LLE with low-dimensional neighborhood representation

No Author Given

No Institute Given

Abstract. The local linear embedding algorithm (LLE) is a non-linear dimension-reducing technique, widely used due to its computational simplicity and intuitive approach. LLE first linearly reconstructs each input point from its nearest neighbors and then preserves these neighborhood relations in the low-dimensional embedding. We show that the reconstruction weights computed by LLE capture the *high*-dimensional structure of the neighborhoods, and not the *low*-dimensional manifold structure. Consequently, the weight vectors are highly sensitive to noise. Moreover, this causes LLE to converge to a *linear* projection of the input, as opposed to its *non-linear* embedding goal. To overcome both of these problems, we propose to compute the weight vectors using a low-dimensional neighborhood representation. We present numerical examples demonstrating both the perturbation and linear projection problems, and the improved outputs using the low-dimensional neighborhood representation.

1 Introduction

The local linear embedding algorithm (LLE) [1] belongs to a class of recently developed, non-linear dimension-reducing algorithms that include Isomap [2], Laplacian Eigenmap [3], Hessian Eigenmap [4], LTSA [5], and MVU [6]. This group of algorithms assumes that the data is sitting on, or next to, an embedded manifold of low dimension within the original high-dimensional space, and attempts to find an embedding that maps the input points to the lower-dimensional space. Here a manifold is defined as a topological space that is locally equivalent to an Euclidean space. LLE was found to be useful in data visualization [1, 7], and image processing applications, such as image denoising [8] and human face detection [9]. It is also applied in different fields of science such as chemistry [10], biology [11], and astrophysics [12].

LLE attempts to recover the domain structure of the input data set in three steps. First, LLE assigns neighbors to each input point. Second, for each input point LLE computes weight vectors that best linearly reconstruct the input point from its neighbors. Finally, LLE finds a set of low-dimensional output points that minimize the sum of reconstruction errors, under some normalization constraints.

In this paper we focus on the computation of the weight vectors in the second step of LLE. We show that LLE's neighborhood description captures the structure of the *high*-dimensional space, and not that of the *low*-dimensional

domain. We show two main consequences of this observation. First, the weight vectors are highly sensitive to noise. This implies that a small perturbation of the input may yield an entirely different embedding. Second, we show that LLE converges to a linear projection of the high-dimensional input when the number of input points tends to infinity. Numerical results that demonstrate our claims are provided.

To overcome these problems, we suggest a simple modification to the second step of LLE, *LLE with low-dimensional neighborhood representation*. Our approach is based on finding the best low-dimensional representation for the neighborhood of each point, and then computing the weights with respect to these low-dimensional neighborhoods. This proposed modification preserves LLE's principle of reconstructing each point from its neighbors. It is of the same computational complexity as LLE and it removes the need to use regularization when the number of neighbors is greater than the input dimension.

We proved that the weights computed by LLE with low-dimensional neighborhood representation are robust against noise. We also proved that when using the modified LLE on input points sampled from an isometrically embedded manifold, the pre-image of the input points achieves a low value of the objective function. The theorems and proofs are omitted, due to lack of space, and will presented elsewhere. We demonstrate an improvement in the output of LLE when using the low-dimensional neighborhood representation for several numerical examples.

There are other works that suggest improvements for LLE. The Efficient LLE [13] and the Robust LLE [14] algorithms both address the problem of outliers by preprocessing the input data. Other versions of LLE, including ISOLLE [15] and Improved LLE [16], suggest different ways to compute the neighbors of each input point in the first step of LLE. The Modified LLE algorithm [17] proposes to improve LLE by using multiple local weight vectors in LLE's second step, thus characterizing the high-dimensional neighborhood more accurately. All of these algorithms attempt to characterize the *high*-dimensional neighborhoods, and not the *low*-dimensional neighborhood structure.

Other algorithms can be considered variants of LLE. Laplacian Eigenmap essentially computes the weight vectors using regularization with a large regularization constant (see discussion on the relation between LLE and Laplacian Eigenmap in [3], Section 5). Hessian Eigenmap [4] characterizes the local input neighborhoods using the null space of the local Hessian operator, and minimizes the appropriate function for the embedding. Closely related is the LTSA algorithm [5], which characterizes each local neighborhood using its local PCA. These last two algorithms attempt to describe the low-dimensional neighborhood. However, these algorithms, like Laplacian Eigenmap, do not use LLE's intuitive approach of reconstructing each point from its neighbors. Our proposed modification provides a low-dimensional neighborhood description while preserving LLE's intuitive approach.

The paper is organized as follows. The description of LLE is presented in Section 2. The discussion of the second step of LLE appears in Section 3. The

suggested modification of LLE is presented in Section 4. In Section 5 we present numerical examples.

2 Description of LLE

The input data $X = \{x_1, \ldots, x_N\}$, $x_i \in \mathbb{R}^D$ for LLE is assumed to be sitting on or next to a d-dimensional manifold $\mathcal{M}$. We refer to X as an $N \times D$ matrix, where each row stands for an input point. The goal of LLE is to recover the underlying d-dimensional structure of the input data X. LLE attempts to do so in three steps.

First, LLE assigns neighbors to each input point x_i. This can be done, for example, by choosing the input point's K-nearest neighbors based on the Euclidian distances in the high-dimensional space. Denote by $\{\eta_j\}$ the neighbors of x_i. Let the neighborhood matrix of x_i be denoted by X_i, where X_i is the $K \times D$ matrix with rows $\eta_j - x_i$.

Second, LLE computes weights w_{ij} that best linearly reconstruct x_i from its neighbors. These weights minimize the reconstruction error function

$$\varepsilon_i(w_i) = \|x_i - \sum_j w_{ij} x_j\|^2 , \tag{1}$$

where $w_{ij} = 0$ if x_j is not a neighbor of x_i, and $\sum_j w_{ij} = 1$. With some abuse of notation, we will also refer to w_i as a $K \times 1$ vector, where we omit the entries of w_i for non-neighbor points. Using this notation, we may write $\varepsilon_i(w_i) = w_i' X_i X_i' w_i$.

Finally, given the weights found above, LLE finds a set of low-dimensional output points $Y = \{y_1, \ldots, y_N\} \in \mathbb{R}^d$ that minimize the sum of reconstruction errors

$$\Phi(Y) = \sum_{i=1}^{n} \|y_i - \sum_j w_{ij} y_j\|^2 , \tag{2}$$

under the normalization constraints $Y'\mathbf{1} = 0$ and $Y'Y = I$, where $\mathbf{1}$ is vector of ones. These constraints force a unique minimum of the function Φ.

The function $\Phi(Y)$ can be minimized by finding the d-bottom non-zero eigenvectors of the sparse matrix $(I-W)'(I-W)$, where W is the matrix of weights. Note that the p-th coordinate ($p = 1, \ldots, d$), found simultaneously for all output points y_i, is equal to the eigenvector with the p-smallest non-zero eigenvalue. This means that the first p coordinates of the LLE solution in q dimensions, $p < q$, are exactly the LLE solution in p dimensions [1, 18]. Equivalently, if an LLE output of dimension q exists, then a solution for dimension p, $p < q$, is merely a linear projection of the q-dimensional solution on the first p dimensions.

When the number of neighbors K is greater than the dimension of the input D, each data point can be reconstructed perfectly from its neighbors, and the local reconstruction weights are no longer uniquely defined. In this case, regularization is needed and one needs to minimize

$$\varepsilon_i^{\mathrm{reg}}(w_i) = \|x_i - \sum_j w_{ij} x_j\|^2 + \delta \|w_i\|^2 . \tag{3}$$

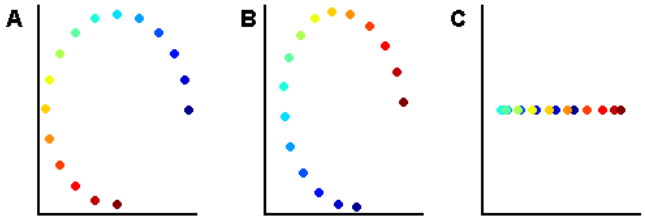

Fig. 1. The input for LLE is the 16-point open ring that appears in (A). The two-dimensional output of LLE is given in (B). LLE finds and preserves the two-dimensional structure of each of the local neighborhoods. The one-dimensional output of LLE appears in (C). The computation was performed using 4-nearest-neighbors, and regularization constant $\Delta = 10^{-9}$.

where δ is a small constant. Saul and Roweis [18] suggested $\delta = \frac{\Delta}{K}\text{trace}(X_i X_i')$ with $\Delta \ll 1$. Regularization can be problematic for the following reasons. When the regularization constant is not small enough, it was shown by Zhang and Wang [17] that the correct weight vectors cannot be well approximated by the minimizer of $\varepsilon_i^{\text{reg}}(w_i)$. Moreover, when the regularization constant is relatively high, it produces weight vectors that tend towards the uniform vectors $w_i = (1/K, \ldots, 1/K)$. Consequently, the solution for LLE with large regularization constant is close to that of Laplacian Eigenmap, and does not reflect a solution based on reconstruction weight vectors (see [3], Section 5). However, in real-world data the dimension of the input is typically greater than the number of neighbors. Hence regularization is usually unnecessary.

3 Preservation of high-dimensional neighborhood structure by LLE

In this section we focus on the computation of the weight vectors, which is performed in the second step of LLE. We first show that LLE characterizes the *high*-dimensional structure of the neighborhood. We explain how this can lead to the failure of LLE in finding a meaningful embedding of the input. Two additional consequences of preservation of the high-dimensional neighborhood structure are discussed. First, LLE's weight vectors are sensitive to noise. Second, LLE's output tends toward a linear projection of the input data when the number of input points tends to infinity. These claims are demonstrated using numerical examples.

We begin by showing that LLE preserves the high-dimensional neighborhood structure. We use the example that appears in Fig 1. The input is a sample from an open ring which is a one-dimensional manifold embedded in $\mathbb{R}^2$. For each point on the ring, we define its neighborhood using its 4 nearest neighbors. Note that its *high*-dimensional ($D = 2$) neighborhood structure is curved, while the

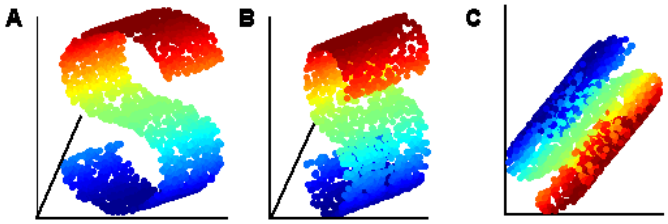

Fig. 2. (A) LLE's input, a 2000-point 'S' curve. (B) The three-dimensional output of LLE. It can be seen that LLE finds the overall three-dimensional structure of the input. (C) The two-dimensional output of LLE.

low-dimensional structure ($d = 1$) is a straight line. The two-dimensional output of LLE (see Fig. 1) is essentially a reconstruction of the input. In other words, LLE's weight vectors preserve the curved shape of each neighborhood.

The one-dimensional output of the open ring is presented in Fig 1C. Recall that the one-dimensional solution is a linear projection of the two-dimensional solution, as explained in section 2. In the open-ring example, LLE clearly fails to find an appropriate one-dimensional embedding, because it preserves the two-dimensional curved neighborhood structure. We now show that this is also true for additional examples.

The 'S' curve input data appears in Fig 2A. Fig 2B shows that the overall three-dimensional structure of the 'S' curve is preserved in the three-dimensional embedding. The two-dimensional output of LLE appears in Fig 2C. It can be seen that LLE does not succeed in finding a meaningful embedding in this case. Fig 3 presents the swissroll, with similar results.

We performed LLE, here and in all other examples, using the LLE Matlab code as it appears on the LLE website [19]. [1] The code that produced the input data for the 'S' curve and the swissroll was also taken from the LLE website. We used the default values of 2000-point samples and 12-nearest-neighbors. For the regularization constant we used $\Delta = 10^{-9}$. It should be noted that using a large regularization constant improved the results. However, as discussed in Section 2, the weight vectors produced in this way do not reflect a solution that is based on reconstruction weight vectors. Instead the vectors tend toward the uniform vector.

We now discuss the sensitivity of LLE's weight vectors $\{w_i\}$ to noise. Figure 4 shows that an arbitrarily small change in the neighborhood can cause a large change in the weight vectors. This result can be understood by noting how the vector w_i is obtained. It can be shown [18] that w_i equals $(X_i X_i')^{-1}\mathbf{1}$, up to normalization. Sensitivity to noise is therefore expected when the condition number of $X_i X_i'$ is large (see [20], Section 2). One way to solve this problem

[1] The changes in the Matlab function *eigs* were taken into account

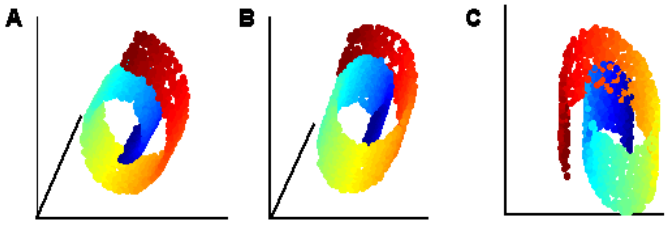

Fig. 3. (A) LLE's input, a 2000-point swissroll. (B) The three-dimensional output of LLE. It can be seen that LLE finds the overall three-dimensional structure of the input. (C) The two-dimensional output of LLE.

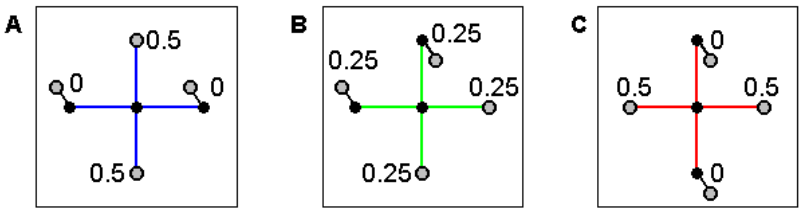

Fig. 4. The effect of a small perturbation on the weight vector computed by LLE. All three panels show the same unperturbed neighborhood, consisting of a point and its four nearest-neighbors (black points), all sitting in the two-dimensional plane. Each panel shows a different small perturbation of the original neighborhood (gray points). All perturbations are in the direction orthogonal to the plane of the original neighborhood. (A) and (C): Both perturbations are in the same direction. (B) Perturbations are of equal size, in opposite directions. The unique weight vector for the center point is denoted for each case. These three different weight vectors vary widely, even though the different perturbations can be arbitrarily small.

is to enforce regularization, with its associated problems (see section 2). In the next section we suggest a simple alternative solution to the sensitivity of LLE to noise.

One more implication of the fact that LLE preserves the high-dimensional neighborhood structure is that LLE's output tends to a linear projection of the input data. Wu and Hu [21] proved for a finite data set that when the reconstruction errors are exactly zero for each of the neighborhoods, and under some dimensionality constraint, the output of LLE must be a linear projection of the input data. Here, we present a simple argument that explains why LLE's output tends to a linear projection when the number of input points tends to infinity, and show numerical examples that strengthen this claim. For simplicity, we assume that the input data is normalized.

Our argument is based on two claims. First, note that LLE's output for dimension d is a linear-projection of LLE's output for dimension D (see Section 2). Second, note that by definition, the LLE output is a set of points Y that minimizes the sum of reconstruction errors $\Phi(Y)$. For normalized input X of dimension D, when the number of input points tends to infinity, each point is well reconstructed by its neighboring points. Therefore the reconstruction error $\varepsilon_i(w)$ tends to zero for each point x_i. This means that the input data X tends to minimize the sum of reconstruction errors $\Phi(Y)$. Hence, the output points Y of LLE for output of dimension D tend to the input points (up to a rotation). The result of these two claims is that any requested solution of dimension $d < D$ tends to a linear projection of the D-dimensional solution, i.e., a linear projection of the input data.

The result that LLE tends to a linear projection is of asymptotical nature. However, numerical examples show that this phenomenon can occur even when the number of points is relatively small. This is indeed the case for the outputs of LLE shown in Figs. 1C, 2C, and 3C, for the open ring, the 'S' curve, and the swissroll, respectively.

4 Low-dimensional neighborhood representation for LLE

In this section we suggest a simple modification of LLE that computes the low-dimensional structure of the input points' neighborhoods. Our approach is based on finding the best representation of rank d (in the l_2 sense) for the neighborhood of each point, and then computing the weights with respect to these d-dimensional neighborhoods. In Section 5 we show numerical examples that justify our suggested modification.

We begin by finding a rank-d representation for each local neighborhood. Recall that X_i is the $K \times D$ neighborhood matrix of x_i, whose j-th row is $\eta_j - x_i$, where η_j is the j-th neighbor of x_i. We assume that the number of neighbors K is greater than d, since otherwise x_i cannot (in general) be reconstructed by its neighbors. We say that X_i^P is the best rank-d representation of X_i, if X_i^P minimizes $\left\|X_i - Y\right\|_2$ over all the $K \times D$ matrices Y of rank d. Let ULV' be the SVD of X_i, where U and V are orthogonal matrices of size $K \times K$ and $D \times D$, respectively, and L is a $K \times D$ matrix, where $L_{jj} = \lambda_j$ are the singular values of X_i for $j = \min(K, D)$, ordered from the largest to the lowest, and $L_{ij} = 0$ for $i \neq j$. We denote

$$U = \left(U_1, U_2\right) \; ; \; L = \begin{pmatrix} L_1, & 0 \\ 0, & L_2 \end{pmatrix} \; ; \; V = \left(V_1, V_2\right) \tag{4}$$

where $U_1 = (u_1, \ldots, u_d)$ and $V_1 = (v_1, \ldots, v_d)$ are the first d columns of U and V, respectively, U_2 and V_2 are the last $K - d$ and $D - d$ columns of U and V respectively, and L_1 and L_2 are of dimension $d \times d$ and $(K - d) \times (D - d)$, respectively. Then by Corollary 2.3-3 of [20], X_i^P can be written as $U_1 L_1 V_1'$.

We now compute the weight vectors for the d-dimensional neighborhood X_i^P. By (1), we need to find w_i that minimize $w_i' X_i^P X_i^{P'} w_i$ (see Section 2). The

solution for this minimization problem is not unique, since by the construction all the vectors spanned by $u_{d+1}, \ldots, u_K$ zero this function. Thus, our candidate for the weight vector is the vector in the span of $u_{d+1}, \ldots, u_K$ that has the smallest l_2 norm. In other words, we are looking for

$$\underset{\substack{w_i \in \text{span}\{u_{d+1},\ldots,u_K\} \\ w_i'\mathbf{1}=1}}{\text{argmin}} \quad \|w_i\|^2 \,. \tag{5}$$

Note that we implicitly assume that $\mathbf{1} \notin \text{span}\{u_1, \ldots, u_d\}$. This is true whenever the neighborhood points are in general position, i.e., no $d+1$ of them lie in a $(d-1)$-dimensional plane. To understand this, note that if $\mathbf{1} \in \text{span}\{u_1, \ldots, u_d\}$ then $(I - \frac{1}{K}\mathbf{1}\mathbf{1}')X_i^P = (I - \frac{1}{K}\mathbf{1}\mathbf{1}')U_1 L_1 V_1'$ is of rank $d-1$. Since $(I - \frac{1}{K}\mathbf{1}\mathbf{1}')X_i$ is the projected neighborhood after centering, we obtained that the dimension of the centered projected neighborhood is of dimension $d-1$, and not d as assumed, and therefore the points are not in general position.

The following Lemma shows how to compute the vector w_i that minimizes (5).

Lemma 1. *Assume that the points of X_i^P are in general position. Then the vector w_i that minimizes* (5) *is given by*

$$w_i = \frac{U_2 U_2' \mathbf{1}}{\mathbf{1}' U_2 U_2' \mathbf{1}} \,. \tag{6}$$

The proof is based on Lagrange multipliers.

Following Lemma 1, we propose a simple modification for LLE based on computing the reconstruction vectors using a d-dimensional neighborhood representation.

Algorithm:
LLE with low-dimensional neighborhood representation

Input: X, an $N \times D$ matrix.
Output: Y, an $N \times d$ matrix.

Procedure:

1. For each point x_i find K-nearest-neighbors and compute the neighborhood matrix X_i.
2. For each point x_i compute the weight vector w_i using the d-dimensional neighborhood representation:
   - Use the SVD decomposition to write $X_i = ULV'$.
   - Write $U_2 = (u_{d+1} \ \ldots, u_K)$.
   - Compute
   $$w_i = \frac{U_2 U_2' \mathbf{1}}{\mathbf{1}' U_2 U_2' \mathbf{1}} \,.$$
3. Compute the d-dimension embedding by minimizing $\Phi(Y)$ (see (2)).

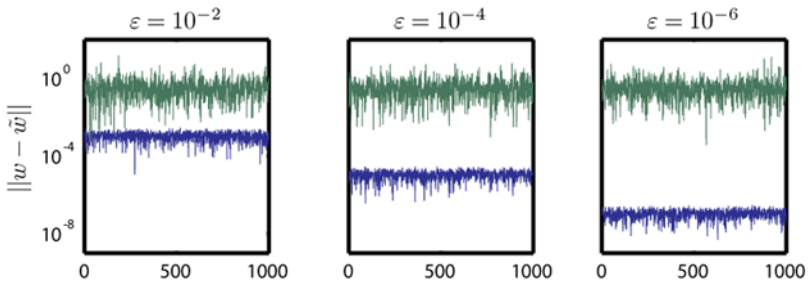

Fig. 5. The effect of neighborhood perturbation on the weight vectors of LLE and of LLE with low-dimensional neighborhood representation. The original neighborhood consists of a point on the two-dimensional grid and its 4-nearest neighbors, as in Fig 4. A 4-dimensional noise matrix εE where $\|E\|_F = 1$ was added to the neighborhood for $\varepsilon = 10^{-2}, 10^{-4}$ and 10^{-6}, with 1000 repetitions for each value of ε. Note that no regularization is needed since $K = D$. The graphs show the distance between the vector $w = \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$ and the vectors computed by LLE (upper, in green) and by LLE with low-dimensional neighborhood representation (lower, in blue). Note the log scale in the y axis.

Note that the difference between this algorithm and LLE is in step (2). We compute the low-dimensional neighborhood representation of each neighborhood and obtain its weight vector, while LLE computes the weight vector for the original high-dimensional neighborhoods. One consequence of this approach is that the weight vectors w_i are less sensitive to perturbation (see Fig 5). Proof of this result is omitted due to lack of space. Another consequence is that the d-dimensional output is no longer a projection of the embedding in dimension q, $q > d$. This is because the weight vectors w_i are computed differently for different values of output dimension d. In particular, the input data no longer minimize Φ, and therefore the linear projection problem does not occur.

From a computational point of view, the cost of this modification is small. For each point x_i, the cost of computing the SVD of the matrix X_i is $\mathcal{O}(DK^3)$. For N neighborhoods we have $\mathcal{O}(NDK^3)$ which is of the same scale as LLE for this step. Since the overall computation of LLE is $\mathcal{O}(N^2D)$, the overhead of the modification has little influence on the running time of the algorithm (see [18], Section 4).

5 Numerical results

In this section we present empirical results for LLE and LLE with low-dimensional neighborhood representation on some data sets. For LLE, we used the Matlab code as appears in LLE website [19]. The code for LLE with low-dimensional neighborhood representation is based on the LLE code and differs only in step (2) of the algorithm.

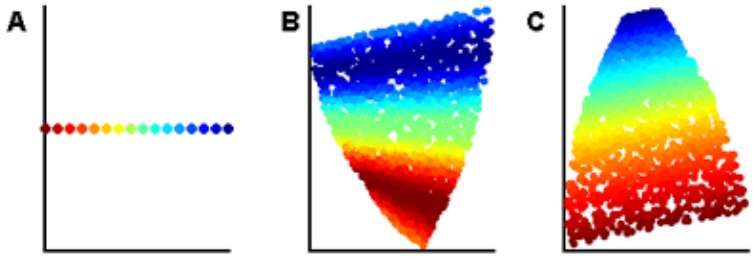

Fig. 6. The output of LLE with low-dimensional neighborhood representation for the open ring, the 'S' curve, and the swissroll, appear in (A),(B), and (C), respectively. Compare to the results of LLE, presented in Figs. 1C, 2C and 3C.

.

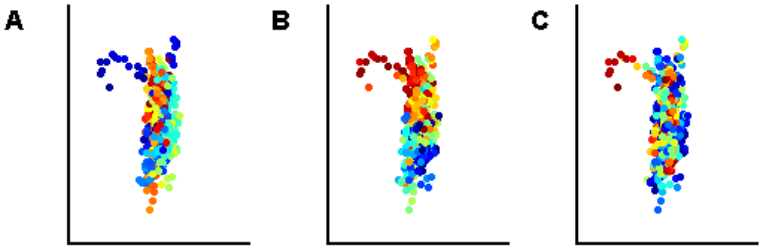

Fig. 7. The first two dimensions out of the three-dimensional output of LLE for the faces database appear in all three panels. (A) is colored according to the right-left pose, (B) is colored according to the up-down pose, and (C) is colored according to the lighting direction.

We ran LLE with low-dimensional neighborhood representation on the data sets of the open ring, the 'S'-curve, and the swissroll that appear in Figs 1-3. We used the same parameters for both LLE and LLE with low-dimensional neighborhood representation ($K = 4$ for the open ring and $K = 12$ for the 'S'-curve and the swissroll). The results appear in Fig 6.

We ran both LLE and LLE with low-dimensional neighborhood representation on 64 by 64 pixel images of a face, rendered with different poses and lighting directions. The 698 images and their respective poses and lighting directions can be found at the Isomap webpage [22]. The results of LLE, with $K = 12$, are given in Fig. 7. We also checked for $K = 8, 16$; in all cases LLE does not succeed in retrieving the pose and lighting directions. The results for LLE with low-dimensional neighborhood representation, also with $K = 12$, appear in Fig 8. The left-right pose and the lighting directions were discovered by LLE with low-dimensional neighborhood representation. We also checked for $K = 8, 16$; the results are roughly the same.

Fig. 8. The output of LLE with low-dimensional neighborhood representation is colored according to the left-right pose. LLE with low-dimensional neighborhood representation also succeeds in finding the lighting direction. The up-down pose is not fully recovered.

References

1. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290** (2000) 2323–2326
2. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290** (2000) 2319–2323
3. Belkin, M., Niyogi, P.: Laplacian Eigenmaps for dimensionality reduction and data representation. Neural Comp. **15** (2003) 1373–1396
4. Donoho, D., Grimes, C.: Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. Proc. Natl. Acad. Sci. U.S.A. **100** (2004) 5591–5596
5. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. SIAM J. Sci. Comp.
6. Weinberger, K.Q., Saul, L.K.: Unsupervised learning of image manifolds by semidefinite programming. International Journal of Computer Vision **70(1)** (2006) 77–90
7. Xu, W., Lifang, X., Dan, Y., Zhiyan, H.: Speech visualization based on locally linear embedding (LLE) for the hearing impaired. In: BMEI (2). (2008) 502–505
8. Shi, R., Shen, I.F., Chen, W.: Image denoising through locally linear embedding. In: CGIV '05: Proceedings of the International Conference on Computer Graphics, Imaging and Visualization, IEEE Computer Society (2005) 147–152
9. Chen, J., Wang, R., Yan, S., Shan, S., Chen, X., Gao, W.: Enhancing human face detection by resampling examples through manifolds. Systems, Man and Cybernetics, Part A, IEEE Transactions on **37** (2007) 1017–1028
10. L'Heureux, P., Carreau, J., Bengio, Y., Delalleau, O., Yue, S.: Locally linear embedding for dimensionality reduction in qsar. J. Comput. Aided Mol. Des. **18** (2004) 475–482

11. Wang, M., Yang, H., Xu, Z., Chou, K.: SLLE for predicting membrane protein types. J. Theor. Biol. **232** (2005) 7–15
12. Xu, X., Wu, F., Hu, Z., Luo, A.: A novel method for the determination of redshifts of normal galaxies by non-linear dimensionality reduction. Spectroscopy and Spectral Analysis **26** (2006) 182–186
13. Hadid, A., Pietikäinen, M.: Efficient locally linear embeddings of imperfect manifolds. (2003) 188–201
14. Chang, H., Yeung, D.Y.: Robust locally linear embedding. Pattern Recognition **39** (2006) 1053–1065
15. Varini, C., Degenhard, A., Nattkemper, T.W.: ISOLLE: LLE with geodesic distance. Neurocomputing **69** (2006) 1768–1771
16. Wang, H., Zheng, J., Yao, Z., Li, L.: Improved locally linear embedding through new distance computing. (2006) 1326–1333
17. Zhang, Z., Wang, J.: MLLE: Modified locally linear embedding using multiple weights. In Schölkopf, B., Platt, J., Hoffman, T., eds.: Advances in Neural Information Processing Systems 19. MIT Press, Cambridge, MA (2007) 1593–1600
18. Saul, L.K., Roweis, S.T.: Think globally, fit locally: unsupervised learning of low dimensional manifolds. J. Mach. Learn. Res. **4** (2003) 119–155
19. Saul, L.K., Roweis, S.T.: Locally Linear Embedding (LLE) website `http://www.cs.toronto.edu/~roweis/lle/`.
20. Golub, G.H., Loan, C.F.V.: Matrix Computations. Johns Hopkins University Press, Baltimore, Maryland (1983)
21. Wu, F., Hu, Z.: The LLE and a linear mapping. Pattern Recognition **39** (2006) 1799–1804
22. Tenenbaum, J.B., de Silva, V., Langford, J.C.: Isomap website `http://isomap.stanford.edu/`.