2015 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 17–20, 2015, BOSTON, USA

ORDINAL EMBEDDING OF UNWEIGHTED KNN GRAPHS VIA SYNCHRONIZATION

Mihai Cucuringu, Joseph Woodworth

UCLA Mathematics Department

ABSTRACT

We consider the problem of embedding unweighted, directed k-nearest neighbor graphs in low-dimensional Euclidean space. The k-nearest neighbors of each vertex provide ordinal information on the distances between points, but not the distances themselves. Relying only on such ordinal information, we recover the coordinates of the points up to arbitrary similarity transformations (rigid transformations and scaling). We make existing approaches scalable by using an instance of a local-to-global algorithm based on group synchronization, recently proposed in the literature in the context of sensor network localization, which we augment with a scale synchronization step. We show our approach compares favorably to the recently proposed Local Ordinal Embedding (LOE) algorithm even in the case of smaller sized problems, and also demonstrate its scalability on large graphs. The above divide-and-conquer paradigm can be of independent interest to the machine learning community when tackling geometric embeddings problems.

Index Terms— k-nearest-neighbor graphs, ordinal constraints, graph embeddings, eigenvector synchronization

1. INTRODUCTION

Embedding unweighted k-nearest neighbor (kNN) graphs is a special case of ordinal or non-metric embedding, where one seeks a spatial embedding of n points $\{\vec{x}_i\}_{i=1}^n$ in $\mathbb{R}^d$ such that

$$\forall (i_1, j_1, i_2, j_2) \in \mathcal{C}, \quad \|\vec{x}_{i_1} - \vec{x}_{j_1}\|_2 < \|\vec{x}_{i_2} - \vec{x}_{j_2}\|_2, \quad (1)$$

where $\mathcal{C}$ denotes the set of ordinal constraints. Ordinal constraints are sometimes also specified as triplets [1]. In the case of unweighted kNN graph embedding, $\mathcal{C} = \mathcal{C}(G) = \left\{(a, b, a, c) \middle| ab \in E(G), ac \notin E(G)\right\}$, where $E(G)$ is the set of directed edges in the kNN graph G.

Graph-based methods are of utmost importance in several modern machine learning methods with applications such as clustering, dimensionality reduction, and ranking. Many such methods rely on weighted graphs, with weights often based on similarity functions, i.e., $w_{ij} = S(x_i, x_j)$. From a practical standpoint, storing unweighted kNN graphs instead would allow for a very sparse representation of the data. If one could recover the source data $\{x_i\}_{i=1}^n$ from unweighted kNN graphs, such a computationally efficient sparser representation would incur no information loss. Because of the extreme sparsity of the representation, this is generally a hard problem. Just recently, a method for recovering data distributions from unweighted kNN graphs was introduced in [2]. Another motivation for this problem comes from an instance of the popular sensor network localization problem, where each sensor is able to transmit only limited connectivity information to a central location (ID names of its k nearest neighbors), but transmits neither the distance measurements nor a complete list of all its neighbors within a given fixed radius.

The original work on this problem dates back to Shepard [3] and Kruskal [4, 5], and lately has been studied intensively in the machine learning literature [1, 6, 7]. In this work, we compare against and extend the recent Local Ordinal Embedding method [8], which enjoys several favorable comparisons with other modern methods. Our key ingredient is a modified version of the As-Synchronized-As-Possible (ASAP) algorithm introduced in [9], which makes existing embedding methods scalable via a divide-and-conquer, non-iterative local to global approach, which reduces computational complexity, allows for massive parallelization of large problems, and increases robustness to noise. The ASAP algorithm introduced in [9], on which we rely in the present paper, renders our approach to reconstruct kNN graphs scalable to graphs with thousands or even tens of thousands of nodes, and is an example of a local-to-global approach that integrates local ordinal information into a global embedding calculation.

We detail in Section 3.1 the exact approach used to decompose the initial kNN graph into many overlapping subgraphs, that we shall refer to as patches from now on. Each resulting patch is then separately embedded in a coordinate system of its own using an ordinal embedding algorithm, such as the recent Local Ordinal Embedding (LOE) algorithm [8]. In the hypothetical scenario when LOE recovers the actual ground truth coordinates of each patch, such local coordinates agree with the global coordinates up to scaling and some unknown rigid motion (such as rotation, reflection and translation), in other words, up to a similarity transformation. However, in most practical instances, it is unreasonable to expect that the LOE algorithm will recover the exact coordinates only from ordinal data. On a related note, we point out the recent work of Kleindessner and von Luxburg [10], who settled a long-known conjecture claiming that, given knowledge of all ordinal constraints of the form $||x_i - x_j|| < ||x_k - x_l||$ be-

978-1-4673-7454-5/15/$31.00

tween an unknown set of points $x_1, \ldots, x_n \in \mathbb{R}$ (for finite n), it is possible to approximately recover the ground truth coordinates of the points up to similarity transformations. Furthermore, the same authors show that the above statement holds even when we only have **local** information such as the distance comparisons between points in small neighborhoods of the graphs, thus giving hope for a local-to-global approach, in the spirit of the one we propose in the present paper.

Our contributions are: **1.** We present a local-to-global approach, which is scalable to very large graphs, and can be computed efficiently and robustly in a distributed manner. **2.** We extend the ASAP framework to the setting of ordinal embeddings, by augmenting it with a scale synchronization step.

The rest of the paper is organized as follows. Section 2 is a summary of existing methods for related embedding problems. Section 3 details the pipeline of the ASAP framework, including the scale synchronization step in Section 3.2. Section 4 shows the results of several experiments and compares to the existing LOE algorithm. We conclude in Section 5.

2. RELATED WORK

2.1. Multidimensional Scaling

Broadly speaking, multidimensional scaling (MDS) refers to a number of related problems and methods. In Classical Multidimensional Scaling (CMDS) [11], one is given all Euclidean Squared-Distance measurements $\Delta_{ij} = \|\vec{x}_i - \vec{x}_j\|_2^2$ on a set of points $X = \{\vec{x}_i\}_{i=1}^n$ and wishes to approximate the points, assuming that they approximately lie in a low-dimensional space $d \ll n$. Note that the solution for the coordinates is unique only up to rigid transformations, and that solutions do not exist for all possible inputs Δ.

One can generalize CMDS to incorporate additional non-negative weights W_{ij} on each distance, useful when some distances are missing, or most distances are noisy, but some are known. The optimization involves minimizing an energy known in the literature as *stress* [12]. One approach to minimize stress is to iteratively minimize a majorizing function of two variables. A further generalization of MDS is non-metric MDS, or Ordinal Embedding, in which the input D is assumed to be an increasing function applied to distance measurements [3]. This may be the case if D represents dissimilarity between points, as opposed to measured distances. The problem can again be expressed with stress functionals and is usually solved with isotonic regresion [5].

2.2. Local Ordinal Embedding

Terada and von Luxburg [8] have recently proposed an algorithm for ordinal embedding and kNN embedding specifically, called Local Ordinal Embedding (LOE), which minimizes a soft objective function that penalizes violated ordinal constraints, with a scale parameter $\delta > 0$ included

$$\min_{X \in \mathbb{R}^{d \times n}} \sum_{i<j,k<l,(i,j,k,l) \in \mathcal{C}} \max\left[0, D_{ij}(X) + \delta - D_{kl}(X)\right]^2. \quad (2)$$

An advantage of this energy in contrast to ones that normalize by the variance of X (to guarantee nondegeneracy) is its relatively simple dependence on X, making the above energy easier to minimize. The authors of [8] introduce algorithms to minimizing the above energy, based on majorization minimization or the (BFGS) approximation of Newton's method.

3. ASAP & SCALE SYNCHRONIZATION FOR ORDINAL EMBEDDINGS

In this section we detail the steps of the ASAP algorithm, central to the divide-and-conquer algorithm we propose for the ordinal embedding problem. Note that the difference between the original ASAP algorithm introduced in [9] and our approach lies in the decomposition method from Section 3.1 and the scale synchronization step from Section 3.2. The ASAP approach starts by decomposing the given graph G into overlapping subgraphs (referred to as *patches*), which are then embedded via the method of choice (in our case LOE). To every local patch embedding, there corresponds a scaling and an element of the Euclidean group Euc(d) of d-dimensional rigid transformations, and our goal is to estimate the scalings and group elements that will properly align all the patches in a globally consistent framework. The local optimal alignments between pairs of overlapping patches yield noisy measurements for the ratios of the above unknown group elements. Finding group elements from noisy measurements of their ratios is also known as the group synchronization problem. for which Singer [13] introduced spectral and semidefinite programming (SDP) relaxations over the group SO(2) of planar rotations, which is a building block for the ASAP algorithm [9]. Table 1 gives an overview of the steps of our approach. The inputs are an ordinal graph (we consider kNN graphs) $G = (V, E)$, where edge $ij \in E$ and non-edge $il \notin E$ indicates that $d_{ij} \leq d_{il}$, the max patch size parameter MPS, the target dimension d, and a base-case ordinal embedding method $OrdEmbed : G \mapsto X \in \mathbb{R}^{d \times n}$ for embedding each patch, such as LOE.

3.1. Break up the kNN graph into patches and embed

The first step we use in breaking the kNN graph into patches is to apply normalized spectral clustering [14] to a symmetrized version of the graph. Normalized spectral clustering partitions the nodes of a graph into $N \ll n$ clusters by performing k-means on the embedding given by the top N eigenvectors of the random-walk normalized graph Laplacian. It is shown [14] that normalized spectral clustering minimizes a relaxation of the normalized graph cut problem. Next, we enlarge the clusters by adding the graph-neighbors of each node, so that the resulting patches have significant overlap, a prerequisite for the ASAP synchronization algorithm. The higher the overlap between the patches, the more robust the pairwise group ratio estimates would be, thus leading overall to a more accurate final global solution. Finally, we use an iterative procedure to remove nodes from the graph relying on tools from rigidity theory, which we omit due to space considerations.

INPUT	$G = (V, E),\ \|V\| = n,\ \|E\| = m,\ MPS,\ d,\ OrdEmbed(\cdot)$
Choose Patches Embed Patches	1. Break G into N overlapping globally rigid patches $P_1, \ldots, P_N$ following the steps in Sec. 3.1. 2. Embed each patch P_i separately based on the ordinal constraints of the corresponding subgraph of G using $OrdEmbed(\cdot)$.
Step 1 Scale	1. If a pair of patches (P_i, P_j) have enough nodes in common, let Λ_{ij} be the median of the ratios of distances realized in the embedding of P_i and their corresponding distances in P_j as in (3); otherwise set $\Lambda_{ij} = 0$. 2. Compute the eigenvector v_1^{Λ} corresponding to the largest eigenvalue of the sparse matrix Λ. 3. Scale each patch P_i by $v_1^{\Lambda}(i)$, for $i = 1, \ldots, n$
Step 2 Rotate & Reflect	1. Align all pairs of patches (P_i, P_j) that have enough nodes in common. 2. Estimate their relative rotation and possibly reflection $H_{ij} \in O(d) \subset \mathbb{R}^{d\times d}$. 3. Build a sparse $dN \times dN$ symmetric matrix $H = (H_{ij})$ where entry ij is itself a matrix in $O(d)$. 4. Define $\mathcal{H} = D^{-1}H$, where D is a diagonal matrix with $D_{1+d(i-1),1+d(i-1)} = \ldots = D_{di,di} = deg(i), i = 1, \ldots, N$, where $deg(i)$ is the node degree of patch P_i. 5. Compute the top d eigenvectors $v_i^{\mathcal{H}}$ of $\mathcal{H}$ satisfying $\mathcal{H}v_i^{\mathcal{H}} = \lambda_i^{\mathcal{H}} v_i^{\mathcal{H}}, i = 1, \ldots, d$. 6. Estimate the global reflection and rotation of patch P_i by the orthogonal matrix $\hat{h}_i$ that is closest to $\widetilde{H}_i$ in Frobenius norm, where $\widetilde{H}_i$ is the submatrix corresponding to the i$^{\text{th}}$ patch in the $dN \times d$ matrix formed by the top d eigenvectors $[v_1^{\mathcal{H}} \ldots v_d^{\mathcal{H}}]$. 7. Update the embedding of patch P_i by applying the above orthogonal transformation $\hat{h}_i$
Step 3 Translate	Solve $m \times n$ overdetermined system of linear equations (5) for optimal translation in each dimension.
OUTPUT	Estimated coordinates $\hat{x}_1, \ldots, \hat{x}_n$

Table 1. Overview of the modified ASAP algorithm that incorporates the scale synchronization step.

3.2. Scale Synchronization

Before applying the original ASAP algorithm to the embedded patches, we introduce an additional step that further improves our approach and is independent of the dimension d. In the *graph realization problem* which motivated ASAP, one is given a graph $G = (V, E)$ and non-negative distance measurement d_{ij} associated with each edge $ij \in E(G)$, and is asked to compute a realization of G in $\mathbb{R}^d$. The distances are readily available to the user and thus the local embedding of each patch is on the same scale as the ground truth. However, in the kNN embedding problem, distances are unknown and the scale of one patch relative to another must be approximated. Any ordinal embedding approach has no way of relating the scaling of the local patch to the global scale. To this end, we augment the ASAP algorithm with a step where we synchronize local scaling information to recover an estimate for the global scaling of each patch, thus overall synchronizing over the group of similarity transformations.

We accomplish this as follows. Given a set of patches, $\{P_i\}_{i=1}^N$, we can create a patch graph in which two patches are connected if and only if they have at least $d + 1$ nodes in common. We then construct a matrix $\Lambda \in \mathbb{R}^{N\times N}$ as

$$\Lambda_{ij} = \begin{cases} \text{median}\left\{ \frac{D_{a,b}^{P_i}}{D_{a,b}^{P_j}} \right\}_{a\neq b\in P_i\cap P_j} & \text{if } P_i \sim P_j,\ i \leq j, \\ 1/\Lambda_{ji} & \text{if } P_i \sim P_j,\ i > j, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where $D_{a,b}^{P_i}$ is the distance between nodes a and b as realized in the embedded patch P_i. The matrix Λ approximates the relative scales between patches. If all distances in all patches were recovered correctly up to scale, and all patches had sufficient overlap with each other, then each row of Λ would be a scalar multiple of the others and each column of Λ would be scalar multiple of the others, thus rendering Λ a rank-1 matrix. For the noisy case, in order to get a consistent estimate of global scaling, we compute the best rank-1 approximation of Λ, given by its leading eigenvector $v_1^{(\Lambda)}$. We use this approximation of global scaling to rescale the embedded patches before running ASAP. Note that the connectivity of the patch graph and the non-negativity of Λ guarantee, via the Perron-Frobenius Theorem, that all entries of $v_1^{(\Lambda)}$ are positive.

3.3. Optimal Rotation, Reflection and Translation

After applying the optimal scaling to each patch embedding, we use the original ASAP algorithm to integrate all patches in a global framework, as illustrated in the pipeline in Figure 1. We estimate, for each patch P_i, an element of the Euclidean group Euc(d) = O(d) $\times\mathbb{R}^d$ which, when applied to that patch embedding P_i, aligns all patches as best as possible in a single coordinate system. In doing so, we start by estimating, for each pair of overlapping patches P_i and P_j, their optimal relative rotation and reflection, i.e., an element H_{ij} of the orthogonal group O(d) that best aligns P_j with P_i. Whenever the patch embeddings perfectly match the ground truth, $H_{ij} = O_i O_j^{-1}$. We refer the reader to [9] for several methods on aligning pairs of patches and computing their relative reflections and rotations $H_{i,j}$. Finding group elements $\{O_i\}_{i=1}^N$ from noisy measurements H_{ij} of their ratios is also known as the group synchronization problem. Since this problem is NP-hard, we rely on the spectral relaxation [13] of

$$\min_{O_1,\ldots,O_N\in O(d)} \sum_{P_i\sim P_j} \|O_i O_j^{-1} - H_{ij}\|_F^2. \tag{4}$$

for synchronization over O(2), and estimate a consistent global rotation of each patch from the top d eigenvectors of the graph Connection Laplacian, as in Step 2.4 in Table 1.

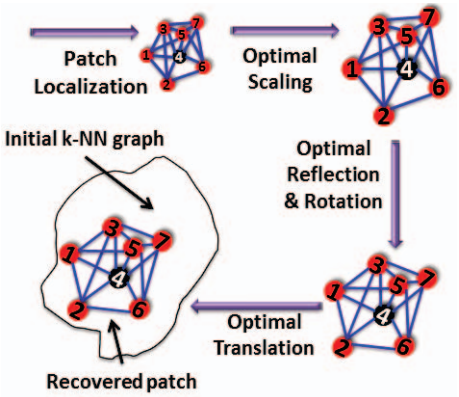

Fig. 1. ASAP and scale synchronization pipeline.

We estimate the optimal translation of each patch by solving, in a least squares sense, d overdetermined linear systems

$$x_i - x_j = x_i^{(k)} - x_j^{(k)}, \quad (i,j) \in E_k, \quad k = 1, \ldots, N, \quad (5)$$

where x_i, respectively $x_i^{(k)}$, denote the unknown location of node i we are solving for, respectively, the known location of node i in the embedding of patch P_k. We refer the reader to [9] for a description of computing the optimal translations.

3.4. Extension to higher dimensions

Although we present experiments here on 2D and 3D data, the ASAP approach extends naturally to higher dimensions. In the 3D case, ASAP has been recently used as a scalable robust approach to the molecule problem in structural biology [15]. For the d-dimensional general case, one can extend ASAP by first using the same approach for scaling synchronization from Section 3.2, then synchronizing over O(d), and finally estimating the optimal translations over R^d by solving d overdetermined systems of linear equations via least-squares. The LOE approach that can be used to obtain the local patch embeddings required by ASAP, has a natural extension to the d-dimensional case, thus rendering the entire pipeline amenable to dealing with higher-dimensional data.

4. EXPERIMENTS

Our experiments compare embeddings of points drawn from two different 2D synthetic densities: piecewise constant half-planes (**PC**), piecewise constant squares (**PCS**), each with $n = \{500, 5000\}$ points, as well as points drawn uniformly from a 3D donut shape (**Donut**) with $n = 500$, and the actual 2D coordinates of $n = 1101$ cities in the US (**US cities**). For a given set of data points, we use its kNN adjacency matrix as input to each ordinal embedding method. We test Laplacian Eigenmaps [16], the LOE BFGS and LOE MM methods [8], and ASAP with LOE BFGS used for the patch embeddings. As LOE was already compared with several methods in [8], attaining better performance than LOE may suggest better performance than a number of relevant methods including Kamada and Kawai [17], and Fruchterman and Reingold [18]. We remark that our approach deals with a different input than that of the **t-SNE** method in [19], which is generally used for embeddings of high dimensional data where some of the constraints are deliberately violated, which is not necessarily the case in our setting. We evaluate the methods based on (wall-clock) runtime and *A-error* ($\mathcal{E}_A$) defined as the percentage of edge disagreements between the kNN adjacency matrix of the proposed embeddeding $\tilde{X}$ and the ground truth

$$\text{error}(\tilde{X}, X): \quad \mathcal{E}_A \stackrel{\text{def}}{=} \frac{1}{n^2} \sum_{i,j=1}^{n} \left| \left(A^k_{\tilde{X}}\right)_{ij} - \left(A^k_X\right)_{ij} \right|, \quad (6)$$

where $A^k_X \in \{0,1\}^{n \times n}$ denotes the adjacency matrix of the corresponding kNN graph. We set varying limits on the number of LOE iterations $\{5, 10, 50, 100, 300, 500\}$, and we use varying maximum patch sizes (MPS) for ASAP . The LOE and ASAP methods give, for each distribution and values n and k, an error-runtime Pareto curve (with low values in both coordinates being best). In Table 2, we establish some shorthand notation for the methods and parameters used in this section. It is worth mentioning that while LOE BFGS and LOE MM are iterative methods which should converge to the best estimate of the solution as the number of iterations increases, ASAP is not iterative and the results of ASAP LOE with a given MPS, do not inform the results of ASAP LOE with another MPS. This aspect, combined with the randomized k-means spectral clustering used to choose patches means that we do not generally expect the recovery errors of ASAP LOE to be monotonically decreasing with MPS or time (as higher MPS generally leads to longer computational time).

Recovery Method	
LE	Laplacian Eigenmaps embedding
LOE MM	Local Ordinal Embedding using majorization minimization
LOE BFGS	Local Ordinal Embedding using BFGS
ASAP LOE	ASAP & LOE BFGS patch embeddings
Parameters	
sparse k	$k = \lceil 2\log(n) \rceil$
dense k	$k = \lceil \sqrt{n \log(n)} \rceil$
MPS	maximum patch size (for ASAP)
Iter.	number of iterations (of LOE)

Table 2. Notation for plotting experimental results.

First, to illustrate the importance of the scale synchronization introduced in Section 3.2, we compare in Figure 2 ASAP synchronized embeddings with and without this step. Clearly, this step significantly improves the recovered solutions.

We show A-error versus runtime for recovering $n = \{500, 5000\}$ points sampled from the PC and PCS distributions for the sparse, respectively dense, regime in Figure 3, respectively Figure 4. Even for lower values of n, we find that ASAP LOE is often either faster than or better-performing than LOE BFGS, or both. This seems to be especially true in the sparse k domain. This is partly due to the massively parallel embedding step in ASAP, which can take advantage of multiple cores as the problem scales.

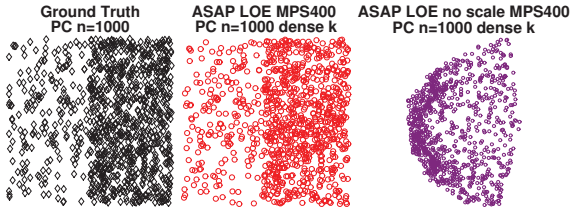

Fig. 2. Left: Ground truth, $n = 1000, k = 14$. Middle: ASAP LOE with scale synchronization: $\mathcal{E}_A = 0.007$. Right: ASAP LOE without scale synchronization: $\mathcal{E}_A = 0.038$.

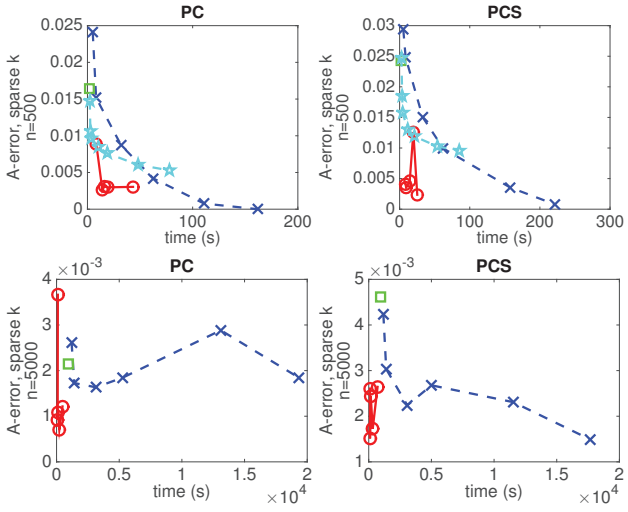

Fig. 3. A-error $\mathcal{E}_A$ vs. time, $n = \{500, 5000\}$, k sparse, ○ ASAP LOE, × LOE BFGS, □ LE , ⋆ LOE MM

To further illustrate how the methods perform, we plot the embeddings of $n = 1000$ point sampled from the 2D densities in Figure 5. In each case, the ASAP LOE with MPS 400 takes less time to run and yields smaller A-error errors than the LOE BFGS with 100 maximum iterations. We only run LOE MM for $n = 500$ because of difficulties we had when trying to get the provided R implementation to run on our Linux-based remote computing resource. We ran into no problems with the LOE BFGS implementation. The computers used have 12 CPU cores which are Intel(R) Xeon(R) X5650 @ 2.67GHz, and have 48GB ram. The R implementation of LOE does not (as far as the authors are aware) take advantage of multiple cores, and runs a single process on a single core. In contrast, our ASAP Matlab implementation uses the Multicore package to divide up the local embedding problems among the available cores.

In Table 3 we show the A-error $\mathcal{E}_A$ vs runtime for ASAP LOE on a data set of $n = 50,000$ points and $k = 22$. While this size is completely prohibitive for LOE BFGS, ASAP LOE produces good results in less than 4 hours.

To demonstrate that this approach is not limited to the 2D case, nor does it only perform well on synthetic data, we plot in Figure 6 the embeddings for points sampled from a 3D donut shape, and actual coordinates of $n = 1101$ US cities. In

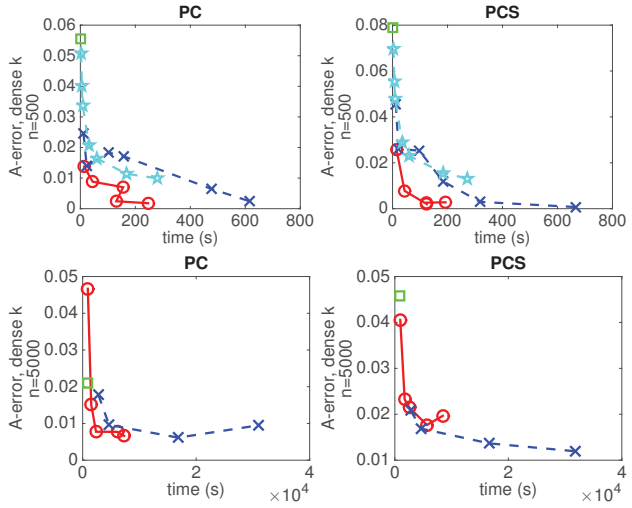

Fig. 4. A-error $\mathcal{E}_A$ vs. time, $n = \{500, 5000\}$, k dense, ○ ASAP LOE, × LOE BFGS, □ LE , ⋆ LOE MM

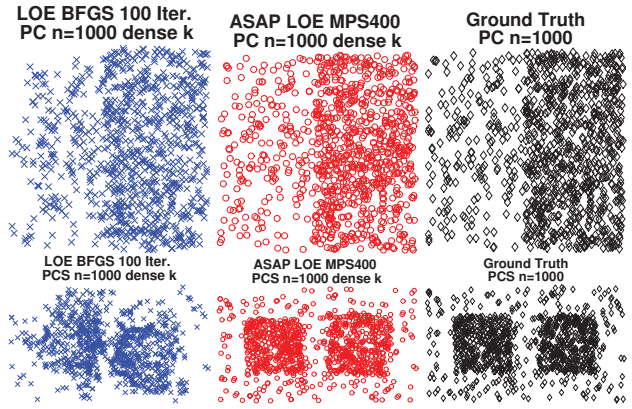

Fig. 5. Embeddings for the PC (top) and PCS (bottom) data sets with $n = 1000$, and k dense. Column 1: LOE BFGS Iter.=100. Column 2: ASAP LOE with MPS = 400 (with each ASAP result obtained is less time than the corresponding LOE result). Column 3: ground truth.

both cases, ASAP LOE with MPS 300 runs faster and yields smaller A-error than LOE BFGS with 500 maximum iterations, the latter of which produces twisted or folded results.

5. SUMMARY AND DISCUSSION

We have demonstrated that the computational efficiency of LOE for the kNN embedding problem can be significantly improved, while maintaining and often improving accuracy in a distributed setting. Our application of the divide-and-conquer ASAP method renders the problem significantly more tractable, distributing the embedding steps, and using fast spectral methods to combine them. We expect that such

MPS	100	300	500
PCS $\mathcal{E}_A$	5.1×10^{-4}	5.6×10^{-4}	1.9×10^{-4}
PC $\mathcal{E}_A$	5.8×10^{-4}	4.7×10^{-4}	3.0×10^{-4}

Table 3. Recovery results for $n = 50,000$ for ASAP LOE.

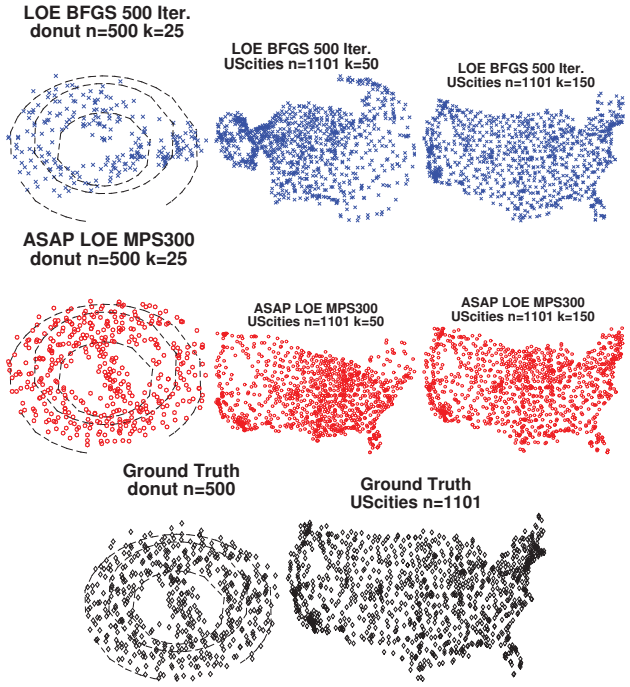

Fig. 6. Embeddings of Donut (3D) and US Cities (2D) data sets. Row 1: LOE BFGS Iter.=500. Row 2: ASAP LOE MPS=300 (with each ASAP result obtained in less time than the corresponding LOE result). Row 3: Ground truth.

improvements will make it possible to use kNN embeddings in a broader range of settings, and that the ASAP framework will be of independent interest to the machine learning community for tackling large geometric embedding problems. We refer the reader to [20] for an extended version of our work, that includes additional experiments, a linear programming formulation for kNN embeddings, and an approach for the density estimation problem based on Total-Variation Maximum Penalized Likelihood Estimation.

6. REFERENCES

[1] S. Agarwal, J. Wills, L. Cayton, G. Lanckriet, D. J. Kriegman, and S. Belongie, "Generalized non-metric multidimensional scaling," in *AISTATS*, 2007, pp. 11–18.

[2] U. von Luxburg and M. Alamgir, "Density estimation from unweighted k-nearest neighbor graphs: a roadmap," in *NIPS*, 2013, pp. 225–233.

[3] R. N. Shepard, "The analysis of proximities: Multidimensional scaling with an unknown distance function i," *Psychometrika*, vol. 27, no. 2, pp. 125–140, 1962.

[4] J. B. Kruskal, "Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis," *Psychometrika*, vol. 29, no. 1, pp. 1–27, 1964.

[5] J. B. Kruskal, "Nonmetric multidimensional scaling: a numerical method," *Psychometrika*, vol. 29, no. 2, pp. 115–129, 1964.

[6] B. Shaw and T. Jebara, "Structure preserving embedding," in *ICML*. ACM, 2009, pp. 937–944.

[7] B. McFee and G. R. Lanckriet, "Partial order embedding with multiple kernels," in *ICML*, 2009, pp. 721–728.

[8] Y. Terada and U. V. von Luxburg, "Local ordinal embedding," in *ICML*, 2014, pp. 847–855.

[9] M. Cucuringu, Y. Lipman, and A. Singer, "Sensor network localization by eigenvector synchronization over the Euclidean group," *ACM Trans. Sen. Netw.*, vol. 8, no. 3, pp. 19:1–19:42, Aug. 2012.

[10] U. von Luxburg and M. Kleindessner, "Uniqueness of ordinal embedding," in *COLT*, 2014, pp. 40–67.

[11] W. S. Torgerson, *Theory and methods of scaling.*, Wiley, 1958.

[12] J. B. Kruskal and M. Wish, *Multidimensional scaling*, vol. 11, Sage, 1978.

[13] A. Singer, "Angular synchronization by eigenvectors and semidefinite programming," *Appl. Comput. Harmon. Anal.*, vol. 30, no. 1, pp. 20–36, 2011.

[14] U. von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[15] M. Cucuringu, A. Singer, and D. Cowburn, "Eigenvector synchronization, graph rigidity and the molecule problem," *Information and Inference*, vol. 1, no. 1, pp. 21–67, 2012.

[16] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, June 2003.

[17] T. Kamada and S. Kawai, "An algorithm for drawing general undirected graphs," *Information processing letters*, vol. 31, no. 1, pp. 7–15, 1989.

[18] T. M. J. Fruchterman and E. M Reingold, "Graph drawing by force-directed placement," *Software: Practice and experience*, vol. 21, no. 11, pp. 1129–1164, 1991.

[19] L. Van der Maaten and K. Weinberger, "Stochastic triplet embedding," in *IEEE Workshop on Machine Learning for Signal Processing (MLSP)*, 2012, pp. 1–6.

[20] M. Cucuringu and J. Woodworth, "Point localization and density estimation from ordinal knn graphs using synchronization," *arXiv:1504.00722*.