Lower Bounds for Embedding the Earth Mover Distance Metric into Normed Spaces

by

Javed K. K. Samuel

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January 2005 February 2005

Author ....
Department of Electrical Engineering and Computer Science
January 28, 2005

Certified by..
Piotr Indyk
Associate Professor
Thesis Supervisor

Accepted by..
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Lower Bounds for Embedding the Earth Mover Distance Metric into Normed Spaces

by

Javed K. K. Samuel

Submitted to the Department of Electrical Engineering and Computer Science
on January 28, 2005, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

This thesis presents a lower bounds for embedding the Earth Mover Distance (EMD) metric into normed spaces. The EMD is a metric over two distributions where one is a mass of earth spread out in space and the other is a collection of holes in that same space. The EMD between these two distributions is defined as the least amount of work needed to fill the holes with earth. The EMD metric is used in a number of applications, for example in similarity searching and for image retrieval. We present a simple construction of point sets in the EMD metric space over two dimensions that **cannot be embedded** from the EMD metric exactly into normed spaces, namely l_1 and the square of l_2. An embedding is a mapping $f : X \rightarrow V$ with X a set of points in a metric space and V a set of points in some normed vector space. When the Manhattan distance is used as the underlying metric for the EMD, it can be shown that this example is isometric to $K_{2,4}$ which has distortion equal to 1.25 when it is embedded into l_1 and 1.1180 when embedded into the square of l_2. Other constructions of points sets in the EMD metric space over three and higher dimensisions are also discussed.

Thesis Supervisor: Piotr Indyk
Title: Associate Professor

Acknowledgments

I would like to express my sincere gratitude and appreciation to my thesis supervisor, Professor Piotr Indyk. Working with him on this thesis project has been a very pleasurable and rewarding experience. His comments and suggestions helped me considerably in successfully completing this thesis. I have learnt a significant amount about embeddings and computational geometry from discussions with him.

I would also like to thank Alex Andoni and Carlo Tomasi for allowing me to use their code to run computational tests. Their code provided an excellent basis to complete this project successfully.

I would also like to thank my family and friends who have continually encouraged me while I pursue my goals. I would like to extend my heartfelt congratulations to everyone who helped in whatever way while I completed this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Earth Mover Distance (EMD) is a metric over two distributions where one is a mass of earth spread out in space and the other is a collection of holes in that same space. The Earth Mover Distance between these two distributions is defined as the least amount of work needed to fill the holes with earth. The Earth Mover Distance of two k-element sets $A, B \subset \Re^d$ is the minimum weight of a perfect matching between A and B; that is $min_{\pi:A\rightarrow B} \sum_{a\in A} D(a, \pi(a))$. The Earth Mover Distance metric is of considerable theoretical interest and it is also a natural metric to use for similarity searching, image retrieval and vector feature comparison.

We present a simple construction of point sets in the EMD metric space in two dimensions and in three dimensions that **cannot be embedded** from the EMD metric exactly into normed spaces, namely l_1 and the square of l_2. An embedding is defined as a mapping $f : X \rightarrow V$ with X a set of points in a metric space and V a set of points in some normed vector space. Low-distortion embeddings are very useful and allow us to reduce more "difficult" metrics such as EMD into problems over "simpler" metrics such as l_1 or l_2. However, it is not always possible to have isometric embeddings for all metric spaces and we discuss examples involving the EMD metric and the normed spaces l_1 and the square of l_2.

Our results are as follows:

1. If we use the Manhattan distance as the underlying metric we construct an example in two dimensions that is isometric to $K_{2,4}$ which has distortion equal to **1.25** when it is embedded into l_1 and equal to **1.1180** when embedded into the square of l_2.

2. We can also construct an example when the Euclidean distance is used as the underlying metric. This example has distortion of **1.1667** when embedded into l_1 and distortion of **1.0854** when embedded into the square of l_2.

3. We can also construct other examples in three dimensions and in higher dimensions that cannot be embedded exactly into l_1 and the square of l_2.

1.1 Definitions

Vector Norm: A vector norm for column vectors $x = [x_j]$ with n coordinates is a "generalized length", and is denoted by $||x||$. It is defined by the four usual properties of the length of vectors in three-dimensional space.

1. $||x||$ is a nonnegative real number.

2. $||x|| = 0$ if and only if $x = 0$.

3. $||kx|| = |k| \times ||x||$ for all k.

4. $||x + y|| \leq ||x|| + ||y||$ - (Triangle inequality)

l_1 *Norm:* A vector norm defined for a vector $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ as $||x||_1 = \sum_{r=1}^{n} ||x_r||$.

Square of l_2 *norm:* A vector norm for $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$ as $||x|| = \sum_{k=1}^{n} ||x_k||^2$.

Dissimilarity measures: This is a quantitative measure of the difference between two distributions. It can also be used to approximate perceptual dissimilarity. Choosing the correct dissimilarity measure has significant implications for image retrieval applications. Some examples of dissimilarity measures [24] include Minkowski-form distance, Jeffrey's divergence and the Earth Mover Distance.

Metric Space: A pair (X, D) where X is a set of points and $D : X \times X \to [0, \infty]$ in a distance function satisfying the following conditions for all $x, y, z \in X$:

1. $D(x, y) = 0$ if and only if $x = y$
2. $D(x, y) = D(y, z)$ - Symmetry relation
3. $D(x, y) + D(y, z) \geq D(x, z)$ - Triangle inequality

Embedding: An embedding is a mapping $f : X \to V$, with X a set of points in a metric space and V a set of points in some normed vector space [11] [18]. Embeddings with low-distortion are used in a variety of fields. One recent example is gel electrophoresis images which is used for surveying the protein contents of cells and it is used for DNA matchings and genetics. Embeddings are also used in biology to compare structures like fingerprints, DNA, etc. They can be embedded into normed vector spaces and then comparisons can become computationally feasible [10].

Distortion: A mapping $f : X \to X'$, where (X, D) and (X', D') are metric spaces, has distortion at most c, where $c \geq 1$, if there is an $r \in (0, \infty)$ such that for all $x, y \in X$.

$$r.D(x, y) \leq D'(f(x), f(y)) \leq cr \cdot D(x, y)$$

Isometric Mapping: An isometric mapping is one φ from a metric space (X, D) to a metric space (Y, ρ) which preserves distances, that is, $\rho(\varphi(x), \varphi(y)) = D(x, y)$ for all $x, y \in X$. We note that isometries are often very restricted and more flexibility is usually gained by allowing the embedding have some distortion > 1 with a corresponding loss of accuracy.

Cut metric: A pseudo-metric D on a set X such that, for some partition $X = A \cup B$, we have $D(x, y) = 0$ if both $x, y \in B$, and $D(x, y) = 1$ otherwise.

Embedding general metrics into l_1: We can use Bourgain's theorem [3] which states that every n-point metric space (X, D) can be embedded in an $O(\log n)$ dimensional Euclidean space with a $O(\log n)$ distortion and an algorithm suggested by Linial et al [15] [13] to embed a general metric into l_1. We can use cut metrics [5] to create a linear program with variables for pairwise distances. If we assume that the triangle inequality holds [16] [17] then we can then solve this linear program to determine the embedding into l_1.

$K_{m,n}$: This is the complete bipartite graph with parts of sizes m and n. By computing the shortest path distance matrix in this graph , we obtain a metric in which the distances between any two points in the same set to be 2, and the distances between one point from each set to be 1.

Lower Bounds for embedding $K_{2,n}$-metric into l_1 norm: The following theorems were proved by Andoni, Indyk et al [1] . They show a lower bound for the embedding of $K_{2,n}$-metric into the l_1 norm. We can construct an example of points in two dimensions under the EMD metric which is isometric to $K_{2,4}$.

Theorem 1. *For any $\epsilon > 0$, there exists n, such that the distortion of any embedding of $K_{2,n}$-metric into l_1 norm is at least $3/2 - \epsilon$.*

Theorem 2. *There exists an embedding f of $K_{2,n}$-metric into l_1 with distortion $3/2$.*

Lower Bounds for embedding $K_{2,n}$-metric into the square of l_2 norm: Andoni, Indyk et al [1] also proved the following theorem. They showed a natural embedding of $K_{2,n}$ metric into the square of l_2, with distortion 3/2.

Theorem 3. *For any $\epsilon > 0$, there exists n, such that the distortion of any embedding of $K_{2,n}$-metric into the square of the l_2 norm is at least $3/2 - \epsilon$.*

Positive Semi-definite Matrix: A positive semi-definite matrix is a self adjoint square matrix and all its eigenvalues are non-negative. A self-adjoint matrix $A = a_{ij}$ is defined as one for which $A = A^H$ where A^H denotes the conjugate transpose. This is equivalent to the condition $a_{ij} = \bar{a}_{ji}$.

Semi-definite Programming: The semi-definite programming problem (SDP) is essentially an ordinary linear program (LP) where the nonnegativity constraint is replaced by a semi-definite constraint on matrix variables. The standard form for the primal problem is

$$min \qquad C \bullet X$$

subject to

$$A_k \bullet X = b_k (k = 1, ..., m); \qquad\qquad X \geq 0$$

where C, A_k and X are symmetric $n \times n$ matrices, b_k is a scalar and $X \geq 0$ means that X, the unknown matrix, must lie in the closed, convex cone of positive semi-definite. Also, $\bullet$ refers to the standard inner product on the space of symmetric matrices. We can use the Semi-Definite Programming package for MATLAB to solve this system of equations.

Ellipsoid Method : This is an algorithm use for nonlinear optimization [20]. We look for either a feasible or an optimum solution of the linear program. First we start with an ellipsoid which we know a priori to contain the solutions. for example a large ball. At each iteration k, we check if the center x_k of the current ellipsoid is a feasible solution. Otherwise, we take a hyperplane containing x_k such that all the solutions lie on one side of this hyperplane. Now we have a half-ellipsoid which contains all solutions. We take the smallest ellipsoid completely containing this half-ellipsoid and continue.

Embedding general metrics X into the square of l_2 norm: We can embed general metrics into the square of l_2 by creating a semi-definite program (SDP) from the distance matrix of the given metric [13] [16] [17]. We can then solve that SDP to obtain the distortion of the embedding of the given metric into the square of l_2.

We have $X = x_1, x_2, ..., x_n$.and f denotes the mapping from X to the square of l_2. Let $f(x_i) = v_i$. We can then change the coordinates such that we have $f(x_1) = v_1 = 0$. Let us now have a matrix A with $A_{ij} = \frac{1}{2}(d_{1i}^2 + d_{1j}^2 - d_{ij}^2)$. We want $A_{ij} = (v_i, v_j)$ which is equivalent to determining whether A can be written as $B^T B$ for

$$B = \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}$$

This decomposition will exist if and only if A is a positive semi-definite matrix and this can be calculated in polynomial time using the Grotschel. Lovasz and Schrijver method [20]. We also need to minimize D, subject to the constraints that all the pairwise distance are distorted by at most D. We can formulate it as an SDP by changing D to a vector whose norm denotes the distortion. We then need to use a semi-definite program package in to solve this system of equations and calculate the required distortion.

$$min(D, D)$$

such that

$$d_{ij}^2 \leq (v_i, v_i) + (v_j, v_j) - 2(v_i, v_j) \leq (D, D)d_{ij}^2 \qquad \forall i, j$$
$$d_{1i}^2 \leq (v_i, v_i) \leq (D, D)d_{1i}^2 \qquad \forall i$$

CIE Lab color space: This is a model used to describe all the colors visible to the human eye. The three parameters in the model represent the luminance of the color (**L**, the smallest L yields black), its position between red and green (**a**, the smallest a yields green) and its position between yellow and blue (**b**, the smallest b yields blue) [7]. It is to be used a device-independent, absolute reference model.

Histograms: A histogram $\{h_i\}$ is a mapping from a set of d-dimensional integers $\mathbf{i}$ to the set of nonnegative reals. These vectors usually represent bins or their centers in a fixed partitioning of the relevant region of the underlying feature space, and the associated reals are a measure of the mass of the distribution that falls into the corresponding bin [22].

Bipartitite graph matching: We are a given a bipartite graph G (this is a set of graph vertices decomposed into two disjoint sets such that no two graph vertices within the same set are adjacent) with the bipartition $V(G) = A \cup B$ and we need to find the maximum matching in G.

Successive Shortest Path Algorithm: We are given a directed graph G, capacities $u : E(G) \rightarrow \Re_+$, numbers $b : V(G) \rightarrow \Re$ with $\sum_{v \in V(G)} b(v) = 0$, and conservative weights $c : E(G) \rightarrow \Re$. We output the minimum cost b-flow f. We use augmentations to determine the output for the successive shortest path.

Vector fields: A vector field is a map $f : \Re^n \Rightarrow \Re^n$ that assigns each x a vector function $f(x)$. Helmholtz's theorem states that a vector field is uniquely specified by giving its divergence and curl within a region and its normal component over the boundary.

Chapter 2

Earth Mover Distance

2.1 Introduction

The Earth Mover Distance [23] provides a mechanism to compute the dissimilarity between two probability distributions in some feature space. A predefined ground distance measure is given between single features. Examples of this "ground distance" include the Euclidean or Manhattan distance. The Earth Mover Distance then "lifts" this distance from these individual features to full distributions. This can also be viewed in the following way. We have two distributions: one is a mass of the earth spread out in space and the other is a collection of holes in that same space. The Earth Mover Distance measures the least amount of work needed to fill the holes with earth. A unit of work is defined as transporting a unit of earth by a unit of the "ground distance".

We can represent a distribution by a set of clusters where each cluster is represented by its mean and by the fraction of the distribution that belongs to that cluster. This is called the signature of the distribution. A signature is a set of the major clusters or modes of the distribution, that is represented by a single point in the underlying space, together with the weight which represents the size of that cluster.

2.2 Transportation Problem

The computation of the Earth Mover Distance is based on the solution to the transportation problem [8]. Suppose that several suppliers, each with a given amount of goods, are required to supply several consumers each with a given limited capacity. For each supplier-consumer pair, the cost of transporting a single unit of goods is given. The transportation problem is then defined as finding the least expensive flow of goods from the suppliers to the consumers that satisfies the consumer's demand. The formulation of the transportation problem is assymetric.

The matching signature in a transportation problem can be defined with one signature being the supplier and the other one as the consumer. We can set the cost for the supplier-consumer pair equal to the ground distance between an element in the first signature and an element in the second. We can see that the solution is then the minimum amount of "work" required to transform one signature into the other.

The transportation problem can be formalized as follows.

- Let $P = (p_1, w_{p_1}), ..., (p_m, w_{p_m})$ be defined as the first signature with m clusters, where p_i is the cluster representative and w_{p_i} is the weight of the cluster.
- Let $Q = (q_1, w_{q_1}), ..., (q_n, w_{q_n})$ is the second signature with n clusters.
- Let $D = d_{ij}$ represent the ground distance matrix where d_{ij} is the ground distance between clusters p_i and q_j.

The aim is to find a flow $F = f_{ij}$ with f_{ij} representing the flow between p_i and q_j that minimizes the overall cost.

$$f_{ij} \geqq 0 \qquad 1 \leqq i \leqq m, 1 \leqq j \leqq n$$

$$\sum_{j=1}^{n} f_{ij} \leqq w_{p_i} \quad 1 \leqq i \leqq m$$

$$\sum_{i=1}^{m} f_{ij} \leqq w_{q_j} \quad 1 \leqq j \leqq n$$

$$\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} = min(\sum_{i=1}^{m} w_{p_i}, \sum_{j=1}^{n} w_{q_i})$$

1. The first constraint allows moving "supplies" from P to Q but not vice versa.

2. The next constraint limits the amount of supplies in P to their weights.

3. The third constraint ensures that the clusters in Q receive no more supplies than their weights.

4. The final constraint forces the supplier to move the maximum amount of supplies possible. This maximum amount is defined as the "total flow."

Once the transportation problem is solved and we have calculated the optimal flow F, the earth mover's distance is defined as the work normalized by the total flow.

$$EMD(P, Q) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij}}$$

The normalization factor is included to avoid favoring smaller signatures in the case of partial matching. However, we only consider embedding the Earth Mover Distance metric into normed spaces, that is l_1 and the square of l_2 with complete matchings.

The Earth Mover Distance has the following advantages.

1. It naturally extends the notion of a distance between single elements to that of a distance between sets, or distributions of elements.

2. It can be applied to the more general variable-size signatures, which subsume histograms. Signatures are more compact and the cost of moving "earth" adequately reflects the notion of the nearness property.

3. It is a true metric if the ground distance is metric and if the total weights of the two signatures are equal. In our case, the ground distance was usually the Manhattan distance or the Euclidean distance. The weights of the two signatures were always equal.

4. It is bounded from below by the distances between the centers of mass of the two signatures for metric ground distances. This lower bound helps to reduce the number of EMD computations in retrieval systems.

5. It matches perceptual similarity better than other measures when the ground distance is meaningful.

2.3 Minimum Weight Matching Problem

In our case for the Earth Mover Distance, the total weight in the two distributions are equal. The result is that the EMD solution has a one to one correspondence with the problem of a bipartite graph matching [14]. We can therefore use a graph algorithm to solve the minimum weight bipartite graph problem and hence obtain a solution to the Earth Mover Distance problem.

We have a metric space (X, D) and two n element sets $A, B \subset X$, the Earth Mover Distance is equivalent to the minimum cost of the perfect matching between A and B.

$$EMD(A, B) = min(\pi : A \to B) \sum_{a \in A} D(a, \pi(a))$$

2.3.1 An $O(n^3)$ algorithm for solving the minimum weight bipartite graph problem.

Let G be a bipartite graph with bipartition $V(G) = A \cup B$. We assume that $|A| = |B| = n$. We add a vertex s and connect it to all the vertices of A, and add another vertex t connected to all vertices of B. We then orient the edges for A to B and from B to t. Let the capacities be their distance in the particular metric and let the new edges have zero cost.

Then any integral $s - t$ flow of value n corresponds to a perfect minimum weight matching with the same cost, and vice-versa. Hence we have reduced the problem to solving the Minimum Cost Flow Problem [19]. We can solve that by applying the Successive Shortest Path Algorithm [6]. This results in a running time of $O(nm + n^3)$. We can solve it slightly faster if we use Dijkstra's algorithm (which is a subroutine of the Successive Shortest Path Algorithm) with Fibonacci heaps, resulting in a running time of $O(nm + n^2 \log n)$.

2.4 Uses of Earth Mover Distance

The Earth Mover Distance is used in a variety of applications, for example several systems use it as the basis for their similarity measures [24]. The EMD can be used for region matching. In that case, its actual effectiveness is dependent on the underlying distance function and the weight given to each region which may become problematic to determine accurately for certain data sets. The EMD is also used in image retrieval systems [4] and in computing differences between vector fields [2].

The EMD has been used successfully in a color-based image retrieval system with color signatures. The EMD performed fairly well compared to other dissimilarity measures such as Minkowski-form distance, Jeffrey divergence, χ_2 statistics and the quadratic form distance. The implementors of the system used the Euclidean distance between the individual colors. This was used primarily distance in the CIE-Lab color space [7] as their underlying because it allows short Euclidean distances to correlate closely with actual human color discrimination. The ground distance used in such a system is of critical importance in evaluating the precision of a query.

An improvement that can be made to a color based retrieval system is to take into account the position of the colors in the image [25]. For example, if we have a blue ball on a red chair in one picture and a red ball on a blue chair in the other picture and we simply use color distributions with EMD, then the two pictures may be considered very similar. However, if we add the actual position of the colors as an additional parameter we will get a more accurate image retrieval system. Therefore, the ground distance would be the Euclidean distance in the CIE-Lab color space plus the (x, y) position of each pixel. This modification resulted in more accurate results at the cost of a slightly more complex ground distance function.

The EMD can also be used on texture signatures [21]. Texture is a more global property of a given image since a single point has no texture. The texture content of an entire image is represented by a distribution of texture features. Usually for the distribution would be simple for an image of one texture. for example an image of clear blue sky. More complex images like the image of the crowd at a sporting event with consist of multiple textures. The texture signature is simplified by only examining the dominant clusters. The ground distance is more complicated to define in this case and the designers developed a two-level EMD approach. They used the l_1 distance between texture features as an approximation for the low level EMD and then used this distance as the ground distance of the high level EMD. Though the EMD cannot be exactly embedded into l_1 for all data sets as we will show in the following section, l_1 distances still serves as a reasonable approximation for EMD.

The EMD can also be used to compute the differences between vector fields [2]. In that case the feature distribution is defined as the characteristics of a vector field. Vector fields themselves have numerous real-world applications which include gravitation and electromagnetism, the velocity vectors of fluid motion, for example airflow over an airplane and the pressure gradients on weather maps. They compute the EMD between every pair of vector fields and position the vector fields on a map such that the distances between the vector fields match their EMD values as accurately as possible.

2.5 Implementation

We used modules from the code written by C. Tomasi [26] to calculate the distance matrix for any distribution of points using the Earth Mover Distance as the underlying metric. The code was implemented in C and was based on the solution to the Transportation problem [9]. We compute the EMD between two distributions, which are represented by signatures. The signatures are sets of weighted features that capture the distributions. The features can be of any type and in any number of dimensions, and were defined as needed. We used primarily features of dimensionality two, three and four in this project. In most cases, the "underlying ground distance" between the points in each set was defined as the Manhattan distance l_l norm. In some cases, the Euclidean distance was used as the ground distance. These are some of the more natural ground distances and since they are true metrics satisfying the equality, symmetry and triangle equality properties it follows that the EMD with these ground distances is also a true metric.

The number of points in each set was always equal and hence a complete matching between sets was always calculated. The code was modified to compute the Earth Mover Distance for all the pairs of the various sets which were then used to compute the required distance matrix. Most data sets consisted of points in two dimensions, three dimensions or four dimensions. The weights for each point in every set was always set to one, and hence all sets had equal total weights. This ensured that we did not have to normalize the EMD calculation for any pair of distributions in the given data set since we could determine a perfect matching.

We then computed and solved the linear equations representing the cut matrices of this particular distance matrix for the embedding into the normed space l_1 [1]. We solved these linear system of equations by the MATLAB's linear programming solver (linprog) to calculate the distortion. This procedure also gave us the values for each of the cut matrices needed to embed that particular data set into l_1. We performed a similar procedure for embedding into the square of l_2. In that case the output was a semi-definite linear program and we used MATLAB's semi-definite programming (SDP) solver to calculate the distortion.

Chapter 3

Embedding of Earth Mover Distance into Normed Spaces in Two Dimensions

3.1 Upper Bounds for Embedding of EMD into l_1

The EMD metric $D_M(P, Q)$ between two point sets is defined as the cost of the minimum weight matching in the weighted matching between points in P and Q where P and Q are two point sets of cardinality s, each in R^k and $V = P \cup Q$. For any pair $p \in P, q \in Q$ the weight of (p, q) is defined as the Manhattan distance between p and q.

3.1.1 Description of the embedding of Indyk-Thaper [12]

Let us assume that the smallest inter-point distance is 1, and also let $\triangle$ be the diameter of V. We can then embed the EMD into l_1 by the following construction.

We first build grids on the space R^k of sides $\frac{1}{2}, 1, 2, 4, ..., 2^i, ...\triangle$ [12]. Let G_i be the grid of side 2^i. The grid G_i is a refinement of grid G_{i+1}. The grid is translated by a vector chosen uniformly at random from $[0, \triangle]^k$.

For each grid G_i, construct a vector $v_i(P)$ with one coordinate per cell, where each coordinate counts the number of points in the corresponding cell. This results in $v_i(P)$ forming a histogram of P. We can define the mapping f by setting $f(P)$ as the vector

$$\frac{v_{-1}(P)}{2}, v_o(P), 2v_1(P), 4v_2(P), ..., 2^i v_i(P), ...$$

We can see that $v(P)$ lives in an $O(\triangle^k)$ dimensional space, but that only $O(\log(\triangle).|P|)$ entries in this vector are non-zero since the vector $v(P)$ is sparse.

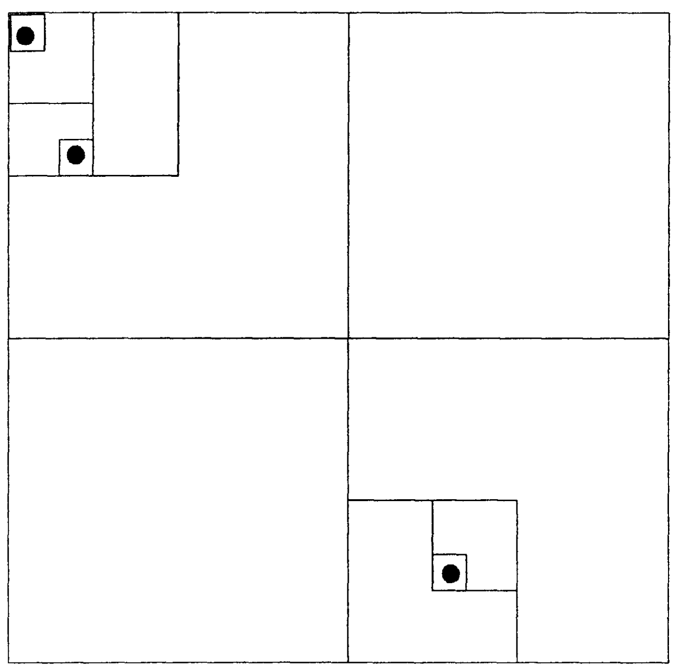

Figure 3-1: Grid Construction with three points

If we label each square as follows: top-left = 0, bottom-left = 1, bottom-right=2, top-right = 3, we then have the coordinates of each point in the above figure be as follows:

- Point 1 = (0, 0, 0 ,0)
- Point 2 = (0, 0, 1, 2)
- Point 3 = (2, 1, 3, 2)

Therefore we have the following values for each grid squares can be calculated as follows which form the vector $v(P)$:
$G_0 = 2, G_1 = 0, G_2 = 1, G_3 = 0$
$G_{00} = 2, G_{01} = 0, G_{02} = 0, G_{03} = 0$
$G_{10} = 0, G_{11} = 0, G_{12} = 0, G_{13} = 0$
$G_{20} = 0, G_{21} = 1, G_{22} = 0, G_{23} = 0$
$G_{30} = 0, G_{31} = 0, G_{32} = 0, G_{33} = 0$

For the remaining G_i, all of remainder are equal to 0 except
$G_{000}, G_{001}, G_{213}, G_{0000}, G_{0012}, G_{2132}$ which are all equal to 1.

It can be seen from that example $v(P)$ is indeed sparse and that most of the entries in $v(P)$ are indeed 0.

This example can be exactly embedded into l_1 and therefore the resulting distortion is exactly 1.

3.1.2 Distortion Bounds

Indyk and Thaper [12] proved the following theorems for an upper bound for the distortion induced by the embedding the EMD metric into l_1.

Theorem 4. *There is a constant C such that for any P, Q , we have $D_M(P, Q) \leq C.|v(P) - v(Q)|_1$.*

Theorem 5. *There is a constant C such that, for a fixed pair P, Q, if we shift the grids randomly, then the expected value of $|v(P) - v(Q)|_1$ is at most $C.D_M(P, Q) log \triangle$.*

They also noted that these theoretical bounds do not provide meaningful practical guarantees and that in practice, the distortion induced by the embedding of the EMD metric into l_1 is much lower.

3.2 Embedding of EMD into the square of l_2

We use the following theorem proved by Linal et al [15] to show how to embed EMD into the square of l_2.

Theorem 6. *An n-point metric space (X, d) may be embedded in a Euclidean space with distortion $\leq c$ if and only if for every matrix Q which is positive, semi-definite and satisfies the $Q \cdot \overrightarrow{1} = \overrightarrow{0}$*

$$\sum_{q_{i,j}>0} q_{i,j} \cdot d_{i,j}^2 + c^2 \sum_{q_{i,j}<0} q_{i,j} \cdot d_{i,j}^2 \leq 0$$

Construction: We can construct the embedding of the EMD into the square of l_2 with the following procedure:

- Let $X = x_1, x_2, ..., x_n$.
- Let f denote the mapping from X to the square of l_2.
- Let $f(x_i) = v_i$. We can then change the coordinates such that we have $f(x_1) = v_1 = 0$.
- Let us now have a matrix A with $A_{ij} = \frac{1}{2}(d_{1i}^2 + d_{1j}^2 - d_{ij}^2)$.

We want $A_{ij} = (v_i, v_j)$ which is equivalent to determining whether A can be written as $B^T B$ for

$$B = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}$$

We know that this decomposition will exist if and only if A is a positive semi-definite matrix and this can be calculated in polynomial time using the Grotschel, Lovasz and Schrijver ellipsoid method [20].

A Semi-Definite Program (SDP) can be written to obtain the value of the best distortion. This SDP is written as a linear program where the "variables" are actually the inner products of the vectors. The solution of the SDP gives us the distortion value.

In our problem, we want to minimize D, subject to the constraints that all the pairwise distance are distorted by at most D. We can formulate it as an SDP by changing D to a vector whose norm denotes the distortion.

$$min(\overrightarrow{D}, \overrightarrow{D})$$

such that

$$\begin{aligned} d_{ij}^2 \leq (v_i, v_i) + (v_j, v_j) - 2(v_i, v_j) \leq (\overrightarrow{D}, \overrightarrow{D}) d_{ij}^2 & \qquad \forall i, j \\ d_{1i}^2 \leq (v_i, v_i) \leq (\overrightarrow{D}, \overrightarrow{D}) d_{1i}^2 & \qquad \forall i \end{aligned}$$

We then used MATLAB's Semi-Definite Programming solver to calculate the distortion.

3.3 Example of embedding EMD into l_1 or square of l_2 with the Manhattan Distance as the underlying metric

Many random distributions of sets of points on the plane are exactly embeddable from EMD to l_1 or the square of l_2. Our goal was to determine a simple example that resulted in distortion for the embedding from the EMD metric to both l_1 and the square of l_2.

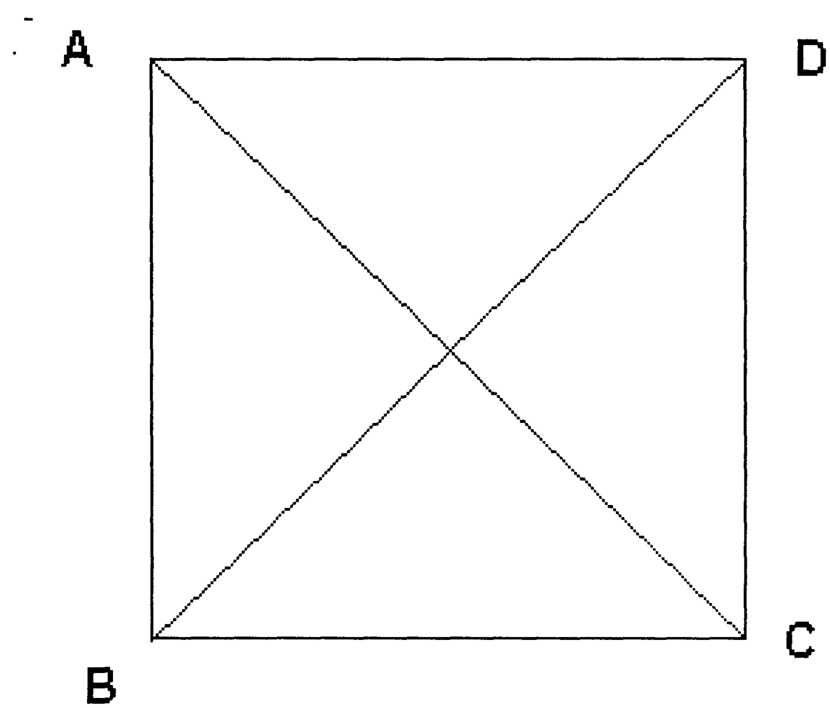

Figure 3-2: Embedding EMD using Manhattan distance - A unit square ABCD

3.3.1 Properties of Example

- We have a unit square ABCD with
  A at (1,0); B at (0,0); C at (0,1); D at (1,1).

- We represent each of the edges as different sets, therefore we have six sets (AC, BD, AB, BC, CD, DA) with two items in each set. The location of each point in each set is simply at the ends of the edge.

- Partition the edges as follows:

  - On the "left side" we have the two diagonals namely:
    { AC, BD } = $\mathcal{A}$.

  - On the "right side" we have the other four horizonal and vertical sides namely:
    { AB, BC, CD, DA } = $\mathcal{B}$.

3.3.2 Proof

We will show that EMD over $\mathcal{A} \cup \mathcal{B}$ is isometric to $K_{2,4}$. That is we show that:

- all distances between $\mathcal{A}$ and $\mathcal{B}$ are 1.

- all distance within edges in set $\mathcal{A}$ and edges in set $\mathcal{B}$ are 2.

The proof is by enumeration of all cases namely:

1. The EMD between the two diagonals AC and BD is 2. ([AC-BD])

2. The EMD between any of the horizontal or vertical sides and either one of the diagonals is 1 since one pair of the nodes overlaps and the other pair is within distance 1. ([AB-AC; AB-BD; BC-AC; BC-BD; CD-AC; CD-BD; DA-AC; DA-BD])

3. The distance between the parallel edges is 2. ([AD-BC; AB-CD])

4. The distances between consecutive edges on the sides bounding the square have distance 2, since one point is shared but the other point is a distance 2 away. ([AB-BC; BC-CD; CD-DA: DA-AB])

The distance matrix is therefore

$$\begin{pmatrix} 0 & 2 & 1 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 2 & 2 & 2 \\ 1 & 1 & 2 & 0 & 2 & 2 \\ 1 & 1 & 2 & 2 & 0 & 2 \\ 1 & 1 & 2 & 2 & 2 & 0 \end{pmatrix}$$

A $K_{2,4}$ graph can be constructed from this example.

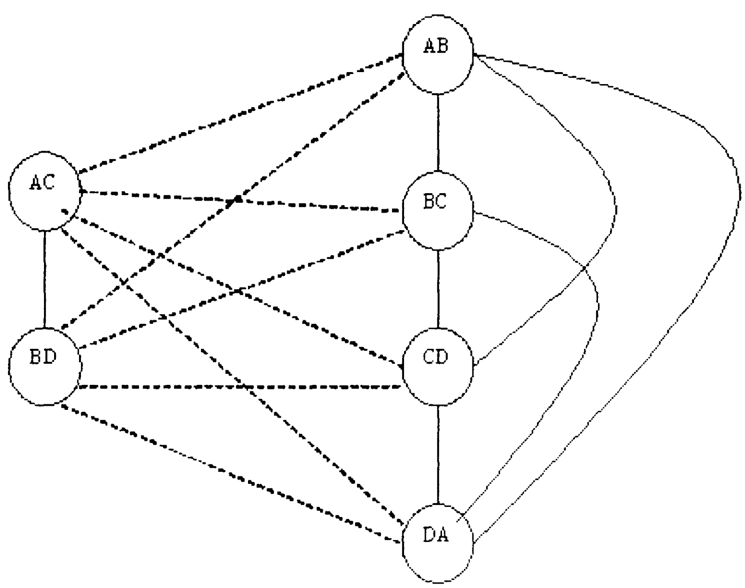

Figure 3-3: Equivalent $K_{2,4}$ graph for vertices of the square. (*Straight lines denote distances = 2; Dotted lines denote distances = 1*)

3.3.3 Computational Results

- We obtain distortion of **1.25** for the embedding of a $K_{2,4}$ graph into l_1 (see Appendix B for equations)
- For the embedding into the square of l_2, the distortion is equal to **1.1180**
- This example can be further simplified and if we remove one of either the vertical or horizontal lines from the square, the distortion for the embedding into l_1 remains at **1.25**, however distortion for the embedding into the square of l_2 decreases to **1.0801**.

3.4 Example of embedding EMD into l_1 or square of l_2 with the Euclidean Distance as underlying metric

If the Euclidean distance is used as the underlying distance metric, then the construction discussed in the previous section no longer provides the highest distortion. However, we can modify the example as follows. This modified example is not equivalent to any $K_{2,n}$ structure but still has distortion greater than 1.

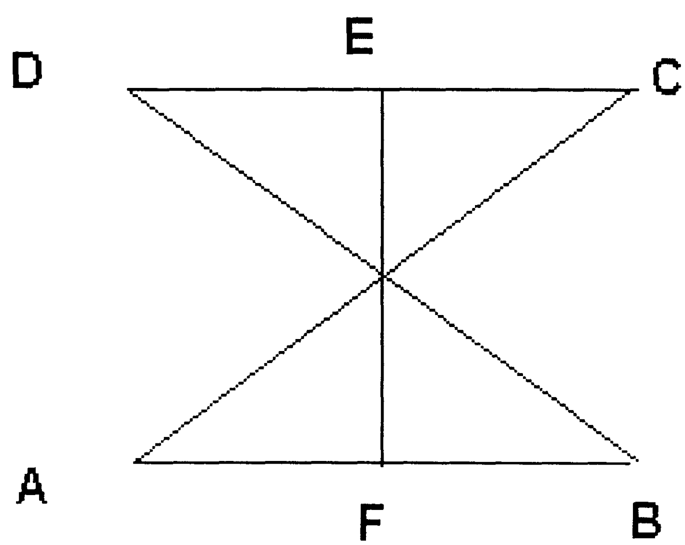

Figure 3-4: Embedding EMD using Euclidean distance - AB and BC = 1 unit

3.4.1 Properties of Example

- The EMD between the two diagonals is 2.
- The EMD between either of the two diagonals and either of the horizontal lines is 1 since one of the points are shared by both lines.
- The EMD between the diagonals the middle line is also equal to 1 since both points are $\frac{1}{2}$ apart from each other.
- The EMD between the horizontal lines is 2.
- The EMD between the horizontal lines and the middle line is equal to $\frac{1}{2} + \sqrt{\frac{1}{2}^2 + 1^2} = \frac{1}{2}(1 + \sqrt{5})$ - *the golden ratio*

The resulting distance matrix is as follows:

$$\begin{pmatrix} 0 & 2 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 2 & \frac{1}{2}(1+\sqrt{5}) \\ 1 & 1 & 2 & 0 & \frac{1}{2}(1+\sqrt{5}) \\ 1 & 1 & \frac{1}{2}(1+\sqrt{5}) & \frac{1}{2}(1+\sqrt{5}) & 0 \end{pmatrix}$$

3.4.2 Computational Results

- Using the method outlined for the embedding of a metric into l_1 we get a distortion of **1.1708**
- For the embedding into the square of l_2, the distortion is equal to **1.0854**
- If the vertical lines were added to the square, the distortion for the embedding into l_1 remains at **1.1708**, while the distortion for the embedding into the square of l_2 increases to **1.1090**.

- If we used the example when the underlying distance was the Manhattan distance we obtain distortion of **1.1213** when embedding it into l_1 and distortion of **1.1035** when embedding it into the square of l_2.

Chapter 4

Embedding of Earth Mover Distance metric into normed spaces in higher dimensions

4.1 Examples in Three Dimensions

We examine several constructions of points in three dimensions and compute their distortion when embedding these points into the normed spaces of l_1 and the square of l_2.

4.1.1 Eleven Edge Cube Example

We can recursively construct an example using the two dimensional example with the points forming the vertices of a cube. If we flatten out these points on the cube unto a two dimensional plane, our construction will look as follows.

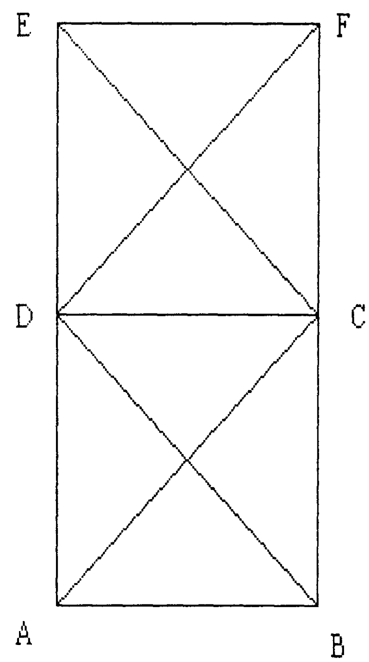

Figure 4-1: Eleven Edge Example of points on a flattened cube for embedding EMD into l_1 or the square of l_2.

The points in this example are:

- A - (0, 0, 0)
- B - (0, 0, 1)
- C - (0, 1, 1)
- D - (0, 1, 0)
- E - (1, 1, 0)
- F - (1, 1, 1)

There are 11 sets located on the 11 edges depicted in the diagram.
AB. AC. AD. BC. BD. CD. CE, CF. DE, DF, EF.

The distance matrix for this example

$$\begin{pmatrix} 0 & 2 & 1 & 1 & 2 & 2 & 2 & 3 & 3 & 4 & 4 \\ 2 & 0 & 1 & 1 & 2 & 2 & 4 & 3 & 3 & 4 & 4 \\ 1 & 1 & 0 & 2 & 1 & 1 & 3 & 2 & 2 & 3 & 3 \\ 1 & 1 & 2 & 0 & 1 & 1 & 3 & 2 & 2 & 3 & 3 \\ 2 & 2 & 1 & 1 & 0 & 2 & 2 & 1 & 1 & 2 & 2 \\ 2 & 2 & 1 & 1 & 2 & 0 & 4 & 3 & 3 & 4 & 2 \\ 2 & 4 & 3 & 3 & 2 & 4 & 0 & 1 & 1 & 2 & 2 \\ 3 & 3 & 2 & 2 & 1 & 3 & 1 & 0 & 2 & 1 & 1 \\ 3 & 3 & 2 & 2 & 1 & 3 & 1 & 2 & 0 & 1 & 1 \\ 4 & 4 & 3 & 3 & 2 & 4 & 2 & 1 & 1 & 0 & 2 \\ 4 & 4 & 3 & 3 & 2 & 2 & 2 & 1 & 1 & 2 & 0 \end{pmatrix}$$

This distance matrix can be sub-divided into the following matrices.

1. A $K_{2,4}$ sub-graph which is formed by the edges *AB, AC, AD, BC, BD, CD*.

$$\begin{pmatrix} 0 & 2 & 1 & 1 & 2 & 2 \\ 2 & 0 & 1 & 1 & 2 & 2 \\ 1 & 1 & 0 & 2 & 1 & 1 \\ 1 & 1 & 2 & 0 & 1 & 1 \\ 2 & 2 & 1 & 1 & 0 & 2 \\ 2 & 2 & 1 & 1 & 2 & 0 \end{pmatrix}$$

2. A $K_{2,3}$ sub-graph which is formed by the edges CE, CF, DE, DF, EF.

$$\begin{pmatrix} 0 & 1 & 1 & 2 & 2 \\ 1 & 0 & 2 & 1 & 1 \\ 1 & 2 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 \end{pmatrix}$$

3. The distance matrix between the $K_{2,4}$ and $K_{2,3}$ partition.

$$\begin{pmatrix} 2 & 3 & 3 & 4 & 4 \\ 4 & 3 & 3 & 4 & 4 \\ 3 & 2 & 2 & 3 & 3 \\ 3 & 2 & 2 & 3 & 3 \\ 2 & 1 & 1 & 2 & 2 \\ 4 & 3 & 3 & 4 & 2 \end{pmatrix}$$

4. The distance matrix between the $K_{2,4}$ and $K_{2,3}$ partition.

$$\begin{pmatrix} 2 & 4 & 3 & 3 & 2 & 4 \\ 3 & 3 & 2 & 2 & 1 & 3 \\ 3 & 3 & 2 & 2 & 1 & 3 \\ 4 & 4 & 3 & 3 & 2 & 4 \\ 4 & 4 & 3 & 3 & 2 & 2 \end{pmatrix}$$

Computational Results

- This structure had a distortion of **1.2857** when embedded into l_1. Though there is no direct isometry to $K_{2,5}$, it has the exact distortion of **1.2857** after embedding into l_1 as well.

- This structure has a distortion of **1.1180** when embedded into the square of l_2.

4.1.2 Fifteen Edge Cube Example

In this example, we have the same points as the above example, except that now we have the additional edges between *AE, AF, BF, BE*.

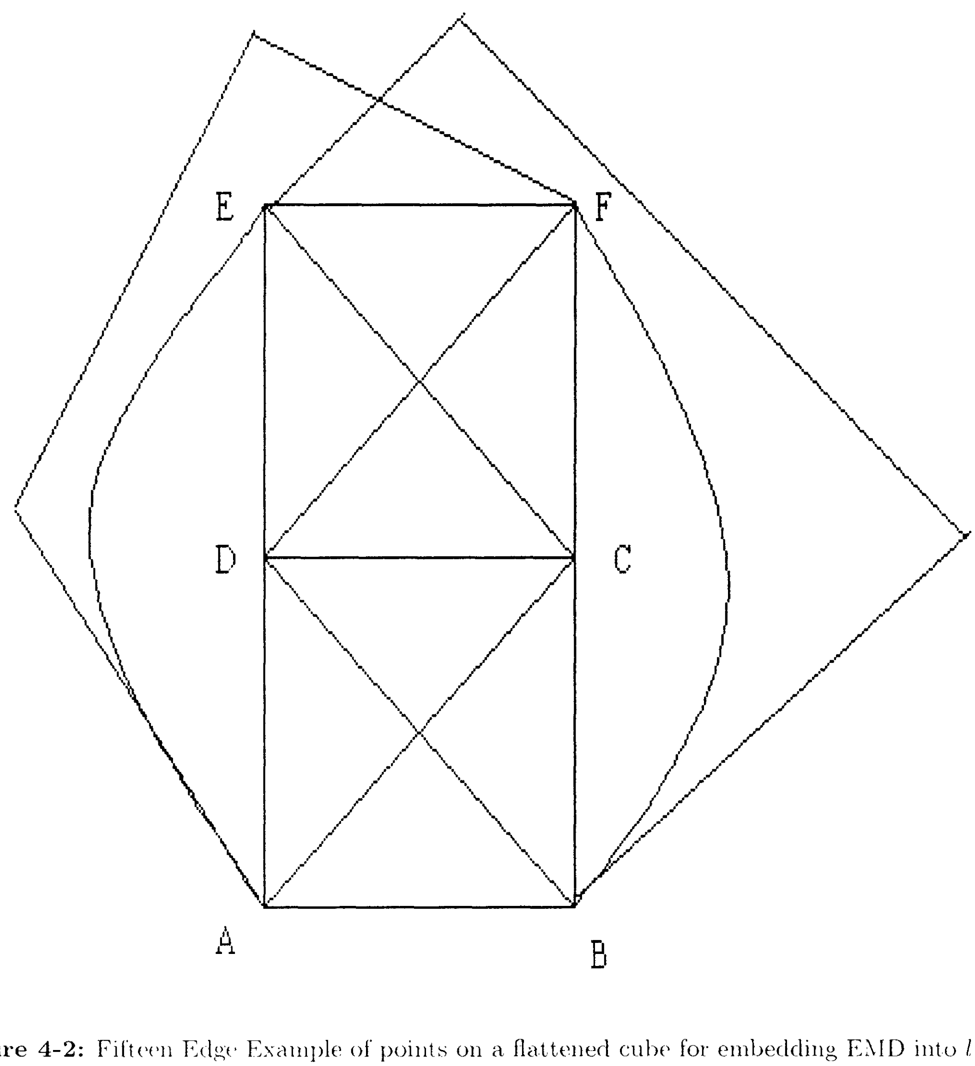

Figure 4-2: Fifteen Edge Example of points on a flattened cube for embedding EMD into l_1 or the square of l_2.

The distance matrix is as follows

$$\begin{pmatrix}
0 & 2 & 1 & 1 & 2 & 2 & 2 & 3 & 3 & 4 & 4 & 1 & 2 & 2 & 3 \\
2 & 0 & 1 & 1 & 2 & 2 & 4 & 3 & 3 & 4 & 4 & 3 & 2 & 2 & 3 \\
1 & 1 & 0 & 2 & 1 & 1 & 3 & 2 & 2 & 3 & 3 & 2 & 1 & 3 & 2 \\
1 & 1 & 2 & 0 & 1 & 1 & 3 & 2 & 2 & 3 & 3 & 2 & 3 & 1 & 2 \\
2 & 2 & 1 & 1 & 0 & 2 & 2 & 1 & 1 & 2 & 2 & 3 & 2 & 2 & 3 \\
2 & 2 & 1 & 1 & 2 & 0 & 4 & 3 & 3 & 4 & 2 & 3 & 2 & 2 & 1 \\
2 & 4 & 3 & 3 & 2 & 4 & 0 & 1 & 1 & 2 & 2 & 1 & 2 & 2 & 3 \\
3 & 3 & 2 & 2 & 1 & 3 & 1 & 0 & 2 & 1 & 1 & 2 & 1 & 3 & 2 \\
3 & 3 & 2 & 2 & 1 & 3 & 1 & 2 & 0 & 1 & 1 & 2 & 3 & 1 & 2 \\
4 & 4 & 3 & 3 & 2 & 4 & 2 & 1 & 1 & 0 & 2 & 3 & 2 & 2 & 3 \\
4 & 4 & 3 & 3 & 2 & 2 & 2 & 1 & 1 & 2 & 0 & 3 & 2 & 2 & 1 \\
1 & 3 & 2 & 2 & 3 & 3 & 1 & 2 & 2 & 3 & 3 & 0 & 1 & 1 & 2 \\
2 & 2 & 1 & 3 & 2 & 2 & 2 & 1 & 3 & 2 & 2 & 1 & 0 & 2 & 1 \\
2 & 2 & 3 & 1 & 2 & 2 & 2 & 3 & 1 & 2 & 2 & 1 & 2 & 0 & 1 \\
3 & 3 & 2 & 2 & 3 & 1 & 3 & 2 & 2 & 3 & 1 & 2 & 1 & 1 & 0
\end{pmatrix}$$

In this case the fifteen edges can be sub-divided into three categories.

- A $K_{2,4}$ sub-graph which is formed by the edges *AB, AC, AD, BC, BD, CD.*
- A $K_{2,3}$ sub-graph which is formed by the edges *CE, CF, DE, DF, EF.*
- A $K_{2,2}$ sub-graph which is formed by the edges *AE. AF, BF, BE.*

Computational Results

We know that both the $K_{2,4}$ and $K_{2,3}$ graphs cannot be embedded exactly into l_1 or the square of l_2. The $K_{2,2}$ sub-graph can be embedded without any distortion into both l_1 or the square of l_2. In this example the interactions of the various subgraphs increases the distortion into l_1 to **1.3000** and the distortion into the square of l_2 is now **1.1396**.

4.1.3 Twenty Edge Cube Example

In this example, we add the following two points G and H at (1, 0, 0) and (1, 1, 0) respectively. We also have the following additional edges between *EG, EH, FG, FH, GH.* The distance matrix now contains an additional $K_{2,3}$ subgraph and the resulting interactions between the four sub-graphs. It was not computationally feasible to embed this graph into l_1 since this running time and required space increases exponentially. However, the embedding into the square of l_2 is polynomial and remains computationally feasible for this 20×20 distance matrix. The distortion of the twenty edge cube into the square of l_2 was calculated to be **1.1644**.

4.1.4 Twenty-Four Edge Cube Example

We have the same eight points A, B, C, D, E, F, G, H located at the eight vertices of the cube. The new edges that we add formed another $K_{2,2}$ sub-graph. The new edges that were added were AG, AH, BG, BH. The distortion of the twenty-four edge cube when embedded into the square of l_2 was calculated to be **1.1717**.

4.1.5 Twenty-Eight Edge Cube Example

We add four more edges to the vertices of the cube which form another $K_{2,2}$ subgraph. The new edges that were added were CG, CH, DG, DH. The distortion of the twenty-eight edge cube after embedding into the square of l_2 was **1.1792**.

4.2 Analysis of Three Dimensional Example

The above examples show how the distortion for the embedding into the square of l_2 increases with each new sub-graph that is added to the construction. There is no obvious translation of these constructions into a standard $K_{m,n}$ graph, but we know that every construction would have at least as high a distortion as the previous example. It was not possible to calculate the distortion for the embedding of these constructions into l_1 since this embedding created an exponential number of constraints which became computational infeasible when we had more than fifteen edges.

The following table shows the distortion created when embedding various $K_{m,n}$ into l_1 and the square of l_2.

Graph	Distortion into l_1	Distortion into square of l_2
$K_{2,2}$	1.0000	1.0000
$K_{2,3}$	1.2500	1.0801
$K_{2,4}$	1.2500	1.1180
$K_{2,5}$	1.2857	1.1402
$K_{2,6}$	1.2857	1.1547
$K_{2,7}$	1.3000	1.1649
$K_{2,8}$	1.3000	1.1726
$K_{2,9}$	1.3077	1.1785
$K_{2,10}$	1.3077	1.1832
$K_{2,11}$	1.3125	1.1871
$K_{2,12}$	1.3125	1.1905
$K_{2,13}$	1.3158	1.1929
$K_{3,3}$	1.2500	1.1547
$K_{4,4}$	1.3333	1.2247
$K_{5,5}$	1.3750	1.2649
$K_{6,6}$	1.4000	1.2910
$K_{7,7}$	1.4167	1.3093

Table 4.1: Distortion of embedding certain $K_{m,n}$ graphs into l_1 and into the square of l_2.

4.3 Conjecture of results in higher dimensions

We can recursively construct examples of points in higher dimensions using these structures as sub-graphs. We analyzed various constructions on points on the tesseract (a four dimensional cube). All of the previous point constructions discussed earlier can easily be constructed on part of the tesseract.

We can also combine these structures to achieve higher distortion. In the previous section, we discussed the fifteen edge example where we had the following six points on six corners of the cube:

$$(0,\ 0,\ 0);\ (0,\ 0,\ 1);\ (0,\ 1,\ 1);\ (0,\ 1,\ 0);\ (1,\ 1,\ 0);\ (1,\ 1,\ 1)$$

We had edges joining every pair of points and then calculated the EMD between each of these edges. These EMD distances formed a 15×15 matrix. This distance matrix had distortion of **1.3000** when embedded into 1_1 and the distortion into the square of l_2 was now **1.1396**. We can modify this example for the four dimensional case. We can do so by creating two copies of this structure and hence obtain the following twelve points:

$$\begin{gathered}(0,\ 0,\ 0,\ 0);\ (0,\ 0,\ 0,\ 1);\ (0,\ 0,\ 1,\ 1);\ (0,\ 0,\ 1,\ 0);\ (0,\ 1,\ 1,\ 0);\ (0,\ 1,\ 1,\ 1);\\ (1,\ 0,\ 0,\ 0);\ (1,\ 0,\ 0,\ 1);\ (1,\ 0,\ 1,\ 1);\ (1,\ 0,\ 1,\ 0);\ (1,\ 1,\ 1,\ 0);\ (1,\ 1,\ 1,\ 1).\end{gathered}$$

In this case, with edges joining every pair of points and then compute the EMD between these edges, the result is a 66×66 distance matrix. This distance matrix can then be theoretically embedded into 1_1 and the square of 1_2. However, it was not computationally feasible to embed this matrix into l_1 with our current embedding. Embedding into the square of l_2 resulted in distortion of **1.2007**.

This example could theoretically be simplified, however we were unable to determine a method for determining which edges and their resulting interactions contributed most to the distortion. As a result, we were unable to determine another example on the tesseract with higher distortion. Another example that could have been analyzed was using the twenty-eight edge example from the earlier section on multiple vertices on the tesseract. We conjecture that recursive constructions using some of the structures described earlier will result in higher distortion in higher dimensions. We were unable to come up with a formal proof for this conjecture.

Chapter 5

Conclusions and Open Problems

We discussed the construction of examples in two and three dimensions that show a lower bound for embedding the Earth Mover Distance (EMD) metric into the normed spaces of l_1 and the square of l_2. The EMD is a very important metric that is used in a number of applications ranging from similarity searching, image retrieval and vector feature comparison. The EMD is defined as the least amount of work needed to move a mass of earth spread out in space into a collection of holes in that same space.

We showed an example in two dimensions with the Manhattan distance defined as the underlying distance metric for the EMD that is isometric to $K_{2,4}$. This example of points then has a distortion of **1.25** when embedded into the normed space l_1 and also a distortion of **1.1180** when embedded into the normed space of the square of l_2. We also constructed an example using the Euclidean distance is used as the underlying metric for the EMD. In that example, there was a distortion of **1.1667** when embedded into the normed space l_1 and distortion of **1.0854** when embedded into the normed space the square of l_2. We discussed other examples of constructions of points in three dimensions on the vertices of a cube and in higher dimensions for example on the vertices of a tesseract that cannot be embedded exactly into l_1 and the square of l_2.

Further research can be done in obtaining general lower bounds for higher dimensions. Also we can try to determine lower bounds for EMD with other underlying metrics in addition to the Manhattan distance and the Euclidean distance.

Appendix A

Code

A.1 Earth Mover Distance Calculations

We used the exisiting code written by C. Tomasi [26] to calculate the Earth Mover Distance for various data sets. We specified the feature data type in the header file with structures. Therefore, for two dimensions we would have the following:

```
typedef struct {
int X,Y;}
feature_t;
```

Similarly, for three dimensions we have

```
typedef struct {
int X,Y,Z;}
feature_t;
```

The signature data type signature_t is defined in the header file as follows:

```
typedef struct
{
  int n;                /* Number of features in the distribution */
  feature_t *Features;  /* Pointer to the features vector */
  float *Weights;       /* Pointer to the weights of the features */
}
  signature_t;
```

We compute an EMD by calling the following:

```
float emd(signature_t *Signature1, signature_t *Signature2,
  float (*Dist)(feature_t *, feature_t *),
  flow_t *Flow, int *FlowSize)
```

where

1. Signature_1, Signature_2: Pointers to the two signatures which we want to compute their distance for.

2. Dist: Pointer to the ground distance function. This is the function that computes the distance between two features.

3. Flow: Pointer to a vector of flow_t (which was defined in the header file) where the resulting flow will be stored. Flow must have n_1 + n_2 - 1 elements, where n_1 and n_2 are the sizes of the two signatures respectively. If NULL, the flow is not returned.

4. FlowSize: In case Flow is not NULL, FlowSize points to a integer where the number of flow elements which is always less or equal to n_1 + n_2 -1 is written.

A.2 Embedding into l_1 and square of l_2

We used existing code written by A. Andoni [1] that computed the embedding of a given metric into the normed space l_1 and the normed space the square of l_2. The input file to this module was the Earth Mover Distance matrix computed from the given data set. In the embedding into l_1 the output was a linear program for MATLAB. This was then solved using the linear programming solver linprog.

```
[x,values]=linprog(f,A,b,Aeq,beq,lb,ub)
```

The values for the cut metrics are constrained to be greater than or equal to zero in order to output a valid embedding. Since the embedding algorithm into l_1 using cut matrices was exponential. it was computationally feasible to embed distance matrices of only up to size 16x16. Therefore, it was not possible to determine the distortion for various constructions of points in three and higer dimensions.

In the embedding into the square of l_2, the output was a semi-definite program which was then solved using MATLAB's Semi-Definite Program package.

```
[A,b,C,blk]=importSQLP('test.sqlp');
```

The embedding algorithm into the square of l_2 using semi-definite programming is polynomial and in that case it was computationally feasible to embed distance matrices of up to size 50x50. This still limited our ability to determine the distortion for constructions of points in four and higher dimensions.

A.3 Java Applet for Construction of Points

In order to provide intuition to help us determine what constructions of points cannot be exactly embedded into l_1 or the square of l_2 a simple Java applet that allowed for basic manipulation of the points on a grid for several distributions (colors) was written. This applet allowed us to determine what structures needed to be present in the example to ensure that we have distortion after the embedding.

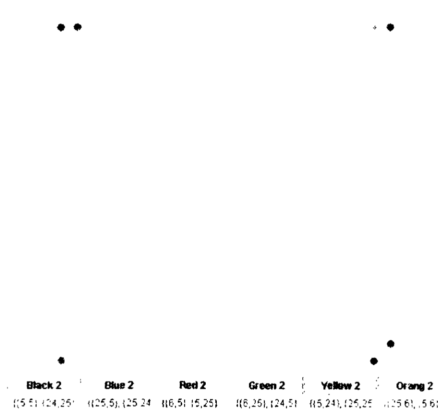

Figure A-1: Screenshot of Applet with example

Appendix B

Equations

For the example of embedding EMD into l_1 using the Manhattan Distance as the underlying metric, we have the following distance matrix.

$$\begin{pmatrix} 0 & 2 & 1 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 2 & 2 & 2 \\ 1 & 1 & 2 & 0 & 2 & 2 \\ 1 & 1 & 2 & 2 & 0 & 2 \\ 1 & 1 & 2 & 2 & 2 & 0 \end{pmatrix}$$

The resulting linear program for this matrix is as follows:

Minimize: x_0

Subject To

$$x_1 + x_5 + x_9 + x_{13} + x_{17} + x_{21} + x_{25} + x_{29} + x_{33} + x_{37} + x_{41} + x_{45} + x_{49} + x_{53} + x_{57} + x_{61} \leq 2$$

$$2x_0 + x_1 + x_5 + x_9 + x_{13} + x_{17} + x_{21} + x_{25} + x_{29} + x_{33} + x_{37} + x_{41} + x_{45} + x_{49} + x_{53} + x_{57} + x_{61} \geq 2$$

$$x_1 + x_3 + x_9 + x_{11} + x_{17} + x_{19} +_{x25} + x_{27} + x_{33} + x_{35} + x_{41} + x_{43} + x_{49} + x_{51} + x_{57} + x_{59} \leq 1$$

$$x_0 x_1 + x_3 + x_9 + x_{11} + x_{17} + x_{19} +_{x25} + x_{27} + x_{33} + x_{35} + x_{41} + x_{43} + x_{49} + x_{51} + x_{57} + x_{59} \geq 1$$

$$x_1 + x_3 + x_5 + x_7 + x_{17} + x_{19} + x_{21} + x_{23} + x_{33} + x_{35} + x_{37} + x_{39} + x_{49} + x_{51} + x_{53} + x_{55} \leq 1$$

$x_0+x_1+x_3+x_5+x_7+x_{17}+x_{19}+x_{21}+x_{23}+x_{33}+x_{35}+x_{37}+x_{39}+x_{49}+x_{51}+x_{53}+x_{55} \geq 1$

$x_1+x_3+x_5+x_7+x_9+x_{11}+x_{13}+x_{15}+x_{33}+x_{35}+x_{37}+x_{39}+x_{41}+x_{43}+x_{45}+x_{47} \leq 1$

$x_0+x_1+x_3+x_5+x_7+x_9+x_{11}+x_{13}+x_{15}+x_{33}+x_{35}+x_{37}+x_{39}+x_{41}+x_{43}+x_{45}+x_{47} \geq 1$

$x_1+x_3+x_5+x_7+x_9+x_{11}+x_{13}+x_{15}+x_{17}+x_{19}+x_{21}+x_{23}+x_{25}+x_{27}+x_{29}+x_{31} \leq 1$

$x_0+x_1+x_3+x_5+x_7+x_9+x_{11}+x_{13}+x_{15}+x_{17}+x_{19}+x_{21}+x_{23}+x_{25}+x_{27}+x_{29}+x_{31} \geq 1$

$x_3+x_5+x_{11}+x_{13}+x_{19}+x_{21}+x_{27}+x_{29}+x_{35}+x_{37}+x_{43}+x_{45}+x_{51}+x_{53}+x_{59}+x_{61} \leq 1$

$x_0+x_3+x_5+x_{11}+x_{13}+x_{19}+x_{21}+x_{27}+x_{29}+x_{35}+x_{37}+x_{43}+x_{45}+x_{51}+x_{53}+x_{59}+x_{61} \geq 1$

$x_3+x_7+x_9+x_{13}+x_{19}+x_{23}+x_{25}+x_{29}+x_{35}+x_{39}+x_{41}+x_{45}+x_{51}+x_{55}+x_{57}+x_{61} \leq 1$

$x_0+x_3+x_7+x_9+x_{13}+x_{19}+x_{23}+x_{25}+x_{29}+x_{35}+x_{39}+x_{41}+x_{45}+x_{51}+x_{55}+x_{57}+x_{61} \geq 1$

$x_3+x_7+x_{11}+x_{15}+x_{17}+x_{21}+x_{25}+x_{29}+x_{35}+x_{39}+x_{43}+x_{47}+x_{49}+x_{53}+x_{57}+x_{61} \leq 1$

$x_0+x_3+x_7+x_{11}+x_{15}+x_{17}+x_{21}+x_{25}+x_{29}+x_{35}+x_{39}+x_{43}+x_{47}+x_{49}+x_{53}+x_{57}+x_{61} \geq 1$

$x_3+x_7+x_{11}+x_{15}+x_{19}+x_{23}+x_{27}+x_{31}+x_{33}+x_{37}+x_{41}+x_{45}+x_{49}+x_{53}+x_{57}+x_{61} \leq 1$

$x_0+x_3+x_7+x_{11}+x_{15}+x_{19}+x_{23}+x_{27}+x_{31}+x_{33}+x_{37}+x_{41}+x_{45}+x_{49}+x_{53}+x_{57}+x_{61} \geq 1$

$x_5+x_7+x_9+x_{11}+x_{21}+x_{23}+x_{25}+x_{27}+x_{37}+x_{39}+x_{41}+x_{43}+x_{53}+x_{55}+x_{57}+x_{59} \leq 2$

$2x_0+x_5+x_7+x_9+x_{11}+x_{21}+x_{23}+x_{25}+x_{27}+x_{37}+x_{39}+x_{41}+x_{43}+x_{53}+x_{55}+x_{57}+x_{59} \geq 2$

$x_5+x_7+x_{13}+x_{15}+x_{17}+x_{19}+x_{25}+x_{27}+x_{37}+x_{39}+x_{45}+x_{47}+x_{49}+x_{51}+x_{57}+x_{59} \leq 2$

$2x_0+x_5+x_7+x_{13}+x_{15}+x_{17}+x_{19}+x_{25}+x_{27}+x_{37}+x_{39}+x_{45}+x_{47}+x_{49}+x_{51}+x_{57}+x_{59} \geq 2$

$x_5+x_7+x_{13}+x_{15}+x_{21}+x_{23}+x_{29}+x_{31}+x_{33}+x_{35}+x_{41}+x_{43}+x_{49}+x_{51}+x_{57}+x_{59} \leq 2$

$2x_0+x_5+x_7+x_{13}+x_{15}+x_{21}+x_{23}+x_{29}+x_{31}+x_{33}+x_{35}+x_{41}+x_{43}+x_{49}+x_{51}+x_{57}+x_{59} \geq 2$

$x_9+x_{11}+x_{13}+x_{15}+x_{17}+x_{19}+x_{21}+x_{23}+x_{41}+x_{43}+x_{45}+x_{47}+x_{49}+x_{51}+x_{53}+x_{55} \leq 2$

$2x_0+x_9+x_{11}+x_{13}+x_{15}+x_{17}+x_{19}+x_{21}+x_{23}+x_{41}+x_{43}+x_{45}+x_{47}+x_{49}+x_{51}+x_{53}+x_{55} \geq 2$

$x_9+x_{11}+x_{13}+x_{15}+x_{25}+x_{27}+x_{29}+x_{31}+x_{33}+x_{35}+x_{37}+x_{39}+x_{49}+x_{51}+x_{53}+x_{55} \leq 2$

$2x_0+x_9+x_{11}+x_{13}+x_{15}+x_{25}+x_{27}+x_{29}+x_{31}+x_{33}+x_{35}+x_{37}+x_{39}+x_{49}+x_{51}+x_{53}+x_{55} \geq 2$

$x_{17}+x_{19}+x_{21}+x_{23}+x_{25}+x_{27}+x_{29}+x_{31}+x_{33}+x_{35}+x_{37}+x_{39}+x_{41}+x_{43}+x_{45}+x_{47} \leq 2$

$2x_0+x_{17}+x_{19}+x_{21}+x_{23}+x_{25}+x_{27}+x_{29}+x_{31}+x_{33}+x_{35}+x_{37}+x_{39}+x_{41}+x_{43}+x_{45}+x_{47} \leq 2$

Bounds

$0 \leq x_1$

$0 \leq x_3$

$$0 \leq x_5$$
$$0 \leq x_7$$
$$0 \leq x_9$$
$$0 \leq x_{11}$$
$$0 \leq x_{13}$$
$$0 \leq x_{15}$$
$$0 \leq x_{17}$$
$$0 \leq x_{19}$$
$$0 \leq x_{21}$$
$$0 \leq x_{23}$$
$$0 \leq x_{25}$$
$$0 \leq x_{27}$$
$$0 \leq x_{29}$$
$$0 \leq x_{31}$$
$$0 \leq x_{33}$$
$$0 \leq x_{35}$$
$$0 \leq x_{37}$$
$$0 \leq x_{39}$$
$$0 \leq x_{41}$$
$$0 \leq x_{43}$$
$$0 \leq x_{45}$$
$$0 \leq x_{47}$$
$$0 \leq x_{49}$$
$$0 \leq x_{51}$$
$$0 \leq x_{53}$$
$$0 \leq x_{55}$$
$$0 \leq x_{57}$$
$$0 \leq x_{59}$$
$$0 \leq x_{61}$$

This linear equation was then solved with MATLAB's linprog function. The solution for this linear program was calculated to be:

$x_1 = 0.2500$

$x_3 = 0.0000$

$x_5 = 0.0000$

$x_7 = 0.0000$

$x_9 = 0.0000$

$x_{11} = 0.0000$

$x_{13} = 0.0000$

$x_{15} = 0.2500$

$x_{17} = 0.0000$

$x_{19} = 0.0000$

$x_{21} = 0.0000$

$x_{23} = 0.2500$

$x_{25} = 0.0000$

$x_{27} = 0.2500$

$x_{29} = 0.0000$

$x_{31} = 0.0000$

$x_{33} = 0.2500$

$x_{35} = 0.0000$

$x_{37} = 0.0000$

$x_{39} = 0.2500$

$x_{41} = 0.0000$

$x_{43} = 0.2500$

$x_{45} = 0.0000$

$x_{47} = 0.0000$

$x_{49} = 0.2500$

$x_{51} = 0.2500$

$x_{53} = 0.0000$

$x_{55} = 0.0000$

$x_{57} = 0.2500$

$x_{59} = 0.0000$

$x_{61} = 0.2500$

The final distortion was equal to **1.25**.

Bibliography

[1] A. Andoni, M. Deza, A. Gupta, P. Indyk, and S. Raskhodikova. Lower bounds for embedding of Edit Distance into normed spaces. *Proceddings of the ACM-Siam Symposium on Discrete Algorithms*, 2004.

[2] Rajesh Batra, Lambertus Hesselink, and Yingmei Lavin. Feature comparisons of vector fields using earth mover's distance. *Proceedings of the conference on Visualization*, pages 105–114, 1999.

[3] J. Bourgain. On Lipschitz of finite metric spaces in Hilbert space. *Israel Journal of Math*, 1985.

[4] Moses Charikar, Kai Li, and Qin Lv. Image similarity search with compact data structures. *Conference on Information and Knowledge Management*, 2004.

[5] M. Deza and M. Laurent. *Geometry of Cuts and Metrics*, volume 15 of *Algorithms and Combinatorics*. Springer-Verlag, 1997.

[6] J Edmonds and R. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM*, 19:248–264, 1972.

[7] M. Fairchild. *Color Appearance Models*. Addison-Wesley, Reading, Massachusetts, 1998.

[8] S. Hiller and G. Liberman. *Introduction to Mathematical Programming*. McGraw-Hill, 1990.

[9] F. Hitchcock. The distribution of a product from several sources to numerous localities. *Journal of Mathematical Physics*, 20:224–230, 1941.

[10] P. Indyk. Algorithmic applications of low-distortion embeddings. *42nd Annual IEE Symposium on Foundations of Computer Science*, pages 10–33, 2001.

[11] P. Indyk and J. Matousek. Low distortion embeddings of finite metric spaces. *CRC Handbook of Discrete and Computational Geometry*, 2003.

[12] P. Indyk and N Thaper. Fast image retrieval via embeddings. *Third International Workshop on Statistical and Computational Theories of Vision*, 2003.

[13] W. Johnson and J. Lindenstauss. Extensions of Lipschitz maps into a Hilbert space. *Contemporary Mathematics*, 1984.

[14] D. König. Graphs and matrices. *Matematikaies Fizikai Lapok*, 38:116–119, 1931.

[15] N. Linial, E. London, and Y Rabinovich. The geometry of graphs and some of its algorithmic applications. *Proceddings of 35th Annual IEEE Symposium on Foundations of Computer Science*, pages 577–591, 1994.

[16] J. Matousek. Bi-Lipschitz embeddings into low-dimensional euclidean spaces. *Comment. Math. Univ Carolin*, 31:589–600, 1990.

[17] J. Matousek. On the distortion required for embedding finite metric spaces into normed spaces. *Israel Journal of Math*, 93:333–344, 1996.

[18] I. Newman and Y. Rabinvoich. A lower bound on the distortion of embedding planar metrics into euclidean space. *Discrete Computational Geometry*, 29:77–81, 2003.

[19] J Orlin. A polynomial time primal network simplex algorithm for minimum cost flows. *Mathematical Programming*, 78:109–129, 1997.

[20] M Grötschel. L. Lovásv, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica I*, pages 169–197, 1981.

[21] Y. Rubner and C. Tomasi. Texture metrics. *Proceddings of the IEEE International Conference on Systems, Man and Cybernetics*, 1998.

[22] Y. Rubner, C. Tomasi, and L. Guibas. The Earth Mover's Distance, multi-dimensional scaling, and color-based image retrieval. *Proceddings of the ARPA Image Understanding Workshop*. 1997.

[23] Y. Rubner, C. Tomasi, and L. Guibas. A metric for distributions with applications to image databases. *International Conference on Computer Vision*, 1998.

[24] Y. Rubner, C. Tomasi, and L. Guibas. The Earth Mover's Distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[25] M. Stricker and M. Orengo. Similarity of color images. *SPIE Conference on Storage and Retrieval for Image and Video Databases III*, 2420:381–392, 1995.

[26] C. Tomasi. Code for the Earth Mover Distance (EMD). *http://www.cs.duke.edu/ tomasi/software/emd.htm*, May 1998.