Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 108C (2017) 635–644

Procedia Computer Science

International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland

Sparse Locally Linear Embedding

Lori Ziegelmeier[1], Michael Kirby[2], and Chris Peterson[2]

[1] Macalaster College, Saint Paul, Minnesota, USA
lziegel1@macalester.edu
[2] Colorado State University, Fort Collins, Colorado, USA
kirby@math.colostate.edu, peterson@math.colostate.edu

Abstract

The Locally Linear Embedding (LLE) algorithm has proven useful for determining structure preserving, dimension reducing mappings of data on manifolds. We propose a modification to the LLE optimization problem that serves to minimize the number of neighbors required for the representation of each data point. The algorithm is shown to be robust over wide ranges of the sparsity parameter producing an average number of nearest neighbors that is consistent with the best performing parameter selection for LLE. Given the number of non-zero weights may be substantially reduced in comparison to LLE, Sparse LLE can be applied to larger data sets. We provide three numerical examples including a color image, the standard swiss roll, and a gene expression data set to illustrate the behavior of the method in comparison to LLE. The resulting algorithm produces comparatively sparse representations that preserve the neighborhood geometry of the data in the spirit of LLE.

Peer-review under responsibility of the scientific committee of the International Conference on Computational Science

Keywords: Locally Linear Embedding, Dimensionality Reduction, Manifold Learning, Optimization, Regularization, Sparsity

1 Introduction

The Locally Linear Embedding (LLE) algorithm is an unsupervised dimensionality reduction algorithm that determines a mapping of data lying in a higher dimensional space to a lower dimensional space while optimizing the maintenance of local spatial relationships [10]. Through this map, the LLE algorithm uncovers a lower dimensional representation with the goal of preserving the topology and neighborhood structure of the original high dimensional data.

The primary parameter of the LLE algorithm is K, the choice of the number of nearest neighbors associated to each data point. This choice greatly affects the embedding results as it determines the local and global representation of the high dimensional data in a lower dimensional space. Furthermore, the LLE algorithm assumes that this value K is fixed for all data points when, in practice, the ability to vary the number of nearest neighbors based on the local features of the data is appealing. Others have observed that it can be difficult to appropriately choose K, and methods have been introduced to aid in this task [9, 2, 8].

1877-0509
Peer-review under responsibility of the scientific committee of the International Conference on Computational Science
10.1016/j.procs.2017.05.171

Sparsity promoting regularization using the ℓ_1 norm is now a well-understood and widely applied tool, see, *e.g.*, [4, 5, 11]. The semi-norm ℓ_0 counts the number of non-zero components of a vector, but given it is non-convex, it is not practical for the analysis of high-dimensional data. The ℓ_1 norm is the closest convex approximation to the ℓ_0 semi-norm and is therefore an attractive proxy for minimization problems [3]. The ℓ_1 norm has been used successfully in a variety of applications such as compressed sensing, error correction, and matrix completion, and it is known to produce sparsity in the decision variables of optimization problems when used as a regularization term in the objective function.

In this paper, we propose a modification to the LLE optimization problem by using the ℓ_1 norm to promote sparsity in the weights used to represent data points by their neighbors. We formulate this as a quadratic programming problem and solve using the primal dual interior point algorithm. The resulting algorithm automatically selects an optimal number of nearest neighbors for each data point. The net result is a greatly reduced number of non-zero weights required to accomplish the dimension reduction using a sparse eigensolver. We apply the algorithm to a color image, the prototype swiss role example, and the analysis of a biological data set related to gene expression and the human immune system's response to infection.

2 The Locally Linear Embedding Algorithm

We begin by briefly summarizing the LLE algorithm [10]. Given a data matrix, X, consisting of p points each of dimension D, the LLE algorithm determines a representation of reduced dimension $d < D$ that retains the local topological structure of the data. This structure is expressed by the relative positions of the data points and is captured by writing each point as the weighted sum of its neighbors. Hence, the first step is to determine the set of neighboring points associated to each of the high-dimensional data points. Typically, this is done by determining *a fixed number* of nearest neighbors K using the Euclidean distance to measure proximity.

Given a set of K-nearest neighbors for each data point, the next step is to express each point optimally in terms of these neighbors. If $\mathbf{x}_i$ denotes the i^{th} point, and N_i denotes the index set of its nearest neighbors, one seeks the values of w_{ij} that solve the optimization problem

$$\begin{aligned} &\underset{w_{ij}}{\text{minimize}} \sum_{i=1}^{p} \|\mathbf{x}_i - \sum_{j \in N_i} w_{ij}\mathbf{x}_j\|_2^2 \\ &\text{subject to} \sum_{j \in N_i} w_{ij} = 1 \end{aligned}$$

for each i where the nonzero weights are supported on N_i.

The algorithm then determines a new set of lower dimensional points $\mathbf{y}_i$ which maintain these relationships between each point and its neighbors by solving the optimization problem

$$\begin{aligned} &\underset{\mathbf{y}_i}{\text{minimize}} \ \sum_{i=1}^{p} \|\mathbf{y}_i - \sum_{j \in N_i} w_{ij}\mathbf{y}_j\|_2^2 \\ &\text{subject to} \ \frac{1}{p}\sum_{i=1}^{p} \mathbf{y}_i\mathbf{y}_i^T = I, \sum_{i=1}^{p} \mathbf{y}_i = 0. \end{aligned}$$

The solution to this optimization problem is given by the eigenvector problem

$$MY^T = Y^T\Lambda \tag{1}$$

where $M = (I - W)^T (I - W)$, and W is the $p \times p$ matrix containing the weights w_{ij}. The optimal solution is given by the eigenvectors associated with the smallest d non-zero eigenvalues. The i^{th} row of Y^T corresponds to $\mathbf{y}_i$.

If a data set is a sufficiently dense sampling of a manifold, then a fundamental assumption of the algorithm is that each data point and its nearest neighbors can be characterized by a locally linear patch of the manifold. Therefore, in the second step of the algorithm, each $\mathbf{x}_i$ is approximated by a linear combination of its neighbors. Data points that are close together in the original higher dimensional space should still be close together after being mapped to lie in the lower dimensional space, thus preserving the topology of the original data set.

3 Selection of Nearest Neighbors

Implementation of the above algorithm quickly reveals that the choice of the number of nearest neighbors K significantly impacts the solution. If K is chosen too large, then two distinct pathologies may occur. As the result of the manifold's curvature, the locally linear assumption of the algorithm may be violated, *i.e.*, the neighborhood is not *flat*, or points may be identified as neighbors in Euclidean space even though their distance may be large when measured along the manifold. For an example, see Figure 3(a), where points in oval A could potentially be represented by nearest neighbors in oval B if K is chosen too large. Alternatively, for K chosen too small, the local patches may be under-sampled and unable to capture the global structure of the data as the manifold may not consist of a single connected component.

A variety of heuristics have been proposed to guide the selection of K. For example, it is possible to check every value of K up to some maximum value and pick an optimal K based on an additional quality criterion, which is straightforward but computationally intensive [9]. Thus, a hierarchical approach is proposed in [9] which implements the first and second steps of the LLE algorithm for $K \in [1, K_{max}]$ where K_{max} has been assigned, computes the reconstruction error between the original high dimensional data and the linear combination of its nearest neighbors for each value of K, and then, for every value of K that minimizes this error, the third step to determine the embedding vectors is performed in order to calculate the residual variance. The optimal K is selected as the one that minimizes the residual variance. This hierarchical method is computationally less costly as the eigenvector problem, the most expensive computation, is only performed for a small number of K values.

However, [2] suggests that the residual variance is not a good measure of the local geometric structure of the data and instead proposes a new measure called Preservation Neighborhood Error–which incorporates both the global manifold behavior and local geometry of the data. [2] also suggests a method to select K_i for each point $\mathbf{x}_i$ locally based on graph theory. Other work by [8] indicates that an optimal K need not be selected, as often a range of K values produces stable reconstructions. This claim is dependent on sampling density and manifold geometry. We have not seen this to be the case for many of the data sets considered in our work.

We now propose a sparse representation of weights allowing for identification of the 'true' nearest neighbors of each data point and a more appropriate local reconstruction, which determines a neighborhood size K_i for each data point automatically.

4 Sparse Locally Linear Embedding

In the standard derivation of the algorithm, each data point $\mathbf{x}_i$ will be represented by the number of nearest neighbors K selected, *i.e.*, the weights w_{ij} used to represent the data point $\mathbf{x}_i$

by its nearest neighbors $\{\mathbf{x}_j : j \in N_i\}$ are nonzero. However, this feature does not necessarily reflect the actual number of points needed for an optimal local representation. Allowing for the neighborhood size to vary provides additional flexibility to the algorithm to match local complexity *e.g.* noise, isolated points, holes in the data. The density and intrinsic dimensionality may differ for the neighborhoods of each point [2], and thus, an appropriate number of nearest neighbors should be selected for each point instead of a single value of K chosen for all points.

By using ℓ_1 regularization to modify the LLE algorithm, sparsity in the optimal decision variables w_{ij} is induced, allowing for an automatic selection of the nearest neighbors. Therefore, the ad hoc nature of the selection of K has been removed and each point $\mathbf{x}_i$ will have its own nearest neighbors, both location and number K_i, determined as part of the optimization problem. To this end, we propose to solve the optimization problem

$$\begin{aligned} \underset{w_{ij}}{\text{minimize}} \quad & \lambda \sum_{j \in N_i} |w_{ij}| f(d(\mathbf{x}_i, \mathbf{x}_j)) + \|\mathbf{x}_i - \sum_{j \in N_i} w_{ij}\mathbf{x}_j\|_2^2 \\ \text{subject to} \quad & \sum_{j \in N_i} w_{ij} = 1. \end{aligned} \tag{2}$$

As with standard LLE before, there is one optimization problem to solve for each i.

We introduce a non-negative function f to afford more control over the level of sparsity in the solution to the optimization problem. We have explored several options for $f(d(\mathbf{x}_i, \mathbf{x}_j))$ including $f(d(\mathbf{x}_i, \mathbf{x}_j)) = ||\mathbf{x}_i - \mathbf{x}_j||_p^m$ where d is assumed to be a metric. We employ $m = 1$ and $p = 2$ in the remainder of this paper. The parameter λ allows one to tune the sparsity of the local representations. Larger values of λ will drive more weights to zero.

The non-differentiable term in the objective function can be moved into the constraints by the approach of introducing nonnegative variables w_{ij}^+ and w_{ij}^- such that $w_{ij} = w_{ij}^+ - w_{ij}^-$ and $|w_{ij}| = w_{ij}^+ + w_{ij}^-$. Our decision variables can be represented as $\mathbf{w}$, the $2K$-dimensional vector of weight terms associated to point $\mathbf{x}_i$, with $\mathbf{w} = (\mathbf{w}^+, \mathbf{w}^-)$.

Each optimization problem can be rewritten as a quadratic program of the form

$$\begin{aligned} \underset{\mathbf{w}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{w}^T Q \mathbf{w} + \mathbf{c}^T \mathbf{w} \\ \text{subject to} \quad & A\mathbf{w} = \mathbf{b}, \mathbf{w} \geq 0. \end{aligned} \tag{3}$$

Removing the constant term will not affect the optimal solution. Equation (3) is in $n = 2K$ variables with equality constraint, $A\mathbf{w} = \mathbf{b}$, an $m \times n$ system where $m = 1$. Note that Q can be written as an outer product and is thus positive semi-definite. Therefore, Equation (3) is a convex optimization problem where any local optimum is also a global optimum [3, 13]. We solve this problem using the Primal Dual Interior Point method by reducing it to a sequence of linear, equality constrained problems and then applying Newton's method; see [13].

The computational complexity of Sparse LLE is as follows: The first step of Sparse LLE is equivalent to the first step of standard LLE to determine the nearest neighbors. This scales in the worst case as $\mathcal{O}(Dp^2)$ where D is the ambient dimension of our input data and p is the number of data points but for certain data distributions can scale as $\mathcal{O}(p \log p)$. Standard LLE can determine the weights by solving a least squares problems to determine the K weights associated to the nearest neighbors of each point which scales as $\mathcal{O}(DpK^3)$. In Sparse LLE, the number of decision variables is $2K$ and there is a single equality constraint. Therefore, the number of operations required to solve for all of the weights is $\mathcal{O}(Dp(4K+1)^3 h)$, where h is the average number of iterations required to solve each problem. This problem can be trivially parallelized. Although this is more computationally intensive than standard LLE, Sparse LLE

has the benefit that nearest neighbors are chosen appropriately for each nearest neighbor. The most computationally expensive step in LLE is to solve the dense eigenvector problem which scales as $\mathcal{O}(dp^2)$ where d is the dimension of the embedding data. Methods to solve sparse symmetric eigenproblems, however, reduce complexity to subquadratic in p. Sparse LLE can dramatically increase the sparsity (see for example, Section 6.2) which improves computational time in this final step.

5 The Algorithm

Here we provide the implementation details of the proposed algorithm. First, one must choose K_{max}, an upper bound for the number of nearest neighbors. In practice, the algorithm appears to be insensitive to this parameter,provided it is large enough. The number of nearest neighbors selected is generally much smaller than this bound. It is tempting to select $K_{max} = p - 1$. While this works, we advocate smaller values to speed up computations. As described above, the sparsity parameter λ can be adjusted to force more weights to zero. Sparsity is admittedly a tradeoff with accuracy. Generally, a value of $\lambda = 0.01$ seems to work well if $K_{max} > D$.[1] The resulting automatic selection of a small number of nearest neighbors i.e. fewer non-zero weights, has potentially significant ramifications for the sparse eigensolvers which are used to solve (1). The parameter ϵ is used to threshold weights that are close to zero. A choice of $\epsilon = 10^{-4}$ worked well in our experiments.

Sparse LLE Algorithm

1. Select K_{max}, λ and ϵ.
2. Find K_{max} nearest neighbors to each point $\mathbf{x}_i$.
3. Find the sparse weights used to write each point as a linear combination of its nearest neighbors $\mathbf{x}_j$ by solving Equation (3) for each point $\mathbf{x}_i$. If $w_{ij} < \epsilon$, set $w_{ij} = 0$.
4. Determine lower dimensional embedding vectors by solving the sparse eigenvector problem given by Equation (1) for Y.

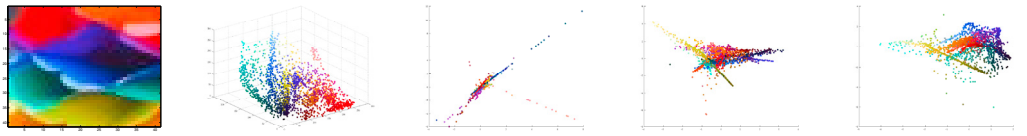
6 Numerical Experiments

We now present three numerical experiments to illustrate the behavior of the proposed Sparse LLE algorithm. We illustrate its ability to locally select an optimal number of nearest neighbors and to preserve the topological structure of the high-dimensional space.

6.1 Color Image

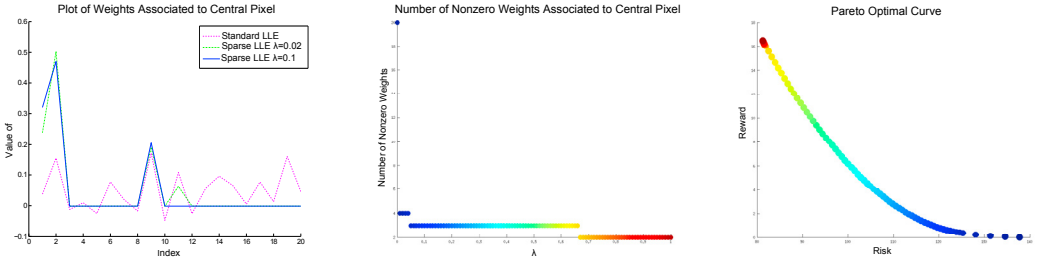
For our first example, we consider the 3-dimensional color data from a square 41×41 color image, Figure 1(a). For this example, we illustrate the sparsity induced on the weights using Sparse LLE focusing on the central pixel within the image to illustrate the characteristics of the algorithm. In the spirit of selecting far more nearest neighbors than is required to represent this data in $\mathbb{R}^3$, we set $K_{max} = 20$.

[1] In the case when it is not possible to choose $K_{max} > D$, a larger value of λ must be selected to place more emphasis on sparsity rather than reconstruction. The final example illustrates this.



(a) Original Image (b) $\mathbb{R}^3$ Data (c) $K = 4$ (d) $K = 6$ (e) $K = 8$

Figure 1: Illustration of the sensitivity of LLE to the choice of number of nearest neighbors K. The figures are the original image, the pixel data plotted in $\mathbb{R}^3$, and the associated manifold coordinates in $\mathbb{R}^2$ using various choices of the parameter K.



(a) Weights Varying λ (b) Nonzero Weights (c) Pareto Optimal

Figure 2: (a) Plot of weights associated to Sparse LLE with $K_{max} = 20$ nearest neighbors and varying λ. Note $\lambda = 0$ is standard LLE. (b) Plot of the number of nonzero weights associated to nearest neighbors of the central pixel versus λ. (c) Pareto optimal curve showing the value of the sparsity (risk) term $\sum_{j\in N_i} |w_{ij}| f(d(\mathbf{x}_i, \mathbf{x}_j))$ versus the reconstruction error (reward) term $\|\mathbf{x}_i - \sum_{j\in N_i} w_{ij}\mathbf{x}_j\|_2^2$ as parameterized by λ. Blue corresponds to smaller and red to larger λ.

A simple experiment illustrates the sensitivity of LLE on the selection of the number of nearest neighbors K. The original image is shown in Figure 1(a), and a plot of the points residing in $\mathbb{R}^3$ extracted as the RGB pixel information is displayed in 1(b). We observe that the choice of the fixed number of nearest neighbors K significantly impacts the data structure in the reduced space of $\mathbb{R}^2$, see Figures 1(c)-1(e).

As discussed above, the main parameter dependence in the proposed sparse algorithm is connected to λ. By increasing λ, we observe that sparsity is induced in the weights.[2] Figure 2(a) displays the numerical values of the 20 weights associated to the central pixel for three choices of parameter λ. There are 20 nonzero weights when $\lambda = 0$, 4 when $\lambda = 0.02$, and 3 when $\lambda = 0.1$. The weights plotted are the actual output of the algorithm, *i.e.*, there has been no thresholding. Figure 2(b) shows how the number of non-zero weights changes as a function of λ. Notice that almost immediately as λ is increased, a small number of non-zero weights result and further, that this number is not very sensitive to changes in λ. Given the raw data lives in $\mathbb{R}^3$, it makes sense that 3 or 4 nearest neighbors would be necessary to reconstruct each data point, remembering that the weights associated to these nearest neighbors must sum to 1.

The sparse LLE optimization problem (2) involves the balancing of the sparsity term $\sum_{j\in N_i} |w_{ij}| f(d(\mathbf{x}_i, \mathbf{x}_j))$ with the reconstruction error $\|\mathbf{x}_i - \sum_{j\in N_i} w_{ij}\mathbf{x}_j\|_2^2$. As more emphasis is placed on the first term, the solution's sparsity increases at the expense of the reconstruction

[2]Note that standard LLE is a special case of Sparse LLE with sparsity parameter $\lambda = 0$.

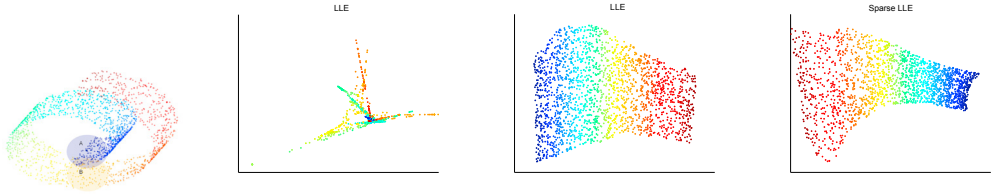
error. Thus, there is a balancing act between these two terms analogous to the *efficient frontier* in the risk-reward analysis of investments. Thus, sparsity may be tuned as a parameter in a manner similar to how risk is tuned for an investment. This balancing act is captured in the pareto optimal curve shown in Figure 2(c). The optimal values of each term in the objective function are displayed, parameterized by λ.

In summary, we observe that small λ induces sparsity in the weights associated to nearest neighbors. Further increasing λ produces a more sparse solution at the expense of increasing the reconstruction error, yet the number of nearest neighbors is robust over a large scale of λ.

6.2 Swiss Roll Example

Now, consider the canonical swiss roll example widely used as an illustrative data set for nonlinear dimensionality reduction algorithms; see, *e.g.*, [10, 2]. The topological structure of this data is not captured with a linear dimensionality reduction technique such as PCA or MDS. It is, however, unraveled by nonlinear techniques such as LLE. The data consists of 2000 random points along the swiss roll in $\mathbb{R}^3$, see Figure 3(a).

Given that the ambient dimension of the data set is three, one might be tempted to set $K = 4$ for the number of nearest neighbors in the standard LLE algorithm. However, this choice produces a poor embedding in $\mathbb{R}^2$ and does not reflect the topological structure of the data; see Figure 3(b). In fact, this choice disconnects the data, as evidenced by the eigenvalues of the Laplacian matrix M defined in Equation (1). The source code on the LLE homepage instead selects $K = 12$ [1]. This choice of K yields a much better embedding; see Figure 3(c).

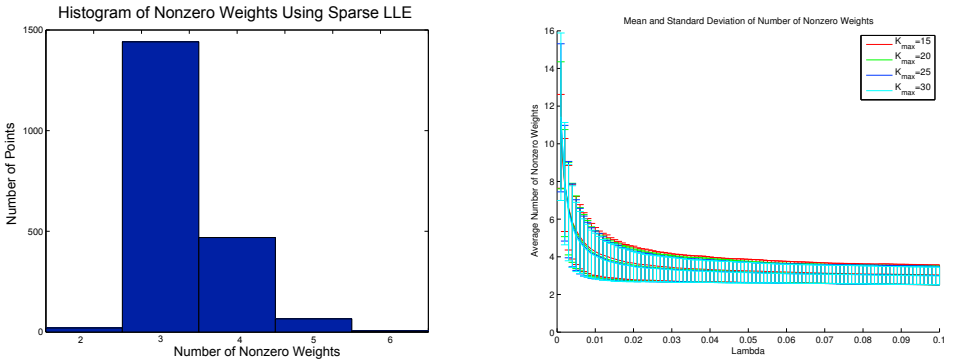


(a) Data set. (b) LLE $K = 4$. (c) LLE $K = 12$. (d) SLLE $K_{max} = 20$.

Figure 3: Points sampled from the swiss roll embedded into $\mathbb{R}^2$ using LLE and Sparse LLE (SLLE) with $\epsilon = 10^{-4}$, and $\lambda = 0.01$.

We now illustrate how the Sparse LLE algorithm automatically determines an appropriate value of K_i for each point $\mathbf{x}_i$ on the swiss roll. As above, we select $K_{max} = 20$ for the maximum number of nearest neighbors allowed and the sparsity parameter $\lambda = 0.01$. Figure 3(d) was generated by zeroing out those entries less than $\epsilon = 10^{-4}$ in W and using the remaining 'nonzero' entries in W in the sparse eigenvector problem. The histogram of the number of nonzero weights associated to each swiss roll point using Sparse LLE with threshholding $\epsilon = 10^{-4}$ is shown in Figure 4(a). For the sparsity parameter $\lambda = 0.01$ only 2 to 6 weights are nonzero for each data point using Sparse LLE even though $K_{max} = 20$ nearest neighbors were allowed. Thus, the Sparse LLE algorithm requires less than one-third the number of weights required by LLE.

The algorithm is repeated for several values of $K_{\max}$ and λ. The results appear robust as shown in Figure 4(b). Observe that for very small λ approximately 12 ± 4 neighbors are selected. This number is consistent with the value of $K = 12$ nearest neighbors recommended for standard LLE for this data. For $\lambda > 0.01$ the solution is robust and indicates that 4 ± 1 neighbors are needed to represent the data.



(a) Sparse LLE $K_{\max} = 20$ $\lambda = 0.01$

(b) Average Nonzero Weights

Figure 4: (a) Histogram of the number of nonzero weights associated to each point on the swiss roll using SLLE with $K_{\max} = 20$ and $\lambda = 0.01$. (b) Mean and standard deviation of the number of nonzero weights associated to nearest neighbors of points in the swiss roll as both λ and K_{max} are varied in Sparse LLE.

Through this experiment, we see that Sparse LLE is not only able to preserve the topological structure of this highly nonlinear data set but also automatically chooses an appropriate number of nearest neighbors for each point in an efficient way, requiring approximately 33% of the number of nearest neighbors used by standard LLE.

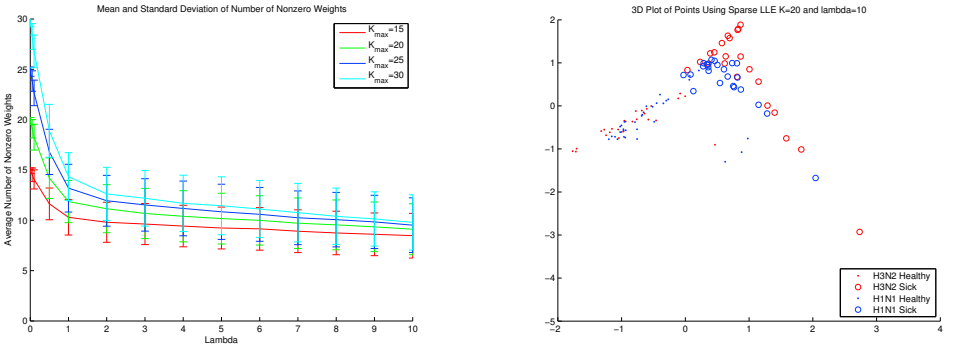
6.3 Gene Expression Influenza Data

The final data set that we consider is gene expression data from the Influenza Challenge described in [12]. In this study, 17 volunteers were exposed to the H1N1 virus, and 19 volunteers were exposed to the H3N2 virus. Blood samples were collected every 8 hours from each subject and then analyzed for gene expression levels. Each measurement consists of expression levels of 12023 distinct genes. In total, the data set consists of 108 points in $\mathbb{R}^{12023}$. Additionally, each subject either became symptomatic or not, *i.e.*, some individuals had an immune response that effectively suppressed symptoms while others felt very sick. For both strains of virus about half of all subjects produced symptoms while others remained asymptomatic. Genes associated with biological pathways that are active in fighting infection express differently depending on the subject's ability to fight the infection.

Geometrically, we envision a very high-dimensional vector space where symptomatic H1N1 and H3N2 subjects are clustered away from the asymptomatic subjects. Since we cannot visualize this data set, we reduce the dimension to $\mathbb{R}^2$ using both LLE and Sparse LLE. In this case, unlike in our other analyses, any choice of nearest neighbors will be much less than the dimension of our ambient space, *i.e.* $K < D$, and thus, our data points will be unable to be perfectly reconstructed by their nearest neighbors.[3] However, Sparse LLE is still able to remain robust across the choice of nearest neighbors for an appropriately large λ.

We perform a parameter space search using standard LLE with $K = 4, \ldots, 40$, and observe once again that reconstructions are highly dependent upon this choice. We consider the "optimal" embedding $K = 10$ as symptomatic and asymptomatic subjects appear to have fairly

[3]This is also true for standard LLE.



(a) Mean and standard deviation of the number of nonzero weights associated to nearest neighbors as both λ and K_{max} are varied in Sparse LLE.

(b) Results of embedding of using Sparse LLE with parameters $K_{max} = 20$, $\lambda = 10$.

Figure 5: Application of sparse LLE to the gene expression data produced by the influenza challenge.

good separation in this reconstruction. A similar analyses is performed with Sparse LLE. Figure 5(a) shows the average number of nonzero weights as well as the standard deviation of varying both K_{max} and λ in Sparse LLE. The curves appear to level off at 10 ± 3 nearest neighbors. Therefore, Sparse LLE automatically identifies 10 as the mean number of nearest neighbors, consistent with the best result found with LLE. We observe in Figure 5(b) that SLLE, like LLE, reveals good separation between symptomatic and asymptomatic subjects at this level. In fact, SLLE has a broad range of λ values for which varying $K_{\max}$ does not significantly affect the output reconstructions.

7 Conclusion

We have proposed a modification to the optimization problem associated with the Locally Linear Embedding algorithm that computes the weights used to reconstruct each data point from its neighbors. The result is a constrained quadratic program with a sparsity parameter on the weights. By varying this parameter away from zero, the number of nearest neighbors required to characterize each point is determined automatically. In the standard LLE algorithm, the structure of the embedding appears highly dependent on the selection of K. We have provided evidence in three examples that Sparse LLE behaves robustly to parameter selection over a wide range of the sparsity parameter. The number of nonzero weights in standard LLE is fixed to be the number of points times the (constant) number of nearest neighbors. In contrast, as illustrated in examples, Sparse LLE may require substantially fewer non-zero weights. This permits the embedding of potentially larger data sets when a sparse eigensolver is used. We have limited our direct comparisons to the addition of the sparsity promoting term in LLE since other non-sparse variations on LLE have been widely published. For instance, it may be interesting to consider how the approach presented here could be adapted to the Hessian LLE algorithm proposed in [6]. We note that another approach, similar in spirit to ours, also employs an ℓ_1 optimization problem for solving the nearest neighbor selection problem [7]. In future work we will explore a comparison of these two methodologies.

Acknowledgments This paper is based on research partially supported by the National Science Foundation under Grants Nos. DMS-1322508 and IIS-1633830. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Locally linear embedding, swiss roll source code. `http://www.cs.nyu.edu/~roweis/lle/code/swissroll.m`. Accessed: 2013-04-24.

[2] Andrés Álvarez-Meza, Juliana Valencia-Aguirre, Genaro Daza-Santacoloma, and Germán Castellanos-Domínguez. Global and local choice of the number of nearest neighbors in locally linear embedding. *Pattern Recognition Letters*, 32(16):2171–2177, 2011.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[4] Paul S Bradley and Olvi L Mangasarian. Feature selection via concave minimization and support vector machines. In *Machine Learning Proceedings of the Fifteenth International Conference (ICML98)*, pages 82–90, 1998.

[5] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions on*, 52(2):489–509, 2006.

[6] David L Donoho and Carrie Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596, 2003.

[7] Ehsan Elhamifar and René Vidal. Sparse manifold clustering and embedding. In *Advances in Neural Information Processing Systems*, pages 55–63, 2011.

[8] Rasa Karbauskait, Olga Kurasova, and Gintautas Dzemyda. Selection of the number of neighbours of each data point for the locally linear embedding algorithm. *Information Technology and Control*, 36(4), 2007.

[9] Olga Kouropteva, Oleg Okun, and Matti Pietikinen. Selection of the optimal parameter value for the locally linear embedding algorithm. In *First International Conference on Fuzzy Systems*, pages 359–363, 2002.

[10] Lawrence K. Saul, Sam T. Roweis, and Yoram Singer. Think globally, fit locally: Unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[11] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[12] Aimee K Zaas, Minhua Chen, Jay Varkey, Timothy Veldman, Alfred O Hero III, Joseph Lucas, Yongsheng Huang, Ronald Turner, Anthony Gilbert, Robert Lambkin-Williams, et al. Gene expression signatures diagnose influenza and other symptomatic respiratory viral infections in humans. *Cell Host & Microbe*, 6(3):207–217, 2009.

[13] Lori B. Ziegelmeier. *Exploiting Geometry, Topology and Optimization for Knowledge Discovery in Big Data*. PhD thesis, Colorado State University, Fort Collins, May 2013.