Approximating additive distortion of embeddings into line metrics

Kedar Dhamdhere*[1]

School of Computer Science,
Carnegie Mellon University
Pittsburgh, PA 15213, USA
kedar@cs.cmu.edu

Abstract. We consider the problem of fitting metric data on n points to a *path* (line) metric. Our objective is to minimize the total additive distortion of this mapping. The total additive distortion is the sum of errors in all pairwise distances in the input data. This problem has been shown to be NP-hard by [13]. We give an $O(\log n)$ approximation for this problem by using Garg *et al.*'s [10] algorithm for the multi-cut problem as a subroutine. Our algorithm also gives an $O(\log^{1/p} n)$ approximation for the L_p norm of the additive distortion.

1 Introduction

One of the most common methods for clustering numerical data is to fit the data to *tree metrics*. A tree metric is defined on vertices of a weighted tree. The distance between two vertices is the sum of the weights of edges on the path between them. Here the main problem is to find a tree metric that represents the input numerical data as closely as possible. This problem, known as *numerical taxonomy*, has applications in various fields of science, such as linguistics and evolutionary biology. For example, in evolutionary biology tree metrics represent the branching process of evolution that leads to the observed distribution of data. Naturally, this problem has received a great deal of attention. (e.g. see [3, 14]).

The problem of fitting data to tree metrics is usually cast as the problem of minimization of $L_p(D,T)$: the L_p norm of *additive distortion* of the output tree T with respect to input data D. The input data is specified as an $n \times n$ matrix, where the entry D_{ij} denotes the distance between points i and j. Let T_{ij} denote the distance between i and j in the output tree metric T. Then the L_p norm of additive distortion is $L_p(D,T) = (\sum_{i,j} |D_{ij} - T_{ij}|^p)^{1/p}$. Such a formulation was first proposed by [5] in 1967. In 1977, Waterman *et al.* [15] showed that if there is a tree metric T coinciding exactly with the input data D, then it can be constructed in linear time. In the case when there is no tree that fits the data perfectly, Agarwala *et al.* [1] used the framework of approximation algorithms to give heuristics with provable guarantees for the problem. They gave a 3-approximation to the L_∞ norm of the additive distortion for fitting the data to

* Supported by NSF ITR grants CCR-0085982 and CCR-0122581.

a tree metric. They reduced the problem to that of fitting the data to *ultrametric*, where each leaf is at the same distance from a common root. For *ultrametrics*, they used an exact polynomial-time algorithm for the L_∞ norm due to Farach *et al.* [8]. Agarwala *et al.* [1] showed that if there is a ρ-approximation algorithm for *ultrametrics* under the L_p norm, then it implies a 3ρ-approximation for tree metrics under the L_p norm. Our work is motivated by this problem.

For a special case of fitting the data to a tree metric under L_1 norm, we make a simple observation. Suppose we know the structure of the tree along with a mapping of the points to the vertices of the tree. Then we can find the edge weights that minimize the L_1 norm of additive distortion using linear programming. However, finding the topology of the tree is the hard part. Therefore, we consider a special case of the problem in which we restrict the structure of the output tree to be a path. For this special case, we give an approximation algorithm. We believe that our techniques can be extended to solve the original problem. The main result of this paper is the following theorem.

Theorem 1. *There is an $O(\log^{1/p} n)$-approximation algorithm for the problem of fitting metric data to a line metric to minimize the L_p norm of additive distortion for $p \geq 1$.*

1.1 Related Work

Note that fitting points to a path metric is equivalent to fitting points on a real line, where the distances in the real line are defined in a natural way. The special case of the problem for the L_∞ norm (i.e. with $p = \infty$) was considered by Håstad *et al.* [11]. They gave a 2-approximation for it.

For fitting points to a line, a well-known result due to Menger (see e.g. [6]) gives the following four point criterion. The four point criterion says that, if every subset of size 4 can be mapped into the real line exactly, then all the points can be mapped into the line exactly. An approximate version of Menger's result was given by Badoiu *et al.* [2]. They proved that if every subset of size 4 can be embedded into the line with the L_∞ norm of the additive distortion being at most ϵ then all the points can be embedded with the L_∞ norm of the additive distortion being at most 6ϵ.

In a related work, Dhamdhere *et al.* [7] and Badoiu *et al.* [2] independently studied average distortion of embedding a metric into path metrics. The objective was to minimize $\sum_{i,j} f_{ij}$, subject to $f_{ij} \geq D_{ij}$ for all i, j, where D_{ij} is the distance between points i and j in the input and f_{ij} is the distance between them in the output path metric. They gave a 14-approximation for the problem. While their problem has additional constraint $f_{ij} \geq D_{ij}$, their objective function is easier than the L_1 norm of the additive distortion. We do not know how to minimize the L_p (or even the L_1) norm of additive distortion under the additional constraint. However, for the special case of $p = \infty$, we would like to point out that the algorithm for *min-excess path problem* due to Blum *et al.* [4] gives a $2 + \epsilon$ approximation. The *min-excess path problem* asks for a path from a source s to a destination t that visits at least k vertices and minimizes the objective $l(path) - d(s, t)$, where $l(path)$ is the length of the path.

1.2 Techniques and Roadmap

In Section 2, we define our problem formally. In Section 3, we show how to reduce the problem to that of finding the best r-restricted mapping. An r-restricted mapping of the input points into a line is one in which the distances of all points in the line from point r are same as that in the input. We show that this problem is equivalent to a two-cost partition problem. In Section 4, we give an approximation for this problem via the multi-cut problem [10].

2 Problem Definition

Consider a set of n points, denoted by $[n] = \{1, 2, \ldots, n\}$. The input data consists of an $n \times n$ matrix $D_{n \times n}$. The entry D_{ij} denotes the distance between points i and j. We assume that all the entries of D are non-negative and that D is symmetric.[1] Furthermore, we assume that $D_{ii} = 0$ for all i.

Let $f : [n] \rightarrow \mathbb{R}$ denote a mapping of the input points to the real line. Distance between images of points i and j in the line is given by $f_{ij} = |f(i) - f(j)|$. The total additive distortion (in the L_1 norm) is given by

$$L_1(D, f) = \sum_{i,j} |D_{ij} - f_{ij}|.$$

Generalizing this, we can write the L_p norm of the additive distortion as

$$L_p(D, f) = \big(\sum_{i,j} |D_{ij} - f_{ij}|^p \big)^{\frac{1}{p}}.$$

The goal is to find a map f that minimizes the $L_1(D, f)$ (or more generally $L_p(D, f)$).

3 Approximation for L_p norm

In this section, we give an approximation algorithm for minimizing the L_p norm of the additive distortion.

In Lemma 1, we will show that it is sufficient to look at r-restricted mapping of the points into the real line. The problem of finding an optimal r-restricted mapping can be cast as a kind of partition problem given the characteristics of the real line.

[1] Our results hold even if the input distances in $D_{n \times n}$ do not satisfy triangle inequality, i.e. even if D is not a "metric".

3.1 r-restricted mappings

Let r be a point in the input. A mapping f of the input points to the real line $\mathbb{R}$ is an r-restricted mapping, if distance on the line of all points from r is same as that in the input. Formally, $D_{ri} = |f(r) - f(i)|$ for all i.

We will denote an r-restricted mapping by f^r. We next show that there is always a "good" r-restricted mapping. This will enable us to focus only on r-restricted mappings which are easier to handle. Agarwala *et al.* [1] prove a similar lemma for tree metrics. We adapt their proof for the case of line metrics.

Lemma 1. *There exists a point* $\mathbf{r}$ *among the input points such that there is an* r*-restricted mapping* f^r *that is within a factor of* 3 *of the optimal mapping for the* L_p *norm of additive distortion, for all* $p \geq 1$.

Proof. Let f^* denote an optimal mapping of the input points to the line for the L_p norm of additive distortion. We will modify the optimal solution to produce a mapping f^i for each point i in the input. To produce the restricted mapping f^i, perturb the distances in f^* so that it becomes i-restricted. In particular, if $f^*(j) \leq f^*(i)$ for some j, then set $f^i(j) = f^*(i) - D_{ij}$ and if $f^*(j) > f^*(i)$, set $f^i(j) = f^*(i) + D_{ij}$. Our mapping f^i maps point i to $f^*(i)$. It maps rest of the points according to their distance from i, while maintaining their order to the left or right of point i in the optimal mapping f^*.

Let ϵ_{jk} denote $|D_{jk} - f^*_{jk}|$. We can write the additive distortion of the optimal mapping as $L_p(D, f^*) = (\sum_{j,k} \epsilon^p_{jk})^{1/p}$. From the construction of the map f^i, it follows that $|f^*_{jk} - f^i_{jk}| \leq \epsilon_{ij} + \epsilon_{ik}$.

Now we bound the additive distortion of f^i in terms of ϵ_{jk}'s. For all j, k we have,

$$\begin{aligned} |D_{jk} - f^i_{jk}| &\leq |f^*_{jk} - f^i_{jk}| + |D_{jk} - f^*_{jk}| \\ &\leq (\epsilon_{ij} + \epsilon_{ik}) + \epsilon_{jk} \qquad (1) \end{aligned}$$

Note that $|x|^p$ is a convex function of x for $p \geq 1$. Therefore, Equation (1) gives us the following:

$$\begin{aligned} |D_{jk} - f^i_{jk}|^p &\leq (\epsilon_{ij} + \epsilon_{ik} + \epsilon_{jk})^p \\ &\leq 3^{p-1}(\epsilon^p_{ij} + \epsilon^p_{ik} + \epsilon^p_{jk}) \qquad (2) \end{aligned}$$

By an averaging argument, we can say that

$$\min_i \{L_p(D, f^i)^p\} \leq \frac{\sum_{i=1}^n L_p(D, f^i)^p}{n}$$

We use Equation (2) to bound the sum

$$\begin{aligned} \sum_{i=1}^n L_p(D, f^i)^p &\leq \sum_{i=1}^n \sum_{j,k} 3^{p-1}(\epsilon^p_{ij} + \epsilon^p_{ik} + \epsilon^p_{jk}) \\ &\leq 3^p n \cdot \sum_{j,k} \epsilon^p_{jk} \\ &= 3^p n \cdot L_p(D, f^*)^p \end{aligned}$$

Therefore, $\min_i L_p(D, f^i) \leq 3 \cdot L_p(D, f^*)$ which proves the result.

3.2 Algorithm

The result of Lemma 1 proves that it is sufficient to consider r-restricted mappings (with a loss of 3 in the approximation factor). Next we describe the algorithm that implements this idea.

Algorithm A:

1. For each point $r = 1, 2, \ldots, n$, find (approximately) the best r-restricted mapping f^r.
2. Output a mapping that has the smallest additive distortion among these mappings.

By Lemma 1, the additive distortion of the output of Algorithm A is within a factor of 3 of the optimal additive distortion. As we point out in Section 5, finding best r-restricted mapping is NP-hard. Therefore, we approximate the optimal a-restricted mapping within a factor of $O(\log^{1/p} n)$. From the following observation it follows that the overall approximation factor of our algorithm will be $O(\log^{1/p} n)$.

Lemma 2. *If ρ is the approximation factor of the algorithm for r-restricted mapping, then the solution produced by Algorithm A will be a 3ρ approximation for the additive distortion.*

4 Approximating r-restricted mappings

Let f be an r-restricted mapping. Without loss of generality, we can assume that $f(r) = 0$. Let $V_1 = \{i \mid f(i) < 0\}$ and $V_2 = \{i \mid f(i) > 0\}$. Note that $[n] = V_1 \cup \{r\} \cup V_2$. Note that, the mapping f is fully characterized by the partition $V_1 \cup V_2$ of $[n] - \{r\}$. Hence, the problem of finding the best r-restricted mapping is equivalent to the problem of finding the partition of $V = [n] - \{r\}$ that has minimum additive distortion. Henceforth, we will think of the problem as that of partitioning the input set of points to minimize the *cost* of the partition, i.e. the additive distortion. For simplicity, we describe the argument for $p = 1$. The other cases ($p > 1$) are similar.

Consider a partition $V_1 \cup V_2$ induced by an r-restricted mapping f. We can write an expression for its *cost* as follows. Consider two points x and y. If they both belong to the same side of the partition, then the contribution of the pair $\{x, y\}$ to the cost of the partition is $c(x, y) = |D_{xy} - f_{xy}| = D_{xy} - |f(x) - f(y)| = |D_{xy} - |D_{rx} - D_{ry}||$. On the other hand, if x and y belong to different sides of the partition, then the contribution is $c'(x, y) = |D_{xy} - f_{xy}| = |D_{xy} - |f(x) - f(y)|| = |D_{rx} + D_{ry} - D_{xy}|$. Note that $c(x, y)$ and $c'(x, y)$ are completely determined from the input matrix $D_{n \times n}$.

Therefore, we can think of the problem as a graph partitioning problem where each edge has two costs $c(\cdot)$ and $c'(\cdot)$ associated with it. The objective function is

$$\sum_{x,y \text{ on same side}} c(x,y) + \sum_{x,y \text{ on different sides}} c'(x,y)$$

4.1 Two-cost Partition Problem

We are given a complete graph $G = (V, E)$ with two cost functions c and c'. We want to find a partition of the vertex set $V = V_1 \cup V_2$ which minimizes $\sum_{i=1,2} \sum_{u,v \in V_i} c(u,v) + \sum_{u \in V_1, v \in V_2} c'(u,v)$.

Note that, if $c(u,v) = 0$ for all u, v, then the problem reduces to finding a minimum cut in the graph. On the other hand, if $c'(u,v) = 0$, then the problem is the well known edge deletion for graph bipartition problem (BIP) [12]. Our algorithm generalizes the algorithm for graph bipartition given by [12, 10]. The basic idea is to create two copies of each vertex to go on different sides of the partition. To ensure that they are on different sides, we designate each pair as a source-sink pair in the multi-cut subroutine.

Algorithm B:

1. Create an auxiliary graph G' from the graph G as follows.
   (a) For each vertex u in the graph G, G' has two vertices: u and u'.
   (b) For each edge (u, v) we create 4 edges in G': $(u, v), (u, v'), (u', v)$ and (u', v').
   (c) The edges in G' have weights, denoted by $l(\cdot, \cdot)$. Set $l(u, v) = l(u', v') = c(u, v)$ and $l(u, v') = l(u', v) = c'(u, v)$.
2. Use an approximation algorithm for the multi-cut problem (E.g., [10]) as a subroutine to find a multi-cut in graph G' with (u, u'), for all u, as the source-sink pairs. Let S be the set of edges in the multi-cut returned by the subroutine.
3. Construct a set of edges T as follows. If $\{u, v\}$ or $\{u', v'\}$ is chosen in S, then include both in T. Similarly, if $\{u, v'\}$ or $\{u', v\}$ is chosen, then include both in T.
4. Find a bipartition $V_1' \cup V_2'$ of vertices of G' so that T contains all the edges going across the partition.[2]
5. Output the partition $V_1 \cup V_2$, where $V_i = V_i' \cap V$.

The intuition behind this algorithm is as follows. For the cut represented by T, we will show that we can get a partition of vertices in graph G' such that only one of u and u' is in one partition. From the partition of G', we get a bipartition of G. The cost of the bipartition of G is related to the cost of multi-cut obtained by above algorithm in the graph G'. We prove this in the next lemma.

[2] We will show how to do this in the proof of Proposition 1.

Lemma 3. *Algorithm B returns a partition $V' = V_1' \cup V_2'$ of graph G', such that if $u \in V_1'$, then $u' \in V_2'$ and vice versa. Moreover, $\sum_{x \in V_1', y \in V_2'} l(x,y)$ is at most twice that of the multi-cut found after step 2 by Algorithm B separating each u from u'.*

Proof. Consider the set S of edges found by the multi-cut subroutine whose removal separates each u from u'. For each edge $(x,y) \in S$, we also include its "mirror" edge in T. i.e. if $(x,y) \in S$, then $(x',y') \in T$ from the graph. Note that, the cost of an edge and its "mirror" edge is same (i.e., $l(x,y) = l(x',y')$). Therefore, the cost of the edges in T is at most twice the cost of edges in S.

Now we show that removal of the edges in T breaks the graph in two parts with the desired property. Consider the graph $G' \backslash T$. Construct a graph H whose vertices represent the connected components in G' after removing the edges in T. Two vertices h_1 and h_2 in H are connected to each other if the corresponding connected components in G' have vertices x and x'.

In Proposition 1, we prove that the graph H is bipartite. Now we can use graph H to construct a partition $V' = V_1' \cup V_2'$ in graph G'. Since the vertices in graph H were connected components in graph G', there are no edges crossing the partition $V_1' \cup V_2'$ in graph G'. Moreover, bipartiteness of graph H means that each pair of vertices x and x' in graph G is split in the partition. The cost of this partition is at most 2 times the cost of the multi-cut.

Proposition 1. *The graph H defined in the proof of Lemma 3 is bipartite.*

Proof. For the sake of contradiction, assume that H has a cycle of odd length. Consider three consecutive vertices u, v and w in this odd cycle. Let v be connected to u and w.

Let x be a vertex of G' that belongs to the connected component u and defines the edge $\{u,v\}$ in graph H. Therefore, x' is the component v. Similarly, let y be a vertex in component w and y' be the corresponding vertex in component v. Since x' and y' are in the same connected component v, there is a path $x' \rightarrow y'$ that lies completely inside the component v. Since we didn't remove any of the edges on the path $x' \rightarrow y'$, all the *mirror* edges haven't been removed either. Therefore the the *mirror* path $x \rightarrow y$ connects x and y. This contradicts the fact that x and y were in different connected components.

This proves that the graph H is a bipartite graph.

Lemma 4. *The cost of the optimal multi-cut is a lower bound on the cost of partition of graph G.*

Proof. Consider a partition $V = V_1 \cup V_2$ of graph G. From this, we can construct a partition of the vertex set of G'. Let $V_1' = V_1 \cup \{x' \mid x \in V_2\}$ and $V_2' = V' \backslash V_1'$. Then, removing all the edges in G' crossing this partition ensures that no vertex x is connected to its counterpart x'. i.e. The set of edges going across the partition is a multi-cut. The cost of these edges is exactly the cost of the partition of G.

Recall that GVY algorithm for multi-cut [10] is an $O(\log k)$ approximation for k terminals. Here we have n terminals. Therefore by Lemmas 3 and 4, we get an $O(\log n)$ approximation for the best r-restricted mapping. Along with Observation 2 give us an $O(\log n)$ approximation for the L_1 norm of additive distortion.

To get an approximation algorithm for the L_p norm, we modify the costs in the two-cost partition problem as follows. Let $c(x,y) = |D_{xy} - |D_{ax} - D_{ay}||^p$ and $c'(x,y) = |D_{ax} + D_{ay} - D_{xy}|^p$. With these costs, Algorithm B gives an $O(\log n)$ approximation for $L_p(D, f^a)^p$. Therefore, for the L_p norm of additive distortion, we get an $O(\log^{1/p} n)$ algorithm.

5 Open questions

We can show that the problem of finding the best r-restricted mapping is NP-hard by reducing the edge deletion for graph bipartition (BIP) [9] to it. Consider a graph G. Let $V(G) = n$. We construct a distance matrix D on $n+1$ points $V(G) \cup \{a\}$. Set the diagonal entries D_{xx} to 0. Set $D_{ax} = 1/2$ for all $x \in V(G)$. For all $\{x,y\} \in E(G)$, set $D_{xy} = 1$. Set the rest of the entries to 1/2. Consider an r-restricted mapping. Let $V(G) = V_1 \cup V_2$ be the partition induced by the r-restricted mapping. Then the cost of the r-restricted mapping is $B(V_1, V_2) + (1/2)(\binom{n}{2} - |E(G)|)$, where $B(V_1, V_2)$ is the number of edges that need to be deleted to obtain V_1 and V_2 as two sides of a bipartite graph. Therefore, finding the optimal r-restricted mapping corresponds to minimizing the number of edges deleted for making the graph G bipartite. This proves that finding the best r-restricted mapping is NP-hard. However, this reduction is not approximation preserving. So it does not preclude the possibility of a PTAS for this problem. Getting even a constant factor approximation would be quite interesting.

In the proof of NP-hardness of r-restricted mapping problem, we used an input matrix D that does not satisfy the triangle inequality. For input matrix D that is a *metric* (i.e. it satisfies the triangle inequality), it might be possible to get a polynomial time algorithm for the best r-restricted mapping.

Agarwala *et al.* [1] have shown that for the problem of fitting data to tree metrics, the best r-restricted tree metric is within a factor of 3 of the optimal solution. However, the problem of approximating the additive distortion of the best r-restricted tree is still open.

Acknowledgments We are grateful to R. Ravi and Anupam Gupta for helpful discussions and valuable suggestions.

References

1. Richa Agarwala, Vineet Bafna, Martin Farach, Babu O. Narayanan, Mike Paterson, and Mikkel Thorup. On the approximability of numerical taxonomy (fitting distances by tree metrics). In *Symposium on Discrete Algorithms*, pages 365–372, 1996.

2. Mihai Badoiu, Piotr Indyk, and Yuri Rabinovich. Approximate algorithms for embedding metrics into low-dimensional spaces. In *Unpublished manuscript*, 2003.
3. J-P. Barthélemy and A. Guénoche. *Trees and proximity representations.* Wiley, New York, 1991.
4. Avrim Blum, Shuchi Chawla, David Karger, Adam Meyerson, Maria Minkoff, and Terran Lane. Approximation algorithms for orienteering and discounted-reward tsp. In *IEEE Symposium on Foundations of Computer Science*, 2003.
5. L. Cavalli-Sforza and A. Edwards. Phylogenetic analysis models and estimation procedures. *American Journal of Human Genetics*, 19:233–257, 1967.
6. M. Deza and M. Laurent. *Geometry of Cuts and Metrics.* Springer-Verlag, Berlin, 1997.
7. K. Dhamdhere, A. Gupta, and R. Ravi. Approximating average distortion for embeddings into line. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, 2004.
8. M. Farach, S. Kannan, and T. Warnow. A robust model for finding optimal evolutionary trees. *Algorithmica*, 13:155–179, 1995.
9. M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W. H. Freeman, San Fransisco, USA, 1979.
10. N. Garg, V. Vazirani, and M. Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.
11. Johan Håstad, Lars Ivansson, and Jens Lagergren. Fitting points on the real line and its application to RH mapping. In *European Symposium on Algorithms*, pages 465–476, 1998.
12. Philip Klein, Ajit Agarwal, R. Ravi, and Satish Rao. Approximation through multicommodity flow. In *IEEE Symposium on Foundations of Computer Science*, pages 726–737, 1990.
13. James B. Saxe. Embeddability of graphs into k-space is strongly np-hard. In *Allerton Conference in Communication, Control and Computing*, pages 480–489, 1979.
14. P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy.* W. H. Freeman, San Fransisco, CA, 1973.
15. M. S. Waterman, T. S. Smith, M. Singh, and W. A. Beyer. Additive evolutionary trees. *Journal of Theoretical Biology*, 64:199–213, 1977.