The Metric Nearness Problem with Applications

Inderjit S. Dhillon* Suvrit Sra† Joel A. Tropp‡

8 June 2003

UTCS Technical Report # TR-03-23

Abstract

Many practical applications in machine learning require pairwise distances among a set of objects. It is often desirable that these distance measurements satisfy the properties of a metric, especially the triangle inequality. Applications that could benefit from the metric property include data clustering and metric-based indexing of databases. In this paper, we present the *metric nearness problem*: Given a dissimilarity matrix, find the "nearest" matrix of distances that satisfy the triangle inequalities. A weight matrix in the formulation captures the confidence in individual dissimilarity measures, including the case of altogether missing distances. For an important class of nearness measures, the problem can be attacked with convex optimization techniques. A pleasing aspect of this formulation is that we can compute globally optimal solutions. Experiments on some sample dissimilarity matrices are presented, including some from biology.

1 Introduction

Imagine that a lazy graduate student has been asked to measure the pairwise distances among a group of objects in a metric space. He doesn't complete the experiment, and he must figure out the remaining numbers before his adviser returns from her conference. Obviously, all the distances need to be consistent, but the student does not know very much about the space in which the objects are embedded. One way to solve his problem is to find the "nearest" complete set of distances that satisfy the triangle inequalities. This procedure respects the measurements that have already been taken while forcing the missing numbers to behave like distances.

More charitably, suppose that the student has finished the experiment, but (measurements being what they are) the numbers do not satisfy the triangle inequality. The student knows that they must represent distances, so he would like to massage the data so that it corresponds with his *a priori* knowledge. Once again, the solution seems to require the "nearest" set of distances that satisfy the triangle inequalities.

Matrix nearness problems [Hig89] offer a natural framework for developing this idea. If there are n points, we may collect the measurements into an $n \times n$ symmetric matrix whose (j, k) entry represents the dissimilarity between the j-th and k-th objects. Then, we seek to approximate this matrix by another whose entries satisfy the triangle inequalities. That is, $m_{ik} \leq m_{ij} + m_{jk}$ for every triple (i, j, k). Any such matrix will represent the distances among n points in some metric space.

*Department of Computer Sciences, University of Texas at Austin. inderjit@cs.utexas.edu
†Department of Computer Sciences, University of Texas at Austin. suvrit@cs.utexas.edu
‡Inst. for Comp. Engr. and Science, University of Texas at Austin. jtropp@ices.utexas.edu

We measure the approximation error with a matrix norm that depends on how the corrected matrix should relate to the input matrix. For example, one might prefer to change a few entries significantly or to change all the entries a little.

We call the problem of approximating general data by metric data the *metric nearness problem*. This problem seems to be unstudied, although the literature contains a few related problems (see Section 1.1). In Section 2, we present a rigorous statement of the metric nearness problem and we argue that it is free of local minima. Then we show how standard methods of mathematical programming can be used to produce a global optimizer when the approximation error is measured using an ℓ_p vector norm. In Section 3, we present an application to clustering. Clustering, like many computational problems, can be solved much more efficiently when the data are drawn from a metric space. By approximating non-metric data with metric data, we can take advantage of fast clustering algorithms, while retaining guarantees on the quality of the results. Clustering algorithms might also benefit from using our method to learn a consistent metric for the data, but we have not completed detailed experiments to test this hypothesis. In Section 4 we provide some examples to illustrate the numerical behavior of solutions to the metric nearness problem. We show that it can actually be used to learn a complete metric, and we discuss an application in biological databases that originally motivated us to consider the problem. We conclude with a discussion of future work in Section 5.

1.1 Related Work

To the best of our knowledge, the metric nearness problem has not been studied earlier. There are, however, several provinces that border on this territory.

The most relevant work is probably a recent paper by Roth, Laub, Buhmann and Müller [RLBM03]. They remark on the importance of having metric data for machine learning applications, and propose a technique for metrizing dissimilarity data. Their method, constant-shift embedding, increases all the dissimilarities by an equal amount to produce a set of Euclidean distances[1]. The size of the translation depends on the data, so the relative and absolute changes to the dissimilarity values can be huge. Our approach to metrizing data is completely different. We seek a consistent set of distances that deviates as little as possible from the original measurements. Using our approach the resulting set of distances can arise from an arbitrary metric space. There is no restriction on obtaining Euclidean distances; we emphasize this point in Example 4.3. In consequence, we expect metric nearness to provide superior denoising. Moreover, our techniques can also learn distances that are missing entirely.

There is at least one other method for learning a metric. A 2003 article of Xing, Ng, Jordan and Russell [XNJR03] proposes a technique for learning a Mahalanobis (i.e. inner-product) distance for data in $\mathbb{R}^s$. That is, a metric $d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(\boldsymbol{x}-\boldsymbol{y})^T \boldsymbol{G}(\boldsymbol{x}-\boldsymbol{y})}$, where $\boldsymbol{G}$ is an $s \times s$ positive semi-definite matrix. The user specifies that various pairs of points are similar or dissimilar. Then the matrix $\boldsymbol{G}$ is computed by minimizing the total *squared* distances between similar points while forcing the total distances between dissimilar points to exceed one. The article provides explicit algorithms for the case where $\boldsymbol{G}$ is diagonal and when $\boldsymbol{G}$ is positive semi-definite. In comparison, the metric nearness problem is not restricted to Mahalanobis distances; it can learn a general discrete metric. It also allows us to use specific distance measurements and to indicate our confidence in those measurements, rather than forcing a binary choice of "similar" or "dissimilar."

Our work is also distinct from metric Multi-Dimensional Scaling (MDS). The fundamental problem of MDS is to find an ensemble of points in a *fixed* metric space, such as a Euclidean space, whose distances most closely approximate a collection of measured distances [KW78, KVMB79].

[1] That is, a set of numbers that can be realized as the pairwise distances among an ensemble of points in a Euclidean space.

A simpler variant of the MDS problem is to approximate the measured distances with distances that could arise from points in the *specified* metric space *without* computing an embedding [MM94]. Like the simple variant, the metric nearness problem does not produce an embedding. In contrast, our approach does not frame any hypotheses about the space underlying the data, except that it is a metric space. Some illuminating distinctions between our problem and MDS are tendered by Example 4.3.

2 Problem Formulation

We define a *dissimilarity matrix* to be a symmetric[2] matrix with non-negative entries and a zero diagonal. We shall denote the set of $n \times n$ dissimilarity matrices by $\mathscr{D}_n$. We define a *distance matrix* to be a dissimilarity matrix whose entries satisfy the triangle inequality. If $\boldsymbol{M}$ is a distance matrix, then $m_{ik} \leq m_{ij} + m_{jk}$ for every triple of distinct indices (i, j, k). These matrices arise from measuring the distances between pairs of points in a pseudo-metric space[3]. We shall write $\mathscr{M}_n$ for the set of $n \times n$ distance matrices. Observe that the triangle inequalities are *linear* constraints, so $\mathscr{M}_n$ is a closed, convex cone.

The nearest metric problem requests a distance matrix $\boldsymbol{M}$ that is closest to a given dissimilarity matrix $\boldsymbol{D}$ with respect to some norm. Specifically,

$$\boldsymbol{M} \in \operatorname*{argmin}_{\boldsymbol{X} \in \mathscr{M}_n} \|\boldsymbol{W} \cdot (\boldsymbol{X} - \boldsymbol{D})\|, \tag{1}$$

where $\boldsymbol{W} \in \mathscr{D}_n$ is a weight matrix and '$\cdot$' denotes the element-wise (Hadamard) product of two matrices.

Proposition 2.1. *The functional in* (1) *always attains its minimum on* $\mathscr{M}_n$*. Moreover, every local minimum is a global minimum. If, in addition, the norm is strictly convex and the weight matrix has no zeros or infinities off its diagonal, then there is a unique global minimum.*

Proof. All references in this proof are to Rockafellar's opus [Roc70].

Norms are convex (p. 131), and $\boldsymbol{W} \cdot (\boldsymbol{X} - \boldsymbol{D})$ is an affine function of $\boldsymbol{X}$. Therefore, the objective function in (1) is convex by Theorem 5.7. The set $\mathscr{M}_n$ is generated by a finite system of linear inequalities, so it is a polyhedral convex set (p. 170). Every local minimum of a convex function on a convex domain is also a global minimum over that domain. Moreover, the set of minima is convex (p. 264).

Suppose that the norm is strictly convex (p. 253). If all the off-diagonal entries of $\boldsymbol{W}$ are non-zero and finite, then elementwise multiplication by $\boldsymbol{W}$ is a bijection on the set of symmetric matrices with zero diagonals. Let $\boldsymbol{X}, \boldsymbol{Y} \in \mathscr{M}_n$ be distinct minimizers of the objective function in (1). We show that $\frac{1}{2}(\boldsymbol{X} + \boldsymbol{Y}) \in \mathscr{M}_n$ yields a strictly smaller value of the objective function. Since elementwise multiplication by $\boldsymbol{W}$ is injective, $\boldsymbol{W} \cdot (\boldsymbol{X} - \boldsymbol{D})$ and $\boldsymbol{W} \cdot (\boldsymbol{Y} - \boldsymbol{D})$ are distinct points. By the strict convexity of the norm,

$$\|\boldsymbol{W} \cdot (\tfrac{1}{2}(\boldsymbol{X} + \boldsymbol{Y}) - \boldsymbol{D})\| < \tfrac{1}{2}\|\boldsymbol{W} \cdot (\boldsymbol{X} - \boldsymbol{D})\| + \tfrac{1}{2}\|\boldsymbol{W} \cdot (\boldsymbol{Y} - \boldsymbol{D})\|.$$

This inequality contradicts the hypothesis that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are distinct minimizers of the objective function. Therefore, at most one minimizer exists.

It remains to show that the objective function attains a minimum on the feasible set. For convenience, we pass to the function $f(\boldsymbol{Z}) = \|\boldsymbol{W} \cdot \boldsymbol{Z}\|$. Notice that if f attains a minimum on

[2]There is an obvious extension of this work to non-symmetric dissimilarities and to quasi-metrics, which are non-symmetric, but we find the present approach more natural.

[3]A pseudo-metric is non-negative, symmetric and satisfies the triangle inequality. As usual, $d(x, x) = 0$, but it is also possible that $d(x, y) = 0$ even when $x \neq y$.

$\mathscr{M}_n - \boldsymbol{D}$, then the objective function in (1) will attain a minimum on $\mathscr{M}_n$. Norms and linear functions are both continuous, so f is a closed convex function (p. 52). Moreover, f is homogeneous of degree one, so we can compute its recession function as (p. 66)

$$(f0^+)(\boldsymbol{Y}) = \lim_{h\to 0}(f(h\,\boldsymbol{Y}) - f(\boldsymbol{0}))/h = \lim_{h\to 0} f(h\,\boldsymbol{Y})/h = f(\boldsymbol{Y}).$$

But f is non-negative, so its only directions of recession are directions in which f is constant (p. 69). Since $\mathscr{M}_n - \boldsymbol{D}$ is a closed, *polyhedral* cone, we may apply Theorem 27.3 to conclude that f attains a minimum on this cone. □

The weight matrix plays an important role in the problem statement. Its values should reflect our relative confidence in the entries of the input matrix $\boldsymbol{D}$. For example, if we are equally sure (or unsure) of all the input dissimilarities, then the off-diagonal entries of $\boldsymbol{W}$ should be constant. When each d_{jk} represents a measurement with variance σ_{jk}^2, then we might choose $w_{jk} = 1/\sigma_{jk}^2$. In the extreme case that we have no information about the dissimilarity between j and k, we can learn it by setting $w_{jk} = 0$. A caveat is that when some weights are zero, it is possible for the corresponding entries of a solution matrix to be arbitrarily large. Nevertheless, every solution $\boldsymbol{M}$ of (1) is a distance matrix that globally minimizes the approximation error. If blowup is a concern, it might be preferable to set $w_{jk} = \varepsilon$, where ε is very small. On the other hand, when an entry is sacrosanct, set $w_{jk} = 1/\varepsilon$. The weight matrix can also be used to minimize the relative approximation error (instead of the absolute error) by setting $w_{jk} = 1/d_{jk}$.

In the rest of the article, we shall only consider approximation with respect to the ℓ_p *vector* norms, which we denote $\|\cdot\|_p$. The associated minimizations are

$$\min_{\boldsymbol{X}\in\mathscr{M}_n}\left[\sum_{j\neq k}\left|w_{jk}\,(x_{jk} - d_{jk})\right|^p\right]^{1/p} \qquad \text{for } 1 \le p < \infty, \text{ and} \tag{2}$$

$$\min_{\boldsymbol{X}\in\mathscr{M}_n}\max_{j\neq k}\left|w_{jk}\,(x_{jk} - d_{jk})\right| \qquad \text{for } p = \infty. \tag{3}$$

Note that the ℓ_p norms are strictly convex for $1 < p < \infty$, and therefore the solution to (2) is unique. There is a basic intuition for choosing p. The ℓ_1 norm equals the absolute sum of all the changes, and the ℓ_∞ only reflects the maximum absolute change, while the other ℓ_p norms interpolate between these extremes. Therefore, a small value of p typically results in a solution that makes a few large changes to the original data, while a large value of p typically yields a solution with many small changes.

Now, we show how to use standard mathematical programming software to find a global minimizer for any of the problems, $1 \le p \le \infty$. The cases $p = 1$, 2, ∞ receive special attention, since they only require linear and quadratic programs.

2.1 Problem Simplifications

Since all the matrices are symmetric and have zero diagonals, we consider only the strict upper triangle in our formulation. This decision reduces the problem size significantly at the cost of some additional notation. Let $N = n(n-1)/2$, the number of entries in the (strict) upper triangle of $\boldsymbol{D} \in \mathscr{D}_n$. Define $\boldsymbol{d} = \operatorname{trivec}(\boldsymbol{D}) \in \mathbb{R}^N$ as the vector formed by stacking the columns of the upper triangle of $\boldsymbol{D}$, beginning at the left. For non-negative vectors $\boldsymbol{x} \in \mathbb{R}^N$, we may also define the inverse operation, $\operatorname{trimtx}(\boldsymbol{x})$, that returns the matrix in $\mathscr{D}_n$ whose upper triangle is formed by unstacking the vector and whose lower triangle reflects the upper triangle.

To reduce computation further, we shall always minimize the functional with respect to the auxiliary variable $\boldsymbol{y} = \operatorname{trivec}(\boldsymbol{X} - \boldsymbol{D})$. If $\boldsymbol{y}^*$ is a minimizer of the modified problem, we may compute a solution to the original problem using $\boldsymbol{M} = \operatorname{trimtx}(\boldsymbol{y}^*) + \boldsymbol{D}$.

The problem formulations also require two matrices. The matrix $\boldsymbol{A}$ encodes the $3\binom{n}{3}$ constraints given by the triangle inequalities so that $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{0}$ if and only if $\operatorname{trimtx}(\boldsymbol{x})$ is a distance matrix. As a consequence of the triangle inequalities, the constraints $\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{0}$, imply $\boldsymbol{x} \geq \boldsymbol{0}$. Section 2.5 lists some properties of $\boldsymbol{A}$. We also convert the weight matrix into an $N \times N$ diagonal matrix with the operation $\boldsymbol{H} = \operatorname{diag}(\operatorname{trivec}(\boldsymbol{W}))$.[4]

2.2 Solution for $p = 2$

First, we consider minimization of (2) for $p = 2$, the simplest case. By squaring the objective function and rewriting it in terms of the variables described in the last subsection, we see that the minimization is equivalent to the semi-definite quadratic program

$$\min \boldsymbol{y}^T \boldsymbol{H}^2 \boldsymbol{y} \qquad \text{subject to} \qquad \boldsymbol{A}\boldsymbol{y} \leq -\boldsymbol{A}\boldsymbol{d}. \tag{4}$$

If $\boldsymbol{y}^*$ is a minimizer of (4), then $\boldsymbol{M} = \operatorname{trimtx}(\boldsymbol{y}^*) + \boldsymbol{D}$ solves the original problem (2) for $p = 2$.

2.3 Solution for $p = 1$

Next, we consider the minimization (2) for $p = 1$. The absolute value makes the problem appear nonlinear. Nevertheless, it can be solved with a linear program by introducing an auxiliary variable $\boldsymbol{z} \in \mathbb{R}^N$ that can be interpreted as the absolute value of $\boldsymbol{H}\boldsymbol{y}$.

$$\min \boldsymbol{1}^T \boldsymbol{z} \qquad \text{subject to} \qquad \begin{array}{rcr} \boldsymbol{A}\boldsymbol{y} & \leq & -\boldsymbol{A}\boldsymbol{d}, \\ \boldsymbol{H}\boldsymbol{y} - \boldsymbol{z} & \leq & \boldsymbol{0}, \\ \boldsymbol{H}\boldsymbol{y} + \boldsymbol{z} & \geq & \boldsymbol{0}. \end{array} \tag{5}$$

2.3.1 Solution for $1 < p < \infty$

For an arbitrary ℓ_p norm, where $p < \infty$, the formulation is identical with the ℓ_1 problem, except that the objective function becomes $\sum_k z_k^p$. This is a convex programming problem with a smooth objective function and linear constraints.

2.4 Solution for $p = \infty$

Finally, we consider (3). This problem can also be solved with a linear program by introducing an auxiliary variable r that can be interpreted as $\|\boldsymbol{H}\boldsymbol{y}\|_\infty$.

$$\min\ r \qquad \text{subject to} \qquad \begin{array}{rcr} \boldsymbol{A}\boldsymbol{y} & \leq & -\boldsymbol{A}\boldsymbol{d}, \\ \boldsymbol{H}\boldsymbol{y} - r\,\boldsymbol{1} & \leq & \boldsymbol{0}, \\ \boldsymbol{H}\boldsymbol{y} + r\,\boldsymbol{1} & \geq & \boldsymbol{0}. \end{array} \tag{6}$$

2.5 The Constraint Matrix

In common to all the formulations is the matrix $\boldsymbol{A}$ that encodes the linear constraints arising from the triangle inequalities. In this section, we observe some interesting characteristics of $\boldsymbol{A}$. For example, the constraint matrix $\boldsymbol{A}_n = \boldsymbol{A}$ depends only on n, the number of points, and it is independent of a specific dissimilarity matrix $\boldsymbol{D}$ (clearly $n > 2$). Some structural properties of $\boldsymbol{A}$ that can be easily verified are listed below.

[4] The $\operatorname{diag}(\boldsymbol{x})$ operation forms a diagonal matrix whose diagonal entries are given by the components of the vector $\boldsymbol{x}$ taken in order.

1. Matrix $\boldsymbol{A}$ has $3\binom{n}{3}$ rows and $\binom{n}{2}$ columns (rows correspond to constraints while columns correspond to the variables).
2. Each row of $\boldsymbol{A}$ has just three non-zeros: $+1, -1, -1$. Thus $\boldsymbol{A}$ is highly sparse.
3. Each column of $\boldsymbol{A}$ has just $3(n-2)$ non-zero entries.
4. Matrix $\boldsymbol{A}$ has full column rank.
5. Experiments indicate that $\boldsymbol{A}$ has three distinct singular values, viz. $\sqrt{3n-4}$, $\sqrt{2n-2}$ and $\sqrt{n-2}$ with respective multiplicities $n(n-3)/2$, $(n-1)$ and 1.
6. A dissimilarity matrix $\boldsymbol{D}$ is a distance matrix if and only if $\boldsymbol{Ad} \leq 0$, where $\boldsymbol{d} = \text{trivec}(\boldsymbol{D})$. This fact can be easily proved using property 2 listed above.

Each row of $\boldsymbol{A}$ represents a triangle inequality on the entries of the strict upper triangle of a dissimilarity matrix. Clearly the strict upper triangle contains $\binom{n}{2}$ entries. There are three triangle inequalities on each collection of three distinct indices (i, j, k), so $\boldsymbol{A}$ has a total of $3\binom{n}{3}$ rows. This verifies Claim 1. Claim 2 follows from the structure of each triangle inequality, $d_{ik} - d_{ij} - d_{jk} \leq 0$.

We now prove Claims 3, 4 and part of Claim 5. To prove Claim 3, observe that each dissimilarity d_{ik} participates in three inequalities with each of $(n-2)$ other dissimilarities, d_{ij} and d_{jk} for $j \neq i, k$. We can prove Claim 4 and part of Claim 5 by considering the matrix $\boldsymbol{G} = \boldsymbol{A}^T\boldsymbol{A}$. It is most convenient to index the entries of $\boldsymbol{G}$ as $g_{\alpha\beta}$, where $\alpha = (i, j)$ and $\beta = (k, \ell)$. The elements of the ordered pairs represent the entries of the strict upper triangle of a dissimilarity matrix. Therefore, $1 \leq i < j \leq n$, and $1 \leq k < \ell \leq n$.

First, we consider the diagonal entries. On account of Claims 2 and 3, it is clear that $g_{\alpha\alpha} = 3(n-2)$. Now, an off-diagonal entry $g_{\alpha\beta} = 0$ unless $k = i, j$ or $\ell = i, j$. There are a total of $2(n-2)$ such β for a fixed α. For each instance, the corresponding columns of $\boldsymbol{A}$ have entries in three common locations. Two of them have opposite signs, while the third has two minus signs. So $g_{\alpha\beta} = -1$. We conclude that each row of $\boldsymbol{G}$ has exactly $2(n-2)$ non-zero entries off the diagonal, each of which equals -1.

Now, we may apply Geršgorin's Theorem [HJ85] to $\boldsymbol{G}$ to discover that its eigenvalues must lie between $(n-2)$ and $5(n-2)$. In particular, $\boldsymbol{G}$ is non-singular, which implies that $\boldsymbol{A}$ has full column rank. It also follows that the smallest non-zero singular value of $\boldsymbol{A}$ must be atleast $\sqrt{n-2}$. To see that $\sqrt{n-2}$ is always a singular value of $\boldsymbol{A}$, observe that $\boldsymbol{A}^T\boldsymbol{A}\mathbf{1} = \boldsymbol{G}\mathbf{1} = (3(n-2) - 2(n-2))\mathbf{1} = (n-2)\mathbf{1}$.[5]

2.6 Some Generalizations and Specializations

Our framework for solving the Metric Nearness problem, lends itself to the incorporation of arbitrary linear constraints, in addition to those imposed by the triangle inequality. Some useful constraints on the solution vector could be:

Box-constraints. These constraints take the form $\boldsymbol{l} \leq \boldsymbol{y} + \boldsymbol{d} \leq \boldsymbol{u}$, where $\boldsymbol{l}$ and $\boldsymbol{u}$ describe lower and upper bounds on the solution vector. As a special case, observe that a constraint of the form $\boldsymbol{y} + \boldsymbol{d} \geq \boldsymbol{e}$ with $\boldsymbol{e} > \mathbf{0}$, can ensure a metric instead of a pseudo-metric solution. Recall that the triangle inequality constraints imply $\boldsymbol{y} + \boldsymbol{d} \geq 0$.

Rank constraints. Suppose we order the elements of $\boldsymbol{D}$ so that $d_{r_1s_1} \leq \cdots \leq d_{r_Ns_N}$, where (r_i, s_i) denote all pairs such that $r_i \neq s_i$. We could describe constraints to enforce this ordering so that $m_{r_1s_1} \leq \cdots \leq m_{r_Ns_N}$. Such rank constraints are usually employed in the non-metric MDS problem (see for e.g. [KW78, KVMB79]).

[5] Alternatively we can observe that $\boldsymbol{A}\mathbf{1} = -\mathbf{1} \Rightarrow \boldsymbol{A}^T\boldsymbol{A}\mathbf{1} = -\boldsymbol{A}^T\mathbf{1} = (n-2)\mathbf{1}$.

An interesting generalization is afforded by relaxing the triangle inequality to a "λ-approximate" triangle inequality, where $\lambda \geq 1$ [FS98, MP03]. In this case, for any triple (i, j, k), and scalar $\lambda \geq 1$, we have $m_{ij} \leq \lambda(m_{ik} + m_{kj})$. Consequently the constraint matrix $\boldsymbol{A}$ is replaced by $\boldsymbol{\Lambda} \cdot \boldsymbol{A}$, where $\boldsymbol{\Lambda} = \lambda \mathbf{1}_{M \times N}$, $M = 3\binom{n}{3}$, $N = \binom{n}{2}$ and $\lambda \geq 1$.[6]

3 An Application to Clustering

The metric nearness problem can be used to develop efficient algorithms for clustering that provide guarantees on the quality of the output in comparison with the optimal clustering. The Max-Cut problem offers an especially attractive example. A *cut* of a graph is a partition of the vertices into two disjoint sets, and the value of a cut is the total weight of all edges that cross the partition. Max-Cut simply asks for the cut of a graph with maximum value. If the size of each edge weight is proportional to the dissimilarity between the two vertices, solving Max-Cut can be interpreted as finding the best clustering of the vertices into two sets.

For a general set of weights, Max-Cut is hard enough [PY91] that the solution cannot be well-approximated in polynomial time (unless $\mathsf{P} = \mathsf{NP}$) [ALM+98]. On the other hand, for weights that do satisfy the triangle inequality, de la Vega and Kenyon have exhibited a randomized algorithm that can approximate the solution arbitrarily well in polynomial time [dlVK01]. That is, for a given $\varepsilon > 0$, their method can (with high probability) compute in polynomial time, a cut whose value is no smaller than $(1 - \varepsilon)$ times the value of the optimal cut. Of course, the time complexity grows quickly as ε shrinks.

Metric nearness plays a surprising role here. First, we approximate the original graph by a metric graph. Then, we use the fast algorithm to produce a nearly optimal cut of the metric graph. The same cut of the original graph also has a nearly optimal value, which can be bounded in terms of the approximation error from the metric nearness problem.

Theorem 3.1. *Suppose that $\boldsymbol{D}$ is a dissimilarity matrix and that $\boldsymbol{M}$ is a distance matrix. If $\mathscr{S}$ is a cut of $\boldsymbol{M}$ whose value exceeds $(1 - \varepsilon)\,\mathrm{maxcut}(\boldsymbol{M})$, then we have the bounds*

$$\mathrm{cut}_{\mathscr{S}}(\boldsymbol{D}) \geq (1 - \varepsilon)\,\mathrm{maxcut}(\boldsymbol{D}) - (1 - \varepsilon/2)\,\|\boldsymbol{M} - \boldsymbol{D}\|_1 \qquad \textit{and} \tag{7}$$

$$\mathrm{cut}_{\mathscr{S}}(\boldsymbol{D}) \geq \frac{1 - \varepsilon}{\|\boldsymbol{M}/\boldsymbol{D}\|_\infty \, \|\boldsymbol{D}/\boldsymbol{M}\|_\infty}\,\mathrm{maxcut}(\boldsymbol{D}), \tag{8}$$

where '/' represents element-wise division and $\|\cdot\|_\infty$ denotes the ℓ_∞ norm that ignores the matrix diagonal. If $m_{jk} = d_{jk} = 0$, then the infinity norm also ignores the (j, k) entry of its argument.

To find the optimal $\boldsymbol{M}$ for bound (7), we simply solve the metric nearness problem with $p = 1$ and $w_{jk} \equiv 1$. The optimal $\boldsymbol{M}$ for (8) cannot be obtained without solving a non-convex optimization problem.

Proof. For a set of vertices $\mathscr{S}$, the value of the corresponding cut is computed by the linear function

$$\mathrm{cut}_{\mathscr{S}}(\boldsymbol{D}) = \sum_{j \in \mathscr{S}} \sum_{k \notin \mathscr{S}} d_{jk}.$$

The maximum cut just optimizes this functional over all subsets $\mathscr{S}$ of the vertex set $\{1, 2, \ldots, n\}$:

$$\mathrm{maxcut}(\boldsymbol{D}) = \max_{\mathscr{S}} \sum_{j \in \mathscr{S}} \sum_{k \notin \mathscr{S}} d_{jk}.$$

[6] Note that we can further relax the triangle inequalities by allowing concession for differing λ's such as those given by $m_{ij} \leq \lambda_{ik} m_{ik} + \lambda_{kj} m_{kj}$ with $\lambda_{ik}, \lambda_{kj} \geq 1$.

Obviously, $\mathrm{cut}_{\mathscr{S}}(\boldsymbol{D}) \leq \mathrm{maxcut}(\boldsymbol{D})$. It can be shown that $\mathrm{maxcut}(|\cdot|)$ is a matrix norm. In particular, it satisfies the triangle inequality for norms. It is also clear that

$$\mathrm{maxcut}(|\boldsymbol{T}|) \leq \tfrac{1}{2} \sum_{j \neq k} |t_{jk}| = \tfrac{1}{2} \|\boldsymbol{T}\|_1$$

for any symmetric matrix $\boldsymbol{T}$ with a zero diagonal.

Let us begin with bound (7). Suppose that $\mathscr{S}$ is a $(1-\varepsilon)$-optimal cut of $\boldsymbol{M}$. Then

$$\begin{aligned}
\mathrm{cut}_{\mathscr{S}}(\boldsymbol{D}) &= \mathrm{cut}_{\mathscr{S}}(\boldsymbol{M}) + \mathrm{cut}_{\mathscr{S}}(\boldsymbol{D} - \boldsymbol{M}) \\
&\geq (1-\varepsilon)\ \mathrm{maxcut}(\boldsymbol{M}) - \mathrm{cut}_{\mathscr{S}}(|\boldsymbol{D} - \boldsymbol{M}|) \\
&\geq (1-\varepsilon)\ \mathrm{maxcut}(\boldsymbol{D} + (\boldsymbol{M} - \boldsymbol{D})) - \tfrac{1}{2} \|\boldsymbol{D} - \boldsymbol{M}\|_1 \\
&\geq (1-\varepsilon)\,(\mathrm{maxcut}(\boldsymbol{D}) - \mathrm{maxcut}(|\boldsymbol{M} - \boldsymbol{D}|)) - \tfrac{1}{2} \|\boldsymbol{M} - \boldsymbol{D}\|_1 \\
&\geq (1-\varepsilon)\ \mathrm{maxcut}(\boldsymbol{D}) - (1-\varepsilon/2)\, \|\boldsymbol{M} - \boldsymbol{D}\|_1.
\end{aligned}$$

The proof for the bound (8) follows a similar outline. First, we implicitly define a relative error matrix $\boldsymbol{E}$ with the relation $\boldsymbol{M} = \boldsymbol{D} \cdot \boldsymbol{E}$. We assume that $m_{jk} = 0$ if and only if $d_{jk} = 0$ to ensure $\boldsymbol{E}$ can be defined. If not, the resulting error bound would be trivial anyway. Let $r = \min\{e_{jk} : d_{jk} \neq 0\}$ and $R = \max\{e_{jk} : d_{jk} \neq 0\}$. For any zero entry of $\boldsymbol{D}$, take the corresponding entry of $\boldsymbol{E}$ in the range $[r, R]$. In the sequel, we use '/' for elementwise division.

Next, observe that

$$\begin{aligned}
\mathrm{cut}_{\mathscr{S}}(\boldsymbol{M}) = \mathrm{cut}_{\mathscr{S}}(\boldsymbol{D} \cdot \boldsymbol{E}) &= \sum_{j \in \mathscr{S}} \sum_{k \notin \mathscr{S}} d_{jk}\, e_{jk} \\
&\leq \max_{j \neq k} e_{jk} \sum_{j \in \mathscr{S}} \sum_{k \notin \mathscr{S}} d_{jk} \\
&\leq \|\boldsymbol{E}\|_\infty\ \mathrm{cut}_{\mathscr{S}}(\boldsymbol{D}).
\end{aligned}$$

Similarly,

$$\mathrm{maxcut}(\boldsymbol{D}) = \mathrm{maxcut}(\boldsymbol{M}/\boldsymbol{E}) \leq \|\boldsymbol{1}/\boldsymbol{E}\|_\infty\ \mathrm{maxcut}(\boldsymbol{M}).$$

Then, we compute

$$\begin{aligned}
\mathrm{cut}_{\mathscr{S}}(\boldsymbol{D}) &\geq \frac{\mathrm{cut}_{\mathscr{S}}(\boldsymbol{M})}{\|\boldsymbol{E}\|_\infty} \\
&\geq \frac{1-\varepsilon}{\|\boldsymbol{E}\|_\infty}\ \mathrm{maxcut}(\boldsymbol{M}) \\
&\geq \frac{1-\varepsilon}{\|\boldsymbol{E}\|_\infty\, \|\boldsymbol{1}/\boldsymbol{E}\|_\infty}\ \mathrm{maxcut}(\boldsymbol{D}).
\end{aligned}$$

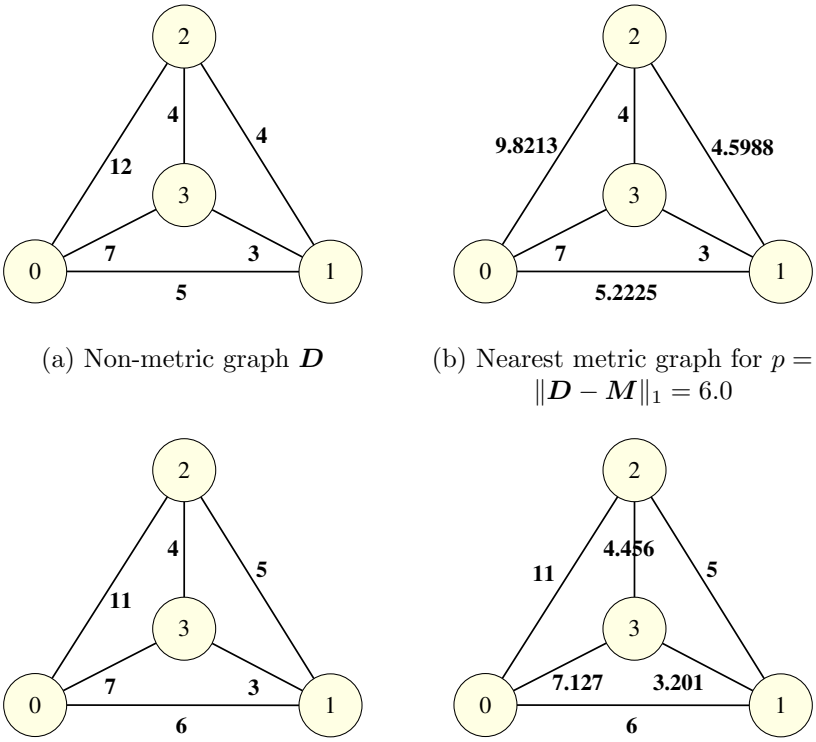
□

This technique can be extended to other types of problems that are computationally easier for metric graphs [Ind99b]. Mettu and Plaxton have also considered fast algorithms for clustering "nearly metric" data, but their approach relies instead on weak versions of the triangle inequality [MP03]. Fast approximation algorithms for various other metric problems such as k-median, MAX-TSP, etc., are discussed in [Ind99a]; our method allows extending these approximation algorithms to non-metric data.

4 Experimental Results

We now present some experiments to exhibit how the solutions of the metric nearness problems behave in practice. First, wee show results for approximating a non-metric graph by a metric graph. Then we highlight distinctions between Metric MDS and Metric Nearness by means of a simple example. Last, we display approximation errors for a set of dissimilarity matrices from biology.

Example 4.1. Figure 1 shows a non-metric graph (whose dissimilarity on adjacency matrix is not a distance matrix) and the corresponding nearest metric graphs as obtained by the methods outlined in Sections 2.2, 2.3 and 2.4. As can be clearly seen the (0–2–3) and (0–1–2) triangles violate the



(a) Non-metric graph $\boldsymbol{D}$

(b) Nearest metric graph for $p = 1$
$\|\boldsymbol{D} - \boldsymbol{M}\|_1 = 6.0$

(c) Nearest metric graph for $p = 2$
$\|\boldsymbol{D} - \boldsymbol{M}\|_F = \sqrt{6}$

(d) Nearest metric graph for $p = \infty$
$\|\boldsymbol{D} - \boldsymbol{M}\|_\infty = 1$

Figure 1: Non-metric graph and its nearest metric graph

triangle inequality in Figure 1(a). Each of the graphs in (b), (c) and (d) is a metric graph that rectifies these violations. Notice that each of the schemes ($p = 1, 2, \infty$) has changed the edge weights (vertex dissimilarities) in a characteristic manner. The linear programming solution ($p = 1$) alters one of the weights by a large amount, whereas the quadratic programming solution has made several changes of moderate magnitude. The solution for $p = \infty$ is quite different: all edge weights have been adjusted, albeit by small amounts. Thus, an application that cannot tolerate large changes should use the ℓ_2 or ℓ_∞ solutions. Even though these graphs are small, they reflect the essential characteristics of the different minimization problems. The weight matrix used gave an equal weight (of unity) to each of the dissimilarity values of graph $\boldsymbol{D}$.

To illustrate how the algorithm can learn missing values consider the same graph as in Figure 1, but imagine that the dissimilarity values between vertices 2–3 and 1–3 are missing. We set the corresponding weight that reflects our confidence (or rather lack of it) in the dissimilarity value, to be zero and initialize the missing dissimilarities to any reasonable value. For example, if we initialize the missing dissimilarity on edge 2–3 to be 0 and that on edge 1–3 to be 3, the approximated graph learned by quadratic programming is the same as the one in Figure 1 (c) except the edge 1–3 gets dissimilarity value 1. We noticed during our experiments that the final value of the learned distance depends on this initialization. Note that when the weight matrix $\boldsymbol{W}$ contains zeros off its diagonal, the minimizer is not unique in general; however, every local minimum is a global minimum (see Proposition 2.1).

4.1 Comparison between Metric MDS and Metric Nearness

In this section we seek to dispel any confusion that might exist in the reader's mind pertaining to the distinction between Metric MDS and Metric Nearness. Assume we are given a distance matrix $\boldsymbol{M}$. The fundamental problem that Metric MDS addresses is that of finding an ensemble of points in a *fixed* metric space (assumed to be a Euclidean space for this example), whose interpoint distance matrix is as close to $\boldsymbol{D}$ as possible. In contrast Metric Nearness seeks to find a nearest distance matrix corresponding to an arbitrary metric space for a given $\boldsymbol{D}$.

A Euclidean distance matrix (EDM) is a distance matrix that represents the interpoint Euclidean distances amongst an ensemble of points in $\mathbb{R}^s$. More often than not, a dissimilarity matrix $\boldsymbol{D} \in \mathscr{D}_n$ is not Euclidean[7]. The following theorem characterizes when a dissimilarity matrix is an EDM. Let $\boldsymbol{E} = -\frac{1}{2}(\boldsymbol{D} \cdot \boldsymbol{D})$, and $\boldsymbol{B} = \boldsymbol{HEH}$, where $\boldsymbol{H} = \boldsymbol{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$.

Theorem 4.2 (Schoenberg [Sch35], see for e.g., [KVMB79, pg. 397]). *Let $\boldsymbol{D}$ be a dissimilarity matrix and define $\boldsymbol{B}$ as above then $\boldsymbol{D}$ is a Euclidean Distance Matrix if and only if $\boldsymbol{B}$ is positive semi-definite.*

Example 4.3 (Metric Nearness differs from Metric MDS). This example is taken from Exercise 14.2.7 of [KVMB79]. Suppose that 1, 2, ..., 7 are regions in a country as depicted in Figure 2. Let the distance matrix be constructed by counting the minimum number of boundaries crossed to pass from region i to region j.

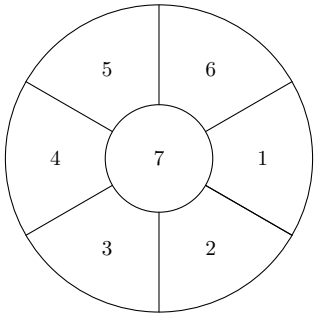


Figure 2: Seven regions in a country.

[7] A nonsurprising empirical observation is that a randomly chosen matrix $\boldsymbol{M} \in \mathscr{M}_n$ is usually not Euclidean. For example, the metric graph in Figure 1(c) is not Euclidean.

The distance matrix for this problem is given by

$$\boldsymbol{M}_1 = \begin{bmatrix} 0 & 1 & 2 & 2 & 2 & 1 & 1 \\ & 0 & 1 & 2 & 2 & 2 & 1 \\ & & 0 & 1 & 2 & 2 & 1 \\ & & & 0 & 1 & 2 & 1 \\ & & & & 0 & 1 & 1 \\ & & & & & 0 & 1 \\ & & & & & & 0 \end{bmatrix}. \tag{9}$$

The matrix $\boldsymbol{M}_1$ is not Euclidean since the matrix $\boldsymbol{B}$ corresponding to it is not positive semi-definite (the eigenvalues of $\boldsymbol{B}$ are $\frac{7}{2}, \frac{7}{2}, \frac{1}{2}, \frac{1}{2}, 0, -\frac{1}{7}, -1$). Let $\boldsymbol{D}$ be the matrix that is the same as $\boldsymbol{M}_1$ except that $d_{23} = d_{32} = 0$. We can view this as a situation where due to some measurement error the $(2,3)$ entry is 0. The dissimilarity matrix $\boldsymbol{D}$ is not a distance matrix (triangle 1–2–3 violates the triangle inequality). Given such a $\boldsymbol{D}$ we can try to obtain distance matrices that are closest in ℓ_1, ℓ_2 and ℓ_∞ norms. Our linear program for ℓ_1 returns a distance matrix $\boldsymbol{M}_2 = \boldsymbol{M}_1$ that is closest to $\boldsymbol{D}$ in ℓ_1 norm. The matrix returned for ℓ_∞ is very different and is not shown. Below is a display of the matrix $\boldsymbol{M}_2$ as obtained by our quadratic program and the matrix $\boldsymbol{M}_3$ that results from a classical Metric MDS solution. We compare our ℓ_2 norm solution to MDS because the classical Metric MDS approximation also yields a solution that is an optimal MDS solution in the ℓ_2 norm.

$$\boldsymbol{M}_2 = \begin{bmatrix} 0 & 1.25 & 1.75 & 2 & 2 & 1 & 1 \\ & 0 & 0.50 & 1.75 & 2 & 2 & 1 \\ & & 0 & 1.25 & 2 & 2 & 1 \\ & & & 0 & 1 & 2 & 1 \\ & & & & 0 & 1 & 1 \\ & & & & & 0 & 1 \\ & & & & & & 0 \end{bmatrix}, \quad \boldsymbol{M}_3 = \begin{bmatrix} 0 & 1.382 & 2.003 & 2.172 & 2.001 & 1.279 & 1.154 \\ & 0 & 1.059 & 2.003 & 2.195 & 2.006 & 1.190 \\ & & 0 & 1.382 & 2.006 & 2.195 & 1.190 \\ & & & 0 & 1.279 & 2.001 & 1.154 \\ & & & & 0 & 1.249 & 1.132 \\ & & & & & 0 & 1.132 \\ & & & & & & 0 \end{bmatrix}.$$

Note that the error $\|\boldsymbol{M}_2 - \boldsymbol{D}\|_F = 1.000$ is smaller than $\|\boldsymbol{M}_3 - \boldsymbol{D}\|_F = 1.9449$. If we compare the error against the original matrix $\boldsymbol{M}_1$, we find that $\|\boldsymbol{M}_2 - \boldsymbol{M}_1\|_F = 1.000$ and $\|\boldsymbol{M}_3 - \boldsymbol{M}_1\|_F = 1.243$.

4.2 A Problem from Biology

We now turn our attention to a problem that arose in connection with biological databases. This problem involves approximating mPAM matrices, which are a derivative of mutation probability (PAM) matrices [DSO78] that arise in protein sequencing applications in biology. The mPAM matrices are 20×20 matrices (since there are 20 amino acids). They represent a certain measure of dissimilarity for an application in protein sequencing. Owing to the manner in which these mPAM matrices are derived, they tend not to be distance matrices. Some applications in biological databases could potentially be accelerated by employment of a metric-based database indexing scheme that can be used if the mPAM matrices were distance matrices [Mir03].

We conducted some experiments in which we approximated various mPAM matrices by their nearest distance matrices. The relative errors of the approximations[8] $\|\boldsymbol{D} - \boldsymbol{M}\|/\|\boldsymbol{D}\|$, are reported in Table 1. We notice that the relative error, as incurred by each of the solutions ($p = 1, 2, \infty$), is comparable. We also observe with increasing mPAM number (from 50 to 300) the relative error decreases. This is a behavior that we propose to investigate in future. One of our future goals is to objectively evaluate the benefit of a metric to the speed of biological database computations.

[8]For more detailed results, see http://www.cs.utexas.edu/users/suvrit/work/metric.

Table 1: Relative errors for mPAM dataset

Dataset	ℓ_1 error	ℓ_2 error	ℓ_∞ error
mPAM50	0.339	0.402	0.278
mPAM100	0.142	0.231	0.206
mPAM150	0.054	0.121	0.151
mPAM250	0.004	0.025	0.042
mPAM300	0.002	0.017	0.056

4.3 Running Time and Implementation Issues

The running time for our implementation was the greatest for ℓ_1 and least for ℓ_2. In fact, the quadratic program was about two orders of magnitude faster than either linear program. Thus, if speed is an issue the recommended norm to minimize would be the ℓ_2 norm. We used the optimization toolbox of MATLAB 6 to solve our linear or quadratic programs. We believe that a hand-crafted implementation of these solvers that takes advantage of the structure and sparsity of the constraint matrix could lead to a much faster implementation.

5 Discussion and Future Work

In this article, we have posed the problem of approximating non-metric data with data that satisfies the triangle inequalities. When the approximation error is measured with an ℓ_p vector norm, standard convex programming software can always compute a global optimum in polynomial time. In particular, we have formulated the ℓ_2 variant as a quadratic program with linear constraints, and we have formulated the ℓ_1 and ℓ_∞ variants as linear programs. For small p, the solution typically involves changing a few of the original data significantly. Meanwhile, solutions for large p typically change all of the original data by a small amount.

A number of avenues for additional research branch out in front of us. It is well-known that the Bellman-Ford algorithm can be used to solve certain types of linear programs [CLR90]. On analogy, we believe that there may be a graph-based method for solving some of the metric nearness problems, especially the ℓ_1 version. This type of algorithm could reduce the cost of solution by several orders of magnitude.

As we have shown, replacing a general graph by a metric graph permits us to use efficient algorithms for approximate clustering problems while retaining bounds on the error incurred. We believe that this idea will extend to other computationally hard graph problems, for example, see [Ind99b]. Consequently metric nearness may have an interesting role to play in many graph algorithms. Clustering algorithms might also benefit from using our method to learn a consistent metric for the data.

A less grand ambition would be to consider metric nearness problem with respect to more general norms or divergences such as Bregman divergences [Bre67]. It might also be valuable to build specialized software for solving the simple optimization problems we have formulated; an off-the-shelf mathematical programming package certainly cannot take advantage of the enormous structure inherent in the metric nearness problem.

Acknowledgements

We would like to thank Professor Dan Miranker and his student Weijia Xu for drawing our attention to the importance of metric-based indexing in biological databases. This research was supported

in part by NSF CAREER Award No. ACI-0093404 and Texas Advanced Research Program grant 003658-0431-2001.

References

[ALM+98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *J. Assoc. Comput. Mach.*, 45:501–555, 1998.

[Bre67] L. M. Bregman. The relaxation method of finding the common point of convex sets and its applications to the solution of problems in convex programming. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967.

[CLR90] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.

[dlVK01] W. F. de la Vega and C. Kenyon. A randomized approximation scheme for Metric MAX-CUT. *J. Comput. Sys. and Sci.*, 63:531–541, 2001.

[DSO78] M. O. Dayhoff, R. M. Schwarz, and B. C. Orcutt. A model of evolutionary change in proteins. *Atlas of Protein Sequence and Structure*, 5(Suppl. 3), 1978.

[FS98] R. Fagin and L. Stockmeyer. Relaxing the triangle inequality in pattern matching. *International Journal of Computer Vision*, 30:219–231, 1998.

[Hig89] N. J. Higham. Matrix nearness problems and applications. In M. J. C. Gower and S. Barnett, editors, *Applications of Matrix Theory*, pages 1–27. Oxford University Press, 1989.

[HJ85] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge Univ. Press, 1985.

[Ind99a] P. Indyk. Sublinear time algorithms for metric space problems. In *31st Symposium on Theory of Computing*, pages 428–434, 1999.

[Ind99b] P. Indyk. A sublinear-time approximation scheme for clustering in metric spaces. In *40th Symposium on Foundations of Computer Science*, 1999.

[KVMB79] J. T. Kent K. V. Mardia and J. M. Bibby. *Multivariate Analysis*. Academic Press, 1979.

[KW78] J. B. Kruskal and M. Wish. *Multidimensional Scaling*. Number 07-011. Sage Publications, 1978. Series: Quantitative Applications in the Social Sciences.

[Mir03] D. P. Miranker. Personal communication, 2003.

[MM94] R. Mathar and R. Meyer. Algorithms in convex anlaysis to fit ℓ_p-distance matrices. *J. Multivariate Anal.*, 51:102–120, 1994.

[MP03] R. R. Mettu and C. G. Plaxton. The online median problem. *SIAM J. Comput.*, 32(3):816–832, 2003.

[PY91] C. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes. *J. Comput. Sys. and Sci.*, 43:425–440, 1991.

[RLBM03] V. Roth, J. Laub, J. M. Buhmann, and K.-R. Müller. Going metric: Denoising pariwise data. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS) 15*, 2003.

[Roc70] R. T. Rockafellar. *Convex Analysis*. Princeton Univ. Press, 1970.

[Sch35] I. J. Schoenberg. Remarks to Maurice Fréchet's article "Sur la definition axiomatique d'une classe d'espace distanciés vectoriellement applicable sur l'espace de Hilbert". *Annals of Mathematics*, 36:724–732, 1935.

[XNJR03] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side constraints. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems (NIPS) 15*, 2003.