(3):411–439, 1995.

[22] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.

[23] P. Indyk. Algorithmic aspects of geometric embeddings. FOCS '01 tutorial. Available at `http://theory.lcs.mit.edu/~indyk/tut.html`.

[24] W.B. Johnson and J. Lindenstrauss (editors). *Handbook of the Geometry of Banach Spaces*. North-Holland, 2003.

[25] M. Karpinski, J. Wirtgen, and A. Zelikovsky. An approximation algorithm for the bandwidth problem on dense graphs. *ECCC* technical report number TR97-017, 1997.

[26] J. Kobler, U. Schöning, and J. Toran. *The Graph Isomorphism Problem: Its Structural Complexity*. Progress in Theoretical Computer Science, Birkhauser, 1993.

[27] D. Kratsch. Finding the minimum bandwidth of an interval graph. *Information and Computation*, 74:140–187, 1987.

[28] N. Linial. Finite metric spaces — combinatorics, geometry and algorithms. In *Proc. of the International Congress of Mathematicians III*, Beijing, 2002, pages 573–586.

[29] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[30] E. Luks. Isomorphism of bounded valence can be tested in polynomial time. *Journal of Computer and System Sciences*, 25:42–65, 1982.

[31] A. Marcus and G. Tardos. Excluded permutation matrices and the Stanley-Wilf conjecture. Manuscript, November 2003. Available at `http://www.renyi.hu/~tardos/submatrix.ps`

[32] J. Matoušek. *Lectures on Discrete Geometry*. Springer, Graduate Texts in Mathematics, Vol. 212, 2002.

[33] G. Miller. Isomorphism testing for graphs of bounded genus. In *Proc. of the 12th Ann. ACM Symp. on Theory of Computing*, 1980, pages 225–235.

[34] B. Monien. The bandwidth minimization problem for caterpillars with hair length 3 is NP-complete. *SIAM J. Algeb. Discr. Meth.*, 7:505–512, 1986.

[35] C.H. Papadimitriou. The NP-completeness of the bandwidth minimization problem. *Computing*, 16(3):263–270, 1976.

[36] C.H. Papadimitriou and S. Safra. The complexity of low-distortion embeddings between point sets. Preprint, February 2004. Available at `http://www.cs.berkeley.edu/~christos`

[37] J.B. Saxe. Dynamic programming algorithms for recognizing small-bandwidth graphs in polynomial time. *SIAM J. Algeb. Discr. Meth.*, 1:363–369, 1980.

[38] W. Unger. The complexity of the approximation of the bandwidth problem. In *Proc. of the 39th Ann. Symp. on Foundations of Computer Science*,

November 1998, pages 82–91.

[39] R.C. Veltkamp and M. Hagedoorn. *State of the Art in Shape Matching*. Technical Report UU-CS-1999-27, U. Utrecht, 1999.

Appendix

Proof of Proposition 2.4(b). We give a simple reduction from the Multiprocessor Scheduling problem, defined as follows:

INPUT: A set of n jobs with positive integer lengths $p_1, \ldots, p_n$, and a number of processors m.

QUESTION: Is there a "flat" schedule, i.e., a partition of the jobs into m classes so that the sum of the lengths of the jobs in each class is exactly $\sum_i p_i/m$?

Multiprocessor Scheduling is well known to be strongly NP-complete [16].

Given an instance as above, we construct a weighted graph G and an unweighted tree T as follows. For each job i, there is a clique of p_i "job" vertices in G. There are also m "processor" vertices, each of which is connected by an edge to all job vertices. Finally, there is a single "anchor" vertex that is connected to all processor vertices. All edges have weight 1, except for the edges adjacent to the anchor vertex, which have weight $1/2$. This completes the description of G. The tree T consists of a root vertex r with m children and $\sum_i p_i$ grandchildren, evenly distributed among the m children.

Notice first that, if there is a flat schedule, then we can map the anchor vertex to the root of the tree, the processor vertices to the children, and for each job i, the p_i corresponding job vertices to children of its assigned processor. It is easy to check that this mapping has expansion 3 (achieved by an edge between a processor vertex and a job vertex assigned to a different processor), and inverse expansion 1 (no edge of T is increased in length under the inverse map). Thus the distortion is 3.

Now we will be done if we can show that, if there is no flat schedule, then the distortion must be at least 4. First it is not too hard to verify that any mapping that does not map the anchor to the root, processor vertices to its children, and job vertices to its grandchildren without separating the vertices of any job must incur an expansion of at least 4. Second, note that the inverse expansion of any bijection must be at least 1. This is because in G there are only m pairs of vertices whose pairwise distances are less than 1, which is much less than the number of edges (each of length 1) in T. Hence the inverse expansion of some edge in T must be at least 1. Putting these observations together confirms that the distortion must be at least 4 when no flat schedule exists. $\qquad \square$