DEPARTMENT OF STATISTICS
University of Wisconsin
1300 University Ave.
Madison, WI 53706

TECHNICAL REPORT NO. 1108

October 5, 2005

Robust Manifold Unfolding with Kernel Regularization

Fan Lu [1]
Department of Statistics, University of Wisconsin, Madison, WI

Yi Lin [2]
Department of Statistics, University of Wisconsin, Madison, WI

Grace Wahba [3]
Department of Statistics, Department of Computer Sciences and Department of Biostatistics and Medical Informatics, University of Wisconsin, Madison, WI

Key Words and Phrases: Kernel Regularization, positive definite matrices, noisy dissimilarity data, convex cone programming

[1]Research supported in part by NSF grant DMS0072292, NSF grant SCI0330538 and NIH grant EY09946.
[2]Research supported in part by NSF Grant DMS0134987.
[3]Research supported in part by NSF grant DMS0072292 and NIH grant EY09946.

Robust Manifold Unfolding with Kernel Regularization

Fan Lu, Yi Lin and Grace Wahba
Department of Statistics
University of Wisconsin, Madison, 53706, USA

October 12, 2005

Abstract

We describe a robust method to unfold a low-dimensional manifold embedded in high-dimensional Euclidean space based on only pairwise distance information (possibly noisy) from the sampled data on the manifold. Our method is derived as one special extension of the recently developed framework called Kernel Regularization, which is originally designed to extract information in the form of a positive definite kernel matrix from possibly crude, noisy, incomplete, inconsistent dissimilarity information between pairs of objects. The special formulation is transformed into an optimization problem that can be solved globally and efficiently using modern convex cone programming techniques. The geometric interpretation of our method will be discussed.

1 Introduction

The dimensionality reduction problem appears in many research fields, where scientists try to conduct exploratory analysis or visualization of multivariate data. One special scenario happens often, when the goal is to find a meaningful/expected low-dimensional structure behind high-dimensional observations, or more precisely, to recover a low-dimensional parameterization of high-dimensional data assuming the data all lie on a low-dimensional manifold. In several recent papers [1, 2, 3, 4, 5], a large family of algorithms has been proposed to solve this particular type of dimensionality reduction problem (hereinafter, manifold-unfolding problem), in the spirit of reconstructing the manifold structure globally, but respecting only local information from the observed data. It is also well known [5, 6, 7, 8] that the solution to the manifold-unfolding problem is closely related to finding a symmetric positive definite kernel (hereinafter "kernel"). Recall that an $N \times N$ kernel obtained from data relating N objects may be used to assign Euclidean coordinates to the N objects in some $p < N$ Euclidean space.

In a recent work [9], a novel framework called Kernel Regularization is proposed to extract information in the form of a kernel matrix from possibly crude, noisy, incomplete, inconsistent dissimilarity or distance-like information between pairs of objects, while controlling a certain complexity measure of the kernel. A special formulation of the framework was applied to configure a set of proteins globally in a Euclidean space based on pairwise dissimilarity information only (see [9]). The configuration (in the form of a kernel) obtained can then be used by any clustering or classification algorithm for further inference if visualization is not the only purpose. Kernel regularization can be used replace of the roles of two traditional techniques, multidimensional scaling (MDS) and

principal component analysis (PCA), in a unified fashion using the special formulation in [9]. However, the manifold-unfolding problem is different from the problem targeted in [9], where global information needs to be preserved.

In this paper, we describe a kernel approach to solve the manifold-unfolding problem using another variation/formulation of the kernel regularization framework. We adopt the same essential idea as in [1, 2, 3, 4, 5] which is, loosely speaking, to respect local information while flattening the global structure. Nevertheless, our method is more flexible than the others in terms of the assumptions on the data. All previous methods assume (at least implicitly) that there is no noise associated with the observed data, while our method allows the existence of noise. One scenario of the noisy-version manifold-unfolding problem is when the observed scattered data points can fall off the 'true' underlying manifold. More precisely, one can assume the direct distance between observed data points and the target manifold follows a compactly supported distribution with zero mean and relatively small variance compared to the global spread of the original manifold . Another possible source of noise is measurement error for either point coordinates or pairwise distances. A related issue here is that to the extent that the desired solution to the manifold-unfolding problem is translation and rotation invariant, a reasonable method should depend only on pairwise distance information. The procedures in algorithms proposed in [2, 3, 4] use more than pairwise distances (at least, procedure-wise), while implementation of our method, like the algorithms in [1, 5], needs only pairwise distances. It is also worth mentioning that, when the distance/dissimilarity information is actually noisy, it might also be non-Euclidean (e.g., the triangle inequality can be violated). Then, in that case, the algorithm in [5] will try to solve an infeasible optimization problem. Nonetheless, our method can naturally handle this noisy situation. Moreover, our algorithm is insensitive to the non-convex case investigated in [4], which causes the algorithms proposed in [1, 2] to fail. So our method is robust for the manifold-unfolding problem in the sense that it can handle both noisy and non-convex data.

This paper is organized as follows. In Section 2, we review the Kernel Regularization framework before proposing a new formulation of it in order to solve the (noisy) manifold-unfolding problem. We also discuss the geometric interpretation of different formulations. In Section 3, we show some simulation results from implementing our method. Finally, we conclude in Section 4 with a summary and discussion of future work.

2 Regularized Kernel Embedding

2.1 Framework of Kernel Regularization

In the same spirit of all regularization methods, the Kernel Regularization method is designed to estimate a target, in our case, a kernel, from observed information, while controlling a certain complexity measure of the resulting estimate to prevent overfitting. The most general framework can be expressed as the following optimization problem:

$$\min_{K \in S_n} L(data, K) + \lambda J(K), \tag{1}$$

where S_N is the convex cone of all real nonnegative definite matrices of dimension N and L is some reasonable loss function on K. J is a kernel penalty (regularizing) functional, and λ is a tuning

parameter balancing fit to the data and the penalty on K. The choice of L obviously depends on the functional/distributional relationship (given or from model assumptions) between the observed data and target kernel, which is usually straightforward after the underlying problem is clear. On the other hand, a reasonable J can only be found after one understands/defines the complexity of the estimated kernel properly for a particular problem. Moreover, computational convenience should be considered when choosing L and J. In most cases, we want use L and J which makes the resulting optimization problem convex.

The Kernel Regularization framework proposed in [9] is motivated by the need to extract useful information from various kinds of dissimilarity information. Given a set of N objects, suppose we have obtained a measure of dissimilarity, d_{ij}, for certain object pairs (i, j). So the dissimilarity measure becomes the proxy to construct the loss function for the target kernel. Also, trace is chosen to be the kernel regularizing function J in order to promote dimension reduction in this case. One special formulation of Kernel Regularization framework in this scenario is the following:

$$\min_{K \in S_N} \sum_{(i,j)\in\Omega} w_{ij} |d_{ij} - B_{ij} \cdot K| + \lambda \,\text{trace}(K), \tag{2}$$

where Ω is the set of pairs for which we utilize dissimilarity information and the w_{ij}s are weights that may, if desired, be associated with particular (i, j) pairs. The natural induced dissimilarity, which is a real squared distance admitting of an inner product, is $\hat{d}_{ij} = K(i,i) + K(j,j) - 2K(i,j) = B_{ij} \cdot K$, where $K(i,j)$ is the (i,j) entry of K and B_{ij} is a symmetric matrix of dimension N with all elements 0 except $B_{ij}(i,i) = B_{ij}(j,j) = 1$, $B_{ij}(i,j) = B_{ij}(j,i) = -1$. The inner (dot) product of two matrices of the same dimensions is defined as: $A \cdot B = \sum_{i,j} A(i,j) \cdot B(i,j) \equiv \text{trace}(A^T B)$.

There are essentially no restrictions on the set of pairs other than requiring that the graph of the objects with pairs connected by edges be connected. A pair may have repeated observations, which just yield an additional term in (1) for each separate observation. If the pair set induces a connected graph, then the minimizer of (1) will have no local minima.

2.2 Deriving the Regularized Kernel Embedding Formulation

Denote the observed squared distance between x_i and x_j by d_{ij}. Let $(i, j) \in \Omega$ if x_j is one of the k-nearest neighbors of x_i, according to this observed squared distance.

We formulate the problem as finding a new positioning of the N points in $\mathcal{R}^p$ so that

(i) the repositioning respects local distance;

(ii) the repositioned points lie in a subspace of $\mathcal{R}^p$ with as low a dimension as possible.

Denote the points after repositioning as y_i, $i = 1, ..., N$. That is, the positioning moves the original point x_i to y_i, $i = 1, ..., N$. These N points are still in $\mathcal{R}^p$, but we hope they lie in a low dimensional subspace in $\mathcal{R}^p$. Denote the distance between y_i and y_j in $\mathcal{R}^p$ by r_{ij}. Notice that the positioning that satisfies these two conditions is not unique. In fact, for any positioning that satisfies (i) and (ii), rotating the coordinate system or shifting the points by a common vector results in a different positioning that also satisfies (i) and (ii). To take care of the shifting problem, we require that the repositioned points are centered at 0_p, the origin of the coordinate system. That is, $\bar{y} \equiv \frac{1}{n} \sum_{i=1}^{N} y_i = 0_p$. As will be seen later, we shall take care of the rotation problem by formulating

the problem in terms of the reproducing kernel matrix generated by the repositioned points, instead of the repositioned points themselves. The reproducing kernel matrix K is the N by N matrix with the element $K(i,j) = (y_i, y_j)$, where $(\cdot,\cdot)$ is the Euclidean inner product in $\mathcal{R}^p$. Notice this matrix is invariant under rotation of the points y_i.

Condition (i) requires that

$$r_{ij}^2 \approx d_{ij}, \quad \forall (i,j) \in \Omega. \tag{3}$$

Now notice that among all possible positioning that satisfies condition (i), the positioning that meets condition (ii) is one in which the distance between pairs of the repositioned points in $\mathcal{R}^p$ are maximized. This is most easily seen in the broken stick example coming up later, but it is also easy to see in general. Therefore we try to maximize $\sum_{i=1}^N \sum_{j=1}^N r_{ij}^2$ subject to (3). To balance the fidelity to local distances and the maximization of distance between all pairs, we use a penalty approach and try to minimize:

$$\sum_{(i,j)\in\Omega} (d_{ij} - r_{ij}^2)^2 - \lambda \sum_{i=1}^N \sum_{j=1}^N r_{ij}^2, \tag{4}$$

where $\lambda > 0$ is a tuning parameter. Alternatively, we can try to minimize

$$\sum_{(i,j)\in\Omega} |d_{ij} - r_{ij}^2| - \lambda \sum_{i=1}^N \sum_{j=1}^N r_{ij}^2, \tag{5}$$

Adopting the matrix inner product definition defined in Section (2.1)

$$\begin{aligned} \sum_{i=1}^N \sum_{j=1}^N r_{ij}^2 &= \sum_{i=1}^N \sum_{j=1}^N K(i,i) + K(j,j) - 2K(i,j) \\ &= 2NI \cdot K - 2 \sum_{i=1}^n \sum_{j=1}^n K(i,j) \\ &= 2NI \cdot K - 2E \cdot K = 2(NI - E) \cdot K, \end{aligned}$$

where I is the N-dimensional identity matrix and E is the N by N matrix with all elements being 1. Plugging $r_{ij}^2 = K(i,i) + K(j,j) - 2K(i,j)$ into (4) or (5), the problem becomes: find K positive semidefinite to minimize

$$\sum_{(i,j)\in\Omega} (d_{ij} - K(i,i) - K(j,j) + 2K(i,j))^2 - 2\lambda(NI - E) \cdot K, \tag{6}$$

or

$$\sum_{(i,j)\in\Omega} |d_{ij} - K(i,i) - K(j,j) + 2K(i,j)| - 2\lambda(NI - E) \cdot K. \tag{7}$$

We can also add weights w_{ij} into the first summation. There is an additional constraint needed to guarantee that the points are centered at 0_p. It is easy to show that if K is positive semidefinite, then $\bar{y} = 0_p$ is equivalent to $Ke = 0_p$, where e is the p by one vector whose elements are all ones, and this constraint can be added to the above optimization problems. This constrained minimization problem can be recast as a convex cone programming problem and there are efficient algorithms

developed in the convex optimization community for solving this type of problems. Notice that under the $Ke = 0_p$ constraint the objective function can be further simplified a little since if K is positive semidefinite, then $Ke = 0_p$ is equivalent to $E \cdot K = 0$. Once the matrix K is obtained, the dimension of the subspace of the repositioned points is $p = rank(K)$. Alternatively, the constraint $Ke = 0_p$ may be omitted and K centered later. This will be discussed further below. We can use the spectral decomposition, i.e., $K = \Gamma' D \Gamma$ where Γ is p by N consisting of p rows of eigenvectors of K, and D is the p by p diagonal matrix of non-zero eigenvalues $\{\lambda_\nu\}$, to get the principal coordinates $Y = D^{1/2}\Gamma$ with columns of Y to be y_i's.

One thing that we need to be careful about is that the neighbor set Ω should not be too small. Otherwise the objective function may diverge to $-\infty$. The necessary and sufficient condition to avoid this situation is that the edges in Ω construct a connected graph for all data points.

Now, it is easy to see that (7) differs from the formulation (2) only in the choice of kernel regularization function $J(K)$. In (7), $J(K) = trace(K) = I \cdot K$, while in (2) $J(K) = -2(NI - E) \cdot K$. It is worth pointing out that they are both linear thus convex in K.

2.3 General Convex Cone Problem

Problems (6) and (7) can all be solved globally (since they are convex in K) and efficiently using modern convex cone programming techniques. We describe here the general convex cone programming problem. This problem, which is central to modern optimization research, involves some unknowns that are vectors in Euclidean space and others that are symmetric matrices. These unknowns are required to satisfy certain equality constraints and are also required to belong to cones of a certain type. The cones have the common feature that they all admit a self-concordant barrier function, which allows them to be solved by interior-point methods that are efficient in both theory and practice [10].

To describe the cone programming problem, we define some notation. Let $\mathcal{R}^p$ be Euclidean p-space, and let P_p be the nonnegative orthant in $\mathcal{R}^p$, that is, the set of vectors in $\mathcal{R}^p$ whose components are all nonnegative. We let Q_q be the second-order cone of dimension q, which is the set of vectors $x = \big(x(1), \ldots, x(q)\big) \in \mathcal{R}^q$ that satisfy the condition $x(1) \geq [\sum_{i=2}^{q} x(i)^2]^{1/2}$. We define S_s to be the cone of symmetric positive semidefinite $s \times s$ matrices of real numbers. Inner products between two vectors are defined in the usual way and we use the dot notation for consistency with the matrix inner product notation.

The general convex cone problem is then:

$$\min_{X_j, x_i, z} \quad \sum_{j=1}^{n_s} C_j \cdot X_j + \sum_{i=1}^{n_q} c_i \cdot x_i + g \cdot z \tag{8}$$

$$\text{s.t.} \sum_{j=1}^{n_s} A_{rj} \cdot X_j + \sum_{i=1}^{n_q} a_{ri} \cdot x_i + g_r \cdot z = b_r, \quad \forall_r$$

$$X_j \in S_{s_j} \ \forall_j; \quad x_i \in Q_{q_i} \ \forall_i; \quad z \in P_p.$$

Here, C_j, A_{rj} are real symmetric matrices (not necessarily positive semidefinite) of dimension s_j, $c_i,\ a_{ri} \in \mathcal{R}^{q_i}$, $g,\ g_r \in \mathcal{R}^p$, $b_r \in \mathcal{R}^1$.

The solution of a general convex cone problem can be obtained numerically using publicly available software such as SDPT3 [11] and DSDP5 [12].

2.4 Regularized Kernel Embedding Formulation for l_1 Loss

To convert the problem of equation (7) into a convex cone programming problem, without loss of generality, we let Ω contain m distinct (i, j) pairs, which we index with $r = 1, 2, \ldots, m$. Let $N \times N$ matrices I and E be defined as before. Define $e_{m,r}$ to be vector of length $2m$ consisting of all zeros except for the rth element being 1 and $(m + r)$th element being -1. If we denote the rth element of Ω as $\big(i(r), j(r)\big)$, and with some abuse of the notation let $i = i(r)$, $j = j(r)$ and $w \in P_{2m}$ with $w(r) = w(r + m) = w_{i(r),j(r)}, \quad r = 1, \ldots, m$, we can formulate the problem of equation (2) as follows:

$$\begin{aligned} &\min_{K \succeq 0, u \geq 0} \; w \cdot u - 2\lambda(NI - E) \cdot K \\ &\text{s.t.} d_{ij} - B_{ij} \cdot K + e_{m,r} \cdot u = 0, \;\; \forall_r, \\ &K \in S_N, \;\; u \in P_{2m}. \end{aligned} \tag{9}$$

The solution to (9) might not be centered. To obtain sensible principal coordinates of the corresponding configuration using the spectral decomposition, we can center the solution kernel following a simple procedure. Define a to be the column with ith entry the average of the ith column of K, c to be the scalar as the mean of all elements in K and e to be vector of suitable dimension consisting of all 1's. A kernel K can be centered simply by: $K_{centered} = K - ae^T - ea^T + cE$. An alternative way to handle this centering step, as discussed before, is to directly impose the centering condition $E \cdot K = 0$, which is actually a linear constraint and can be directly incorporated into the convex cone formulation (9). Then, we can also simplify the kernel regularization function from $-2(NI - E) \cdot K$ to a reduced form $-2(NI) \cdot K$. However in our experiment with the examples here, the optimization problem without the $E \cdot K = 0$ constraint converges faster.

The Regularized Kernel Embedding formulation with square loss can also be easily obtained after simple modification of the square formulation in the appendix of [9].

2.5 'Newbie' Formulation

A very useful 'newbie' algorithm was developed in [9] to find the coordinates for new data points (newbies) within the previously constructed configuration. The corresponding newbie problem is essentially the minimization of the sum of losses involving the newbie only. We adopt the same idea here for the manifold-unfolding problem except we restrict the summation even further to a reasonable neighborhood of the newbie. The neighborhood construction problem will be discussed in general in the following section. However the algorithm remains the same.

2.6 Choosing Neighbors

Choosing neighbors for each sampled data point is a very important/tricky step for almost all methods including ours that are in the spirit of 'thinking globally while fitting locally'. However, it is not discussed in detail in previous papers. A simple way, which is adopted by most algorithms, is to choose k nearest neighbors for all data points. Then the neighbor-choosing problem degenerates into a neighborhood-size-choosing problem. In exploratory studies, where the truth is not known, the only thing one can do might be to start from a 'suitable' neighborhood size based some prior knowledge or intuition, and then vary the size to see how the results change. As we discussed previously, the neighborhood size has to be big enough so that the neighbor edges and all points consist

of a connected graph. On the other hand, if the neighborhood size is too big, we are respecting more than just the local structure, and also, the computation cost usually goes up quickly. A good choice of k and the sensitivity of the results to k depend on the density and distribution of the sampled points on the manifold. In previous papers, with dense enough samples for the examples, the authors simply choose moderate neighborhood sizes. In this paper, we also adopt this approach in the simulated examples.

Nonetheless, in some cases, fixing a neighborhood-size might not be a good approach to setup a connected graph especially when the sampling is very uneven across the underlying manifold or the manifold has very different curvature from place to place. For the formulation proposed in this work, we have another, possibly more stable way to tackle this issue. We can impose a compactly supported kernel around each data point to generate weights for all other points. Only those points who get non-zero weights become candidates to be neighbors for a particular data point, and their weights will be used to multiply the corresponding loss terms in (6) and (7). A threshold number can also to set so that every point only keeps no more than that number of closest neighbors from all candidates. A suitable bandwidth of the kernel can be selected based on the average closest-neighbor distance. The intuition of this approach is to give higher confidence to the distances between closer neighbors.

2.7 Parameter λ

The tuning parameter λ controls a balance between the twin goals we want to achieve – as λ increases, the average squared distance between points far apart is allowed to increase, thus enhancing "flattening", while as λ decreases, the solution is driven towards more closely respecting the observed local structure.

For an exploratory study, where the truth is not known, a sequence of λs within an appropriate range (usually in *log* scale) will give different results. Then prior knowledge may help to choose a good λ. For example, if it is known that there is not much noise within the data and a low dimensional embedding is preferred, one can gradually increase λ to get rid of insignificant dimensions until the sum of the losses exceed some limit.

However, if this manifold-learning task is just a part of bigger problem, e.g., clustering or classification, where we know the truth for training data, it will be natural to tune λ simultaneously with other possible tuning parameters, using standard tuning techniques like cross validation.

3 Unfolding Simulated Examples

3.1 Procrustes Measures

For the simulated data, the truth is known. A reasonable measure of the distance/similarity between two kernel matrices is needed to characterize the goodness of fit for different estimates. In some related literature, it is called Procrustes measure.

A suitable measure proposed in [13] is based on the the positional differences after matching two gram matrix under translation, rotation and reflection. Suppose A and B are two centered gram matrices, then the measure is calculated as follows:

$$G(A, B) = trace(A) + trace(B) - 2trace(A^{1/2}BA^{1/2})^{1/2}. \tag{10}$$

The normalized version of this measure is simply:

$$\gamma_p(A, B) = G(A, B)/(trace(A)trace(B))^{1/2}. \tag{11}$$

Alternatively, if we care only about the pairwise distance information, we can introduce another measure (as defined in [9]):

$$\gamma_d(A, B) = \sum_{i<j} |\hat{d}_{ijA} - \hat{d}_{ijB}| / \sum_{i<j} \frac{1}{2}(\hat{d}_{ijA} + \hat{d}_{ijB}), \tag{12}$$

where $\hat{d}_{ijA}$ and $\hat{d}_{ijB}$ are pairwise distance between object i and j, induced by A and B respectively.

3.2 Unfolding the Swiss Roll with a Window Punched Out

The first simulated example is a Swiss roll manifold with a rectangular window punched out close to the center. This example was used in Donoho and Grimes (2003) to show how a non-convex feature (the punched-out window) can cause some previous methods like ISOMAP [1] and LLE [2] to fail.

The following results are obtained on a random sample of 770 points, each point with 6 neighbors. There is no noise in this example. Figure 1 gives the scatter plot of original data points sampled on the manifold (except within the punched-out window). Figure 2 is the true parameterization, and its "rolled-up" version gives Figure 1. Figure 3 is the solution to our formulation with (9), with the tuning parameter $\lambda = 7e - 7$. In Figure 4, the eigensequence of the corresponding solution kernel is plotted in descending order on a log scale. We can clearly see the fact that the first two eigenvalues stand out significantly in magnitude compared with the rest of the eigenvalues, indicating a 2D embedding. (The last eigenvalue in Figure 4 is the computer version of the zero eigenvalue that goes with the constant function.) The principal coordinates in Figure 3 are constructed using these two significant eigenvalues and corresponding eigenvectors.

3.3 Unfolding the Noisy Wisconsin Roll

This example is specially designed to show the robustness of our method, especially compared with the method proposed recently in [5], which has a basic idea very similar to ours. We consider two types of noise, which are imposed on the pairwise distances between neighbors after the all neighbors are selected. In this example, the data points are sampled on a 'Wisconsin roll', which is a Swiss roll except there is a window in the shape of letter 'W' punched out (thus no points can be sampled with in it) which can be seen clearly if the roll is flatten out.

To impose the first type of noise, twenty percent of the selected pairwise distances are multiplied by a uniform random number over the interval from 0.85 to 1.15. The second type of noise is introduced to all chosen $d_{ij}s$ (between chosen neighbors) by binning them into 15 equal sized bins over the interval from the minimum to the maximum among these $d_{ij}s$. The value of each d_{ij} is then replaced by the center of the bin that it belongs to. It is an analog of the scenario where only ranks are provided as the distance/dissimilarity measure.

A random sample of 861 points was used for this example with the neighborhood size set to be $k = 6$. In both noisy situations, our method successfully (with λ in a proper range) converges to a global optimum with only two significant dimensions. See eigensequence plots Figure 7 and Figure

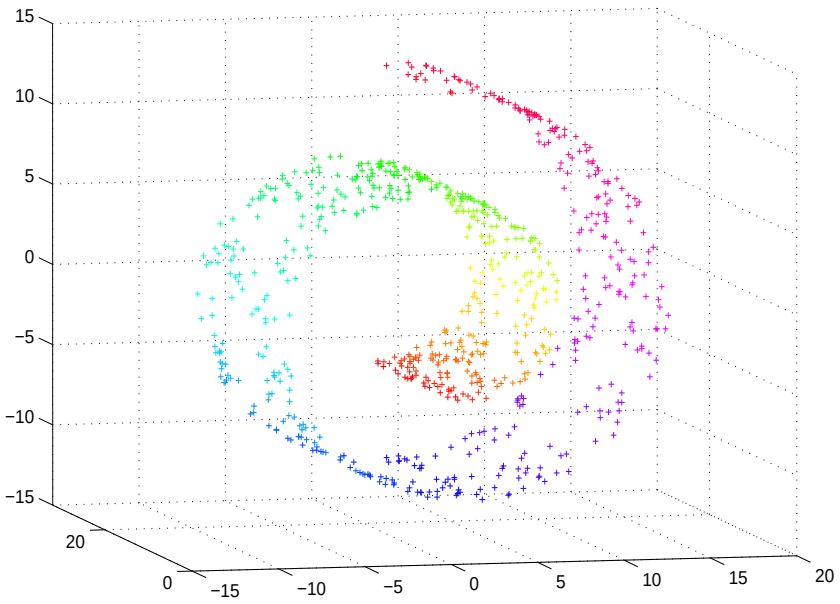

Figure 1: Swiss Roll: Scatter plot of original data points.

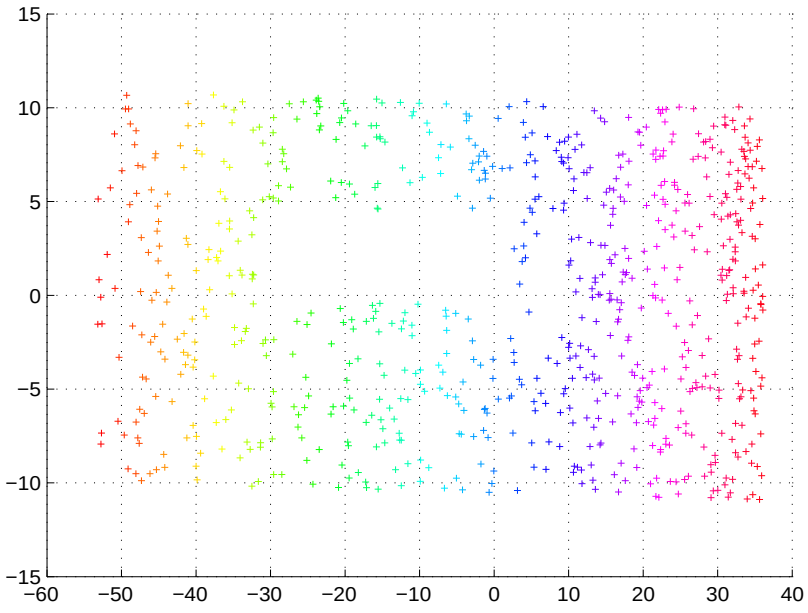

Figure 2: Swiss Roll: True parameterization. Rolled up version gives Figure 1.

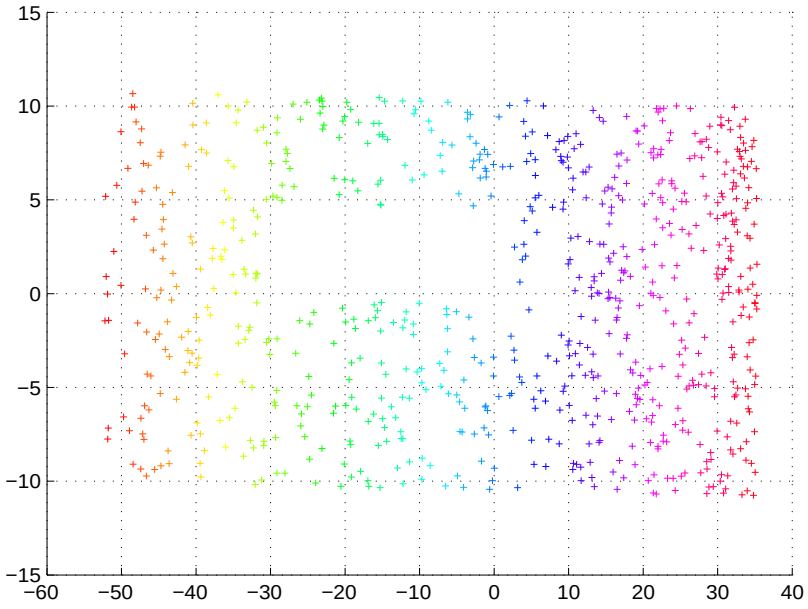

Figure 3: Swiss Roll Unrolled: Regularized Kernel Embedding using (9), $\lambda = 7e - 7$, first two principal coordinates.

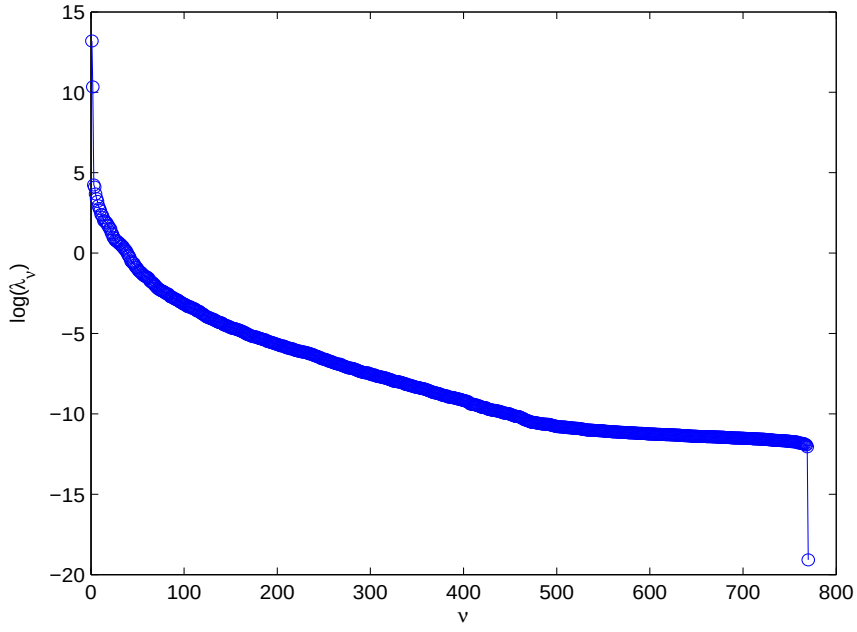

Figure 4: Swiss Roll: Eigensequence of the solution kernel, $\lambda = 7e - 7$. Note log scale.

9. The Procrustes measure shows our solution is very close to the truth (See Table 1), although the recovered embeddings shown in Figure 6 and Figure 8 are distorted a little bit from the truth (see Figure 5) due to the imposed noise.

Table 1: Procrustes Measure between Result and Truth

	1st type of noise case	2st type of noise case
γ_p	0.0055	0.0030
γ_d	0.0154	0.0112

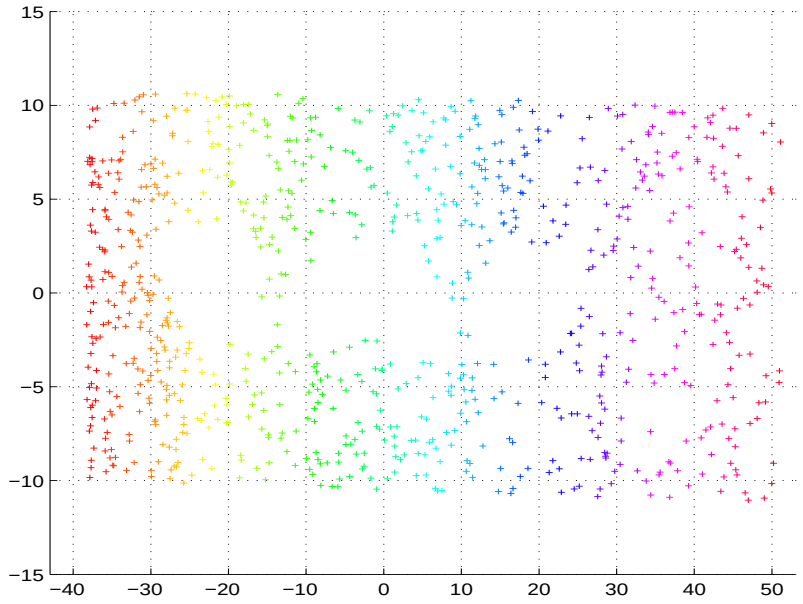

Figure 5: Wisconsin Roll: True parameterization. Observations come from rolled up version after adding noise.

On the contrary, the algorithm in [5] fails to converge because it tries to solve an infeasible primal problem for which the dual is unbounded. For the solvers we used, DSDP5 reported " DSDP: Dual Unbounded, Primal Infeasible" and SDPT3 reported "Stop: primal problem is suspected of being infeasible". These results are expected, because when a certain level of noise is directly imposed on the distance information, it is very likely that no Euclidean metric can fit the noisy distance data (for instance if the triangle inequality is violated somewhere). Then problem set-up in [5] is infeasible in the sense that no solution can satisfy all the constraints simultaneously.

3.4 Unfolding a Broken Stick

In this section we describe a toy example for the purpose of highlighting the difference between our method and the method proposed in [5]. The primary difference between the two methods is that

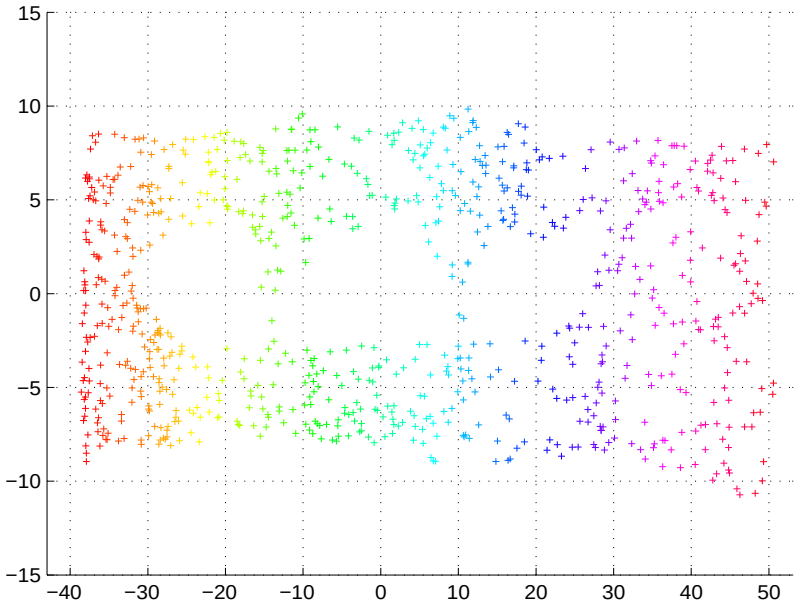

Figure 6: Wisconsin Roll with first type of noise, unrolled. Regularized Kernel Embedding using (9), $\lambda = 0.002$, first two principal coordinates.

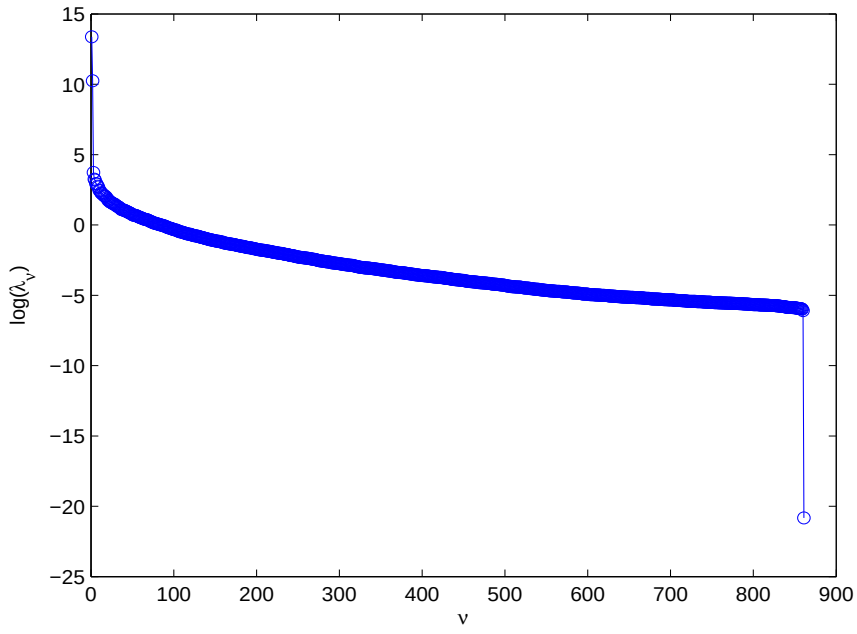

Figure 7: Wisconsin Roll: Eigensequence of the solution kernel, first type of noise, $\lambda = 0.002$.

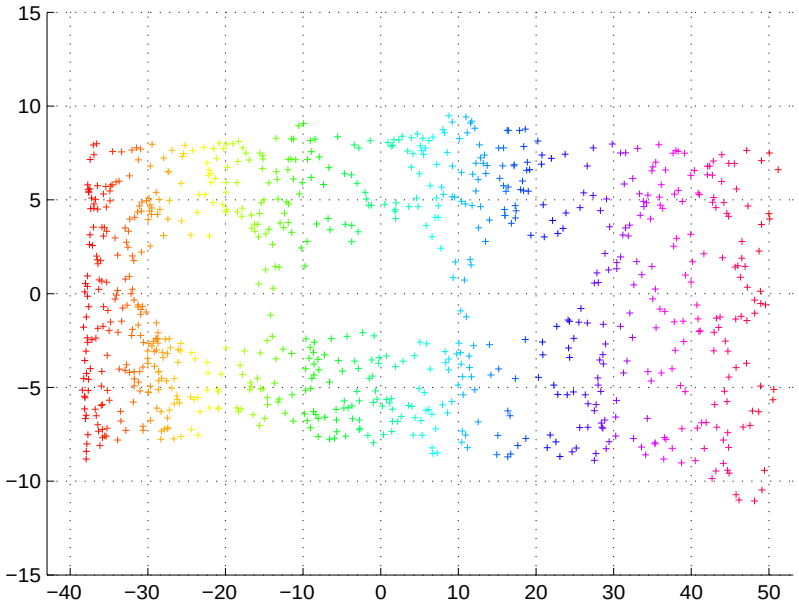

Figure 8: Wisconsin Roll with second type of noise, unrolled. Regularized Kernel Embedding using (9), $\lambda = 0.0025$, first two principal coordinates.

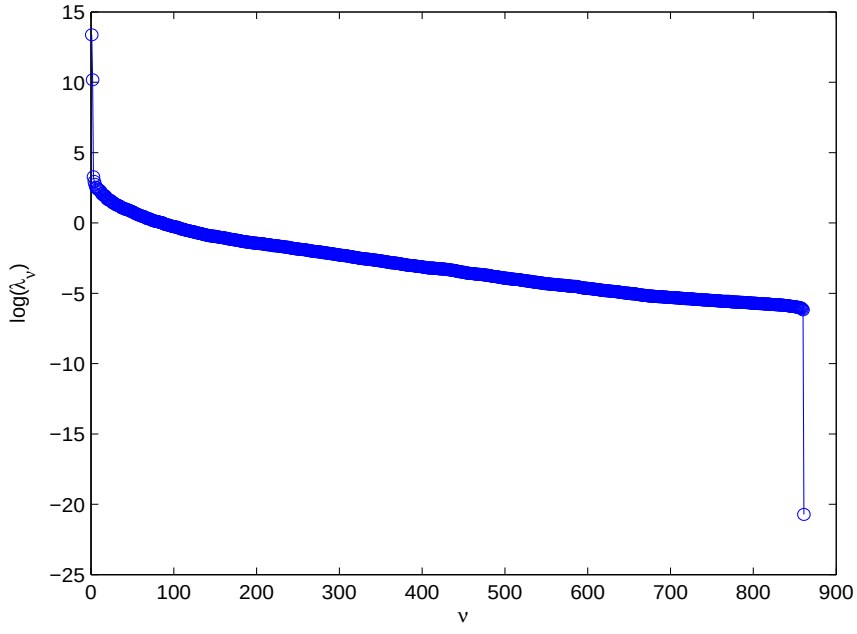

Figure 9: Wisconsin Roll: Eigensequence of the solution kernel, second type of noise, $\lambda = 0.0025$.

for the method in [5] local distances are enforced rigidly while here we relax that requirement. We want to show that this relaxation can be very important for manifold-unfolding problems even in the cases without noise.

The data points are randomly sampled on two branches of a 'broken stick' (see Figure 10). One branch is from the origin to the point $(1, 1)$ and the other is from $(1, 1)$ to $(2, 0)$. We force the sample to include the point $(1, 1)$.

The manifold unfolding goal here is to flatten out the stick. If any of the pairs for which distance is selected to fit, has one member from the left branch and the other member from the right branch (for example, see the black line in Figure 10), then the method in [5] will not be able to flatten the stick. For our method, a small λ will not flatten the stick either, but a sufficiently large λ will. The result from employing the method in [5] with $k = 5$ is almost visually indistinguishable from the plot in Fig 10. With $k = 5$ and λ too small ($\lambda = 1e - 5$) , our method also fails to flatten the stick but recovers the original broken stick. Two outstanding eigenvalues are obtained as can be seen in the upper left corner of Figure 11. However, with λ sufficiently large ($\lambda = 0.3$) we see only one outstanding eigenvalue, and so we obtain the one dimensional flattened stick on the lower right corner of Figure 11. As expected, within our regularized kernel embedding framework, the smoothness/dimensionality is controlled by the smoothing/tuning parameter λ.

4 Discussion and Future Work

In this paper, we developed a robust manifold learning method as a variation of the RKE framework proposed in [9]. It is worth mentioning that, if we choose to impose the centering constraint $E \cdot K = 0$ (although we can do without this) in problems (6) and (7), the kernel regularization function for manifold unfolding becomes $J(K) = -2(NI - E) \cdot K = -2NI \cdot K = -2N\text{trace}(K)$. Interestingly, in [9], the kernel regularization function we use to promote dimension reduction is trace instead of the negative trace (with a constant multiplier) here. So, different signs in front of trace actually both promote dimension reduction but in different scenarios.

More interesting problems come up when there are multiple source of information that are believed to share the same underlying low-dimensional structure. Our method can be naturally extended to that case. Also, it is often unrealistic to assume the given distance information is actually Euclidean. Then, a non-metric variation of our method, i.e., only rank information among all distances will be used, can be very useful. Last but not the least, we will explore the weighting scheme as we discussed in section (2.6) to select neighbors in order to achieve higher stability and robustness.

Acknowledgments

We thank Stephen J. Wright for many helpful discussions. This work has been partly supported by NSF Grant DMS0072292, NIH Grant EY09946 and NSF Grant SCI-0330538 (Fan Lu), NSF Grant DMS0134987 (Yi Lin), and NSF Grant DMS0072292 and NIH Grant EY09946 (Grace Wahba).

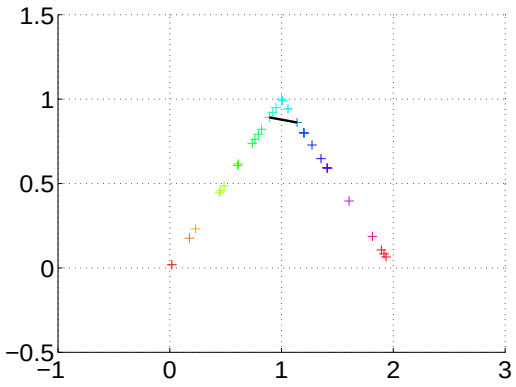

Figure 10: Broken Stick: Original data. Heavy black line joins a pair of points with the members from different sides of the break.

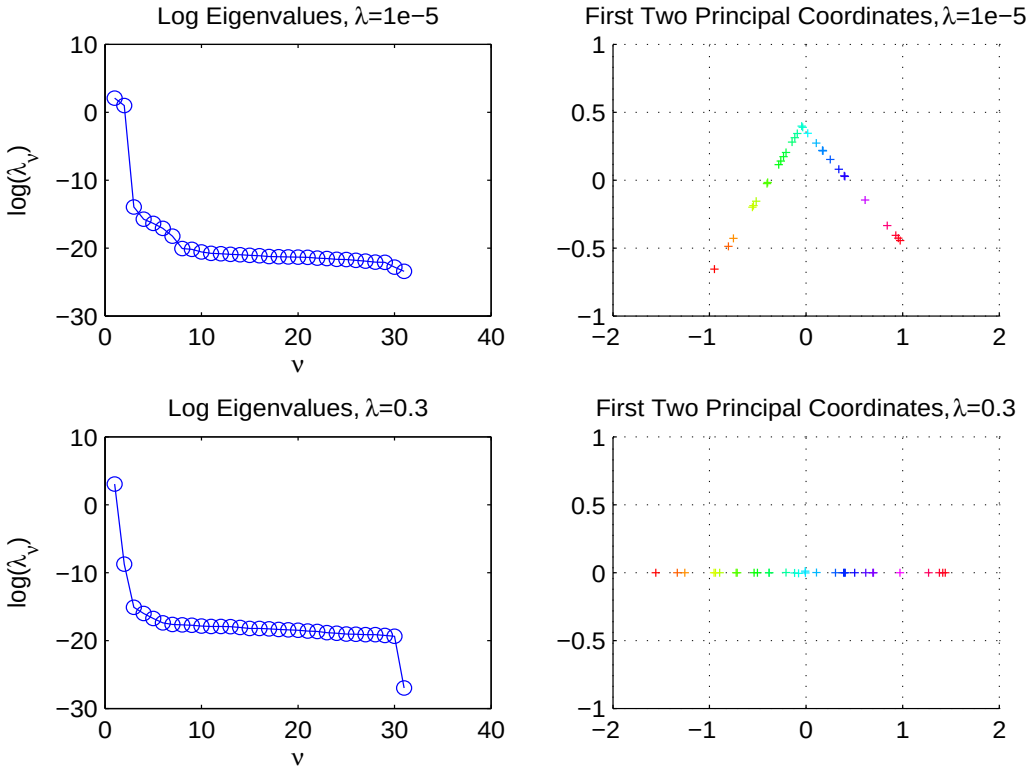

Figure 11: Broken Stick: Effect of λ on the Regularized Kernel Embedding using (9). Small λ does not flatten the stick, but a larger λ does.

References

[1] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science* **290**, 2319-2323, 2000.

[2] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science* **290**, 2323-2326, 2000.

[3] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation* **15(6)**, 1373-1396, 2003.

[4] D. L. Donoho and C. E. Grimes, "Hessian eigenmaps: locally linear embedding techniques for highdimensional data," *Proceedings of the National Academy of Arts and Sciences* **100**, 5591-5596, 2003.

[5] K. Q. Weinberger, F. Sha, and L. K. Saul, "Learning a kernel matrix for nonlinear dimensionality reduction," *Proceedings of the Twenty First International Conference on Machine Learning (ICML-04)*, 839-846, Banff, Canada, 2004.

[6] J. Ham, D. D. Lee, S. Mika, and B. Scholkopf, "A kernel view of the dimensionality reduction of manifolds," *Proceedings of the Twenty First International Conference on Machine Learning (ICML-04)*, 369-376, Banff, Canada, 2004.

[7] B. Scholkopf, A. J. Smola, and K.-R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation* **10**, 1299-1319, 1998.

[8] C. K. I. Williams, "On a connection between kernel PCA and metric multidimensional scaling." In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems* **13**, 675-681, Cambridge, MA, 2001. MIT Press.

[9] F. Lu, S. Keles, S. Wright and G. Wahba, "Framework for Kernel Regularization With Application to Protein Clustering," *Proceedings of the National Academy of Sciences* **102**, 12332-12337.

[10] Y. Nesterov and A. Nemirovskii, *Interior-Point Polynomial Algorithms in Convex Programming*, SIAM Studies in Applied Mathematics, **13**, 1993.

[11] R. H. Tütüncü, K. C. Toh and M. J. Todd, "Solving semidefinite-quadratic-linear programs using SDPT3," *Mathematical Programming* **95**, 189-217, 2003.

[12] S. J. Benson and Y. Ye, "DSDP5: A Software Package Implementing the Dual-Scaling Algorithm for Semidefinite Programming," Technical Report ANL/MCS-TM-255, Mathematics and Computer Science Division, Argonne National Laboratory.

[13] R. Sibson, "Studies in the Robustness of Multidimensional Scaling: Procrustes Statistics," *Journal of the Royal Statistical Society. Series B* **40**, 234-238, 1978