Unsupervised Dimensionality Estimation and Manifold Learning in high-dimensional Spaces by Tensor Voting

Philippos Mordohai and Gérard Medioni
University of Southern California
email: {mordohai,medioni}@iris.usc.edu

Abstract

We address dimensionality estimation and nonlinear manifold inference starting from point inputs in high dimensional spaces using tensor voting. The proposed method operates locally in neighborhoods and does not involve any global computations. It is based on information propagation among neighboring points implemented as a voting process. Unlike other local approaches for manifold learning, the quantity propagated from one point to another is not a scalar, but is in the form of a tensor that provides considerably richer information. The accumulation of votes at each point provides a reliable estimate of local dimensionality, as well as of the orientation of a potential manifold going through the point. Reliable dimensionality estimation at the point level is a major advantage over competing methods. Moreover, the absence of global operations allows us to process significantly larger datasets. We demonstrate the effectiveness of our method on a variety of challenging datasets.

1 Introduction

A number of algorithms for manifold learning from unorganized points have been recently proposed in the machine learning literature. These include kernel PCA [Schölkopf *et al.*, 1998], locally linear embedding (LLE) [Roweis and Saul, 2000], Isomap [Tenenbaum *et al.*, 2000] and charting [Brand, 2003], that are briefly described in Section 2. They aim at reducing the dimensionality of the input space in a way that preserves some geometric or statistic properties of the data. Isomap, for instance, attempts to preserve the geodesic distances between all points as the manifold is "unfolded" and mapped to a space of lower dimension. A common assumption is that the desired manifold consists of locally linear patches. We relax this assumption by only requiring that manifolds be smooth almost everywhere. Smoothness in this context is the property of the manifold's orientation to vary gradually as one moves from point to point. This property is encoded in the votes that each point casts to its neighbors.

The representation of each point is in the form of a second order, symmetric, nonnegative definite tensor whose eigenvalues and eigenvectors fully describe the dimensionality and orientation of each point [Medioni *et al.*, 2000]. The votes are also tensors and their accumulation is a process that is considerably more powerful than the accumulation of scalars or vectors in terms of both the types of structure that can be described, as well as its robustness to noise. A point casts a vote to a neighboring point that carries an estimate of the orientation of the receiver given the orientation of the voter. Even in the case of unoriented inputs, meaningful votes can be cast as a function of the relative position of the two points. The result of voting is a new tensor at each point that encompasses the weighted contribution of the point's neighbors.

The dimensionality d of the manifold at each point in a D-dimensional space is indicated by the maximum gap in the eigenvalues of the tensor, while the top d eigenvectors span the normal space of the manifold. The first property makes dimensionality estimation a procedure with no parameters that require tuning. These estimates are reliable without averaging, as shown is Section 4, and allow us to handle data with varying dimensionality. This is arguably one of the most important contributions of our approach. Moreover, the presence of outliers, boundaries, intersections or multiple manifolds pose no additional difficulties, due to the unsupervised learning ability of our method. Furthermore, non-manifolds, such as hyper-spheres, can be inferred since we do not attempt to estimate a global "unfolding".

We do not attempt to estimate a mapping from the input space to the embedding space, as in [Brand, 2003] [Teh and Roweis, 2003]. Even though certain advantages can be derived from such a mapping, it is not always feasible, as in the case of hyper-spheres or multiple manifolds, and it is not necessary for many tasks, including interpolation. Valid paths on the manifold can be followed using the estimated tangent subspaces at the input points by projecting the desired direction on them and moving on the projection.

Since all operations are local, time complexity is $O(Dnlogn)$ where n is the number of points, enabling us to handle datasets with orders of magnitude more points than the current state of the art without incurring unreasonable computational cost. In terms of storage, the requirements at each point are $O(D^2)$. Our current implementation is probably limited to D in the order of a few hundreds. As all manifold learning methods, we operate under the assumption that the data are randomly sampled from the manifold in a way that is close to uniform, or at least does not favor certain parts or directions of the manifold.

The paper is organized as follows: the next section briefly reviews related work; Section 3 describes the tensor voting framework; Section 4 contains experimental results on classic datasets; finally, Section 5 concludes the paper and discusses the directions of our future work.

2 Related Work

In this section we briefly present recent approaches for learning low dimensional embeddings from points in high dimensional spaces. Most of them are extensions of linear techniques, such as Principal Component Analysis (PCA) and Multi-Dimensional Scaling (MDS), based on the assumption that nonlinear manifolds can be approximated by locally linear patches. In contrast to other methods, Schölkopf *et al.* [1998] propose kernel PCA that attempts to find linear patches using PCA in a space of typically higher dimensionality than the input space. Correct kernel selection can reveal the low dimensional structure of the input data. For instance a second order polynomial kernel can "flatten" quadratic manifolds.

Roweis and Saul [2000] address the problem by Locally Linear Embedding (LLE). Processing involves computing reconstruction weights for each point from its neighbors. The same weights should also reconstruct the point from its neighbors in the low dimensional embedding space, since the neighborhoods are assumed to be linear. The dimensionality of the embedding however has to be given as a parameter since it cannot always be estimated from the data [Saul and Roweis, 2003]. LLE assures that local neighborhood structure is preserved during dimensionality reduction.

Isomap Tenenbaum *et al.* [2000] approximates geodesic distances between points by graph distances. Then, MDS is applied on the geodesic instead of Euclidean distances to compute an embedding that forces nearby points to remain nearby and faraway points to remain that way. A variation of Isomap that can be applied to data with intrinsic curvature, but known distribution, and a faster alternative that only uses a few landmark point for distance computations have been proposed in [de Silva and Tenenbaum, 2003]. Isomap and its variants are limited to convex datasets.

Belkin and Niyogi [2003] compute the graph Laplacian of the adjacency graph of the input data, which is an approximation of the Laplace-Beltrami operator on the manifold, and exploit its locality preserving properties that were first observed in the field of clustering. The Laplacian eigenmaps algorithm can be viewed as a generalization of LLE, since the two become identical when the weights of the graph are chosen according to the criteria of the latter. The dimensionality of the manifold however cannot be reliably estimated from the data, as in most of the work reviewed here. Donoho and Grimes [2003] propose a similar scheme which computes the Hessian instead of the Laplacian. The authors claim that the Hessian is better suited for detecting linear patches on the manifold. The major contribution of this approach is that it proposes a global method, which, unlike Isomap, can be applied to non-convex datasets.

Weinberger and Saul [2004] address the problem of manifold learning by enforcing local isometry. The lengths of the sides of triangles of neighboring points are preserved during the embedding. These constraints can be expressed in terms of pairwise distances and the optimal embedding can be found by semi-definite programming. The method is more computationally demanding than LLE and Isomap, but can reliably estimate the underlying dimensionality of the inputs by locating the largest gap between the eigenvalues of the Gram matrix of the outputs. Similarly to our approach, this estimate does not require a threshold, as do approaches that estimate the residual error as a function of the dimensionality of the fitted model.

An algorithm that computes a mapping, and not just an embedding, of the data is described in [Brand, 2003]. The intrinsic dimensionality of the manifold is estimated by examining the rate of growth of the number of points contained in hyper-spheres as a function of the radius. Linear patches, areas of curvature and noise can be discriminated using the proposed measure. Affine transformations that align the coordinate systems of the linear patches are computed at the second stage. This defines a global coordinate system for the embedding and thus a mapping between the input space and the embedding space.

Finally, we briefly review approaches that estimate intrinsic dimensionality without any attempts to learn local structure. One such approach was presented in [Bruske and Sommer, 1998]. An optimally topology preserving map (OTPM) is constructed for a subset of the data, which is produced after vector quantization. Then PCA is performed for each node of the OTPM under the assumption that the data are locally linear. The average of the number of significant singular values at the nodes is the estimate of the intrinsic dimensionality. Kégl [2005] estimates the capacity dimension of the manifold, which does not depend on the distribution of the data and is equal to the topological dimension. An efficient approximation based on packing numbers is proposed. The algorithm takes into account dimensionality variations with scale and is based on a geometric property of the data, rather than successive projections to increasingly higher-dimensional subspaces until a certain percentage of the data is explained. Costa and Hero [2004] estimate the intrinsic dimension of manifold and the entropy of the samples using geodesic-minimal-spanning trees. The method is similar to Isomap and thus produces a single global estimate. Levina and Bickel [2005] compute maximum likelihood estimates of dimensionality by examining the number of neighbors included in spheres whose radius is selected in such a way that the density of the data can be assumed constant, and enough neighbors are included. These requirements cause an underestimation of the dimensionality when it is very high.

The difference between our approach and those of [Bruske and Sommer, 1998][Brand, 2003] [Kégl, 2005] [Weinberger and Saul, 2004] [Costa and Hero, 2004] and [Levina and Bickel, 2005] is that it produces reliable dimensionality estimates at the point level, which do not have to be averaged over the entire dataset. We are also able to estimate the local orientation of the manifold without being restricted to simple linear models.

3 Tensor Voting

In this section we first review the tensor voting framework [Medioni *et al.*, 2000], which can infer structures from sparse and noisy data via a local voting process. Results have been published mostly in 2- and 3-D domains, but the framework can be extended to higher dimensions, as in [Tang *et al.*, 2001]. We begin by describing the basics of the framework: data representation and voting. Then, we present the contribution of this paper to the methodology, which is an efficient implementation, in terms of both space and time, of tensor voting in high-dimensional spaces. We are able to operate in spaces where that seemed impractical or impossible based on the formulation of [Medioni *et al.*, 2000].

3.1 Data Representation

The representation at each point is in the form of a second order, symmetric, nonnegative definite tensor. The tensor can also be viewed as a matrix or an ellipsoid. It represents the **normal** space at the point. For instance, a hyper-plane has one normal $\vec{n}$, which can be encoded in tensor form as $\vec{n}\vec{n}^T$. A structure with a normal space of rank d has d normals and is represented by a tensor of the form:

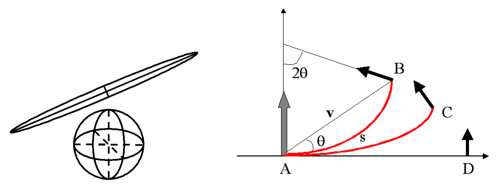
$$T = \sum_d \vec{n}_d \vec{n}_d^T \qquad (1)$$

A point without orientation information can be equivalently viewed as one having all possible normals and is encoded as the identity matrix. A tensor in this form represents an equal preference for all orientations and is called a "ball tensor", since the corresponding ellipsoid is a sphere or hyper-sphere. On the other hand, a tensor that contains only one orientation is called a "stick tensor". Stick tensors represent the hyper-plane of a D-dimensional space, which has one normal and $D-1$ tangents.

Depending on the type of structure the point belongs to, it has a different number of normals and tangents. The tensor's eigensystem encodes this information. Eigenvectors that belong to the tangent space correspond to zero eigenvalues, while those that belong to the normal space correspond to nonzero eigenvalues (typically equal to 1). Therefore, points with known orientation can be encoded in this representation by appropriately constructed tensors, as in (1). Points with unknown orientation are represented by identity matrices (ball tensors) See Fig. 1(a) for example tensors in 3-D.

On the other hand, given a second order, symmetric, nonnegative definite tensor, the type of structure encoded in it can be found by examining its eigensystem. Any tensor that has these properties can be decomposed as in the following equation:

$$\begin{aligned} T &= \sum_{d=1}^{D} \lambda_d \hat{e}_d \hat{e}_d^T = \\ &= (\lambda_1 - \lambda_2)\hat{e}_1\hat{e}_1^T + (\lambda_2 - \lambda_3)(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T) \\ &\quad + ... + \lambda_D(\hat{e}_1\hat{e}_1^T + \hat{e}_2\hat{e}_2^T + ... + \hat{e}_d\hat{e}_d^T) = \\ &\sum_{d=1}^{D-1}[(\lambda_d - \lambda d + 1)\sum_{k=1}^{d}\hat{e}_d\hat{e}_d^T] + \lambda_D(\hat{e}_1\hat{e}_1^T + ... + \hat{e}_d\hat{e}_d^T) \end{aligned} \qquad (2)$$



(a) Example tensors (b) Vote generation

Figure 1: Tensor Voting. (a) The shape of the tensor indicates the type of structure at the location, while its size indicates the confidence of this information. The top tensor has a strong preference of orientation and has higher confidence than the bottom one, which has no preference for any orientation. (b) Vote generation as a function of the relative position of a stick voter and the receiver and the orientation of the voter.

where λ_d are the eigenvalues in descending order and $\hat{e}_d$ are the corresponding eigenvectors. The tensor simultaneously encodes *all* possible types of structure. The confidence we have in the type that has d normals is encoded in the difference $\lambda_d - \lambda_{d+1}$.

3.2 The Voting Process

The tensor voting framework was designed to enforce constraints, such as proximity, co-linearity and co-curvilinearity with human perception in 2- and 3-D in mind. These constraints still hold in high dimensional spaces as long as one is looking for structures that are smooth almost everywhere. In this section we describe how information is propagated from one point, the voter, to another, the receiver.

During the voting process, each point casts votes to each of its neighboring points. The votes are also second-order, symmetric, nonnegative definite tensors. They encode the orientation the receiver would have, if the voter and receiver were in the same structure. We begin by examining the case of a voter that is associated with a stick tensor of unit length. The magnitude of a vote cast by the stick tensor decays with respect to the length of the smooth circular path connecting the voter and receiver. The circle was selected since it maintains constant curvature. It degenerates to a straight line if the vector connecting the voter and receiver is orthogonal to the normal of the voter. The orientation of the vote is towards the center of the circle defined by the two points and the orientation of the voter. The vote is generated according to the following equation and then transformed to the appropriate coordinate system.

$$\mathbf{S}(s,l,\theta) = e^{-(\frac{s^2+c\kappa^2}{\sigma^2})} \begin{bmatrix} -sin(2\theta) \\ cos(2\theta) \end{bmatrix} [-sin(2\theta) \;\; cos(2\theta)]$$

$$s = \frac{\theta\|\vec{v}\|}{sin(\theta)}, \qquad \kappa = \frac{2sin(\theta)}{\|\vec{v}\|} \qquad (3)$$

s is the length of the arc between the voter and receiver, and κ is its curvature (see Fig. 1(b)), σ is the scale of voting, and c is a constant. No vote is generated if the angle θ is more than 45^o. Also, the field is truncated to the extend where the magnitude of the vote is more than 3% of the magnitude of the voter.

Since tensor voting is a function of the position of the voter and the receiver and the orientation of the voter, vote generation by a stick tensor occurs in a 2-D subspace defined by the vector $\vec{v}$ that connects the two points and the normal of the voter. Therefore, stick voting is fully defined by Fig. 1(b) and Eq. 3, regardless of the dimension of the input space. According to [Medioni *et al.*, 2000], the votes cast by other types of tensors were pre-computed and stored in look-up tables, called voting fields. The presence of a normal space of rank more than one is simulated by a rotating stick tensor that spans the space. Since no closed form solution exists, the integration is numerically approximated. This process becomes impractical as the dimensionality of the space increases. Space requirements for storing D D-dimensional voting fields with k samples per axis are $O(Dk^D)$. At the same time, the advantage of re-using pre-computed votes vanishes as the dimensionality increases. We propose instead a novel, efficient way to directly compute votes. Since integration is infeasible, we seek an approximate solution.

3.3 Efficient Voting Scheme

The new scheme, proposed here, generates votes that contain the orientation information that should be communicated to the receiver, but not the uncertainty. The latter, in the original framework, was conveyed by the ball component of the vote that was due to the integration of votes cast by the rotating stick tensor. For instance, according to [Medioni *et al.*, 2000], a voting tensor that encodes a curve in 3-D (that has two normals and a tangent) casts a vote that has a dominant curve component and also some uncertainty. Here, we propose to propagate only the curve orientation at the receiver given the orientation of the voter. The uncertainty in the new formulation comes only from the accumulation of votes that are not perfectly aligned.

Let $\vec{v}$ be the vector connecting the voting and receiving points. It can be decomposed into $\vec{v}_t$ in the tangent space of the voter (null in the case of a ball voter) and $\vec{v}_n$ in the normal space. The new vote generation process is based on the observation that curvature in Eq. 3 is not a factor when θ is zero, or in other words, if the voting stick is orthogonal to $\vec{v}_n$. We can exploit this by defining a new basis for the normal space of the voter that includes $\vec{v}_n$. Then, the vote is constructed as the tensor addition of the votes cast by stick tensors parallel to the new basis vectors. Among those votes, only the one generated by the stick tensor parallel to $\vec{v}_n$ is not parallel to the normal space of the voter. See Fig. 2 for an illustration in 3-D. This scheme is applicable without changes in the case of ball voters. Since two unoriented points define a straight line in any space, the vote from a ball tensor should have $D-1$ normals and one tangent. Since $\vec{v}_t = \vec{0}$, the tensor parallel to $\vec{v}_n$ does not cast a vote due to the 45^o cut-off rule. The remaining $D-1$ tensors in the new basis cast votes that are equal in magnitude and span the space orthogonal to $\vec{v}$, and thus represent a straight line connecting the two points.

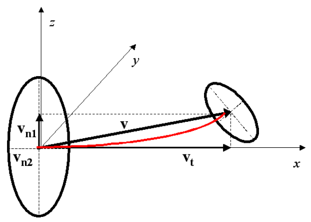


Figure 2: New scheme for vote generation. The voter here is a tensor with two normals in 3-D. The vector connecting the voter and receiver is decomposed into $\vec{v}_n$ and $\vec{v}_t$ that lie in the normal and tangent space of the voter respectively. A new basis that includes $\vec{v}_n$ is defined for the normal space and each basis component casts a stick vote. Only the vote generated by the orientation parallel to $\vec{v}_n$ is not parallel to the normal space. Tensor addition of the stick votes produces the combined vote.

Tensors with unequal eigenvalues are decomposed before voting according to Eq. 2. Then, each component votes separately and the vote is weighted by $\lambda_d - \lambda_{d+1}$, except the ball component whose vote is weighted by λ_D. This implementation of tensor voting is more efficient in terms of space, since there is no need to store voting fields, and in terms of time since we have devised a direct method for computing votes that replaces the numerical integration of [Medioni *et al.*, 2000]. The new computation requires at most D computations of a stick vote according to Eq. 3, followed by the addition of all the votes, instead of integration over D dimensions. Experiments in 2- and 3-D, that cannot be included here due to space limitations, validate the accuracy of the new vote generation scheme.

3.4 Vote Analysis

Votes are accumulated at each point by tensor addition, which is equivalent to the addition of matrices. After voting is completed, the eigensystem of the new tensor is analyzed and the tensor is decomposed as in Eq. 2. In 3-D, the inference of the type of structure that a point belongs is accomplished as follows. The difference between the two largest eigenvalues encodes surface confidence, with a surface normal given by $\vec{e}_1$. The difference between the second and third eigenvalue encodes curve confidence, with a the normals to the curve being $\vec{e}_1$ and $\vec{e}_2$. The smallest eigenvalue encodes junction confidence. Outliers receive little and inconsistent support from their neighborhood and can be identified by their low eigenvalues. Generalization to higher dimensions is straightforward, considering that there are D manifold types in D dimensions. The dimensionality d is estimated by finding the maximum difference $\lambda_d - \lambda_{d+1}$.

4 Experimental Results

In this section we present experimental results in dimensionality estimation and local structure inference. All inputs consist of un-oriented points and are encoded as ball tensors.

Swiss Roll The first experiment is on the "Swiss Roll" dataset, which is available at http://isomap.stanford.edu/. It contains $20,000$ points on a 2-D manifold in 3-D (Fig. 3). We perform a simple evaluation of the quality of the orientation estimates by projecting the nearest neighbors of each

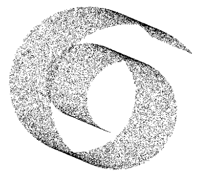


Figure 3: The "Swiss Roll" dataset in 3-D

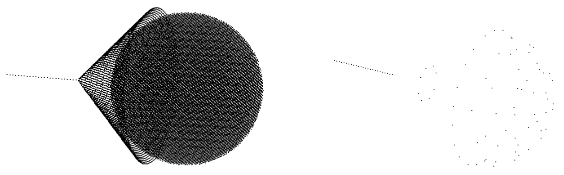
point on the estimated tangent space and measuring the percentage of the distance that has been recovered. The accuracy of dimensionality estimation at each point, the percentage of recovered distances for the 8 nearest neighbors and execution times on a Pentium 4 at 2.8 GHz, as a function of σ, can be seen in Table 1. Performance is the same at boundaries. The number of votes cast by each point ranges from 187 for $\sigma^2 = 50$ to 5440 for $\sigma^2 = 1000$. The accuracy is high for a large range of σ.

σ^2	Correct Dim. (%)	Dist. Rec. (%)	Time (sec)
50	93.10	91.75	11
100	99.24	93.07	22
200	99.91	93.21	50
300	99.96	93.20	81
500	99.95	93.18	144
700	99.88	93.13	202
1000	99.67	93.02	307

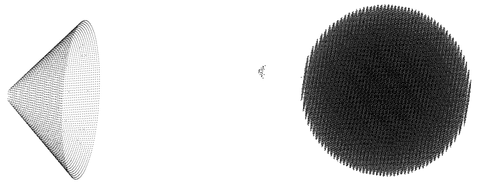
Table 1: Rate of correct dimensionality estimation and execution times as functions of σ for the "Swiss Roll" dataset

Structures with varying dimensionality The second dataset is in 4-D and contains points sampled from three structures: a line, a 2-D cone and a 3-D hyper-sphere. The hyper-sphere is a structure with three degrees of freedom. It cannot be unfolded unless we remove a small part from it. Figure 4(a) shows the first three dimensions of the data. The dataset contains a total $135,864$ points, voting is performed with $\sigma^2 = 200$ and takes 2 hours and 7 minutes. Figures 4(c-d) show the points classified according to their dimensionality. Performing the same analysis as above for the accuracy of the tangent space estimation, 91.04% of the distances of the 8 nearest neighbors of each point lie on the tangent space, even though both the cone and the hyper-sphere have intrinsic curvature and cannot be accurately approximated by linear models. All the methods presented in Sec. 2 would fail for this dataset because of the presence of structures with different dimensionalities and because the hyper-sphere cannot be unfolded.

Data in high dimensions The datasets for this experiment were generated by sampling a few thousand points with low intrinsic dimensionality (3 or 4) and mapping them to a medium dimensional space (14- to 16-D) using linear and quadratic functions. The generated points were then rotated and embedded in a 50- to 150-D space while uniform noise was added to all dimensions. The accuracy of dimensionality estimation after tensor voting can be seen in Table 2.



(a) Input (b) 1-D points



(c) 2-D points (d) 3-D points

Figure 4: Data of varying dimensionality in 4-D. The first three axes of the input and the classified points are shown.

Intrinsic Dim.	Linear Mappings	Quadratic Mappings	Space Dim.	Dim. Est. (%)
4	10	6	50	93.6
3	8	6	100	97.4
4	10	6	100	93.9
3	8	6	150	97.3

Table 2: Rate of correct dimensionality estimation for high dimensional data

5 Conclusions

We have presented an approach to manifold learning that offers certain advantages over the state-of-the-art. First, we are able to obtain accurate estimates of the intrinsic dimensionality at the point level. Moreover, since the dimensionality is found as the maximum gap in the eigenvalues of the tensor at the point, no thresholds are needed. In most other approaches the dimensionality has to be provided, or, at best, an average intrinsic dimensionality is estimated for the entire dataset, as in [Bruske and Sommer, 1998] [Brand, 2003] [Kégl, 2005] [Weinberger and Saul, 2004] [Costa and Hero, 2004]. Second, even though tensor voting on the surface looks like a local covariance estimation, the fact that the votes are tensors and not scalars reveals more information for each point. The properties of the tensor representation, which can handle the simultaneous presence of multiple orientations, allow the reliable inference of the normal and tangent space at each point. Due to its unsupervised nature, tensor voting is very robust against outliers, as demonstrated in [Medioni *et al.*, 2000] for the 2- and 3-D case. This property holds in higher dimensions, where random noise is even more scattered. Lack of space did not allow us to include outlier rejection results here.

An important advantage of tensor voting is the absence of global computations, which makes time complexity $O(Dnlogn)$. This enables us to process datasets with very large number of points. Computation time scales linearly with the number of points assuming that more points are added to the dataset in a way that the density remains constant. In this case, the number of votes cast per point remains constant and time requirements grow linearly. Complexity is adversely affected by the dimensionality of the space D, since eigen-decompositions of $D \times D$ tensors have to be performed. For most practical purposes, the number of points has to be considerably larger than the dimensionality of the space $(n \gg D)$ to allow structure inference. Computational complexity, therefore, is reasonable with respect to the largest parameter.

It should also be noted that the votes attenuate with distance and curvature. This is a more intuitive formulation than using the N nearest neighbors with equal weights, since some of them may be too far or belong to a different part of the structure. For both the ϵ distance and the N nearest neighbor formulations, however, the distance metric in the input space has to be meaningful. This is also the case for all the methods presented in Sec. 2.

The representation by tensors allows us to deal with structures that are not necessarily manifolds. These include non-manifolds, such as hyper-spheres, intersecting structures and datasets containing structures of different dimensionality. A mapping from the input to the embedding space cannot be computed in any of these cases, but it is not necessary for many tasks including interpolation, function approximation and local orientation (gradient) estimation. One can also view our algorithm as a preprocessing stage that facilitates further processing. Its outputs can be used as inputs for tasks such as dimensionality reduction and mapping to a low dimensional space, if desired.

Two issues require further investigation. The first is a more thorough analysis of the required properties of the data distribution. In line with other methods for manifold learning ([Roweis and Saul, 2000][Tenenbaum *et al.*, 2000][Brand, 2003]), we assume that the data are distributed regularly is space. A uniform distribution is ideal but not necessary, as shown in Sec. 4, especially in the last experiment where the points are mapped to the high dimensional space nonlinearly. The second issue is that of data sufficiency for structure inference. As the dimensionality of the space and the intrinsic dimensionality of the structure grow, the number of points necessary to infer the latter correctly also grows. We intend to investigate the minimum requirements in terms of the number of points for inferring a certain type of structure in a space of high dimensions.

Acknowledgement

This research has been supported by the U.S. National Science Foundation grant IIS 03 29247.

References

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 15*, pages 961–968. MIT Press, Cambridge, MA, 2003.

J. Bruske and G. Sommer. Intrinsic dimensionality estimation with optimally topology preserving maps. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(5):572–575, May 1998.

J. Costa and A.O. Hero. Geodesic entropic graphs for dimension and entropy estimation in manifold learning. *IEEE Trans. on Signal Process.*, 52(8):2210–2221, Aug. 2004.

V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, Cambridge, MA, 2003.

D. Donoho and C. Grimes. Hessian eigenmaps: new tools for nonlinear dimensionality reduction. In *Proceedings of National Academy of Science*, pages 5591–5596, 2003.

B. Kégl. Intrinsic dimension estimation using packing numbers. In *Advances in Neural Information Processing Systems 15*, pages 681–688. MIT Press, Cambridge, MA, 2005.

E. Levina and P. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems 17*, pages 777–784. MIT Press, Cambridge, MA, 2005.

G. Medioni, M.S. Lee, and C.K. Tang. *A Computational Framework for Segmentation and Grouping*. Elsevier, New York, 2000.

S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

B. Schölkopf, A.J. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

C.K. Tang, G. Medioni, and M.S. Lee. N-dimensional tensor voting and application to epipolar geometry estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(8):829–844, August 2001.

Y.W. Teh and S. Roweis. Automatic alignment of local representations. In *Advances in Neural Information Processing Systems 15*, pages 841–848, Cambridge, MA, 2003. MIT Press.

J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

K.Q. Weinberger and L.K. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *Proc. Int. Conf. on Computer Vision and Pattern Recognition*, pages II: 988–995, 2004.