Robust Euclidean Embedding

Lawrence Cayton LCAYTON@CS.UCSD.EDU
Sanjoy Dasgupta DASGUPTA@CS.UCSD.EDU
Department of Computer Science and Engineering, University of California, San Diego
9500 Gilman Dr. La Jolla, CA 92093

Abstract

We derive a robust Euclidean embedding procedure based on semidefinite programming that may be used in place of the popular classical multidimensional scaling (cMDS) algorithm. We motivate this algorithm by arguing that cMDS is not particularly robust and has several other deficiencies. General-purpose semidefinite programming solvers are too memory intensive for medium to large sized applications, so we also describe a fast subgradient-based implementation of the robust algorithm. Additionally, since cMDS is often used for dimensionality reduction, we provide an in-depth look at reducing dimensionality with embedding procedures. In particular, we show that it is NP-hard to find optimal low-dimensional embeddings under a variety of cost functions.

1. Introduction

In this paper, we work with the standard Euclidean embedding problem: given a matrix D of interpoint dissimilarities, find a configuration of points whose interpoint Euclidean distances match the dissimilarities closely. This problem and its relatives have been studied extensively in various research communities, including psychology, operations research, and machine learning. The major applications of Euclidean embedding include visualization, dimensionality reduction, and adapting non-Euclidean dissimilarity measures (*e.g.* KL-divergence) to algorithms that require vector data as input. Of course, the usefulness of an embedding depends strongly on the fidelity of the embedded distances to the input dissimilarities.

In the machine learning community, the algorithm of choice for embedding seems to be classical multidimensional scaling (cMDS). Its popularity results from being relatively fast, parameter-free, easy to implement, and optimal for its cost function. In this work, we look carefully at the algorithm and argue that cMDS has some problematic features as well. In particular, we argue that the cost function is not robust and is conceptually awkward.

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006. Copyright 2006 by the author(s)/owner(s).

We propose a robust alternative to cMDS, robust Euclidean embedding (REE), that retains many of the desirable features of cMDS, but avoids some of its pitfalls. We show that the global minimum of the REE cost function can be found using a semidefinite program (SDP). Though this is heartening, standard SDP-solvers can only manage the embedding program for around 100 points. So that REE can be used on more reasonably sized datasets, we present a subgradient-based implementation of the algorithm.

Dimensionality reduction is an important application of cMDS, though, as we will review, cMDS performs dimensionality reduction by projecting the embedded points on their principal components. We show that the problem of reducing the dimensionality while preserving distances as well as possible is NP-hard for a variety of cost functions. We further show that a standard dimensionality reduction heuristic—minimizing the trace—clashes with the cost function of REE, meaning that, unfortunately, dimensionality reduction cannot be performed as a part of the embedding.

1.1. Related Work

The Euclidean embedding problem has been extensively researched, so let us pause to relate this paper to previous work. The theoretical computer science community has produced a large literature on embeddings; much of this literature traces back to (Linial et al., 1995), where the use of semidefinite programs for embedding was perhaps first discussed. Motivated largely by geometric, combinatorial, and algorithmic concerns, this body of work focuses on bounding the distortion of embeddings between various spaces (often finite metrics and normed spaces). In contrast, this work is tailored to machine learning and is therefore focused on issues of data analysis such as cost function selection, robustness, and efficiency. The operations research and optimization communities have also investigated the Euclidean embedding problem; that line of work is described in detail in (Dattorro, 2005). The relationship of convexity and semidefinite programming to Euclidean embedding has been explored in this body of work, but our perspective on dimensionality reduction, our critique of cMDS, and our subgradient algorithm have not been previously developed. Finally, the psychometrics community has been working with multidimensional scaling for over half a century, but has mostly ignored issues of convexity and algorithmic optimality; see (Borg & Groenen, 2005) for a recent overview.

1.2. Organization

The paper is organized as follows. We review and critique cMDS in section 2. In section 3, we analyze several cost functions for embedding with an emphasis on robustness. In section 4 we describe the robust embedding procedure. Section 5 contains a discussion of dimensionality reduction and NP-hardness and section 6 concludes the paper with several experiments.

1.3. Notation and Conventions

- Unless otherwise specified, we work only with *squared* dissimilarities and distances.
- $\mathbb{D}^n = \{D \in \mathbf{R}^{n\times n} | D^\top = D,\ \operatorname{diag}(D) = \mathbf{0}\}$ is the set of (squared) dissimilarity matrices.
- $\mathbb{EDM}^n$ is the set of $n \times n$ (squared) Euclidean distance matrices.
- For a matrix M, $\|M\|_1 = \sum_{ij} |M_{ij}|$ and $\|M\|_2$ denotes the Frobenius norm.
- The distance matrix associated with a Gram matrix B is denoted $\operatorname{dist}(B)$ and is defined as

$$[\operatorname{dist}(B)]_{ij} = B_{ii} + B_{jj} - B_{ij} - B_{ji}.$$

2. Classical MDS

In this section, we briefly review classical MDS and note a couple of its shortcomings. The algorithm appears in figure 1.

Let us consider cMDS as a 2-stage procedure: during the first stage (steps 1-4), the dissimilarities are embedded into Euclidean space; during the second, the dimensionality is reduced by disregarding the tail-end coordinates of X. Because the coordinate axes of X are aligned with the eigenvectors of $XX^\top$, the dimensionality reduction stage is performing principal components analysis. We focus on the embedding stage of cMDS.

```
cMDS
input: $D \in \mathbb{D}^n$ (dissimilarities), $k \in \{1, \dots, n\}$
  1. Set $B := -\frac{1}{2}HDH$, where $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$ is the centering matrix.
  2. Compute the spectral decomposition of $B$: $B = U\Lambda U^\top$.
  3. Form $\Lambda_+$ by setting $[\Lambda_+]_{ij} := \max\{\Lambda_{ij}, 0\}$.
  4. Set $X := U\Lambda_+^{1/2}$.
  5. Return $[X]_{n\times k}$.
```

Figure 1. Classical multidimensional scaling.

Classical MDS is based on a well-known characterization of Euclidean distance matrices (EDMs):

> D is an Euclidean distance matrix if, and only if, $B = -\frac{1}{2}HDH$ is positive semidefinite (PSD).

In such cases, this matrix B is the Gram (inner-product) matrix for a configuration with interpoint distances D. Given an EDM as input, cMDS converts it to the corresponding Gram matrix B using the above characterization (*i.e.* it sets $B = -\frac{1}{2}HDH$) and then decomposes it to get a configuration of points.

This reconstruction is perfect for EDMs, but what if D is not Euclidean? Then the goal of cMDS is to find an EDM D^* that approximates it well. Finding this D^*—*i.e.* projecting D onto $\mathbb{EDM}^n$—is the core of the embedding problem; once it is found, we may easily recover the corresponding configuration. Classical MDS performs this projection in a round-about way: it takes $B = -\frac{1}{2}HDH$, which will not be positive semidefinite, and projects it onto the cone of positive semidefinite matrices (step 4). In other words, rather than directly projecting D onto $\mathbb{EDM}^n$, cMDS maps D to a matrix of similarities, B, and projects it onto the PSD cone. Figure 2 depicts this process.

The optimization problem being solved by cMDS is $\min_{D^*\in\mathbb{EDM}^n} \|HDH - HD^*H\|_2^2$. We will discuss this cost function in more detail in the next section.

There are a couple of shortcomings of cMDS that make it unappealing for some machine learning tasks. First, it is difficult to handle the case when some of the dissimilarities are unavailable. Second, one cannot explicitly down-weight the error on some dissimilarities.

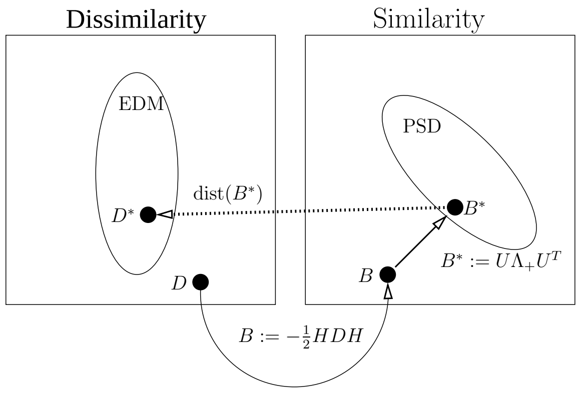

Figure 2. The cMDS projection of D onto the EDM cone.

This functionality is useful in applications where some dissimilarities are unimportant, or are very rough estimates. For example, Isomap is an application of cMDS in which the large distances are often crude estimates, but the small distances are treated as accurate.

3. Cost Functions for Embedding

The cost function of classical MDS is

$$f(D^*) = \|HDH - HD^*H\|_2^2 \tag{1}$$

with domain $\mathbb{EDM}^n$. In this section, we discuss this cost function and several similar ones from a robustness perspective. We also briefly remark on some conceptual problems with the cMDS cost function.

3.1. Robustness

The robustness of a statistic concerns how well it can tolerate a few noisy points (Huber, 1981). More precisely, suppose that the distribution of interest is $\mathcal{P}$ and that there is a noise process modelled by $\mathcal{P}_e$. Then, the robustness of a statistic relates to how much it changes relative to ϵ for a sample drawn from $(1-\epsilon)\mathcal{P} + \epsilon\mathcal{P}_e$. Here, we mean robustness in a similar, though less formal, spirit. We are interested in the effect of a small number of noisy entries in the input on the resulting configuration. That is, suppose that D is a Euclidean distance matrix, but a few of its entries are corrupted by a noise process. How does the embedding compare to the true underlying configuration?

We take a first step towards robustness by simply changing the norm of the cost function of cMDS (1). The ℓ_2-norm is known to be sensitive to outliers; for example, the poor robustness properties of the mean are closely related to the behavior of the ℓ_2-norm. So, we simply substitute the ℓ_1-norm for the ℓ_2-norm in (1), yielding

$$f(D^*) = \|HDH - HD^*H\|_1. \tag{2}$$

Even though the ℓ_1-norm is generally more robust than the ℓ_2, this particular cost function is still quite sensitive to noise. We now show that the HDH transformation is a second aspect of the cMDS cost function that makes the procedure not robust. The terms of HDH are

$$[HDH]_{ij} = D_{ij} - \frac{1}{n}\sum_i D_{ij} - \frac{1}{n}\sum_j D_{ij} + \frac{1}{n^2}\sum_{ij} D_{ij}.$$

Consider what happens if we take an EDM D and form another matrix E which is identical to D except the the dissimilarity between k and l is incremented by δ. Though only two terms differ between D and E (the kl-th term and the lk-th), all n^2 terms between HDH and HEH differ. In particular,

$$[HEH]_{ij} - [HDH]_{ij} = \begin{cases} \delta(1-1/n)^2 & \text{if } (i,j) = (k,l) \\ \delta(1/n^2 - 1/n) & \text{if } i = k \text{ or } j = l \\ \delta(1/n^2) & \text{otherwise.} \end{cases}$$

To summarize, the kl-th and lk-th terms are modified by $\Theta(\delta)$, the terms in the k-th and l-th columns and rows are modified by $\Theta(\delta/n)$, and the rest of the terms by $\Theta(\delta/n^2)$. The noise is spread throughout the matrix, even though the only terms of $B = -\frac{1}{2}HDH$ that bear on D_{kl} are B_{kk}, B_{ll}, B_{kl}, and B_{lk}. Put differently, though the noise occurs only on a single dissimilarity, it contaminates the entire matrix B at varying levels.

The result of this discussion is that we need to change the cost function so that it does not use the HDH transformation. The natural choice is

$$f(D^*) = \|D - D^*\|_1, \tag{3}$$

which is the cost function that we work with for the remainder of this paper. In section 4 we give an embedding algorithm optimizing (3). We demonstrate the described contamination effects in the experiments.

3.2. Other Cost Function Issues

Besides the issues just discussed, the $\frac{1}{2}HDH$ transform is conceptually a bit curious. The transformation makes sense when D is truly Euclidean, but is not obviously principled for $D \notin \mathbb{EDM}^n$. For $D \notin \mathbb{EDM}^n$, the cMDS cost function is awkward to interpret; it suggests that $-\frac{1}{2}HDH$ is a noise-corrupted version of the true Gram matrix B^*. Yet, the input to cMDS is D, so it seems more natural to assume that D is noise corrupted—*i.e.* $\|D - D^*\|_2$ seems better grounded conceptually. Moreover, though the set of mean-centered Gram matrices and the set of EDMs are in one-to-one correspondence,[1] optimizing $\|HDH - HD^*H\|_2$ is *not* equivalent to optimizing $\|D - D^*\|_2$ (Dattorro, 2005).

Future analysis notwithstanding, the cost function of cMDS seems like it is essentially a convenient approximation to $\|D - D^*\|_2$ that can be solved optimally and relatively quickly. Because of numerous advances in convex optimization since the invention of cMDS, $\|D - D^*\|_p$ can now be solved optimally for some values of p; (Gaffke & Mathar, 1989; Glunt et al., 1990) develop an interesting alternating projection algorithm for $p = 2$. Here we show how $\|D - D^*\|_1$ can be minimized using a semidefinite program.

4. Robust Euclidean Embedding

In this section, we show how to find an embedding using the robust cost function developed in the last section. Specifically, we show that the cost function can be optimized using a semidefinite program (SDP). Since general-purpose SDP solvers are unable to handle datasets with more than a hundred points or so, we also present a fast subgradient method for optimizing the robust cost function. We refer to the embedding algorithm developed for the robust metric as Robust Euclidean Embedding (REE).

Again, the optimization problem to be solved by REE is $\min_{D^* \in \mathbb{EDM}^n} \|D - D^*\|_1$. We generalize this slightly to a weighted cost function:

$$\min_{D^* \in \mathbb{EDM}^n} \sum_{ij} W_{ij} \left| D_{ij} - D^*_{ij} \right| . \tag{4}$$

We now show that, like the cMDS cost function, we can solve this one optimally.

4.1. Semidefinite Program Formulation

Semidefinite programs has proved useful in a number of recent machine learning applications because of the equivalence of Gram matrices and positive semidefinite matrices. Here, we show that the robust optimization problem (4) is easily formulated as a SDP. The program follows.

$$\begin{array}{rl} \min_{\xi, B} & \sum_{ij} W_{ij} \xi_{ij} \\ \text{subject to:} & -\xi_{ij} \le D_{ij} - [\text{dist}(B)]_{ij} \le \xi_{ij} \quad \forall i, j; \\ & \sum_{ij} B_{ij} = 0; \\ & B \succeq 0; \\ & \xi_{ij} \ge 0 \;\; \forall i, j. \end{array}$$

[1]The characterization of EDMs in terms of positive semidefinite matrices, mentioned in section 2, gives the correspondence.

In the program, W is a user-specified matrix of weights, ξ is the error matrix, and B is the Gram matrix. The term $\sum_{ij} B_{ij} = 0$ is a regularization constraint that removes the degree of freedom caused by the translation invariance of the problem. Once the program is solved, the corresponding configuration of points is $X = U\Lambda^{1/2}$, where $U\Lambda U^\top$ is the spectral decomposition of B.

There are a number of SDP-solvers available that can solve the above program. Our experiments were conducted using SDPT3 (Toh et al., 1999). However, general purpose SDP-solvers are too memory intensive for even modestly sized programs. We found that SDPT3 could only handle the REE program when $n \approx 100$ or less. To handle larger problem sizes, we implemented a fast first-order method, as we now describe.

4.2. A Subgradient Algorithm

The program for robust embeddings has a non-differentiable cost function and nonlinear constraints, so we cannot hope to solve the program using a standard gradient descent procedure. With a *subgradient* in place of a gradient, and a projection woven into each iteration, however, we can solve the program.

A subgradient of a (potentially non-differentiable) convex function f at x is any function g satisfying

$$f(y) \ge f(x) + g(x)(y - x)$$

for all y. We can use a subgradient to minimize a convex function much like gradients are used to optimize differentiable functions.

Since our variable B is constrained to be positive semidefinite, we must project it back into the positive semidefinite cone with each update we make. The procedure is straightforward: we update the Gram matrix by sliding along the subgradient, then project the updated matrix back onto the positive semidefinite cone. We repeat this process until the algorithm converges to a good solution.

A subgradient for $\sum_{ij} W_{ij} |D_{ij} - [\text{dist}(B)]_{ij}|$ is

$$[G(B)]_{ij} = \begin{cases} W_{ij} \mathbf{I}\left([\text{dist}(B)]_{ij} < D_{ij}\right) & \text{if } i \ne j; \\ \sum_k W_{ik} \mathbf{I}([\text{dist}(B)]_{ik} > D_{ik}) & \text{if } i = j. \end{cases}$$

($\mathbf{I}$ denotes the indicator function returning 1 or -1.)

Subgradient methods require a step size parameter; there are some standard choices for which convergence to an optimal solution is guaranteed (Bertsekas, 1999). We found that $\alpha_i = \frac{c}{\sqrt{i}}$ worked well in experiments.

```
Robust Euclidean Embedding
(subgradient implementation)
input: $D$, $W \in \mathbf{R}^{n \times n}$

  1. Set $B^0 \in \mathbf{R}^{n \times n}$ randomly.
  2. for $k = 1, 2, \ldots$
       • Set $B := B^{k-1} - \alpha_k G(B^{k-1})$.
       • Spectrally decompose $B$: $B = U \Lambda U^\top$.
       • Set $[\Lambda_+]_{ij} := \max\{\Lambda_{ij}, 0\}$.
       • $B^k := U \Lambda_+ U^\top$.
  3. Pick $k$ minimizing $\left(\sum_{ij} W_{ij} |D_{ij} - \mathrm{dist}(B^k)|\right)$.
  4. Return $X := U \Lambda^{1/2}$, where $U \Lambda U^\top$ is the spectral
     decomposition of $B^k$.
```

Figure 3. A subgradient algorithm for robust embedding.

The subgradient-based algorithm for robust embedding is shown in figure 3.

4.3. Comparison with Classical MDS

The REE algorithm improves upon classical MDS in several ways. First, it does not rely upon a conceptually awkward transformation that increases sensitivity to noise. The use of the ℓ_1 metric in the cost function further bolsters the new algorithm's robustness. We show some simple experiments in section 6 that illustrate the differences in robustness.

The robust embedding algorithm is also substantially more flexible because it can handle non-uniform weighting schemes on the cost function. If some D_{ij} is unavailable, we can set D_{ij} to an arbitrary value and set the weight W_{ij} to zero. If some of the D_{ij} are cruder estimates than others, we can explicitly down weight the error on those elements.

The cost of using REE is time complexity: each iteration of the subgradient algorithm requires $O(n^3)$ computations, whereas cMDS requires $O(n^3)$ total computations.

4.4. Speedups

Sampling techniques may be used to speed REE up dramatically. In particular, we may use the Nyström method to obtain a fast approximation to the robust embedding algorithm. The idea of the Nyström method is to embed only a small submatrix of the dissimilarity matrix using the robust embedding algorithm, then embed the rest of the matrix using a simple formula based on the eigenvalues of the Gram matrix found by the embedding procedure. Details of the Nyström method as applied to cMDS can be found in (de Silva & Tenenbaum, 2004; Platt, 2005) and can be easily adjusted for REE.

5. Dimensionality Reduction

Often, an embedding is used for visualization or another application that requires the configuration to be low-dimensional. We have ignored the issue of dimensionality up until now; indeed the dimensionality of the configuration found by the robust procedure is often quite high. In this section, we focus on finding an optimal Euclidean embedding in k-dimensional space.

As we noted in section 2, classical MDS finds an embedding and then reduces the dimensionality via principal components analysis (PCA). Though we could use PCA in conjunction with REE as well, it would be preferable to take the desired dimensionality into account when performing the embedding. If we use PCA after the embedding, the resulting configuration is not guaranteed to be optimal for the REE cost function.

We show that one cannot hope to find a low-dimensional embedding that optimizes the REE cost function because it is a NP-hard problem. We generalize this hardness result to a wide variety of cost functions, including $\|D - D^*\|_2$. We further show that a popular rank reduction technique clashes with the REE program.

5.1. Hardness of Low-Dimensional Embedding

Here is the problem we start with. Note that we work with dissimilarities that are *not* squared in this section.

ℓ_1 EUCLIDEAN EMBEDDING
Input: A dissimilarity matrix $D = (d_{ij})$.
Output: An embedding into the line: $x_1, x_2, \ldots \in \mathbf{R}$
Goal: Minimize $\sum_{i,j} |d_{ij} - |x_i - x_j||$.

We show that this problem is NP-hard by reducing from a variant of not-all-equal 3SAT. Other hardness results (Håstad et al., 1998; Saxe, 1979) apply only to a ℓ_∞ distortion measure, that is, $\max_{i,j} |d_{ij} - |x_i - x_j||$, which is not as appealing in statistical applications because it places complete trust in every dissimilarity coefficient. (Dhamdhere et al., 2004) investigates embedding under average-case distortion, but uses a different cost function than ours and only considers *non-contracting* embeddings—*i.e.* embeddings whose interpoint distances are at least as large as the input dissimilarities. Though they demonstrate a hardness result, their problem setup is substantially different from ours.

We reduce from the following problem.
RESTRICTED NAE 3SAT
Input: A Boolean formula in 3CNF, such that each clause has exactly three literals, and each pair of literals appears together in at most one clause.
Question: Is there an assignment to the variables such that each clause has either one or two satisfied literals?

Here is the reduction: we are given an instance

$$\phi(x_1, x_2, \ldots, x_n) = C_1 \wedge C_2 \wedge \cdots \wedge C_m$$

of RESTRICTED NAE 3SAT. Assume without loss of generality that no clause contains both a literal and its negation. We will now construct an instance of ℓ_1 EUCLIDEAN EMBEDDING, and show that there is some $C^* = \text{poly}(n)$ such that

> If ϕ is NAE-satisfiable, then there is an embedding of cost $\le C^*$; otherwise the best embedding has cost at least $C^* + 1/4$.

The embedding problem has $M+2n$ points: two points per variable, called x_i and $\overline{x}_i$, and M additional points $A_1, \ldots, A_M$ (where $M = 64n^2(C^* + 1/4)$).

Define (symmetric) distances between them as follows:

- $d(A_i, A_j) = 0$ for all i, j
- $d(A_i, x_j) = d(A_i, \overline{x}_j) = 1/2$ for all i, j
- $d(x_i, \overline{x}_i) = 1$ for all i
- If two literals appear together in a clause, the distance between them is 1 (note: any pair of literals appears together in at most one clause).
- All other interpoint distances are 3/4.

Claim 1. *If ϕ is NAE-satisfiable, there is an embedding with cost $C^* = n(n-1) - m/4$.*

Proof: This can be seen by placing positive literals at 1/2 and negative literals at $-1/2$. □

Claim 2. *If ϕ is not NAE-satisfiable, then any embedding has cost at least $C^* + 1/4$.*

Proof: (Sketch: details are in the full version of the paper.) Briefly, we assume without loss of generality that in the embedding, the origin coincides with the median of the A_i. If the embedding has cost less than $C^* + 1/4$, the following can be established in sequence:

1. All the A_i must lie within $\frac{1}{32n^2}$ of the origin;
2. Each x_i must lie within $\frac{1}{16n^2}$ of 1/2 or $-1/2$;
3. There must be n literals near 1/2 and n near $-1/2$, and neither of these two clusters can contain both a literal and its negation.

The corresponding assignment then NAE-satisfies the original formula. □

5.1.1. OTHER COST FUNCTIONS

The hardness result can be extended to distortion functions of the form $\sum_{i,j} g\Big(f(d_{ij}) - f(|x_i - x_j|)\Big)$ We assume that f, g are

1. symmetric;
2. monotonically increasing in the absolute values of their arguments;
3. Lipschitz on $[0, 1]$ with constant λ_U, that is, for $x, y \in [0, 1]$, $|f(x) - f(y)| \le \lambda_U |x - y|$; and
4. similarly lower-bounded: for some $\lambda_L > 0$, for any $x, y \in [0, 1]$, $|f(x) - f(y)| \ge \lambda_L |x - y| \max\{x, y\}$.

Notice that $f(x), g(x) \in \{x, x^2\}$ satisfy these conditions with $\lambda_U = 2, \lambda_L = 1$, meaning that $\|D - D^*\|_1$ and $\|D - D^*\|_2$ are both hard to minimize over one-dimensional embeddings. The details of the reduction are deferred to the full paper.

5.2. Trace Heuristic

Though the previous hardness result is new, it is well known that rank constraints are not convex and hence cannot be introduced into SDPs. A popular and often effective heuristic is to append a weighted trace term on to the cost function. The weighted trace function is the convex envelope of the rank—*i.e.* it is the best convex underestimate of the rank function, which is why it may aid in rank reduction.

We replace the cost function defined previously with $\min \sum_{ij} W_{ij} \xi_{ij} + \rho \cdot \text{trace}(B)$, where ρ is parameter that is adjusted manually to achieve the desired rank of B. When we experimented with the trace heuristic, we observed the following curious behavior.

- When $\rho \le n$, the trace factor did not affect the rank of B.
- When $\rho > n$, the SDP solver would return the zero-rank trivial matrix $B = \mathbf{0}\mathbf{0}^\top$.

When $\rho \le n$, the trace term does not seem to be weighted heavily enough to have a substantial effect on the program. We analyze the dual to explain the behavior of the SDP when $\rho > n$. The dual is

$$\begin{array}{rl} \max & \sum_{ij} D_{ij} S_{ij} \\ \text{subject to:} & S_{ij} \in [-1, +1] \text{ for } i \neq j; \\ & S\mathbf{1} = \rho\mathbf{1}; \\ & S \succeq 0. \end{array}$$

The objective of the solver is to find a weighting S of the non-negative dissimilarities D subject to the constraints. If we were to remove the last two constraints

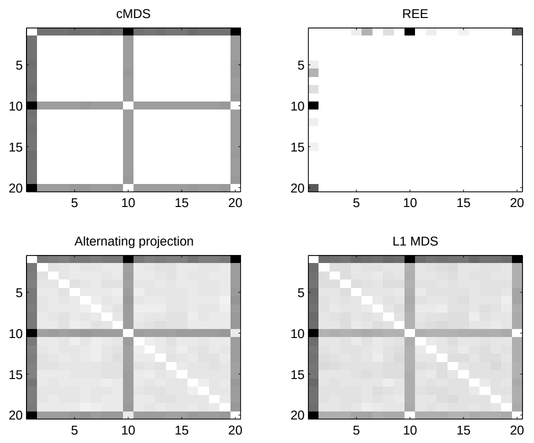

Figure 4. The error residual matrices for embeddings found by various algorithms. Darker shades indicate higher error.

($S\mathbf{1} = \rho\mathbf{1}$ and $S \succeq 0$), we could trivially solve the program by setting $S_{ij} = 1$ for all $i \neq j$ and S_{ii} arbitrarily (the setting of S_{ii} does not affect the cost function since $D_{ii} = 0$). The behavior of the full program whenever $\rho \geq n$ is essentially the same: the cost function is maximized by setting $S_{ij} = 1$ for $i \neq j$, and, with S_{ii} set to $\rho - (n-1)$, the PSD constraint and the row-sum constraints are satisfied. Note that this setting depends on $\rho \geq n$; otherwise, the S described will not be PSD.

What does this analysis imply for the primal? Recall that the primal has the constraint

$$-\xi_{ij} \leq D_{ij} - [\mathrm{dist}(B)]_{ij} \leq \xi_{ij}. \quad (5)$$

We need both inequalities so that $\sum_{ij} \xi_{ij}$ equals $\|D - \mathrm{dist}(B)\|_1$. Analysis of the dual variables reveals that when $S_{ij} = 1$ for all $i \neq j$, the Lagrange multipliers that enforce the $D_{ij} - [\mathrm{dist}(B)]_{ij} \leq \xi_{ij}$ inequalities are all equal to zero. This means that the only constraints on ξ_{ij} are $D_{ij} - [\mathrm{dist}(B)]_{ij} \geq -\xi_{ij}$ and $\xi_{ij} \geq 0$. We can trivially minimize $\sum_{ij} \xi_{ij}$ under these constraints (and the constraint that $B \succeq 0$) by setting $B = \mathbf{0}\mathbf{0}^\top$ and $\xi = \mathbf{0}\mathbf{0}^\top$.

6. Experiments

6.1. Robustness Demonstration

In section 3, we argued that cMDS is not robust in the sense that it propagates error in a single dissimilarity to points throughout the embedding. To exhibit this behavior, we took a $D \in \mathbb{EDM}^n$, corrupted only a couple of entries, and computed embeddings using cMDS and REE. We also compare the ℓ_1 version of cMDS, and the alternating projections algorithm (which minimizes $\|D - D^*\|_2$).

To generate D, we simply picked 20 random points in $\mathbf{R}^{20}$, computed the interpoint Euclidean distances, and added a large constant to two of the distances. Figure 4 shows the error residuals of the embeddings (with respect to the uncorrupted D). In the embedding found by cMDS, the placement of the x_{10} and x_{20} is incorrect with respect to all of the other points. The embeddings found by ℓ_1 cMDS and alternating projection exhibit a similar behavior; in addition, the error has crept into a number of entries besides x_{10} and x_{20}. REE, in contrast, has near-zero residuals for almost all of the uncorrupted dissimilarities.

To confirm the behavior exhibited by this experiment, we re-ran it 1000 times and calculated statistics. Each time we generated a new random set of points, calculated the interpoint distances D, selected two entries at random, and corrupted these entries by a random constant in the range $[0, \|D\|_2]$. We then embedded the corrupted distances using cMDS and REE. The statistic of interest is the number of entries of D that were distorted by the corruption-and-embedding process by more than one percent. The following table shows the mean and standard deviation of this statistic.

	mean	std. deviation
REE	12.0	3.4
cMDS	144.2	17.8

6.2. Visualization

Visualization is one of the primary uses of cMDS. Here, we look at a standard MDS visualization example: map reconstruction (Kruskal & Wish, 1978). The goal is to draw a map based on the interpoint distances between 10 US cities. These distances are approximately Euclidean, so cMDS finds a good embedding. We introduce noise into the dissimilarities by doubling the distance between Los Angeles and New York; the remainder of the dissimilarities are untouched. We consider the two-dimensional embedding found by cMDS and the two-dimensional configuration found by first embedding the dissimilarities using REE and then reducing the dimensionality to two using PCA. Figure 5 shows the results. REE clearly outperforms cMDS.

6.3. Embedding the Shape Distance of Images

Here, we present an example using real data that is not obviously better served by a robust algorithm. Embedding provides a quick way to adapt algorithms that take vectors as input to non-Euclidean dissimilarity measures; here we embed such dissimilarities.

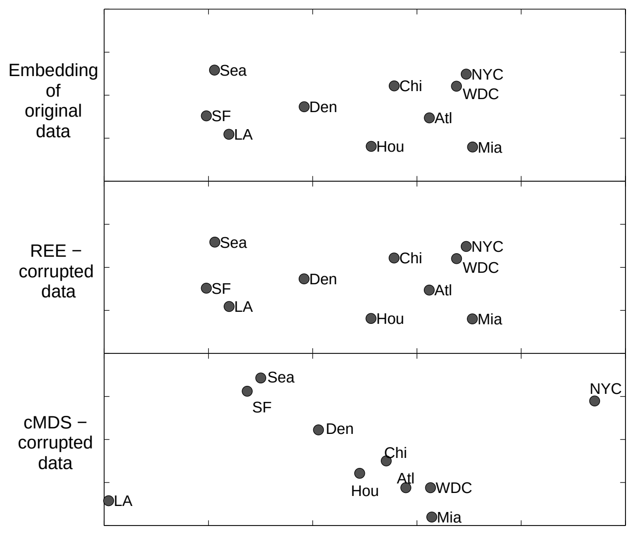

Figure 5. Map reconstruction.

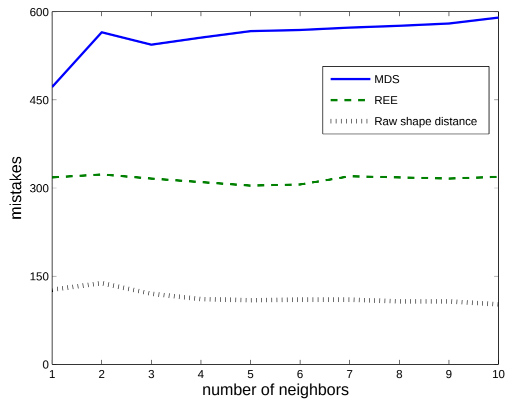

Figure 6. Nearest neighbor classification error.

The *shape distance* is a dissimilarity measure used to compare images that has proved useful for classification and retrieval (Belongie et al., 2002). In particular, it has been successfully applied to the problem of classifying the MNIST handwritten digits with a nearest neighbor algorithm.

We embedded the shape distance of 1000 handwritten digits (all 10 digits were included) taken from the MNIST dataset using both cMDS and REE. We then classified each embedded point according to its k nearest Euclidean neighbors among the remaining 999 points. Figure 6.3 shows the graph of the number of errors versus k for the cMDS and REE embeddings. The nearest-neighbor classifier based on the REE embedding substantially outperforms the cMDS-based classifier for each value of k, suggesting that REE embedding represents the shape distance more faithfully than the cMDS embedding. Note that neither embedding is competitive with the nearest-neighbor algorithm on the raw shape distances. However, our aim was merely to compare the performance of REE and cMDS on a real dataset using an agnostic evaluation criterion.

Acknowledgements

Thanks to Sameer Agarwal for help with computing the shape distances and to the anonymous reviewers for their suggestions.

References

Belongie, S., Malik, J., & Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Pattern Anal. Mach. Intell.*, *24*.

Bertsekas, D. P. (1999). *Nonlinear progamming*. Athena Scientific. 2nd edition.

Borg, I., & Groenen, P. (2005). *Modern multidimensional scaling*. Springer. 2nd edition.

Dattorro, J. (2005). *Convex optimization and euclidean distance geometry*. Meboo publishing.

de Silva, V., & Tenenbaum, J. (2004). Sparse multidimensional scaling using landmark points. Manuscript.

Dhamdhere, K., Gupta, A., & Ravi, R. (2004). Approximation algorithms for minimizing average distortion. *Proc. 21st STACS*.

Gaffke, N., & Mathar, R. (1989). A cyclic projection algorithm via duality. *Metrika*, *36*, 29–54.

Glunt, W., Hayden, L., Hong, S., & Wells, J. (1990). An alternating projection algorithm for computing the nearest euclidean distance matrix. *SIAM J. Matrix Anal. Appl.*, *11*, 589–600.

Håstad, J., Ivansson, L., & Lagergren, J. (1998). Fittings points on the real line and its application to rh mapping. *ESA* (pp. 465–476).

Huber, P. (1981). *Robust statistics*. Wiley Interscience.

Kruskal, J., & Wish, M. (1978). *Multidimensional scaling*. Sage Publications.

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

Platt, J. (2005). Fastmap, metricmap, and landmark mds are all nyström algorithms. *Proc. 10th AISTATS*.

Saxe, J. (1979). Embeddability of weighted graphs in k-space is strongly np-hard. *Proc. Allerton Conference on Circuit and System theory*.

Toh, K. C., Todd, M. J., & Tutuncu, R. (1999). SDPT3 — a Matlab software package for semidefinite programming. *Opt. Methods and Software*, *11*.