Journal of Machine Learning Research 13 (2012) 1609-1638 Submitted 10/10; Revised 1/12; Published 5/12

A Unifying Probabilistic Perspective for Spectral Dimensionality Reduction: Insights and New Models

Neil D. Lawrence[*] N.LAWRENCE@DCS.SHEF.AC.UK
Department of Computer Science
University of Sheffield
Regent Court
Portobello
S1 4DP
United Kingdom

Editor: Tony Jebara

Abstract

We introduce a new perspective on spectral dimensionality reduction which views these methods as Gaussian Markov random fields (GRFs). Our unifying perspective is based on the maximum entropy principle which is in turn inspired by maximum variance unfolding. The resulting model, which we call maximum entropy unfolding (MEU) is a nonlinear generalization of principal component analysis. We relate the model to Laplacian eigenmaps and isomap. We show that parameter fitting in the locally linear embedding (LLE) is approximate maximum likelihood MEU. We introduce a variant of LLE that performs maximum likelihood exactly: Acyclic LLE (ALLE). We show that MEU and ALLE are competitive with the leading spectral approaches on a robot navigation visualization and a human motion capture data set. Finally the maximum likelihood perspective allows us to introduce a new approach to dimensionality reduction based on L1 regularization of the Gaussian random field via the graphical lasso.

1. Introduction

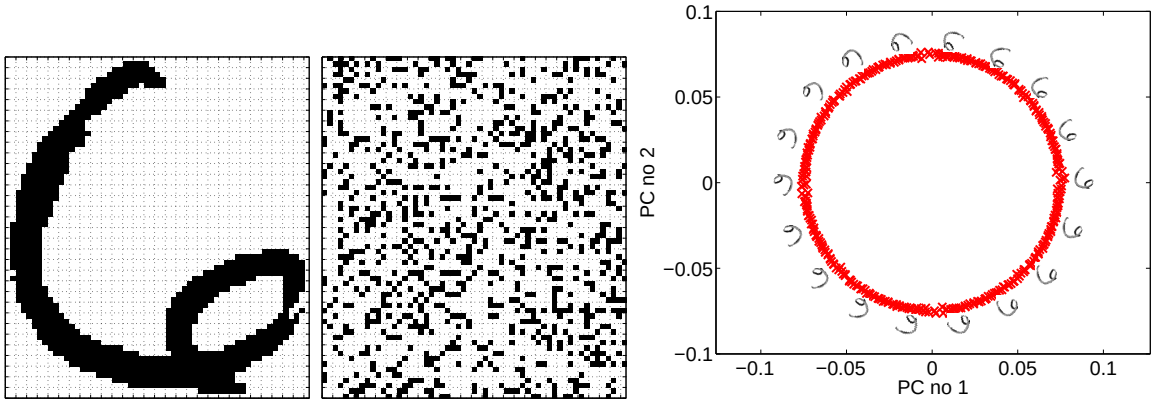
A representation of an object for processing by computer typically requires that object to be summarized by a series of features, represented by numbers. As the representation becomes more complex, the number of features required typically increases. Examples include: the characteristics of a customer in a database; the pixel intensities in an image; a time series of angles associated with data captured from human motion for animation; the energy at different frequencies (or across the cepstrum) as a time series for interpreting speech; the frequencies of given words as they appear in a set of documents; the level of expression of thousands of genes, across a time series, or for different diseases.

With the increasing complexity of the representation, the number of features that are stored also increases. Data of this type is known as high dimensional data.

Consider the simple example of a handwritten six. The six in Figure 1 is represented on a grid of pixels which is 64 rows by 57 columns, giving a datum with 3,648 dimensions. The space in which this digit sits contains far more than the digit. Imagine a simple probabilistic model of the digit which assumes that each pixel in the image is independent and is on with a given probability. We can sample from such a model (Figure 1(b)).

Even if we were to sample every nanosecond from now until the end of the universe we would be highly unlikely to see the original six. The space covered by this model is very large but fundamentally the data lies on low dimensional embedded space. This is illustrated in Figure 1(c). Here a data set has been constructed by rotating the digit 360 times in one degree intervals. The data is then projected onto its first two principal components. The spherical structure of the rotation is clearly visible in the projected data. Despite the

∗. Also in the Sheffield Institute of Translational Neuroscience.



(a) (b) (c)

Figure 1: The storage capacity of high dimensional spaces. (a) A six from the USPS digit data set. (b) A sample from a simple independent pixel model of the six. There are $2^{3,648}$ possible images. Even with an enormous number of samples from such a model we would never see the original six. (c) A data set generated by rotating the original six from (a) 360 times. The data is projected onto its first two principal components. These two principal components show us that the data lives on a circle in this high dimensional space. There is a small amount of noise due to interpolation used in the image rotation. Alongside the projected points we show some examples of the rotated sixes.

data being high dimensional, the underlying structure is low dimensional. The objective of dimensionality reduction is to recover this underlying structure.

Given a data set with n data points and p features associated with each data point, dimensionality reduction involves representing the data set using n points each with a reduced number, q, of features, with $q < p$. Dimensionality reduction is a popular approach to dealing with high dimensional data: the hope is that while many data sets seem high dimensional, it may be that their intrinsic dimensionality is low like the rotated six above.

1.1 Spectral Dimensionality Reduction

Spectral approaches to dimensionality reduction involve taking a data set containing n points and forming a matrix of size $n \times n$ from which eigenvectors are extracted to give a representation of the data in a low dimensional space. Several spectral methods have become popular in the machine learning community including isomap (Tenenbaum et al., 2000), locally linear embeddings (LLE, Roweis and Saul, 2000), Laplacian eigenmaps (Belkin and Niyogi, 2003) and maximum variance unfolding (MVU, Weinberger et al., 2004). These approaches (and kernel principal component analysis, kernel PCA, Schölkopf et al., 1998) are closely related. For a kernel perspective on the relationships see Ham et al. (2004) and Bengio et al. (2004b,a). Our focus in this work is unifying the methods from a classical multidimensional scaling (CMDS, Mardia et al., 1979) perspective.

In classical multidimensional scaling an $n \times n$ symmetric distance matrix, whose elements contain the distance between two data points, is converted to a similarity matrix and visualized through its principal eigenvectors. Viewed from the perspective of CMDS the main difference between the spectral approaches developed in the machine learning community is in the distance matrices they (perhaps implicitly) proscribe.

In this paper we introduce a probabilistic approach to constructing the distance matrix: maximum entropy unfolding (MEU). We describe how isomap, LLE, Laplacian eigenmaps and MVU are related to MEU using the unifying perspective of Gaussian random fields and CMDS.

The parameters of the model are fitted through maximum likelihood in a Gaussian Markov random field (GRF). The random field specifies dependencies between *data points* rather than the more typical approach which specifies dependencies between *data features*. We show that the locally linear embedding algorithm is an approximation to maximum entropy unfolding where pseudolikelihood is maximized as an approximation to the model likelihood. Our probabilistic perspective inspires new dimensionality reduction algorithms. We introduce an exact version of locally linear embedding based on an acyclic graph structure that maximizes the true model likelihood (acyclic locally linear embedding, ALLE). We also consider approaches to learning the structure of the GRF through graphical regression (Friedman et al., 2008). By L1 regularization of the dependencies we explore whether learning the graph structure (rather than prespecifying by nearest neighbour) improves performance. We call the algorithm Dimensionality reduction through Regularization of the Inverse covariance in the Log Likelihood (DRILL).

Our methods are based on maximum likelihood. Normally maximum likelihood algorithms specify a distribution which factorizes over the data points (each data point is independent given the model parameters). In our models the likelihood factorizes over the features (each feature from the data set is independent given the model parameters). This means that maximum likelihood in our model is consistent as the number of features increases, $p \to \infty$ rather than the number of data points. Alternatively, the parameters of our models become better determined as the number of features increase, rather than the number of data. This can be interpreted as a *blessing* of dimensionality rather than the more usual 'curse of dimensionality.' This has significant implications for learning in high dimensional data (known as the large p small n regime) which run counter to received wisdom.

In Section 2 we derive our model through using standard assumptions from the field of dimensionality reduction and the maximum entropy principle (Jaynes, 1986). We then relate the model to other popular spectral approaches for dimensionality reduction and show how the parameters of the model can be fitted through maximum likelihood. This allows us to regularize the system with sparse priors and seek MAP solutions that restrict the inter point dependencies. Finally, we demonstrate the model (with comparisons) on two real world data sets. First though, we will review classical multidimensional scaling which provides the general framework through which these approaches can be related (see also Ham et al., 2004; Bengio et al., 2004b,a).

1.2 Classical Multidimensional Scaling

Given an $n \times n$ matrix of similarities, $\mathbf{K}$, or dissimilarities, $\mathbf{D}$, between a set of data points, multidimensional scaling considers the problem of how to represent these data in a low dimensional space. One way of doing this is to associate a q dimensional latent vector with each data point, $\mathbf{y}_{i,:}$, and define a set of dissimilarities between each latent point, $\delta_{i,j} = \left\|\mathbf{x}_{i,:} - \mathbf{x}_{j,:}\right\|_2^2$ (where $\|\cdot\|_2$ represents the L2-norm) to give a matrix $\mathbf{\Delta}$. Here we have specified the squared distance between each point as the dissimilarity.[1]

If the error for the latent representation is then taken to be the sum of absolute values between the dissimilarity matrix entries,

$$E(\mathbf{X}) = \sum_{i=1}^{n}\sum_{j=1}^{i-1}\left\|d_{i,j} - \delta_{i,j}\right\|_1, \tag{1}$$

and we assume that the data dissimilarities also represent a squared Euclidean distance matrix (perhaps computed in some high, maybe infinite, dimensional space) then the optimal *linear* dimensionality reduction is given by the following procedure (Mardia et al., 1979, pg. 400),

1. Convert the matrix of dissimilarities to a matrix of similarities by taking $\mathbf{B} = -\frac{1}{2}\mathbf{HDH}$ where $\mathbf{H} = \mathbf{I} - n^{-1}\mathbf{1}\mathbf{1}^\top$ is a centering matrix.

2. Extract the first q principal eigenvectors of $\mathbf{B}$.

1. It is more usual to specify the distance directly as the dissimilarity, however, for our purposes it will be more convenient to work with squared distances.

3. Setting $\mathbf{X}$ to these principal eigenvectors (appropriately scaled) gives a global minimum for the error function (1).

The centering matrix $\mathbf{H}$ is so called because when applied to data in the form of a design matrix, $\mathbf{Y} \in \Re^{n\times p}$, that is, one where each row is a data point and each column is a data set feature, the centred data matrix is recovered,

$$\begin{aligned}\hat{\mathbf{Y}} =& \mathbf{YH}\\ =& \mathbf{Y} - n^{-1}\mathbf{Y}\mathbf{1}\mathbf{1}^\top,\\ =& \mathbf{Y} - \boldsymbol{\mu}\mathbf{1}^\top\end{aligned}$$

where $\boldsymbol{\mu} = n^{-1}\mathbf{Y}\mathbf{1}$ is the empirical mean of the data set.

2. Maximum Entropy Unfolding

Classical multidimensional scaling provides the optimal *linear* transformation of the space in which the squared distances are expressed. The key contribution of recently developed spectral approaches in machine learning is to compute these distances in a space which is nonlinearly related to the data thereby ensuring a *nonlinear* dimensionality reduction algorithm. From a machine learning perspective this is perhaps clearest for kernel PCA (Schölkopf et al., 1998). In kernel PCA the squared distances are computed between points in a Hilbert space and related to the original data through a kernel function,

$$d_{i,j} = k(\mathbf{y}_{i,:}, \mathbf{y}_{i,:}) - 2k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:}) + k(\mathbf{y}_{j,:}, \mathbf{y}_{j,:}). \tag{2}$$

For the linear kernel function, $k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:}) = \mathbf{y}_{i,:}^\top \mathbf{y}_{j,:}$ this reduces to the squared Euclidean distance, but for nonlinear kernel functions such as $k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:}) = \exp(-\gamma \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2)$ the distances are nonlinearly related to the data space. They are recognized as squared distances which are computed in a "feature space" (see, e.g., Ham et al., 2004; Bengio et al., 2004b,a). If we equate the kernel matrix, $\mathbf{K}$, to the similarity matrix in CMDS then this equation is also known as the *standard transformation* between a similarity and distance (Mardia et al., 1979).

Kernel PCA (KPCA) recovers an $\mathbf{x}_{i,:}$ for each data point and a mapping from the data space to the $\mathbf{X}$ space. Under the CMDS procedure we outlined above the eigenvalue problem is performed on the centered kernel matrix,

$$\mathbf{B} = \mathbf{HKH},$$

where $\mathbf{K} = [k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:})]_{i,j}$. This matches the procedure for the KPCA algorithm (Schölkopf et al., 1998).[2] However, for the commonly used exponentiated quadratic kernel,

$$k(y_{i,:}, y_{j,:}) = \exp(-\gamma \left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2),$$

KPCA actually *expands* the feature space rather than reducing the dimension (see Weinberger et al., 2004, for some examples of this). Unless data points are repeated the exponentiated quadratic kernel always leads to a full rank matrix, $\mathbf{K}$, and correspondingly a rank $n-1$ centred kernel matrix, $\mathbf{B}$. To exactly reconstruct the squared distances computed in feature space all but one of the eigenvectors of $\mathbf{B}$ need to be retained for our latent representation, $\mathbf{X}$. If the dimensionality of the data, p, is smaller than the number of data points, n, then we have a latent representation for our data which has higher dimensionality than the original data.

The observation that KPCA does not reduce the data dimensionality motivated the maximum variance unfolding algorithm (MVU, Weinberger et al., 2004). The idea in MVU is to learn a kernel matrix that will allow for dimensionality reduction. This is achieved by only considering *local relationships* in the data. A set

2. For stationary kernels, kernel PCA also has an interpretation as a particular form of *metric* multidimensional scaling, see Williams (2001) for details.

of neighbors is defined (e.g., by k-nearest neighbors) and only distances between neighboring data points are respected. These distances are specified as constraints, and the other elements of the kernel matrix are filled in by maximizing its trace, $\mathrm{tr}(\mathbf{K})$, that is, the *total variance* of the data in feature space, while respecting the distance constraints and keeping the resulting matrix centered. Maximizing $\mathrm{tr}(\mathbf{K})$ maximizes the interpoint squared distances for all points that are unconnected in the neighborhood graph, thereby unravelling the manifold.

In this paper we consider an alternative maximum entropy formalism of this problem. Since entropy is related to variance, we might expect a similar result in the quality of the resulting algorithm, but since maximum entropy also provides a probability distribution we should also obtain a probabilistic model with all the associated advantages (dealing with missing data, extensions to mixture models, fitting parameters by Bayesian methods, combining with other probabilistic models). Importantly, our interpretation will also enable us to relate our algorithm to other well known spectral techniques as they each turn out to approximate maximum entropy unfolding in some way.

2.1 Constraints from D Lead to a Density on Y

The maximum entropy formalism (see, e.g., Jaynes, 1986) allows us to derive a probability density given only a set of constraints on expectations under that density. These constraints may be derived from observation. In our case the observations will be squared distances between data points, but we will derive a density over $\mathbf{Y}$ directly (not over the squared distances). We will do this by looking to constrain the expected squared inter-point distances, $d_{i,j}$, of any two samples, $\mathbf{y}_{i,:}$ and $\mathbf{y}_{j,:}$, from the density. This means that while our observations may be only of the squared distances, $d_{i,j}$, the corresponding density will be over the data space that gives rise to those distances, $p(\mathbf{Y})$. Of course, once we have found the form of probability density we are free to directly model in the space $\mathbf{Y}$ or make use only of the squared distance constraints. Direct modeling in $\mathbf{Y}$ turns out to be equivalent to maximum likelihood. However, since we do not construct a density over the squared distance matrix, modeling based on that information alone should be thought of as maximum entropy under distance constraints rather than maximum likelihood.

In the maximum entropy formalism, we specify the density by a free form maximization of the entropy subject to the imposed expectation constraints. The constraints we use will correspond to the constraints applied to maximum variance unfolding: the expectations of the squared distances between two neighboring data points sampled from the model.

2.2 Maximum Entropy in Continuous Systems

The entropy of a continuous density is normally defined as the limit of a discrete system. The continuous distribution is discretized and we consider the limit as the discrete bin widths approach zero. However, as that limit is taken the term dependent on the bin width approaches ∞. Normally this is dealt with by ignoring that term and referring to the remaining term as differential entropy. However, the maximum entropy solution for this differential entropy turns out to be undefined. Jaynes (1986) proposes an alternative *invariant measure* to the entropy. For maximum entropy in continuous systems we maximize the negative Kullback Leibler divergence (KL divergence, Kullback and Leibler, 1951) between a base density, $m(\mathbf{Y})$, and the density of interest, $p(\mathbf{Y})$,

$$H = -\int p(\mathbf{Y}) \log \frac{p(\mathbf{Y})}{m(\mathbf{Y})} \mathrm{d}\mathbf{Y}.$$

Maximizing this measure is equivalent to minimizing the KL divergence between $p(\mathbf{Y})$ and the base density, $m(\mathbf{Y})$. Any choice of base density can be made, but the solution will be pulled towards the base density (through the minimization of the KL divergence). We choose a base density to be a very broad, spherical, Gaussian density with covariance $\gamma^{-1}\mathbf{I}$. This adds a new parameter, γ, to the system, but it will turn out that this parameter has little affect on our analysis. Typically it can be taken to zero or assumed small. The density

that minimizes the KL divergence under the constraints on the expectations is then

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\text{tr}\left(\gamma\mathbf{Y}\mathbf{Y}^\top\right)\right)\exp\left(-\frac{1}{2}\sum_i \sum_{j\in\mathcal{N}(i)} \lambda_{i,j} d_{i,j}\right),$$

where $\mathcal{N}(i)$ represents the set of neighbors of data point i, and $\mathbf{Y} = [\mathbf{y}_{1,:}, \ldots, \mathbf{y}_{n,:}]^\top \in \Re^{n\times p}$ is a *design matrix* containing our data. Note that we have introduced a factor of $-1/2$ in front of our Lagrange multipliers,[3] $\{\lambda_{i,j}\}$, for later notational convenience. We now define the matrix $\mathbf{\Lambda}$ to contain $\lambda_{i,j}$ if i is a neighbor of j and zero otherwise. This allows us to write the distribution[4] as

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\text{tr}\left(\gamma\mathbf{Y}\mathbf{Y}^\top\right) - \frac{1}{4}\text{tr}\left(\mathbf{\Lambda}\mathbf{D}\right)\right).$$

We now introduce a matrix $\mathbf{L}$, which has the form of a graph Laplacian. It is symmetric and constrained to have a null space in the constant vector, $\mathbf{L}\mathbf{1} = \mathbf{0}$. Its off diagonal elements are given by $-\mathbf{\Lambda}$ and its diagonal elements are given by

$$\ell_{i,i} = \sum_{j\in\mathcal{N}(i)} \lambda_{i,j}$$

to enforce the null space constraint. The null space constraint enables us to write

$$p(\mathbf{Y}) = \frac{|\mathbf{L}+\gamma\mathbf{I}|^{\frac{1}{2}}}{\tau^{\frac{np}{2}}}\exp\left(-\frac{1}{2}\text{tr}\left((\mathbf{L}+\gamma\mathbf{I})\mathbf{Y}\mathbf{Y}^\top\right)\right), \tag{3}$$

where for convenience we have defined $\tau = 2\pi$. We arrive here because the distance matrix is zero along the diagonal. This allows us to set the diagonal elements of $\mathbf{L}$ as we please without changing the value of $\text{tr}(\mathbf{L}\mathbf{D})$. Our choice to set them as the sum of the off diagonals gives the matrix a null space in the constant vector enabling us to use the fact that

$$\mathbf{D} = \mathbf{1}\text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)^\top - 2\mathbf{Y}\mathbf{Y}^\top + \text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)\mathbf{1}^\top$$

(where the operator $\text{diag}(\mathbf{A})$ forms a vector from the diagonal of $\mathbf{A}$) to write

$$-\text{tr}(\mathbf{\Lambda}\mathbf{D}) = \text{tr}(\mathbf{L}\mathbf{D}) = \text{tr}\left(\mathbf{L}\mathbf{1}\text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)^\top - 2\mathbf{L}\mathbf{Y}\mathbf{Y}^\top + \text{diag}\left(\mathbf{Y}\mathbf{Y}^\top\right)\mathbf{1}^\top\mathbf{L}\right) = -2\text{tr}\left(\mathbf{L}\mathbf{Y}\mathbf{Y}^\top\right),$$

which in turn allows us to recover (3). This probability distribution is a *Gaussian random field*. It can also be written as

$$p(\mathbf{Y}) = \prod_{j=1}^{p}\frac{|\mathbf{L}+\gamma\mathbf{I}|^{\frac{1}{2}}}{\tau^{\frac{n}{2}}}\exp\left(-\frac{1}{2}\mathbf{y}_{:,j}^\top(\mathbf{L}+\gamma\mathbf{I})\mathbf{y}_{:,j}\right),$$

which emphasizes the independence of the density across data features.

2.3 Gaussian Markov Random Fields

Multivariate Gaussian densities are specified by their mean, $\boldsymbol{\mu}$, and a covariance matrix, $\mathbf{C}$. A standard modeling assumption is that data is drawn independently from identical Gaussian densities. For this case the likelihood of the data, $p(\mathbf{Y})$, will be factorized across the individual data points,

$$p(\mathbf{Y}) = \prod_{i=1}^{n} p(\mathbf{y}_{i,:}) = \prod_{i=1}^{n}\mathcal{N}(\mathbf{y}_{i,:}|\boldsymbol{\mu},\mathbf{C})$$

3. We use λ for both Lagrange multipliers and eigenvalues, we hope that the meaning is clear from the context of use.
4. In our matrix notation the Lagrange multipliers and distances are appearing twice inside the trace, in matrices that are constrained symmetric, $\mathbf{\Lambda}$ and $\mathbf{D}$. The factor of $\frac{1}{4}$ replaces the factor of $\frac{1}{2}$ in the previous equation to account for this "double counting."

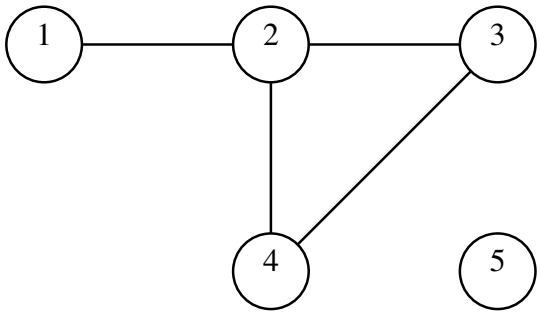


Figure 2: Graph representing conditional relationships between $p = 5$ features from a Gaussian Markov random field. Here the 5th feature is independent of the others. Feature 1 is conditionally dependent on 2, feature 2 is conditional dependent on 1, 3 and 4. Feature 3 is conditionally dependent on 2 and 4, and feature 4 is conditionally dependent on 2 and 3.

and the mean and covariance of the Gaussian are estimated by maximizing the log likelihood of the data. The covariance matrix is symmetric and positive definite. It contains $\frac{p(p+1)}{2}$ parameters. However, if the number of data points, n, is small relative to the number of features p, then the parameters may not be well determined. For this reason we might seek a representation of the covariance matrix which has fewer parameters. One option is a low rank representation,

$$\mathbf{C} = \mathbf{W}\mathbf{W}^\top + \mathbf{D},$$

where $\mathbf{D}$ is a diagonal matrix and $\mathbf{W} \in \Re^{p\times q}$. This is the representation underlying factor analysis, and if $\mathbf{D} = \sigma^2\mathbf{I}$, probabilistic principal component analysis (PPCA, Tipping and Bishop, 1999). For PPCA there are $pq+1$ parameters in the covariance representation.

An alternative approach, and one that is particularly popular in spatial systems, is to assume a sparse *inverse* covariance matrix, known as the precision matrix, or information matrix. In this representation we consider each feature to be a vertex in a graph. If two vertices are unconnected they are conditionally independent in the graph. In Figure 2 we show a simple example graph where the precision matrix is

$$\mathbf{K}^{-1} = \mathbf{J} = \begin{bmatrix} j_{1,1} & j_{1,2} & 0 & 0 & 0 \\ j_{2,1} & j_{2,2} & j_{2,3} & j_{2,4} & 0 \\ 0 & j_{3,2} & j_{3,3} & j_{3,4} & 0 \\ 0 & j_{4,2} & j_{4,3} & j_{4,4} & 0 \\ 0 & 0 & 0 & 0 & j_{5,5} \end{bmatrix}.$$

Zeros correspond to locations where there are no edges between vertices in the graph.

If each feature is constrained to only have K neighbors in the graph, then the inverse covariance (and correspondingly the covariance) is only parameterized by $Kp+p$ parameters. So the GRF provides an alternative approach to reducing the number of parameters in the covariance matrix.

2.4 Independence Over Data Features

The Gaussian Markov random field (GRF) for maximum entropy unfolding is unusual in that the independence is being expressed over data features (in the p-dimensional direction) instead of over data points (in the n-dimensional direction). This means that our model assumes that data *features* are independently and identically distributed (i.i.d.) given the model parameters. The standard assumption for Gaussian models is that data *points* we are expressing conditional probability densities between data points are i.i.d. given the parameters. This specification cannot be thought of as "the wrong way around" as it is merely a consequence

of the constraints we chose to impose on the maximum entropy solution. If those constraints are credible, then this model is also credible. This is not the first model proposed for which the independence assumptions are reversed. Such models have been proposed formerly in the context of semi-supervised learning (Zhu et al., 2003), probabilistic nonlinear dimensionality reduction (Lawrence, 2004, 2005) and in models that aim to discover structural form from data (Kemp and Tenenbaum, 2008).

2.5 Maximum Likelihood and Blessing of Dimensionality

Once the form of a maximum entropy density is determined, finding the Lagrange multipliers in the model is equivalent to maximizing the likelihood of the model, where the Lagrange multipliers are now considered to be parameters. The theory underpinning maximum likelihood is broad and well understood, but much of it relies on assuming independence across data points rather than data features. For example, maximum likelihood with independence across data points can be shown to be consistent by viewing the objective as a sample based approximation to the Kullback-Leibler (KL) divergence between the true data generating density, $\tilde{p}(\mathbf{y})$, and our approximation $p(\mathbf{y}|\boldsymbol{\theta})$ which in turn depends on parameters, $\boldsymbol{\theta}$. Taking the expectations under the generating density this KL divergence is written as

$$\mathrm{KL}\left(\tilde{p}(\mathbf{y}) \,\|\, p(\mathbf{y}|\boldsymbol{\theta})\right) = \int \tilde{p}(\mathbf{y}) \log \tilde{p}(\mathbf{y}) \mathrm{d}\mathbf{y} - \int \tilde{p}(\mathbf{y}) \log p(\mathbf{y}|\boldsymbol{\theta}) \mathrm{d}\mathbf{y}.$$

Given n sampled data points from $\tilde{p}(\mathbf{y})$, $\{\mathbf{y}_{i,:}\}$ we can write down a sample based approximation to the KL divergence in the form

$$\mathrm{KL}\left(\tilde{p}(\mathbf{y}) \,\|\, p(\mathbf{y})\right) \approx -\frac{1}{n}\sum_{i=1}^{n} \log p(\mathbf{y}_{i,:}|\boldsymbol{\theta}) + \text{const.},$$

where the constant term derives from the entropy of the generating density, which whilst unknown, does not depend on our model parameters. Since the sample based approximation is known to become exact in the large sample limit, and the KL divergence has a global minimum of zero *only* when the generating density and our approximation are identical, we know that, *if* the generating density falls within our chosen class of densities, maximum likelihood will reveal it in the large data limit. The global maximum of the likelihood will correspond to a global minimum of the KL divergence. Further, we can show that as we approach this limit, *if* the total number of parameters is fixed, our parameter values, $\boldsymbol{\theta}$, will become better determined (see, e.g., Wasserman, 2003, pg. 126). Since the number of parameters is often related to data dimensionality, p, this implies that for a given data dimensionality, p, we require a large number of data points, n, to have confidence we are approaching the large sample limit and our model's parameters will be well determined. We refer to this model set up as the *sampled-points* formalism.

The scenario described above does not apply for the situation where we have independence across data features. In this situation we construct an alternative consistency argument, but it is based around a density which describes correlation between data points instead of data features. This model is independent across data features. Models of this type can occur quite naturally. Consider the following illustrative example from cognitive science (Kemp and Tenenbaum, 2008). We wish to understand the relationship between different animals as more information about those animals' features is uncovered. There are 33 species in the group, and information is gained by unveiling features of the animals. A model which assumes independence over animals would struggle to incorporate additional feature information (such as whether or not the animal has feet, or whether or not it lives in the ocean). A model which assumes independence across features handles this situation naturally. However, to show the consistency of the model we must now think of our model as a generative model for data features, $\tilde{p}(\mathbf{y}')$, rather than data points. Our approximation to the KL divergence still applies,

$$\mathrm{KL}\left(\tilde{p}(\mathbf{y}') \,\|\, p(\mathbf{y}')\right) \approx -\frac{1}{p}\sum_{j=1}^{p} \log p(\mathbf{y}_{:,i}|\boldsymbol{\theta}) + \text{const.},$$

but now the sample based approximation is based on independent samples of features (in the animal example, whether or not it has a beak, or whether the animal can fly), instead of samples of data points. This model

will typically have a parameter vector that increases in size with the data set size, n (in that sense it is non-parametric), rather than the data dimensionality, p. The model is consistent as the number of features becomes large, rather than data points. For our Gaussian random field, the number of parameters increases linearly with the number of data points, but does not increase with the number of data (each datum requires $O(K)$ parameters to connect with K neighbors). However, as we increase features there is no corresponding increase in parameters. In other words as the number of features increases there is a clear *blessing of dimensionality*. We refer to this model set up as the *sampled-features* formalism.

There is perhaps a deeper lesson here in terms of how we should interpret such consistency results. In the sampled-points formalism, as we increase the number of data points, the parameters become better determined. In the sampled-features formalism, as we increase the number of features, the parameters become better determined. However, for consistency results to hold, the class of models we consider must include the actual model that generated the data. If we believe that "Essentially, all models are wrong, but some are useful" (Box and Draper, 1987, pg. 424) we may feel that encapsulating the right model within our class is a practical impossibility. Given that, we might pragmatically bias our choice somewhat to ensure utility of the resulting model. From this perspective, in the large p small n domain, the sampled-features formalism is attractive. A practical issue can arise though. If we wish to compute the likelihood of an out of sample data-point, we must first estimate the parameters associated with that new data point. This can be problematic. Of course, for the sampled-points formalism the same problem exists when you wish to include an out of sample data-feature in your model (such as in the animals example in Kemp and Tenenbaum, 2008). Unsurprisingly, addressing this issue for spectral methods is nontrivial (Bengio et al., 2004b).

2.5.1 PARAMETER GRADIENTS

We can find the parameters, $\mathbf{\Lambda}$, through maximum likelihood on the Gaussian Markov random field given in (3). Some algebra shows that the gradient of each Lagrange multiplier is given by,

$$\frac{\mathrm{d}\log p(\mathbf{Y})}{\mathrm{d}\lambda_{i,j}} = \frac{1}{2}\left\langle d_{i,j}\right\rangle_{p(\mathbf{Y})} - \frac{1}{2}d_{i,j},$$

where $\left\langle\right\rangle_{p(\cdot)}$ represents an expectation under the distribution $p(\cdot)$. This result is a consequence of the maximum entropy formulation: the Lagrange multipliers have a gradient of zero when the constraints are satisfied. To compute gradients we need the expectation of the squared distance given by

$$\left\langle d_{i,j}\right\rangle = \left\langle y_{i,:}^\top y_{i,:}\right\rangle - 2\left\langle y_{i,:}^\top y_{j,:}\right\rangle + \left\langle y_{j,:}^\top y_{j,:}\right\rangle,$$

which we can compute directly from the covariance matrix of the GRF, $\mathbf{K} = (\mathbf{L}+\gamma\mathbf{I})^{-1}$,

$$\left\langle d_{i,j}\right\rangle = \frac{p}{2}\left(k_{i,i} - 2k_{i,j} + k_{j,j}\right).$$

This is immediately recognized as a scaled version of the standard transformation between distances and similarities (see (2)). This relationship arises naturally in the probablistic model. Every GRF has an associated interpoint distance matrix. It is this matrix that is being used in CMDS. The machine learning community might interpret this as the relationship between distances in "feature space" and the kernel function. Note though that here (and also in MVU) each individual element of the kernel matrix *cannot* be represented only as a function of the corresponding two data points (i.e., we cannot represent them as $k_{i,j} = k(\mathbf{y}_{i,:}, \mathbf{y}_{j,:})$, where each $k_{i,j}$ is a function *only* of the i and jth data points). Given this we feel it is more correct to think of this matrix as a covariance matrix induced by our specification of the random field rather than a true Mercer kernel. We use the notation $k_{i,j}$ to denote an element of such a covariance (or similarity matrix) and only use $k(\cdot,\cdot)$ notation when the value of the similarity matrix can be explicitly represented as a Mercer kernel.

The Base Density Parameter. One role of the base density parameter, γ, is to ensure that the precision matrix is positive definite. Recall that the Laplacian has a null space in the constant vector, implying that $\mathbf{K1} = \gamma^{-1}$,

which becomes infinite as $\gamma \to 0$. This reflects an insensitivity of the covariance matrix to the data mean, and this in turn arises because that information is lost when we specify the expectation constraints only through interpoint distances. In practise though, $\mathbf{K}$ is always centred before its eigenvectors are extracted, $\mathbf{B} = \mathbf{HKH}$, resulting in $\mathbf{B1} = \mathbf{0}$ so γ has no effect on the final visualization. In some cases, it may be necessary to set γ to a small non-zero value to ensure stability of the inverse $\mathbf{L} + \gamma\mathbf{I}$. In these cases we set it to $\gamma = 1 \times 10^{-4}$ but in many of the comparisons we make to other spectral algorithms below we take it to be zero.

Number of Model Parameters. If K neighbors are used for each data point there are $O(Kn)$ parameters in the model, so the model is nonparametric in the sense that the number of parameters increases with the number of data. For the parameters to be well determined we require a large number of features, p, for each data point, otherwise we would need to regularize the model (see Section 3). This implies that the model is well primed for the so-called "large p small n domain."

Once the maximum likelihood solution is recovered the data can be visualized, as for MVU and kernel PCA, by looking at the eigenvectors of the centered covariance matrix $\mathbf{HKH}$. We call this algorithm maximum entropy unfolding (MEU).

Positive Definite Constraints. The maximum variance unfolding (MVU) algorithm maximizes the trace of the covariance matrix (given by the sum of its eigenvalues, $\{\lambda_i\}$),

$$\mathrm{tr}(\mathbf{K}) = \sum_{i=1}^{n} \lambda_i,$$

subject to constraints on the elements of $\mathbf{K}$ arising from the squared distances. These constraints are linear in the elements of $\mathbf{K}$. There is a further constraint on $\mathbf{K}$, that it should be positive semi-definite. This means MVU can be optimized through a a semi-definite program. In contrast MEU cannot be optimized through a semi-definite program because the objective is not linear in $\mathbf{K}$. This implies we need to find other approaches to maintaining the positive-definite constraint on $\mathbf{K}$. Possibilities include exploiting the fact that if the Lagrange multipliers are constrained to be positive the system is "attractive" and this guarantees a valid covariance (see, e.g., Koller and Friedman, 2009, pg. 255). Although now (as in a suggested variant of the MVU) the distance constraints would be inequalities. Another alternative would be to constrain $\mathbf{L}$ to be diagonally dominant through adjusting γ. We will also consider two further approaches in Section 2.7 and Section 3.

Non-linear Generalizations of PCA. Kernel PCA provides a non-linear generalization of PCA. This is achieved by 'kernelizing' the principal coordinate analysis algorithm: replacing data point inner products with a kernel function. Maximum variance unfolding and maximum entropy unfolding also provide non linear generalizations of PCA. For these algorithms, if we increase the neighborhood size to $K = n - 1$, then all squared distances implied by the GRF model are constrained to match the observed inter data point squared distances and $\mathbf{L}$ becomes non-sparse. Classical multidimensional scaling on the resulting squared distance matrix is known as principal coordinate analysis and is equivalent to principal component analysis (see Mardia et al., 1979).[5]

2.6 Relation to Laplacian Eigenmaps

Laplacian eigenmaps is a spectral algorithm introduced by Belkin and Niyogi (2003). In the Laplacian eigenmap procedure, a neighborhood is first defined in the data space. Typically this is done through nearest neighbor algorithms or defining all points within distance ε of each point to be neighbors. In Laplacian eigenmaps a symmetric sparse (possibly weighted) adjacency matrix, $\mathbf{A} \in \Re^{n \times n}$, is defined whose i, jth element, $a_{i,j}$ is non-zero if the ith and jth data points are neighbors. Belkin and Niyogi argue that a good *one*

5. In this case CMDS proceeds by computing the eigendecomposition of the centred negative squared distance matrix, which is the eigendecomposition of the centred inner product matrix as is performed for principal coordinate analysis.

dimensional embedding is one where the latent points, $\mathbf{X}$ minimize

$$E(\mathbf{X}) = \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}(x_i - x_j)^2.$$

For a multidimensional embedding we can rewrite this objective in terms of the squared distance between two latent points, $\delta_{i,j} = \left\|\mathbf{x}_{i,:} - \mathbf{x}_{j,:}\right\|_2^2$, as

$$E(\mathbf{X}) = \frac{1}{4}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j}\delta_{i,j}.$$

The motivation behind this objective function is that neighboring points have non-zero entries in the adjacency matrix, therefore their inter point squared distances in latent space need to be minimized. In other words points which are neighbors in data space will be kept close together in the latent space. The objective function can be rewritten in matrix form as

$$E(\mathbf{X}) = \frac{1}{4}\mathrm{tr}\left(\mathbf{A}\boldsymbol{\Delta}\right).$$

Squared Euclidean distance matrices of this type can be rewritten in terms of the original vector space by introducing the Laplacian matrix. Introducing the degree matrix, $\mathbf{D}$, which is diagonal with entries, $d_{i,i} = \sum_j \mathbf{A}_{i,j}$ the Laplacian associated with the neighborhood graph can be written

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

and the error function can now be written directly in terms of the latent coordinates,

$$E(\mathbf{X}) = \frac{1}{2}\mathrm{tr}\left(\mathbf{L}\mathbf{X}\mathbf{X}^\top\right)$$

by exploiting the null space of the Laplacian ($\mathbf{L1} = \mathbf{0}$) as we saw in Section 2.1.

Let us consider the properties of this objective. Since the error function is in terms of interpoint distances, it is insensitive to translations of the embeddings. The mean of the latent projections is therefore undefined. Further, there is a trivial solution for this objective. If the latent points are all placed on top of one another the interpoint distance matrix will be all zeros. To prevent this collapse Belkin and Niyogi suggest that each dimension of the latent representation is constrained,

$$\mathbf{x}_{:,i}^\top \mathbf{D}\mathbf{x}_{:,i} = 1.$$

Here the degree matrix, $\mathbf{D}$, acts to scale each data point so that points associated with a larger neighborhood are pulled towards the origin.

Given this constraint the objective function is minimized for a q dimensional space by the generalized eigenvalue problem,

$$\mathbf{L}\mathbf{u}_i = \lambda_i \mathbf{D}\mathbf{u}_i,$$

where λ_i is an eigenvalue and $\mathbf{u}_i$ is its associated eigenvector. The smallest eigenvalue is zero and is associated with the constant eigenvector. This eigenvector is discarded, whereas the eigenvectors associated with the next q smallest eigenvalues are retained for the embedding. So we have,

$$\mathbf{x}_{:,i} = \mathbf{u}_{i+1} \quad \text{for} \quad i = 1..q$$

if we assume that eigenvalues are ordered according to magnitude with the smallest first.

Note that the generalized eigenvalue problem underlying Laplacian eigenmaps can be readily converted to the related, symmetric, eigenvalue problem.

$$\hat{\mathbf{L}}\mathbf{v}_i = \lambda_i \mathbf{v}_i \tag{4}$$

where $\hat{\mathbf{L}}$ is the *normalized Laplacian matrix*,

$$\hat{\mathbf{L}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$$

and the relationship between the eigenvectors is through scaling by the degree matrix, $\mathbf{v}_i = \mathbf{D}^{\frac{1}{2}}\mathbf{u}_i$ (implying $\mathbf{v}_i^\top \mathbf{v}_i = 1$). The eigenvalues remain unchanged in each case.

2.6.1 PARAMETERIZATION IN LAPLACIAN EIGENMAPS

In Laplacian eigenmaps the adjacency matrix can either be unweighted (Belkin and Niyogi refer to this as the simple-minded approach) or weighted according to the distance between two data points,

$$a_{i,j} = \exp\left(-\frac{\left\|\mathbf{y}_{i,:} - \mathbf{y}_{j,:}\right\|_2^2}{2\sigma^2}\right), \tag{5}$$

which is justified by analogy between the discrete graph Laplacian and its continuous equivalent, the Laplace Beltrami operator (Belkin and Niyogi, 2003).

2.6.2 RELATING LAPLACIAN EIGENMAPS TO MEU

The relationship of MEU to Laplacian eigenmaps is starting to become clear. In Laplacian eigenmaps a graph Laplacian is specified across the data points just as in maximum entropy unfolding. In classical multidimensional scaling, as applied in MEU and MVU, the eigenvectors associated with the largest eigenvalues of the centred covariance matrix,

$$\mathbf{B} = \mathbf{H}\left(\mathbf{L} + \gamma\mathbf{I}\right)^{-1}\mathbf{H} \tag{6}$$

are used for visualization. In Laplacian eigenmaps the smallest eigenvectors of $\mathbf{L}$ are used, disregarding the eigenvector associated with the null space.

Note that if we define the eigendecomposition of the covariance in the GRF as

$$\mathbf{K} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$$

it is easy to show that the eigendecomposition of the associated Laplacian matrix is

$$\mathbf{L} = \mathbf{U}\left(\mathbf{\Lambda}^{-1} - \gamma\mathbf{I}\right)\mathbf{U}^\top.$$

We know that the smallest eigenvalue of $\mathbf{L}$ is zero with a constant eigenvector. That implies that the largest eigenvalue of $\mathbf{K}$ is γ^{-1} and is associated with a constant eigenvector. However, we do not use the eigenvectors of $\mathbf{K}$ directly. We first apply the centering operation in (6). This projects out the constant eigenvector, but leaves the remaining eigenvectors and eigenvalues intact.

To make the analogy with Laplacian eigenmaps direct we consider the formulation of its eigenvalue problem with the normalized graph Laplacian as given in (4). Substituting the normalized graph Laplacian into our covariance matrix, $\mathbf{K}$, we see that for Laplacian eigenmaps we are visualizing a Gaussian random field with a covariance as follows,

$$\mathbf{K} = (\hat{\mathbf{L}} + \gamma\mathbf{I})^{-1}.$$

Naturally we could also consider a variant of the algorithm which used the unnormalized Laplacian directly, $\mathbf{K} = (\mathbf{L} + \gamma\mathbf{I})^{-1}$. Under the Laplacian eigenmap formulation that would be equivalent to preventing the collapse of the latent points by constraining $\mathbf{x}_{:,i}^\top\mathbf{x}_{:,i} = 1$ instead of $\mathbf{x}_{:,i}^\top\mathbf{D}\mathbf{x}_{:,i} = 1$.

This shows the relationship between the eigenvalue problems for Laplacian eigenmaps and CMDS. The principal eigenvalues of $\mathbf{K}$ will be the smallest eigenvalues of $\mathbf{L}$. The very smallest eigenvalue of $\mathbf{L}$ is zero and associated with the constant eigenvector. However, in CMDS this would be removed by the centering operation and in Laplacian eigenmaps it is discarded. Once the parameters of the Laplacian have been set CMDS is being performed to recover the latent variables in Laplacian eigenmaps.

2.6.3 LAPLACIAN EIGENMAPS SUMMARY

The Laplacian eigenmaps procedure does not fit parameters through maximum likelihood. It uses analogies with the continuous Laplace Beltrami operator to set them via the Gaussian-like relationship in (5). This means that the local distance constraints are not a feature of Laplacian eigenmaps. The implied squared distance matrix used for CMDS will not preserve the interneighbor distances as it will for MVU and MEU. In

fact since the covariance matrix is never explicitly computed it is not possible to make specific statements about what these distances will be in general. However, Laplacian eigenmaps gains significant computational advantage by not representing the covariance matrix explicitly. No matrix inverses are required in the algorithm and the resulting eigenvalue problem is sparse. This means that Laplacian eigenmaps can be applied to much larger data sets than would be possible for MEU or MVU.

2.7 Relation of MEU to Locally Linear Embedding

The locally linear embedding (LLE Roweis and Saul, 2000) is a dimensionality reduction that was originally motivated by the idea that a non-linear manifold could be approximated by small linear patches. If the distance between data points is small relative to the curvature of the manifold at a particular point, then the manifold encircling a data point and its nearest neighbors may be approximated locally by a linear patch. This idea gave rise to the locally linear embedding algorithm. First define a local neighborhood for each data point and find a set of linear regression weights that allows each data point to be reconstructed by its neighbors. Considering the ith data point, $\mathbf{y}_{i,:}$ and a vector of reconstruction weights, $\mathbf{w}_{:,i}$, associated with that data point a standard least squares regression objective takes the form,

$$E(\mathbf{w}_{:,i}) = \frac{1}{2}\left\|\mathbf{y}_{i,:} - \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:} w_{j,i}\right\|_2^2, \tag{7}$$

for each data point. Here the sum over the reconstruction weights, $\mathbf{w}_{:,j}$ is restricted to data points, $\left\{\mathbf{y}_{j,:}\right\}_{j\in\mathcal{N}(()i)}$, which are in the neighborhood of the data point of interest, $\mathbf{y}_{i,:}$. Roweis and Saul point out that the objective function in (7) is invariant to rotation and rescaling of the data. If we rotate each data vector in (7) the objective does not change. If data are rescaled, for example, multiplied by a factor α, then the objective is simply rescaled by a factor α^2. However, the objective is not invariant to translation. For example if we were to translate the data, $\hat{\mathbf{y}}_{i,:} = \mathbf{y}_{i,:} - \boldsymbol{\mu}$, where $\boldsymbol{\mu}$ could be the sample mean of our data set (or any other translation), we obtain the following modified objective,

$$E(\mathbf{w}_{:,i}) = \frac{1}{2}\left\|\hat{\mathbf{y}}_{i,:} + \boldsymbol{\mu} - \sum_{j\in\mathcal{N}(i)} \hat{\mathbf{y}}_{j,:} w_{j,i} - \boldsymbol{\mu} \sum_{j\in\mathcal{N}(i)} w_{j,i}\right\|_2^2,$$

which retains a dependence on $\boldsymbol{\mu}$. Roweis and Saul point out that if we constrain $\sum_{j\in\mathcal{N}(i)} w_{j,i} = 1$ the terms involving $\boldsymbol{\mu}$ cancel and we recover the original objective. Imposing this constraint on the regression weights (which can also be written $\mathbf{w}_{:,i}^\top \mathbf{1} = 1$), ensures the objective is translation invariant.

To facilitate the comparison with the maximum entropy unfolding algorithm we now introduce an alternative approach to enforcing translation invariance. Our approach generalizes the LLE algorithm. First of all we introduce a new matrix $\mathbf{M}$. The sparsity pattern of this matrix should match that of $\mathbf{W}$ for off diagonal elements. We then set the diagonal elements of each row of $\mathbf{M}$ to be the negative sum of the off diagonal columns, so we have $m_{i,i} = -\sum_{j\in\mathcal{N}(i)} w_{j,i}$. We can then rewrite the objective in (7) as,

$$E(\mathbf{w}_{:,i}) = \frac{1}{2}\left\|\mathbf{Y}^\top \mathbf{m}_{:,i}\right\|_2^2 = \mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i},$$

which is identical to (7) if $m_{i,i}$ is further constrained to 1. However, even if this constraint is not imposed, the translational invariance is retained. This is clear if we rewrite the objective in terms of the non-zero elements of $\mathbf{m}_{:,i}$,

$$\begin{aligned} E(\mathbf{m}_{:,i}) &= \frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} + \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:}\frac{m_{j,i}}{m_{i,i}}\right\|_2^2 \\ &= \frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} - \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:} w_{j,i}\right\|_2^2 \end{aligned}$$

where

$$w_{j,i} = -\frac{m_{j,i}}{m_{i,i}}$$

and by definition of $m_{i,i}$ we have $\sum_{j\in\mathcal{N}(i)} w_{j,i} = 1$. We now see that up to a scalar factor, $m_{i,i}^2$, this equation is identical to (7).

This form of the objective also shows us that $m_{i,i}$ has the role of scaling each data point's contribution to the overall objective function (rather like the degree, $d_{i,i}$ would do in the unnormalized variant of Laplacian eigenmaps we discussed in Section 2.6.2).

The objective function is a least squares formulation with particular constraints on the regression weights, $\mathbf{m}_{:,i}$. As with all least squares regressions, there is an underlying probabilistic interpretation of the regression which suggests Gaussian noise. In our objective function the variance of the Gaussian noise for the ith data point is given by $m_{i,i}^{-2}$. We can be a little more explicit about this by writing down the error as the negative log likelihood of the equivalent Gaussian model. This then includes a normalization term, $\log m_{i,i}^2$, which is zero in standard LLE where $m_{i,i}^2 = 1$,

$$\begin{aligned}
E(\mathbf{w}_{:,i}) &= -\log \mathcal{N}\left(\mathbf{y}_{i,:} | \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:}\hat{m}_{j,i}, m_{i,i}^{-2}\right) \\
&= \frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} - \sum_{j\in\mathcal{N}(i)} \mathbf{y}_{j,:}\hat{m}_{j,i}\right\|_2^2 - \frac{1}{2}\log m_{i,i}^2 + \text{const} \\
&= \frac{1}{2}\mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i} - \frac{1}{2}\log m_{i,i}^2 + \text{const.}
\end{aligned} \tag{8}$$

The overall objective is the sum of the objectives for each column of $\mathbf{W}$. Under the probabilistic interpretation this is equivalent to assuming independence between the individual regressions. The objective can be written in matrix form as

$$E(\mathbf{W}) = \frac{1}{2}\sum_{i=1}^{n} \mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i} - \frac{1}{2}\sum_{i=1}^{n}\log m_{i,i}^2 + \text{const.} \tag{9}$$

Recalling that our definition of $\mathbf{M}$ was in terms of $\mathbf{W}$, we now make that dependence explicit by parameterizing the objective function only in terms of the non-zero elements of $\mathbf{W}$. To do this we introduce a 'croupier matrix' $\mathbf{S}_i \in \Re^{n\times k_i}$, where k_i is the size of the i data point's neighborhood. This matrix will distribute the non-zero elements of $\mathbf{W}$ appropriately into $\mathbf{M}$. It is defined in such a way that for the ith data point we have $\mathbf{m}_{:,i} = \mathbf{S}_i\mathbf{w}_i$, where we use the shorthand $\mathbf{w}_i = \mathbf{w}_{\mathcal{N}(i),i}$. In other words $\mathbf{w}_i$ is the vector of regression weights being used to reconstruct the ith data point. It contains the non-zero elements from the ith column of $\mathbf{W}$. The matrix $\mathbf{S}_i$ is constructed by setting all elements in its ith row to -1 (causing $m_{i,i}$ to be the negative sum of the elements of $\mathbf{w}_i$ as defined). Then we set $s_{\ell,j}$ to 1 if ℓ is the jth neighbor of the data point i and zero otherwise. We can then rewrite the objective function for the data set as

$$E(\mathbf{W}) = \frac{1}{2}\sum_{i=1}^{n} \mathbf{w}_i^\top \mathbf{S}_i^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{S}_i\mathbf{w}_i - \frac{1}{2}\sum_{i=1}^{n}\log \mathbf{w}_i^\top \mathbf{1}\mathbf{1}^\top \mathbf{w}_i + \text{const},$$

A fixed point can be found by taking gradients with respect to $\mathbf{w}_i$,

$$\frac{\mathrm{d}E(\mathbf{W})}{\mathrm{d}\mathbf{w}_i} = \mathbf{S}_i^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{S}_i\mathbf{w}_i - \frac{1}{\mathbf{w}_i^\top \mathbf{1}}\mathbf{1}$$

which implies that the direction of $\mathbf{w}_i$ is given by

$$\mathbf{w}_i \propto \mathbf{C}_i^{-1}\mathbf{1}$$

where $\mathbf{C}_i = \mathbf{S}_i^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{S}_i$ has been called the "local covariance matrix" by Roweis and Saul (2000), removing the croupier matrix we can express the local covariance matrix in the same form given by Roweis and Saul (2000),

$$\mathbf{C}_i = \sum_{j \in \mathcal{N}(i)} (\mathbf{y}_{j,:} - \mathbf{y}_{i,:})(\mathbf{y}_{j,:} - \mathbf{y}_{i,:})^\top.$$

For standard LLE the magnitude of the vector $\mathbf{w}_i$ is set by the fact that $\mathbf{1}^\top \mathbf{w}_i = 1$. In our alternative formulation we can find the magnitude of the vector through differentiation of (8) with respect to $m_{i,i}^2$ leading to the following fixed point

$$m_{i,i}^{-2} = \left\| \mathbf{y}_{i,:} - \sum_{j \in \mathcal{N}(i)} \mathbf{y}_{j,:} \hat{m}_{j,i} \right\|_2^2,$$

where $\hat{m}_{j,i} = -m_{j,i}/m_{i,i}$. This update shows explicitly that $m_{i,i}$ estimates the precision with which each individual regression problem is solved.

2.7.1 DETERMINING THE EMBEDDING IN LLE

If the data is truly low dimensional, then we might expect that the local linear relationships between neighbors continue to hold for a data set $\mathbf{X}$, of lower dimensionality, $q < p$, than the original data $\mathbf{Y}$. The next step in the LLE procedure is to find this data set. We do this by minimizing the objective function in (9) with respect to this new, low dimensional data set. Writing the objective in terms of this reduced dimensional data set, $\mathbf{X}$, we have

$$\begin{aligned} E(\mathbf{X}) &= \frac{1}{2} \sum_{i=1}^{n} \mathbf{m}_{:,i}^\top \mathbf{X}\mathbf{X}^\top \mathbf{m}_{:,i} + \text{const} \\ &= \frac{1}{2} \text{tr}\left(\mathbf{M}\mathbf{M}^\top \mathbf{X}\mathbf{X} \right) + \text{const} \\ &= \frac{1}{2} \sum_{i=1}^{n} \mathbf{x}_{i,:}^\top \mathbf{M}\mathbf{M}^\top \mathbf{x}_{i,:} + \text{const}. \end{aligned}$$

Clearly the objective function is trivially minimized by setting $\mathbf{X} = \mathbf{0}$, so to avoid this solution a constraint is imposed that $\mathbf{X}^\top \mathbf{X} = \mathbf{I}$. This leads to an eigenvalue problem of the form

$$\mathbf{M}\mathbf{M}^\top \mathbf{u}_i = \lambda_i \mathbf{u}_i.$$

Here the smallest $q+1$ eigenvalues are extracted. The smallest eigenvector is the constant eigenvector and is associated with an eigenvalue of zero. This is because, by construction, we have set $\mathbf{M}\mathbf{M}^\top \mathbf{1} = \mathbf{0}$. The next q eigenvectors are retained to make up the low dimensional representation so we have

$$\mathbf{x}_{:,i} = \mathbf{u}_{i+1} \quad \text{for} \quad i = 1..q.$$

Extracting the latent coordinates in LLE is extremely similar to the process suggested in Laplacian eigenmaps, despite different motivations. Though in the LLE case the constraint on the latent embeddings is not scaled by the degree matrix. The procedure is also identical to that used in classical multidimensional scaling, and therefore matches that used in MVU and MEU, although again the motivation is different. Rather than distance matching, as suggested for CMDS, in LLE we are looking for a 'representative,' low dimensional, data set.

2.7.2 RELATING LLE TO MEU

We can see the similarity now between LLE and the Laplacian eigenmaps. If we interpret $\mathbf{M}\mathbf{M}^\top$ as a Laplacian we notice that the eigenvalue problem being solved for LLE to recover the embedding is similar to that being solved in Laplacian eigenmaps. The key difference between LLE and Laplacian eigenmaps is the manner in which the Laplacian is parameterized.

When introducing MEU we discussed how it is necessary to constrain the Laplacian matrix to be positive definite (see Section 2.5.1). One way of doing this is to assume the Laplacian factorizes as

$$\mathbf{L} = \mathbf{M}\mathbf{M}^\top$$

where $\mathbf{M}$ is non-symmetric. If $\mathbf{M}$ is constrained so that $\mathbf{M}^\top \mathbf{1} = \mathbf{0}$ then we will also have $\mathbf{L}\mathbf{1} = \mathbf{0}$. As we saw in the last section, this constraint is easily achieved by setting the diagonal elements $m_{i,i} = -\sum_{j \in \mathcal{N}(i)} m_{j,i}$. Then if we force $m_{j,i} = 0$ if $j \notin \mathcal{N}(i)$ we will have a Laplacian matrix which is positive semidefinite without need for any further constraint on $\mathbf{M}$. The sparsity pattern of $\mathbf{L}$ will, however, be different from the pattern of $\mathbf{M}$. The entry for $\ell_{i,j}$ will only be zero if there are no shared neighbors between i and j.

We described above how the parameters of LLE, $\mathbf{W}$, are chosen to reflect locally linear relationships between neighboring data points. Here we show that this algorithm is actually approximate maximum likelihood in the MEU model. Indeed LLE turns out to be the specific case of maximum entropy unfolding where:

1. The diagonal sums, $m_{i,i}$, are further constrained to unity.

2. The parameters of the model are optimized by maximizing the *pseudolikelihood* of the resulting GRF.

As we described in our introduction to LLE, traditionally the reconstruction weights, $\mathbf{w}_i$, are constrained to sum to 1. If this is the case then by our definition of $\mathbf{M}$ we can write $\mathbf{M} = \mathbf{I} - \mathbf{W}$. The sparsity pattern of $\mathbf{W}$ matches $\mathbf{M}$, apart from the diagonal of $\mathbf{W}$ which is set to zero. These constraints mean that $(\mathbf{I} - \mathbf{W})^\top \mathbf{1} = \mathbf{0}$. The LLE algorithm (Roweis and Saul, 2000) proscribes that the smallest eigenvectors of $(\mathbf{I} - \mathbf{W})(\mathbf{I} - \mathbf{W})^\top = \mathbf{M}\mathbf{M}^\top = \mathbf{L}$ are used with the constant eigenvector associated with the eigenvalue of 0 being discarded. This matches the CMDS procedure as applied to the MEU model, where the eigenvectors of $\mathbf{L}$ are computed with the smallest eigenvector discarded through the centering operation.

2.7.3 PSEUDOLIKELIHOOD APPROXIMATION

To see how pseudolikelihood in the MEU model results in the LLE procedure we firstly review the pseudolikelihood approximation (Besag, 1975).

The Hammersley-Clifford theorem (Hammersley and Clifford, 1971) states that for a Markov random field (of which our Gaussian random field is one example) the joint probability density can be represented as a factorization over the cliques of the graph. In the Gaussian random field underlying maximum entropy unfolding the cliques are defined by the neighbors of each data point and the relevant factorization is

$$p(\mathbf{Y}) \propto \prod_{i=1}^{n} p(\mathbf{y}_{i,:} | \mathbf{Y}_{\setminus i}), \tag{10}$$

where $\mathbf{Y}_{\setminus i}$ represents all data other than the ith point and in practice each conditional distribution is typically only dependent on a sub-set of $\mathbf{Y}_{\setminus i}$ (as defined by the neighborhood). As we will see, these conditional distributions are straightforward to write out for maximum entropy unfolding, particularly in the case where we have assumed the factorization of the Laplacian, $\mathbf{L} = \mathbf{M}\mathbf{M}^\top$.

The pseudolikelihood assumes that the proportionality in (10) can be ignored and that the approximation

$$p(\mathbf{Y}) \approx \prod_{i=1}^{n} p(\mathbf{y}_{i,:} | \mathbf{Y}_{\setminus i})$$

is valid.

To see how the decomposition into cliques applies in the factorizable MEU model first recall that

$$\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^\top \mathbf{M}\mathbf{M}^\top\right) = \sum_{i=1}^{n} \mathbf{m}_{:,i}^\top \mathbf{Y}\mathbf{Y}^\top \mathbf{m}_{:,i}$$

so for the MEU model we have[6]

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^\top\mathbf{M}\mathbf{M}^\top\right)\right) = \prod_{i=1}^{n}\exp\left(-\frac{1}{2}\mathbf{m}_{:,i}^\top\mathbf{Y}\mathbf{Y}^\top\mathbf{m}_{:,i}\right).$$

This provides the necessary factorization for each conditional density which can now be rewritten as

$$p(\mathbf{y}_{i,:}|\mathbf{Y}_{\setminus i}) = \left(\frac{m_{i,i}^2}{\tau}\right)^{\frac{p}{2}}\exp\left(-\frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:} + \sum_{j\in\mathcal{N}(i)}\frac{m_{j,i}}{m_{i,i}}\mathbf{y}_{j,:}\right\|_2^2\right).$$

Optimizing the pseudolikelihood is equivalent to optimizing the conditional density for each of these cliques independently,

$$\log p(\mathbf{Y}) \approx \sum_{i=1}^{n}\log p(\mathbf{y}_{i,:}|\mathbf{Y}_{\setminus i}),$$

which is equivalent to solving n independent regression problems with a constraint on the regression weights that they sum to one. This is exactly the optimization suggested in (9). In maximum entropy unfolding the constraint arises because the regression weights are constrained to be $w_{j,i}/m_{i,i}$ and $m_{i,i} = \sum_{j\in\mathcal{N}(i)} w_{j,i}$. In standard LLE a further constraint is placed that $m_{i,i} = 1$ which implies none of these regression problems should be solved to a greater precision than another. However, as we derived above, LLE is also applicable even if this further constraint is not imposed.

Locally linear embeddings make use of the pseudolikelihood approximation to parameter determination Gaussian random field. Underpinning this is a neat way of constraining the Laplacian to be positive semidefinite by assuming a factorized form. The pseudolikelihood also allows for relatively quick parameter estimation by ignoring the partition function from the actual likelihood. This again removes the need to invert to recover the covariance matrix and means that LLE can be applied to larger data sets than MEU or MVU. However, the sparsity pattern in the Laplacian for LLE will not match that used in the Laplacian for the other algorithms due to the factorized representation.

2.7.4 WHEN IS THE PSEUDOLIKELIHOOD VALID IN LLE?

The pseudolikelihood was motivated by Besag (1975) for computational reasons. However, it obtains speed ups whilst sacrificing accuracy: it does not make use of the correct form of the normalization of the Gaussian random field. For a Gaussian model the normalization is the determinant of the covariance matrix,

$$|\mathbf{K}| = |\mathbf{L} + \gamma\mathbf{I}|^{-1}.$$

However, under particular circumstances the approximation is exact. Here we quickly review an occasion when this occurs.

Imagine if we force $\mathbf{M}$ to be lower triangular, that is, we have a Cholesky form for our factorization of $\mathbf{L} = \mathbf{M}\mathbf{M}^\top$. The interpretation here is now that $\mathbf{M}$ is a weighted adjacency matrix from a *directed acyclic graph*. When constructing the LLE neighborhood the triangular form for this matrix can be achieved by first imposing an ordering on the data points. Then, when seeking the nearest K neighbors for i, we only consider a candidate data point j if $j > i$. In the resulting directed acyclic graph the neighbors of each data point are its *parents*.[7] The weighting of the edge between node j and its parent, i, is given by the (i,j)th element of $\mathbf{M}$. To enforce the constraint that $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$ the diagonal elements of $\mathbf{M}$ are given by the negative sum of the off diagonal elements from each column (i.e., the sum of their parents). Note that the last data point (index n) has no parents and so the (n,n)th element of $\mathbf{M}$ is zero.

6. Here we have ignored the term arising from the base density, $\mathrm{tr}\left(\gamma\mathbf{Y}\mathbf{Y}^\top\right)$. It also factorizes, but it does not affect the dependence of the pseudolikelihood on $\mathbf{W}$.

7. Note that parents having a *lower* index than children is the reverse of the standard convention. However, here it is necessary to maintain the structure of the Cholesky decomposition.

Now we use the fact that the log determinant of $\mathbf{L}$ is given by $\log\left|\mathbf{M}\mathbf{M}^\top\right| = \sum_i \log m_{i,i}^2$ if $\mathbf{M}$ is lower triangular. This means that for the particular structure we have imposed on the covariance the true log likelihood *does* factorize into n independent regression problems,

$$\begin{aligned}\log p(\mathbf{Y}) &= \frac{\left|\mathbf{M}\mathbf{M}^\top\right|^{\frac{1}{2}}}{\tau^{\frac{n}{2}}}\exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}\mathbf{M}\mathbf{M}^\top\right)\right)\\ &= \prod_i^n \frac{m_{i,i}^2}{\tau^{\frac{1}{2}}}\exp\left(-\frac{m_{i,i}^2}{2}\left\|\mathbf{y}_{i,:}+\sum_{j\in\mathcal{N}(i)}\frac{m_{j,i}}{m_{i,i}}\mathbf{y}_{j,:}\right\|_2^2\right).\end{aligned}$$

The representation corresponds to a Gaussian random field which is constructed from specifying the directed relationship between the nodes in the graph. We can derive the Gaussian random field by considering a series of conditional relationships,

$$p(\mathbf{y}_{i,:}|\mathbf{Y}_{\setminus i}) = p(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:})$$

where our notation here is designed to indicate that the model is constrained so that the density associated with each data point, $\mathbf{y}_{i,:}$, is only dependent on data points with an index greater than i, a matrix we denote with $\mathbf{Y}_{j>i,:}$. This constraint is enforced by our demand that the only potential neighbors (parents in the directed graph) are those data points with an index greater than i. The undirected system can now be produced by taking the conditional densities of each data point,

$$p(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:}) = \mathcal{N}\left(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:}^\top\mathbf{m}_{j>i,i}, m_{i,i}^{-2}\mathbf{I}\right),$$

and multiplying them together[8]

$$p(\mathbf{Y}) = \prod_{i=1}^n p(\mathbf{y}_{i,:}|\mathbf{Y}_{j>i,:}),$$

to form the joint density. Note that the nth data point has no parents so we can write $p(\mathbf{y}_{n,:}|\mathbf{Y}_{j>n,:}) = p(\mathbf{y}_{n,:})$. However, since we defined $m_{j,j} = -\sum_{i>j} m_{i,j}$ the model as it currently stands associates an infinite variance with this marginal density ($m_{n,n} = 0$). This is a consequence of the constraint $\mathbf{M}^\top\mathbf{1} = \mathbf{0}$. The problem manifests itself when computing the log determinant of $\mathbf{L}$, $\log\left|\mathbf{M}\mathbf{M}^\top\right| = \sum_i \log m_{i,i}^2$ to develop the log likelihood. The last term in this sum is now $\log m_{n,n}^2 = \log 0$. As for the standard model this is resolved if we include the $\gamma\mathbf{I}$ term from the base density when computing the determinant, but this destroys the separability of the determinant computation. If the likelihood is required the value $m_{n,n}$ could be set to a small value, or optimized, relaxing the constraint on $\mathbf{M}$.

We call the algorithm based on the above decomposition acyclic locally linear embedding (ALLE). A weakness for the ALLE is the need to specify an ordering for the data. The ordering specifies which points can be neighbors and different orderings will lead to different results. Ideally one might want to specify the sparsity pattern in $\mathbf{L}$ and derive the appropriate sparsity structure for $\mathbf{M}$. However, given a general undirected graph it is not possible, in general, to find an equivalent directed acyclic graph. This is because co-parents in the directed graph gain an edge in the undirected graph, but the weight associated with this edge cannot be set independently of the weights associated with the edges between those co-parents and their children.

2.7.5 LLE AND PCA

LLE is motivated by considering local linear embeddings of the data, although interestingly, as we increase the neighborhood size to $K = n-1$ we do not recover PCA, which is known to be the optimal linear embedding of the data under linear Gaussian constraints. The fact that LLE is optimizing the pseudolikelihood makes it clear why this is the case. In contrast the MEU algorithm, which LLE approximates, does recover PCA when $K = n-1$. The ALLE algorithm also recovers PCA.

8. The idea of modelling a joint distribution by approximating its conditionals was described by Li and Stephens (2003) in the context of haplotype modeling. Generic density estimators based on this idea appeared earlier (e.g., Frey et al., 1996), using sigmoid belief networks (Neal, 1992). Larochelle and Murray (2011) recently extended these type of models with latent variables for modeling binary data in the neural autoregressive distribution estimator.

2.8 Relation to Isomap

The isomap algorithm (Tenenbaum et al., 2000) more directly follows the CMDS framework. In isomap (Tenenbaum et al., 2000) a sparse graph of distances is created between all points considered to be neighbors. This graph is then filled in for all non-neighboring points by finding the shortest distance between any two neighboring points in the graph (along the edges specified by the neighbors). The resulting matrix is then element-wise squared to give a matrix of square distances which is then processed in the usual manner (centering and multiplying by -0.5) to provide a similarity matrix for multidimensional scaling. Compare this to the situation for MVU and MEU. Both MVU and MEU can be thought of as starting with a sparse graph of (squared) distances. The other distances are then filled in by either maximizing the trace of the associated covariance or maximizing the entropy. Importantly, though, the interneighbor distances in this graph are preserved (through constraints imposed by Lagrange multipliers) just like in isomap. For both MVU and MEU the covariance matrix, $\mathbf{K}$, is guaranteed positive semidefinite because the distances are implied by an underlying covariance matrix that is constrained positive definite. For isomap the shortest path algorithm is effectively approximating the distances between non-neighboring points. This can lead to an implied covariance matrix which has negative eigenvalues (see Weinberger et al., 2004). The algorithm is still slower than LLE and Laplacian eigenmaps because it requires a dense eigenvalue problem and the application of a shortest path algorithm to the graph provided by the neighbors.

3. Estimating Graph Structure

The relationship between spectral dimensionality reduction algorithms and Gaussian random fields now leads us to consider a novel approach to dimensionality reduction. Recently it has been shown that the structure of a Gaussian random field can be estimated through using L1 shrinkage on the parameters of the inverse covariance (see Hastie et al., 2009, Chapter 17). These sparse graph estimators are attractive as the regularization allows some structure determination. In other words, rather than relying entirely on the structure provided by the K nearest neighbors in data space, we can estimate this structure from the data. We call the resulting class of approaches Dimensionality reduction through Regularization of the Inverse covariance in the Log Likelihood (DRILL).

Before introducing the method, we need to first re-derive the maximum entropy approach by constraining the second moment of neighboring data points to equal the empirical observation instead of the expected inter data point squared distances. We first define the empirically observed second moment observation to be

$$\mathbf{S} = \mathbf{Y}\mathbf{Y}^\top$$

so if two points, i and j are neighbors then we constrain

$$s_{i,j} = \left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} \right\rangle,$$

where $s_{i,j}$ is the i,jth element of $\mathbf{S}$. If we then further constrain the diagonal moments,

$$\left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} \right\rangle = s_{i,i} \tag{11}$$

then the expected squared distance between two data points, will be given by

$$\left\langle d_{i,j} \right\rangle = \left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{i,:} \right\rangle - 2\left\langle \mathbf{y}_{i,:}^\top \mathbf{y}_{j,:} \right\rangle + \left\langle \mathbf{y}_{j,:}^\top \mathbf{y}_{j,:} \right\rangle = s_{i,i} - 2s_{i,j} + s_{j,j}.$$

So the expected interpoint squared distance will match the empirically observed interpoint squared distance from the data. In other words, whilst we have formulated the constraints slightly differently, the final model will respect the same interpoint squared distance constraints as our original formulation of maximum entropy unfolding.

The maximum entropy solution for the distribution has the form

$$p(\mathbf{Y}) \propto \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^\top(\mathbf{\Lambda}+\gamma\mathbf{I})\right)\right),$$

where now the matrix of Lagrange multipliers matches the sparsity structure of the underlying neighborhood graph but also contains diagonal elements to enforce the constraint from (11). Writing the full log likelihood in terms of the matrix $\mathbf{S}$ we have

$$\log p(\mathbf{Y}) = -\frac{pn}{2}\log\tau + \frac{p}{2}\log|\mathbf{\Lambda}+\gamma\mathbf{I}| - \frac{1}{2}\mathrm{tr}\left(\mathbf{S}(\mathbf{\Lambda}+\gamma\mathbf{I})\right),$$

Once again, maximum likelihood in this system is equivalent to finding the Lagrange multipliers so, given the structure from the neighborhood relationships, we simply need to maximize the likelihood to solve the system. That will lead to an implied covariance matrix,

$$\mathbf{K} = (\mathbf{\Lambda}+\gamma\mathbf{I})^{-1},$$

which once again should be centred, $\mathbf{B} = \mathbf{HKH}$, and the principal eigenvectors extracted to visualize the embedding. Here, though, we are proposing some additional structure learning. If elements of the inverse covariance are regularized appropriately the model can perform some additional structure learning. In particular recent work on application of L1 priors on the elements of the inverse covariance (see, e.g., Banerjee et al., 2007; Friedman et al., 2008) allows us to apply a L1 regularizer to the inverse covariance and learn the elements of $\mathbf{\Lambda}$ efficiently. The objective function for this system is now

$$E(\mathbf{\Lambda}) = -\log p(\mathbf{Y}) + \sum_{i<j}\left\|\lambda_{i,j}\right\|_1$$

There has been a great deal of recent work on maximizing objectives of this form. In our experiments we used the graphical lasso algorithm (Friedman et al., 2008) which converts the optimization into a series of iteratively applied lasso regressions.

4. Experiments

The models we have introduced are illustrative and draw on the connections between existing methods. The advantages of our approaches are in the unifying perspective they give and their potential to exploit the characteristics of the probabilistic formulation to explore extensions based on missing data, Bayesian formulations etc.. However, for illustrative purposes we conclude with a short experimental section.

For our experiments we consider two real world data sets. Code to recreate all our experiments is available online. We applied each of the spectral methods we have reviewed along with MEU using positive constraints on the Lagrange multipliers (denoted MEU) and the DRILL described in Section 3. To evaluate the quality of our embeddings we follow the suggestion of Harmeling (2007) and use the GP-LVM likelihood (Lawrence, 2005). The higher the likelihood the better the embedding. Harmeling conducted exhaustive tests over different manifold types (with known ground truth) and found the GP-LVM likelihood was the best indicator of the manifold quality amoungst all the measures he tried. Our first data set consists of human motion capture data.

4.1 Motion Capture Data

The data consists of a 3-dimensional point cloud of the location of 34 points from a subject performing a run. This leads to a 102 dimensional data set containing 55 frames of motion capture. The subject begins the motion from stationary and takes approximately three strides of run. We hope to see this structure in the visualization: a starting position followed by a series of loops. The data was made available by Ohio State University. The data is characterized by a cyclic pattern during the strides of run. However, the angle of

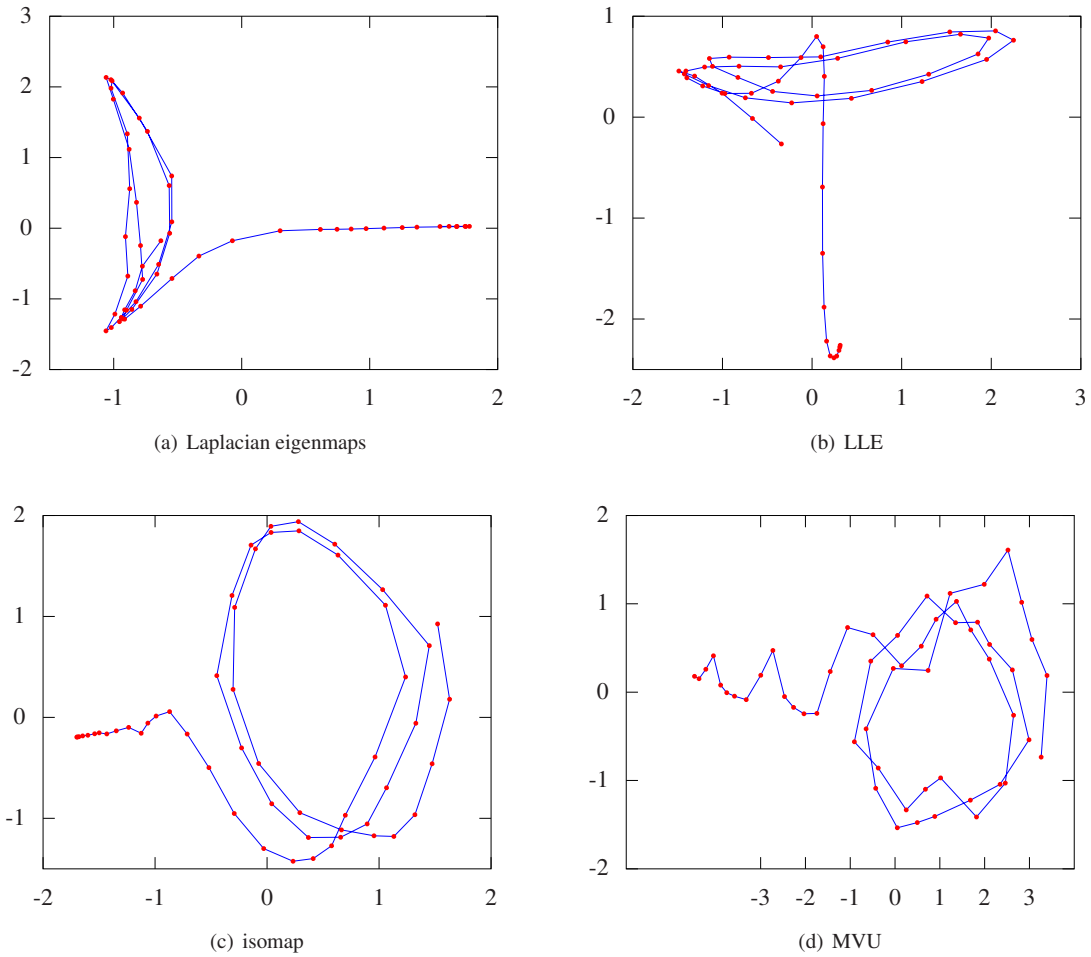
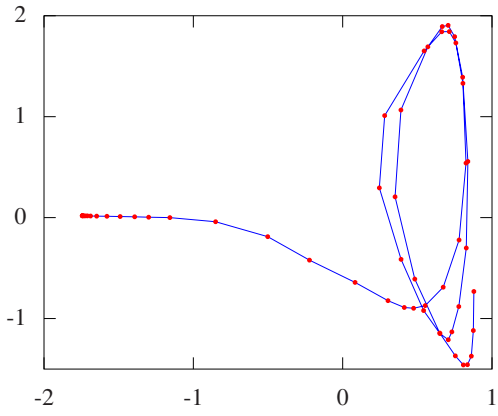


Figure 3: Motion capture data visualized in two dimensions for each algorithm we reviewed using 6 nearest neighbors. Models capture either the cyclic structure or the structure associated with the start of the run or both parts.

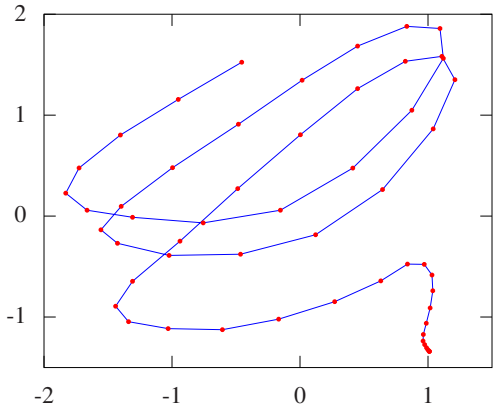
inclination during the run changes so there are slight differences for each cycle. The data is very low noise, as the motion capture rig is designed to extract the point locations of the subject to a high precision.

The two dominant eigenvectors are visualized in Figures 3–4 and the quality of the visualizations under the GP-LVM likelihood is given in Figure 7(a).

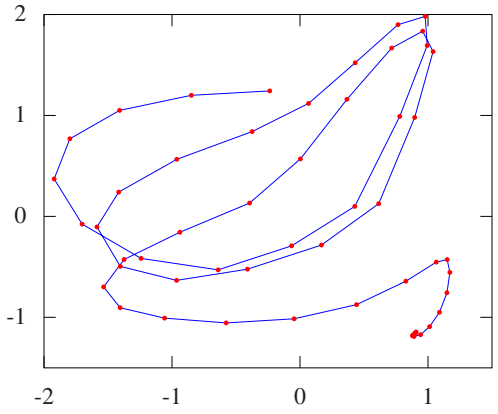
There is a clear difference in quality between the methods that constrain local distances (ALLE, MVU, isomap, MEU and DRILL) which are much better under the score than those that do not (Laplacian eigenmaps and LLE). Amongst the distance preserving methods isomap is the best performer under the GPLVM score, followed by ALLE, MVU, DRILL and MEU. The MEU model here preserves the positive definiteness of the covariance by constraining the Lagrange multipliers to be positive (an 'attractive' network as discussed in Section 2.5.1). It may be that this departure from the true maximum entropy framework explains its relatively poorer performance..



(a) MEU



(b) Acyclic LLE



(c) DRILL

Figure 4: Motion capture data visualized in two dimensions for models derived from the maximum entropy perspective. Again for each algorithm we used 6 nearest neighbors.

4.2 Robot Navigation Example

The second data set we use is a series of recordings from a robot as it traces a square path in a building. The robot records the strength of WiFi signals in an attempt to localize its position (see Ferris et al., 2007, for an application). Since the robot moves only in two dimensions, the inherent dimensionality of the data should be two: the reduced dimensional space should reflect the robot's movement. The WiFi signals are noisier than the motion capture data, so it makes an interesting contrast. The robot completes a single circuit after entering from a separate corridor, so it is expected to exhibit "loop closure" in the resulting map. The data consists of 215 frames of measurement, each frame consists of the WiFi signal strength of 30 access points.

The results for the range of spectral approaches are shown in Figures 5–6 with the quality of the methods scored in Figure 7(b). Both in the visualizations and in the GP-LVM scores we see a clear difference in quality for the methods that preserve local distances (i.e., again isomap, ALLE, MVU, MEU and DRILL are better than LLE and Laplacian eigenmaps). Amongst the methods that do preserve local distance relationships MEU seems to smooth the robot path more than the other three approaches. Given that it has the lowest score of the four distance preserving techniques this smoothing may be unwarranted. MVU appears to have an overly noisy representation of the path.

4.3 Learning the Neighborhood

Our final experiments test the ability of L1 regularization of the random field to learn the neighborhood. We firstly considered the motion capture data and used the DRILL with a large neighborhood size of 20 and L1 regularization on the parameters. As we varied the regularization coefficient we found a maximum under the GP-LVM score (Figure 8(a)). The visualization associated with this maximum is shown in Figure 8(b) this may be compared with Figure 4(c) which used 6 neighbors. Finally we investigated whether L1 regularization alone could recover a reasonable representation of the data. We again considered the motion capture data but initialized all points as neighbors. We then applied L1 regularization to learn a neighborhood structure. Again a maximum under the GP-LVM score was found (Figure 9(a)) and the visualization associated with this maximum is shown Figure 9(b).

The structural learning prior was able to improve the model fitted with 20 neighbors considerably until its performance was similar to that of the the six neighbor model shown in Figure 4(c). However, L1 regularization alone was not able to obtain such a good performance, and was unable to tease out the starting position from the rest of the run in the final visualization. It appears that structural learning using L1-priors for sparsity is not on its own enough to find an appropriate neighborhood structure for this data set.

5. Discussion and Conclusions

We have introduced a new perspective on dimensionality reduction algorithms based around maximum entropy. Our starting point was the maximum variance unfolding and our end point was a novel approach to dimensionality reduction based on Gaussian random fields and lasso based structure learning. We hope that this new perspective on dimensionality reduction will encourage new strands of research at the interface between these areas.

One feature that stands out from our unifying perspective (see also Ham et al., 2004; Bengio et al., 2004b,a) is the three separate stages used in existing spectral dimensionality algorithms.

1. A neighborhood between data points is selected. Normally k-nearest neighbors or similar algorithms are used.

2. Interpoint distances between neighbors are fed to the algorithms which provide a similarity matrix. The way the entries in the similarity matrix are computed is the main difference between the different algorithms.

3. The relationship between points in the similarity matrix is visualized using the eigenvectors of the similarity matrix.

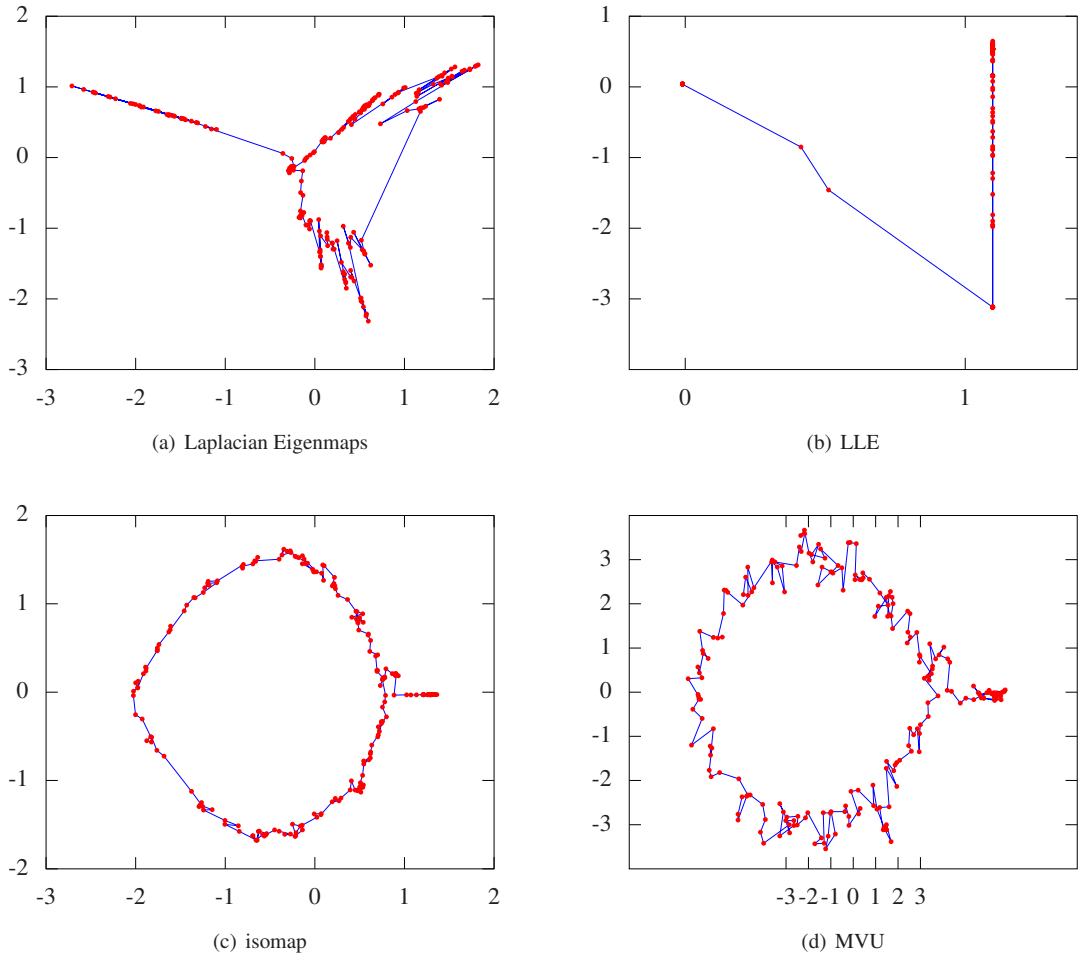
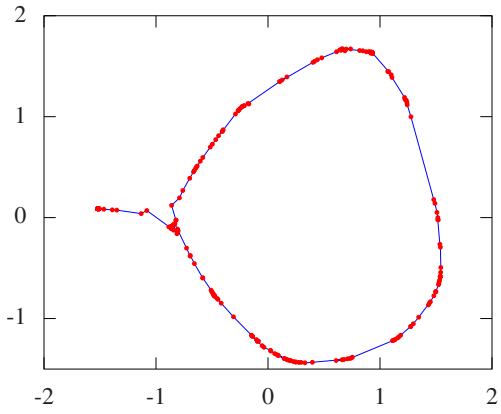


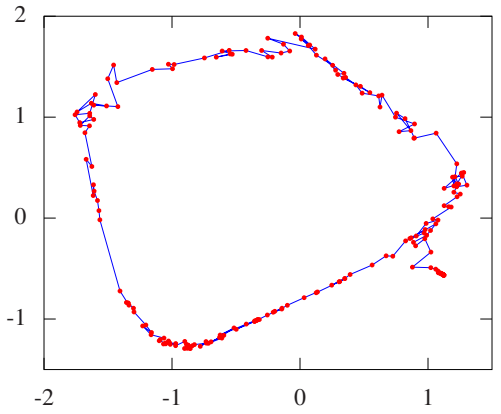
Figure 5: Visualization of the robot WiFi navigation data for different spectral algorithms we reviewed with seven neighbors used to construct graphs. LE and LLE struggle to captured the loop structure (perhaps because of the higher level of noise). Several of the models also show the noise present in the WiFi signals.

Our unifying perspective shows that actually each of these steps is somewhat orthogonal. The neighborhood relations need not come from nearest neighbors, we can use structural learning algorithms such as that suggested in DRILL to learn the interpoint structure. The main difference between the different approaches to spectral dimensionality reduction is how the entries of the similarity matrix are determined. Maximum variance unfolding looks to maximize the trace under the distance constraints from the neighbours. Our new algorithms maximize the entropy or, equivalently, the likelihood of the data. Locally linear embedding maximizes an approximation to our likelihood. Laplacian eigenmaps parameterize the inverse similarity through appealing to physical analogies. Finally, isomap uses shortest path algorithms to compute interpoint distances and centres the resulting matrix to give the similarities.

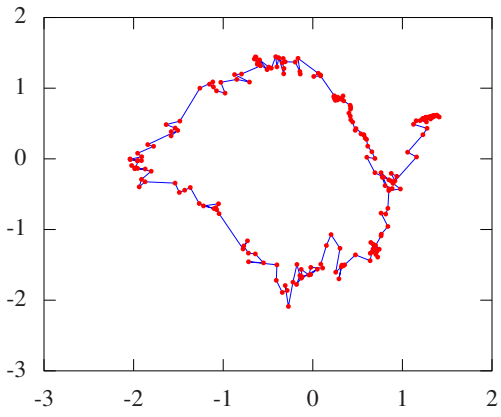
The final step of the algorithm attempts to visualize the similarity matrices using their eigenvectors. However, it simply makes use of one possible objective function to perform this visualization. Considering that underlying the similarity matrix, $\mathbf{K}$, is a sparse Laplacian matrix, $\mathbf{L}$, which represents a Gaussian-Markov



(a) MEU

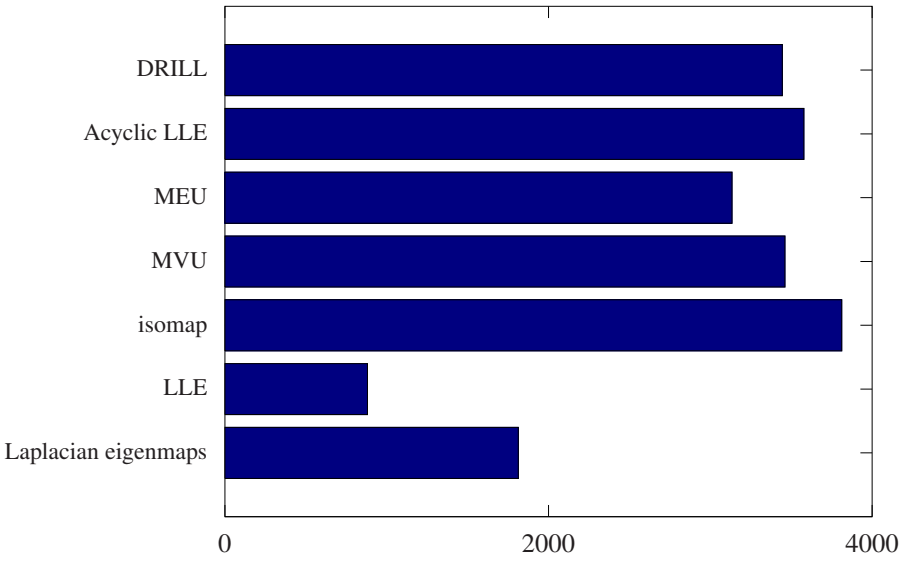


(b) Acyclic LLE

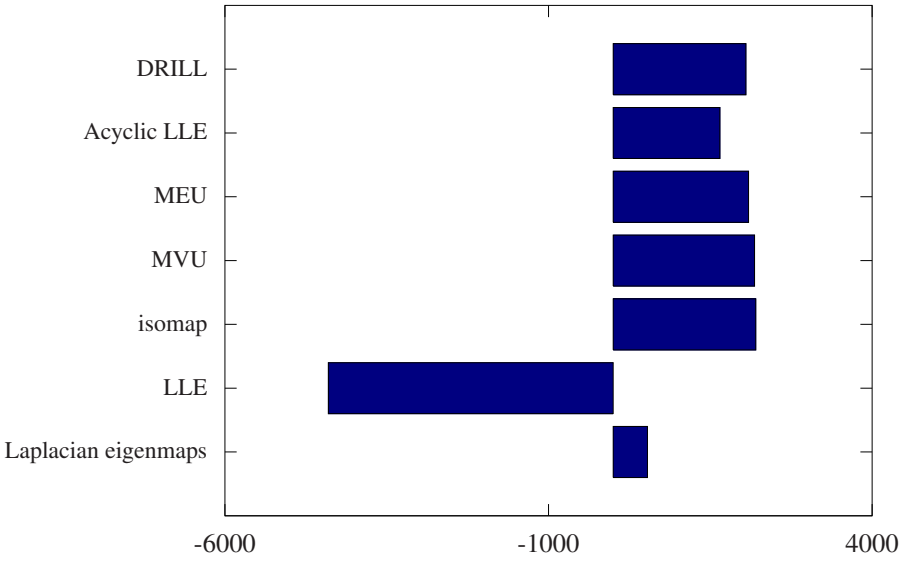


(c) DRILL

Figure 6: Visualization of the robot WiFi navigation data using algorithms based on maximum entropy. Again seven neighbors are used.

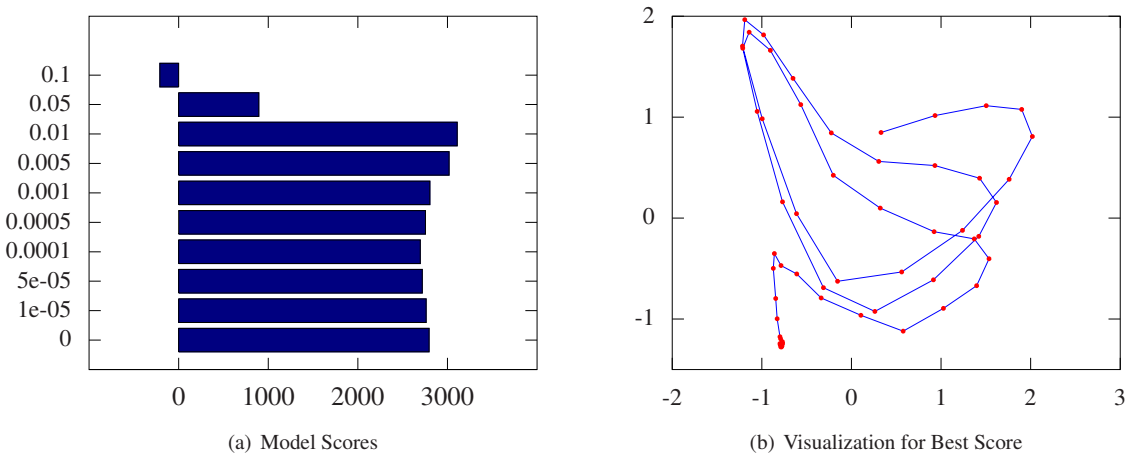


(a) Scores on Motion Capture Data



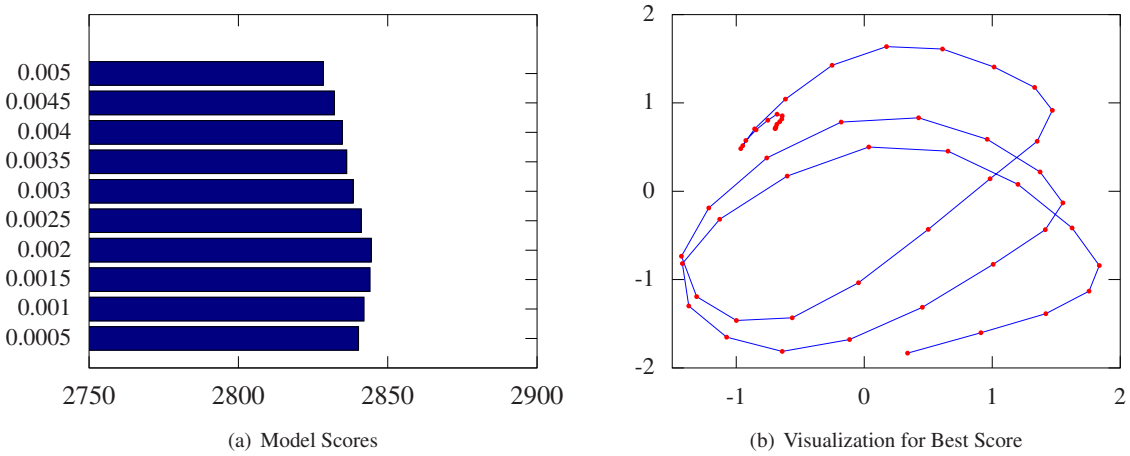
(b) Scores on Robot Navigation Data

Figure 7: Model scores for the different spectral approaches. (a) the motion capture data visualizations, (b) the robot navigation example visualizations.



(a) Model Scores (b) Visualization for Best Score

Figure 8: Structure learning for the DRILL algorithm on the motion capture data set. A model with 20 neighbors was fitted to the data. L1 regularization was used to reduce the number of neighbors associated with each data point. (a) shows the model score for the different L1 regularization parameters and (b) shows the visualization that corresponded to the best score (regularization parameter is 0.01).



(a) Model Scores (b) Visualization for Best Score

Figure 9: Full structure learning for the DRILL algorithm. Here all points are considered neighbors, the structure of the model is then recovered by L1 regularization. (a) shows the model score associated with the different L1 regularization parameters and (b) shows the visualization corresponding to the best score (regularization parameter 0.002).

random field, we can see this final step as visualizing that random field. There are many potential ways to visualize that field and the eigenvectors of the precision is just one of them. In fact, there is an entire field of graph visualization proposing different approaches to visualizing such graphs. However, we could even choose not to visualize the resulting graph. It may be that the structure of the graph is of interest in itself. Work in human cognition by Kemp and Tenenbaum (2008) has sought to fit Gaussian graphical models to data in natural structures such as trees, chains and rings. Visualization of such graphs through reduced

dimensional spaces is only likely to be appropriate in some cases, for example planar structures. For this model only the first two steps are necessary.

One advantage to conflating the three steps we have identified is the possibility to speed up the complete algorithm. For example, conflating the second and third step allows us to speed up algorithms through never explicitly computing the similarity matrix. Using the fact that the principal eigenvectors of the similarity are the minor eigenvalues of the Laplacian and exploiting fast eigensolvers that act on sparse matrices very large data sets can be addressed. However, we still can understand the algorithm from the unifying perspective while exploiting the computational advantages offered by this neat shortcut.

5.1 Gaussian Process Latent Variable Models

Finally, there are similarities between maximum entropy unfolding and the Gaussian process latent variable model (GP-LVM). Both specify a Gaussian density over the training data and in practise the GP-LVM normally makes an assumption of independence across the features. In the GP-LVM a Gaussian process is defined that maps from the latent space, $\mathbf{X}$, to the data space, $\mathbf{Y}$. The resulting likelihood is then optimized with respect to the latent points, $\mathbf{X}$. Maximum entropy unfolding leads to a Gauss Markov Random field, where the conditional dependencies are between neighbors. In one dimension, a Gauss Markov random field can easily be specified by a Gaussian process through appropriate covariance functions. The Ornstein-Uhlbeck covariance function is the unique covariance function for a stationary Gauss Markov process. If such a covariance was defined in a GP-LVM with a one dimensional latent space

$$k(x,x') = \exp(-\|x-x\prime\|_1)$$

then the inverse covariance will be sparse with only the nearest neighbors in the one dimensional latent space connected. The elements of the inverse covariance would be dependent on the distance between the two latent points, which in the GP-LVM is optimized as part of the training procedure. The resulting model is strikingly similar to the MEU model, but in the GP-LVM the neighborhood is learnt by the model (through optimization of $\mathbf{X}$), rather than being specified in advance. The visualization is given directly by the resulting $\mathbf{X}$. There is no secondary step of performing an eigendecomposition to recover the point positions. For larger latent dimensions and different neighborhood sizes, the exact correspondence is harder to establish: Gaussian processes are defined on a continuous space and the Markov property we exploit in MEU arises from discrete relations. But the models are still similar in that they proscribe a Gaussian covariance across the data points which is derived from the spatial relationships between the points.

5.2 Notes

The plots in this document were generated using MATweave (`http://staffwww.dcs.shef.ac.uk/people/N.Lawrence/matweave.html`). Code was run using Octave version 3.2.4 on `x86_64-pc-linux-gnu` architecture. Results were generated on 23/10/2010. All plots and results can be reproduced from the underlying LaTeX document which includes the MATLAB/Octave source code.

Acknowledgments

Conversations with John Kent, Chris Williams, Brenden Lake, Joshua Tenenbaum and John Lafferty have influenced the thinking in this paper.

References

Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research*, 2007.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003. doi: 10.1162/089976603321780317.

Yoshua Bengio, Olivier Delalleau, Jean-Francois Palement, Nicolas Le Roux, Marie Ouimet, and Pascal Vincent. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural Computation*, 16(10): 2197–2219, 2004a.

Yoshua Bengio, Jean-Francois Paiement, Pascal Vincent, Olivier Delalleau, Nicolas Le Roux, and Marie Ouimet. Out-of-sample extensions for LLE, isomap, MDS, eigenmaps, and spectral clustering. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 177–184, Cambridge, MA, 2004b. MIT Press.

Julian Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3):179–195, 1975.

George E. P. Box and Norman R. Draper. *Empirical Model-Building and Response Surfaces*. John Wiley and Sons, 1987. ISBN 0471810339.

Brian D. Ferris, Dieter Fox, and Neil D. Lawrence. WiFi-SLAM using Gaussian process latent variable models. In Manuela M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 2480–2485, 2007.

Brendan J. Frey, Geoffrey E. Hinton, and Peter Dayan. Does the wake-sleep algorithm learn good density estimators? In David Touretzky, Michael Mozer, and Mark Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 661–670, Cambridge, MA, 1996. MIT Press.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, Jul 2008. doi: 10.1093/biostatistics/kxm045.

Jihun Ham, Daniel D. Lee, Sebastian Mika, and Bernhard Schölkopf. A kernel view of dimensionality reduction of manifolds. In Russell Greiner and Dale Schuurmans, editors, *Proceedings of the International Conference in Machine Learning*, volume 21. Omnipress, 2004.

John M. Hammersley and Peter Clifford. Markov fields on finite graphs and lattices. Technical report, 1971. URL `http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf`.

Stefan Harmeling. Exploring model selection techniques for nonlinear dimensionality reduction. Technical Report EDI-INF-RR-0960, University of Edinburgh, 2007.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer-Verlag, 2nd edition, 2009.

Edwin T. Jaynes. Bayesian methods: General background. In J. H. Justice, editor, *Maximum Entropy and Bayesian Methods in Applied Statistics*, pages 1–25. Cambridge University Press, 1986.

Charles Kemp and Joshua B. Tenenbaum. The discovery of structural form. *Proc. Natl. Acad. Sci. USA*, 105 (31), 2008.

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. ISBN 978-0-262-01319-2.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:79–86, 1951.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. *JMLR: W&CP*, 15: 29–37, 2011.

Neil D. Lawrence. Gaussian process models for visualisation of high dimensional data. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 329–336, Cambridge, MA, 2004. MIT Press.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.

Na Li and Matthew Stephens. Modeling linkage disequilibrium and identifying recombination hotspots using single-nucleotide polymorphism data. *Genetics*, 165:2213–2233, 2003. URL `http://www.genetics.org/cgi/content/abstract/165/4/2213`.

Kantilal V. Mardia, John T. Kent, and John M. Bibby. *Multivariate Analysis*. Academic Press, London, 1979. ISBN 0-12-471252-5.

Radford M. Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 56:71–113, 1992.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. doi: 10.1126/science.290.5500.2323.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998. doi: 10.1162/089976698300017467.

Joshua B. Tenenbaum, Virginia de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000. doi: 10.1126/science.290.5500.2319.

Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 6(3):611–622, 1999. doi: doi:10.1111/1467-9868.00196.

Larry A. Wasserman. *All of Statistics*. Springer-Verlag, New York, 2003. ISBN 9780387402727.

Kilian Q. Weinberger, Fei Sha, and Lawrence K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In Russell Greiner and Dale Schuurmans, editors, *Proceedings of the International Conference in Machine Learning*, volume 21, pages 839–846. Omnipress, 2004.

Christopher K. I. Williams. On a connection between kernel PCA and metric multidimensional scaling. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 675–681, Cambridge, MA, 2001. MIT Press.

Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Semi-supervised learning: From Gaussian fields to Gaussian processes. Technical Report CMU-CS-03-175, Carnegie Mellon University, 2003.