Chapter 1

GEOMETRIC METHODS FOR FEATURE EXTRACTION AND DIMENSIONAL REDUCTION

A Guided Tour

Christopher J.C. Burges
Microsoft Research

Abstract We give a tutorial overview of several geometric methods for feature extraction and dimensional reduction. We divide the methods into projective methods and methods that model the manifold on which the data lies. For projective methods, we review projection pursuit, principal component analysis (PCA), kernel PCA, probabilistic PCA, and oriented PCA; and for the manifold methods, we review multidimensional scaling (MDS), landmark MDS, Isomap, locally linear embedding, Laplacian eigenmaps and spectral clustering. The Nyström method, which links several of the algorithms, is also reviewed. The goal is to provide a self-contained review of the concepts and mathematics underlying these algorithms.

Keywords: Feature Extraction, Dimensional Reduction, Principal Components Analysis, Distortion Discriminant Analysis, Nyström method, Projection Pursuit, Kernel PCA, Multidimensional Scaling, Landmark MDS, Locally Linear Embedding, Isomap

Introduction

Feature extraction can be viewed as a preprocessing step which removes distracting variance from a dataset, so that downstream classifiers or regression estimators perform better. The area where feature extraction ends and classification, or regression, begins is necessarily murky: an ideal feature extractor would simply map the data to its class labels, for the classification task. On the other hand, a character recognition neural net can take minimally preprocessed pixel values as input, in which case feature extraction is an inseparable part of the classification process (LeCun and Bengio, 1995). Dimensional reduction - the (usually non-invertible) mapping of data to a lower dimensional space - is

closely related (often dimensional reduction is used as a step in feature extraction), but the goals can differ. Dimensional reduction has a long history as a method for data visualization, and for extracting key low dimensional features (for example, the 2-dimensional orientation of an object, from its high dimensional image representation). The need for dimensionality reduction also arises for other pressing reasons. (Stone, 1982) showed that, under certain regularity assumptions, the optimal rate of convergence[1] for nonparametric regression varies as $m^{-p/(2p+d)}$, where m is the sample size, the data lies in $\mathcal{R}^d$, and where the regression function is assumed to be p times differentiable. Consider 10,000 sample points, for $p = 2$ and $d = 10$. If d is increased to 20, the number of sample points must be increased to approximately 10 million in order to achieve the same optimal rate of convergence. If our data lie (approximately) on a low dimensional manifold $\mathcal{L}$ that happens to be embedded in a high dimensional manifold $\mathcal{H}$, modeling the projected data in $\mathcal{L}$ rather than in $\mathcal{H}$ may turn an infeasible problem into a feasible one.

The purpose of this review is to describe the mathematics and ideas underlying the algorithms. Implementation details, although important, are not discussed. Some notes on notation: vectors are denoted by boldface, whereas components are denoted by x_a, or by $(\mathbf{x}_i)_a$ for the a'th component of the i'th vector. Following (Horn and Johnson, 1985), the set of p by q matrices is denoted M_{pq}, the set of (square) p by p matrices by M_p, and the set of symmetric p by p matrices by S_p (all matrices considered are real). $\mathbf{e}$ with no subscript is used to denote the vector of all ones; on the other hand $\mathbf{e}_a$ denotes the a'th eigenvector. We denote sample size by m, and dimension usually by d or d', with typically $d' \ll d$. δ_{ij} is the Kronecker delta (the ij'th component of the unit matrix). We generally reserve indices i, j, to index vectors and a, b to index dimension.

We place feature extraction and dimensional reduction techniques into two broad categories: methods that rely on projections (Section 1) and methods that attempt to model the manifold on which the data lies (Section 2). Section 1 gives a detailed description of principal component analysis; apart from its intrinsic usefulness, PCA is interesting because it serves as a starting point for many modern algorithms, some of which (kernel PCA, probabilistic PCA, and oriented PCA) are also described. However it has clear limitations: it is easy to find even low dimensional examples where the PCA directions are far from optimal for feature extraction (Duda and Hart, 1973), and PCA ignores correlations in the data that are higher than second order. Section 2 starts with an overview of the Nyström method, which can be used to extend, and link, several of the algorithms described in this chapter. We then examine some methods for dimensionality reduction which assume that the data lie on a low dimensional manifold embedded in a high dimensional space $\mathcal{H}$, namely lo-

cally linear embedding, multidimensional scaling, Isomap, Laplacian eigenmaps, and spectral clustering.

1. Projective Methods

If dimensional reduction is so desirable, how should we go about it? Perhaps the simplest approach is to attempt to find low dimensional *projections* that extract useful information from the data, by maximizing a suitable objective function. This is the idea of projection pursuit (Friedman and Tukey, 1974). The name 'pursuit' arises from the iterative version, where the currently optimal projection is found in light of previously found projections (in fact originally this was done manually[2]). Apart from handling high dimensional data, projection pursuit methods can be robust to noisy or irrelevant features (Huber, 1985), and have been applied to regression (Friedman and Stuetzle, 1981), where the regression is expressed as a sum of 'ridge functions' (functions of the one dimensional projections) and at each iteration the projection is chosen to minimize the residuals; to classification; and to density estimation (Friedman et al., 1984). How are the interesting directions found? One approach is to search for projections such that the projected data departs from normality (Huber, 1985). One might think that, since a distribution is normal if and only if all of its one dimensional projections are normal, if the least normal projection of some dataset is still approximately normal, then the dataset is also necessarily approximately normal, but this is not true; Diaconis and Freedman have shown that most projections of high dimensional data are approximately normal (Diaconis and Freedman, 1984) (see also below). Given this, finding projections along which the density departs from normality, if such projections exist, should be a good exploratory first step.

The sword of Diaconis and Freedman cuts both ways, however. If most projections of most high dimensional datasets are approximately normal, perhaps projections are not always the best way to find low dimensional representations. Let's review their results in a little more detail. The main result can be stated informally as follows: consider a model where the data, the dimension d, and the sample size m depend on some underlying parameter ν, such that as ν tends to infinity, so do m and d. Suppose that as ν tends to infinity, the fraction of vectors which are not approximately the same length tends to zero, and suppose further that under the same conditions, the fraction of pairs of vectors which are not approximately orthogonal to each other also tends to zero[3]. Then ((Diaconis and Freedman, 1984), theorem 1.1) the empirical distribution of the projections along any given unit direction tends to $N(0, \sigma^2)$ weakly in probability. However, if the conditions are not fulfilled, as for some long-tailed distributions, then the opposite result can hold - that is, most projections are

not normal (for example, most projections of Cauchy distributed data[4] will be Cauchy (Diaconis and Freedman, 1984)).

As a concrete example[5], consider data uniformly distributed over the unit $n+1$-sphere $\mathcal{S}^{n+1}$ for odd n. Let's compute the density projected along any line $\mathcal{I}$ passing through the origin. By symmetry, the result will be independent of the direction we choose. If the distance along the projection is parameterized by $\xi \equiv \cos\theta$, where θ is the angle between $\mathcal{I}$ and the line from the origin to a point on the sphere, then the density at ξ is proportional to the volume of an n-sphere of radius $\sin\theta$: $\rho(\xi) = C(1-\xi^2)^{\frac{n-1}{2}}$. Requiring that $\int_{-1}^{1} \rho(\xi) d\xi = 1$ gives the constant C:

$$C = 2^{-\frac{1}{2}(n+1)} \frac{n!!}{(\frac{1}{2}(n-1))!} \tag{1.1}$$

Let's plot this density and compare against a one dimensional Gaussian density fitted using maximum likelihood. For that we just need the variance, which can be computed analytically: $\sigma^2 = \frac{1}{n+2}$, and the mean, which is zero. Figure 1.1 shows the result for the 20-sphere. Although data uniformly distributed on $\mathcal{S}^{20}$ is far from Gaussian, its projection along any direction is close to Gaussian for all such directions, and we cannot hope to uncover such structure using one dimensional projections.

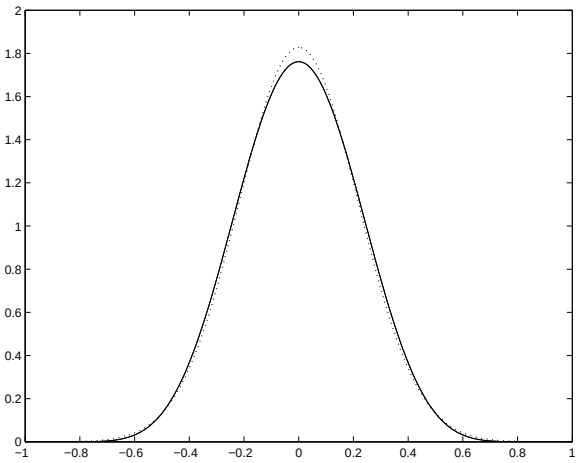

Figure 1.1. Dotted line: a Gaussian with zero mean and variance 1/21. Solid line: the density projected from data distributed uniformly over the 20-sphere, to any line passing through the origin.

The notion of searching for non-normality, which is at the heart of projection pursuit (the goal of which is dimensional reduction), is also the key idea

underlying independent component analysis (ICA) (the goal of which is source separation). ICA (Hyvärinen et al., 2001) searches for projections such that the probability distributions of the data along those projections are statistically independent: for example, consider the problem of separating the source signals in a linear combinations of signals, where the sources consist of speech from two speakers who are recorded using two microphones (and where each microphone captures sound from both speakers). The signal is the sum of two statistically independent signals, and so finding those independent signals is required in order to decompose the signal back into the two original source signals, and at any given time, the separated signal values are related to the microphone signals by two (time independent) projections (forming an invertible 2 by 2 matrix). If the data is normally distributed, finding projections along which the data is uncorrelated is equivalent to finding projections along which it is independent, so although using principal component analysis (see below) will suffice to find independent projections, those projections will not be useful for the above task. For most other distributions, finding projections along which the data is statistically independent is a much stronger (and for ICA, useful) condition than finding projections along which the data is uncorrelated. Hence ICA concentrates on situations where the distribution of the data departs from normality, and in fact, finding the maximally non-Gaussian component (under the constraint of constant variance) will give you an independent component (Hyvärinen et al., 2001).

1.1 Principal Components Analysis (PCA)

1.1.1 PCA: Finding an Informative Direction. Given data $\mathbf{x}_i \in \mathcal{R}^d$, $i = 1, \cdots, m$, suppose you'd like to find a direction $\mathbf{v} \in \mathcal{R}^d$ for which the projection $\mathbf{x}_i \cdot \mathbf{v}$ gives a good one dimensional representation of your original data: that is, informally, the act of projecting loses as little information about your expensively-gathered data as possible (we will examine the information theoretic view of this below). Suppose that unbeknownst to you, your data in fact lies along a line $\mathcal{I}$ embedded in $\mathcal{R}^d$, that is, $\mathbf{x}_i = \boldsymbol{\mu} + \theta_i \mathbf{n}$, where $\boldsymbol{\mu}$ is the sample mean[6], $\theta_i \in \mathcal{R}$, and $\mathbf{n} \in \mathcal{R}^d$ has unit length. The sample variance of the projection along $\mathbf{n}$ is then

$$v_n \equiv \frac{1}{m} \sum_{i=1}^{m} ((\mathbf{x}_i - \boldsymbol{\mu}) \cdot \mathbf{n})^2 = \frac{1}{m} \sum_{i=1}^{m} \theta_i^2 \tag{1.2}$$

and that along some other unit direction $\mathbf{n}'$ is

$$v_n' \equiv \frac{1}{m} \sum_{i=1}^{m} ((\mathbf{x}_i - \boldsymbol{\mu}) \cdot \mathbf{n}')^2 = \frac{1}{m} \sum_{i=1}^{m} \theta_i^2 (\mathbf{n} \cdot \mathbf{n}')^2 \tag{1.3}$$

Since $(\mathbf{n}\cdot\mathbf{n}')^2 = \cos^2\phi$, where ϕ is the angle between $\mathbf{n}$ and $\mathbf{n}'$, we see that the projected variance is maximized if and only if $\mathbf{n} = \pm\mathbf{n}'$. Hence in this case, finding the projection for which the projected variance is maximized gives you the direction you are looking for, namely $\mathbf{n}$, *regardless of the distribution of the data along* $\mathbf{n}$, as long as the data has finite variance. You would then quickly find that the variance along all directions orthogonal to $\mathbf{n}$ is zero, and conclude that your data in fact lies along a one dimensional manifold embedded in $\mathcal{R}^d$. This is one of several basic results of PCA that hold for arbitrary distributions, as we shall see.

Even if the underlying physical process generates data that ideally lies along $\mathcal{I}$, noise will usually modify the data at various stages up to and including the measurements themselves, and so your data will very likely not lie exactly along $\mathcal{I}$. If the overall noise is much smaller than the signal, it makes sense to try to find $\mathcal{I}$ by searching for that projection along which the projected data has maximum variance. If in addition your data lies in a two (or higher) dimensional subspace, the above argument can be repeated, picking off the highest variance directions in turn. Let's see how that works.

1.1.2 PCA: Ordering by Variance. We've seen that directions of maximum variance can be interesting, but how can we find them? The variance along unit vector $\mathbf{n}$ (Eq. (1.2)) is $\mathbf{n}'C\mathbf{n}$ where C is the sample covariance matrix. Since C is positive semidefinite, its eigenvalues are positive or zero; let's choose the indexing such that the (unit normed) eigenvectors $\mathbf{e}_a,\ a = 1,\ldots,d$ are arranged in order of decreasing size of the corresponding eigenvalues λ_a. Since the $\{\mathbf{e}_a\}$ span the space, we can expand $\mathbf{n}$ in terms of them: $\mathbf{n} = \sum_{a=1}^{d}\alpha_a\mathbf{e}_a$, and we'd like to find the α_a that maximize $\mathbf{n}'C\mathbf{n} = \mathbf{n}'\sum_a \alpha_a C\mathbf{e}_a = \sum_a \lambda_a\alpha_a^2$, subject to $\sum_a \alpha_a^2 = 1$ (to give unit normed $\mathbf{n}$). This is just a convex combination of the λ's, and since a convex combination of any set of numbers is maximized by taking the largest, the optimal $\mathbf{n}$ is just $\mathbf{e}_1$, the principal eigenvector (or any one of the set of such eigenvectors, if multiple eigenvectors share the same largest eigenvalue), and furthermore, the variance of the projection of the data along $\mathbf{n}$ is just λ_1.

The above construction captures the variance of the data along the direction $\mathbf{n}$. To characterize the remaining variance of the data, let's find that direction $\mathbf{m}$ which is both orthogonal to $\mathbf{n}$, and along which the projected data again has maximum variance. Since the eigenvectors of C form an orthonormal basis (or can be so chosen), we can expand $\mathbf{m}$ in the subspace $\mathcal{R}^{d-1}$ orthogonal to $\mathbf{n}$ as $\mathbf{m} = \sum_{a=2}^{d}\beta_a\mathbf{e}_a$. Just as above, we wish to find the β_a that maximize $\mathbf{m}'C\mathbf{m} = \sum_{a=2}^{d}\lambda_a\beta_a^2$, subject to $\sum_{a=2}^{d}\beta_a^2 = 1$, and by the same argument, the desired direction is given by the (or any) remaining eigenvector with largest eigenvalue, and the corresponding variance is just that eigenvalue. Repeating this argument gives d orthogonal directions, in order of monotonically decreas-

ing projected variance. Since the d directions are orthogonal, they also provide a complete basis. Thus if one uses all d directions, no information is lost, and as we'll see below, if one uses the $d' < d$ principal directions, then the mean squared error introduced by representing the data in this manner is minimized. Finally, PCA for feature extraction amounts to projecting the data to a lower dimensional space: given an input vector $\mathbf{x}$, the mapping consists of computing the projections of $\mathbf{x}$ along the $\mathbf{e}_a$, $a = 1, \ldots, d'$, thereby constructing the components of the projected d'-dimensional feature vectors.

1.1.3 PCA Decorrelates the Samples. Now suppose we've performed PCA on our samples, and instead of using it to construct low dimensional features, we simply use the full set of orthonormal eigenvectors as a choice of basis. In the old basis, a given input vector $\mathbf{x}$ is expanded as $\mathbf{x} = \sum_{a=1}^{d} x_a \mathbf{u}_a$ for some orthonormal set $\{\mathbf{u}_a\}$, and in the new basis, the same vector is expanded as $\mathbf{x} = \sum_{b=1}^{d} \tilde{x}_b \mathbf{e}_b$, so $\tilde{x}_a \equiv \mathbf{x} \cdot \mathbf{e}_a = \mathbf{e}_a \cdot \sum_b x_b \mathbf{u}_b$. The mean $\boldsymbol{\mu} \equiv \frac{1}{m} \sum_i \mathbf{x}_i$ has components $\tilde{\mu}_a = \boldsymbol{\mu} \cdot \mathbf{e}_a$ in the new basis. The sample covariance matrix depends on the choice of basis: if C is the covariance matrix in the old basis, then the corresponding covariance matrix in the new basis is $\tilde{C}_{ab} \equiv \frac{1}{m} \sum_i (\tilde{x}_{ia} - \tilde{\mu}_a)(\tilde{x}_{ib} - \tilde{\mu}_b) = \frac{1}{m} \sum_i \{\mathbf{e}_a \cdot (\sum_p x_{ip} \mathbf{u}_p - \boldsymbol{\mu})\} \{\sum_q x_{iq} \mathbf{u}_q - \boldsymbol{\mu}) \cdot \mathbf{e}_b\} = \mathbf{e}'_a C \mathbf{e}_b = \lambda_b \delta_{ab}$. Hence in the new basis the covariance matrix is diagonal and the samples are uncorrelated. It's worth emphasizing two points: first, although the covariance matrix can be viewed as a geometric object in that it transforms as a tensor (since it is a summed outer product of vectors, which themselves have a meaning independent of coordinate system), nevertheless, the notion of correlation is basis-dependent (data can be correlated in one basis and uncorrelated in another). Second, PCA decorrelates the samples whatever their underlying distribution; it does not have to be Gaussian.

1.1.4 PCA: Reconstruction with Minimum Squared Error. The basis provided by the eigenvectors of the covariance matrix is also optimal for dimensional reduction in the following sense. Again consider some arbitrary orthonormal basis $\{\mathbf{u}_a,\ a = 1, \ldots, d\}$, and take the first d' of these to perform the dimensional reduction: $\tilde{\mathbf{x}} \equiv \sum_{a=1}^{d'} (\mathbf{x} \cdot \mathbf{u}_a) \mathbf{u}_a$. The chosen $\mathbf{u}_a$ form a basis for $\mathcal{R}^{d'}$, so we may take the components of the dimensionally reduced vectors to be $\mathbf{x} \cdot \mathbf{u}_a$, $a = 1, \ldots, d'$ (although here we leave $\tilde{\mathbf{x}}$ with dimension d). Define the reconstruction error summed over the dataset as $\sum_{i=1}^{m} \|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|^2$. Again assuming that the eigenvectors $\{\mathbf{e}_a\}$ of the covariance matrix are ordered in order of non-increasing eigenvalues, choosing to use those eigenvectors as basis vectors will give minimal reconstruction error. If the data is not centered, then the mean should be subtracted first, the dimensional reduction performed, and the mean then added back[7]; thus in this case, the dimensionally reduced data will still lie in the subspace $\mathcal{R}^{d'}$, but that subspace will be offset from the

origin by the mean. Bearing this caveat in mind, to prove the claim we can assume that the data is centered. Expanding $\mathbf{u}_a \equiv \sum_{p=1}^{d} \beta_{ap}\mathbf{e}_p$, we have

$$\frac{1}{m}\sum_i \|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|^2 = \frac{1}{m}\sum_i \|\mathbf{x}_i\|^2 - \frac{1}{m}\sum_{a=1}^{d'}\sum_i (\mathbf{x}_i \cdot \mathbf{u}_a)^2 \qquad (1.4)$$

with the constraints $\sum_{p=1}^{d} \beta_{ap}\beta_{bp} = \delta_{ab}$. The second term on the right is

$$-\sum_{a=1}^{d'} \mathbf{u}_a' C \mathbf{u}_a = -\sum_{a=1}^{d'}(\sum_{p=1}^{d} \beta_{ap}\mathbf{e}_p')C(\sum_{q=1}^{d} \beta_{aq}\mathbf{e}_q) = -\sum_{a=1}^{d'}\sum_{p=1}^{d} \lambda_p \beta_{ap}^2$$

Introducing Lagrange multipliers ω_{ab} to enforce the orthogonality constraints (Burges, 2004), the objective function becomes

$$F = \sum_{a=1}^{d'}\sum_{p=1}^{d} \lambda_p \beta_{ap}^2 - \sum_{a,b=1}^{d'} \omega_{ab}\left(\sum_{p=1}^{d} \beta_{ap}\beta_{bp} - \delta_{ab}\right) \qquad (1.5)$$

Choosing[8] $\omega_{ab} \equiv \omega_a \delta_{ab}$ and taking derivatives with respect to β_{cq} gives $\lambda_q \beta_{cq} = \omega_c \beta_{cq}$. Both this and the constraints can be satisfied by choosing $\beta_{cq} = 0\ \forall q > c$ and $\beta_{cq} = \delta_{cq}$ otherwise; the objective function is then maximized if the first d' largest λ_p are chosen. Note that this also amounts to a proof that the 'greedy' approach to PCA dimensional reduction - solve for a single optimal direction (which gives the principal eigenvector as first basis vector), then project your data into the subspace orthogonal to that, then repeat - also results in the global optimal solution, found by solving for all directions at once. The same is true for the directions that maximize the variance. Again, note that this argument holds however your data is distributed.

1.1.5 PCA Maximizes Mutual Information on Gaussian Data. Now consider some proposed set of projections $W \in M_{d'd}$, where the rows of W are orthonormal, so that the projected data is $\mathbf{y} \equiv W\mathbf{x}$, $\mathbf{y} \in \mathcal{R}^{d'}$, $\mathbf{x} \in \mathcal{R}^d$, $d' \leq d$. Suppose that $\mathbf{x} \sim \mathcal{N}(0, C)$. Then since the $\mathbf{y}$'s are linear combinations of the $\mathbf{x}$'s, they are also normally distributed, with zero mean and covariance $C_y \equiv (1/m)\sum_i^m \mathbf{y}_i\mathbf{y}_i' = (1/m)W(\sum_i^m \mathbf{x}_i\mathbf{x}_i')W' = WCW'$. It's interesting to ask how W can be chosen so that the mutual information between the distribution of the $\mathbf{x}$'s and that of the $\mathbf{y}$'s is maximized (Baldi and Hornik, 1995; Diamantaras and Kung, 1996). Since the mapping W is deterministic, the conditional entropy $H(\mathbf{y}|\mathbf{x})$ vanishes, and the mutual information is just $I(\mathbf{x}, \mathbf{y}) = H(\mathbf{y}) - H(\mathbf{y}|\mathbf{x}) = H(\mathbf{y})$. Using a small, fixed bin size, we can approximate this by the differential entropy,

$$H(\mathbf{y}) = -\int p(\mathbf{y}) \log_2 p(\mathbf{y}) d\mathbf{y} = \frac{1}{2}\log_2(e(2\pi)^{d'}) + \frac{1}{2}\log_2 \det(C_y) \quad (1.6)$$

This is maximized by maximizing $\det(C_y) = \det(WCW')$ over choice of W, subject to the constraint that the rows of W are orthonormal. The general solution to this is $W = UE$, where U is an arbitrary d' by d' orthogonal matrix, and where the rows of $E \in M_{d'd}$ are formed from the first d' principal eigenvectors of C, and at the solution, $\det(C_y)$ is just the product of the first d' principal eigenvalues. Clearly, the choice of U does not affect the entropy, since $\det(UECE'U') = \det(U)\det(ECE')\det(U') = \det(ECE')$. In the special case where $d' = 1$, so that E consists of a single, unit length vector $\mathbf{e}$, we have $\det(ECE') = \mathbf{e}'C\mathbf{e}$, which is maximized by choosing $\mathbf{e}$ to be the principal eigenvector of C, as shown above. (The other extreme case, where $d' = d$, is easy too, since then $\det(ECE') = \det(C)$ and E can be any orthogonal matrix). We refer the reader to (Wilks, 1962) for a proof for the general case $1 < d' < d$.

1.2 Probabilistic PCA (PPCA)

Suppose you've applied PCA to obtain low dimensional feature vectors for your data, but that you have also somehow found a partition of the data such that the PCA projections you obtain on each subset are quite different from those obtained on the other subsets. It would be tempting to perform PCA on each subset and use the relevant projections on new data, but how do you determine what is 'relevant'? That is, how would you construct a mixture of PCA models? While several approaches to such mixtures have been proposed, the first such probabilistic model was proposed by (Tipping and Bishop, 1999A; Tipping and Bishop, 1999B). The advantages of a probabilistic model are numerous: for example, the weight that each mixture component gives to the posterior probability of a given data point can be computed, solving the 'relevance' problem stated above. In this section we briefly review PPCA.

The approach is closely related to factor analysis, which itself is a classical dimensional reduction technique. Factor analysis first appeared in the behavioral sciences community a century ago, when Spearman hypothesised that intelligence could be reduced to a single underlying factor (Spearman, 1904). If, given an n by n correlation matrix between variables $x_i \in \mathcal{R}$, $i = 1, \cdots, n$, there is a single variable g such that the correlation between x_i and x_j vanishes for $i \neq j$ given the value of g, then g is the underlying 'factor' and the off-diagonal elements of the correlation matrix can be written as the corresponding off-diagonal elements of $\mathbf{z}\mathbf{z}'$ for some $\mathbf{z} \in \mathcal{R}^n$ (Darlington). Modern factor analysis usually considers a model where the underlying factors $\mathbf{x} \in \mathcal{R}^{d'}$ are Gaussian, and where a Gaussian noise term $\boldsymbol{\epsilon} \in \mathcal{R}^d$ is added:

$$\begin{aligned} \mathbf{y} &= W\mathbf{x} + \boldsymbol{\mu} + \boldsymbol{\epsilon} \\ \mathbf{x} &\sim \mathcal{N}(0, \mathbf{1}) \\ \boldsymbol{\epsilon} &\sim \mathcal{N}(0, \Psi) \end{aligned} \tag{1.7}$$

Here $\mathbf{y} \in \mathcal{R}^d$ are the observations, the parameters of the model are $W \in M_{dd'}$ ($d' \leq d$), Ψ and $\boldsymbol{\mu}$, and Ψ is assumed to be diagonal. By construction, the $\mathbf{y}$'s have mean $\boldsymbol{\mu}$ and 'model covariance' $WW' + \Psi$. For this model, given $\mathbf{x}$, the vectors $\mathbf{y} - \boldsymbol{\mu}$ become uncorrelated. Since $\mathbf{x}$ and $\boldsymbol{\epsilon}$ are Gaussian distributed, so is $\mathbf{y}$, and so the maximum likelihood estimate of $E[\mathbf{y}]$ is just $\boldsymbol{\mu}$. However, in general, W and Ψ must be estimated iteratively, using for example EM. There is an instructive exception to this (Basilevsky, 1994; Tipping and Bishop, 1999A). Suppose that $\Psi = \sigma^2\mathbf{1}$, that the $d - d'$ smallest eigenvalues of the model covariance are the same and are equal to σ^2, and that the sample covariance S is equal to the model covariance (so that σ^2 follows immediately from the eigendecomposition of S). Let $\mathbf{e}^{(j)}$ be the j'th orthonormal eigenvector of S with eigenvalue λ_j. Then by considering the spectral decomposition of S it is straightforward to show that $W_{ij} = \sqrt{(\lambda_j - \sigma^2)}\mathbf{e}_i^{(j)}$, $i = 1, \cdots, d$, $j = 1, \cdots, d'$, if the $\mathbf{e}^{(j)}$ are in principal order. The model thus arrives at the PCA directions, but in a probabilistic way. *Probabilistic* PCA (PPCA) is a more general extension of factor analysis: it assumes a model of the form (1.7) with $\Psi = \sigma^2\mathbf{1}$, but it drops the above assumption that the model and sample covariances are equal (which in turn means that σ^2 must now be estimated). The resulting maximum likelihood estimates of W and σ^2 can be written in closed form, as (Tipping and Bishop, 1999A)

$$W_{ML} = U(\Lambda - \sigma^2\mathbf{1})R \tag{1.8}$$

$$\sigma^2_{ML} = \frac{1}{d - d'}\sum_{i=d'+1}^{d} \lambda_i \tag{1.9}$$

where $U \in M_{dd'}$ is the matrix of the d' principal column eigenvectors of S, Λ is the corresponding diagonal matrix of principal eigenvalues, and $R \in M_{d'}$ is an arbitrary orthogonal matrix. Thus σ^2 captures the variance lost in the discarded projections and the PCA directions appear in the maximum likelihood estimate of W (and in fact re-appear in the expression for the expectation of $\mathbf{x}$ given $\mathbf{y}$, in the limit $\sigma \to 0$, in which case the $\mathbf{x}$ become the PCA projections of the $\mathbf{y}$). This closed form result is rather striking in view of the fact that for general factor analysis we must resort to an iterative algorithm. The probabilistic formulation makes PCA amenable to a rich variety of probabilistic methods: for example, PPCA allows one to perform PCA when some of the data is missing components; and d' (which so far we've assumed known) can itself be estimated using Bayesian arguments (Bishop, 1999). Returning to the problem posed at the beginning of this Section, a mixture of PPCA models, each with weight $\pi_i \geq 0$, $\sum_i \pi_i = 1$, can be computed for the data using maximum likelihood and EM, thus giving a principled approach to combining several local PCA models (Tipping and Bishop, 1999B).

1.3 Kernel PCA

PCA is a linear method, in the sense that the reduced dimension representation is generated by linear projections (although the eigenvectors and eigenvalues depend non-linearly on the data), and this can severely limit the usefulness of the approach. Several versions of nonlinear PCA have been proposed (see e.g. (Diamantaras and Kung, 1996)) in the hope of overcoming this problem. In this section we describe a more recent algorithm called kernel PCA (Schölkopf et al., 1998). Kernel PCA relies on the "kernel trick", which is the following observation: suppose you have an algorithm (for example, k'th nearest neighbour) which depends only on dot products of the data. Consider using the same algorithm on transformed data: $\mathbf{x} \rightarrow \mathbf{\Phi}(\mathbf{x}) \in \mathcal{F}$, where $\mathcal{F}$ is a (possibly infinite dimensional) vector space, which we will call feature space[9]. Operating in $\mathcal{F}$, your algorithm depends only on the dot products $\mathbf{\Phi}(\mathbf{x}_i) \cdot \mathbf{\Phi}(\mathbf{x}_j)$. Now suppose there exists a (symmetric) 'kernel' function $k(\mathbf{x}_i, \mathbf{x}_j)$ such that for all $\mathbf{x}_i,\ \mathbf{x}_j \in \mathcal{R}^d$, $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{\Phi}(\mathbf{x}_i) \cdot \mathbf{\Phi}(\mathbf{x}_j)$. Then since your algorithm depends only on these dot products, you never have to compute $\mathbf{\Phi}(\mathbf{x})$ explicitly; you can always just substitute in the kernel form. This was first used by (Aizerman et al., 1964) in the theory of potential functions, and burst onto the machine learning scene in (Boser et al., 1992), when it was applied to support vector machines. Kernel PCA applies the idea to performing PCA in $\mathcal{F}$. It's striking that, since projections are being performed in a space whose dimension can be much larger than d, the number of useful such projections can actually exceed d, so kernel PCA is aimed more at feature extraction than dimensional reduction.

It's not immediately obvious that PCA is eligible for the kernel trick, since in PCA the data appears in expectations over products of individual components of vectors, not over dot products between the vectors. However (Schölkopf et al., 1998) show how the problem can indeed be cast entirely in terms of dot products. They make two key observations: first, that the eigenvectors of the covariance matrix in $\mathcal{F}$ lie in the span of the (centered) mapped data, and second, that therefore no information in the eigenvalue equation is lost if the equation is replaced by m equations, formed by taking the dot product of each side of the eigenvalue equation with each (centered) mapped data point. Let's see how this works. The covariance matrix of the mapped data in feature space is

$$C \equiv \frac{1}{m} \sum_{i=1}^{m} (\mathbf{\Phi}_i - \boldsymbol{\mu})(\mathbf{\Phi}_i - \boldsymbol{\mu})^T \tag{1.10}$$

where $\mathbf{\Phi}_i \equiv \mathbf{\Phi}(\mathbf{x}_i)$ and $\boldsymbol{\mu} \equiv \frac{1}{m} \sum_i \mathbf{\Phi}_i$. We are looking for eigenvector solutions $\mathbf{v}$ of

$$C\mathbf{v} = \lambda \mathbf{v} \tag{1.11}$$

Since this can be written $\frac{1}{m}\sum_{i=1}^{m}(\mathbf{\Phi}_i-\boldsymbol{\mu})[(\mathbf{\Phi}_i-\boldsymbol{\mu})\cdot\mathbf{v}] = \lambda\mathbf{v}$, the eigenvectors $\mathbf{v}$ lie in the span of the $\mathbf{\Phi}_i - \boldsymbol{\mu}$'s, or

$$\mathbf{v} = \sum_i \alpha_i(\mathbf{\Phi}_i - \boldsymbol{\mu}) \tag{1.12}$$

for some α_i. Since (both sides of) Eq. (1.11) lie in the span of the $\mathbf{\Phi}_i - \boldsymbol{\mu}$, we can replace it with the m equations

$$(\mathbf{\Phi}_i - \boldsymbol{\mu})^T C\mathbf{v} = \lambda(\mathbf{\Phi}_i - \boldsymbol{\mu})^T\mathbf{v} \tag{1.13}$$

Now consider the 'kernel matrix' K_{ij}, the matrix of dot products in $\mathcal{F}$: $K_{ij} \equiv \mathbf{\Phi}_i \cdot \mathbf{\Phi}_j, \quad i,j = 1,\ldots,m$. We know how to calculate this, given a kernel function k, since $\mathbf{\Phi}_i \cdot \mathbf{\Phi}_j = k(\mathbf{x}_i, \mathbf{x}_j)$. However, what we need is the *centered* kernel matrix, $K^C_{ij} \equiv (\mathbf{\Phi}_i - \boldsymbol{\mu}) \cdot (\mathbf{\Phi}_j - \boldsymbol{\mu})$. Happily, any m by m dot product matrix can be centered by left- and right- multiplying by the projection matrix $P \equiv \mathbf{1} - \frac{1}{m}\mathbf{e}\mathbf{e}'$, where $\mathbf{1}$ is the unit matrix in M_m and where $\mathbf{e}$ is the m-vector of all ones (see Section 2.2 for further discussion of centering). Hence we have $K^C = PKP$, and Eq. (1.13) becomes

$$K^C K^C \boldsymbol{\alpha} = \bar{\lambda} K^C \boldsymbol{\alpha} \tag{1.14}$$

where $\boldsymbol{\alpha} \in \mathcal{R}^m$ and where $\bar{\lambda} \equiv m\lambda$. Now clearly any solution of

$$K^C \boldsymbol{\alpha} = \bar{\lambda}\boldsymbol{\alpha} \tag{1.15}$$

is also a solution of (1.14). It's straightforward to show that any solution of (1.14) can be written as a solution $\boldsymbol{\alpha}$ to (1.15) plus a vector $\boldsymbol{\beta}$ which is orthogonal to $\boldsymbol{\alpha}$ (and which satisfies $\sum_i \beta_i(\mathbf{\Phi}_i - \boldsymbol{\mu}) = 0$), and which therefore does not contribute to (1.12); therefore we need only consider Eq. (1.15). Finally, to use the eigenvectors $\mathbf{v}$ to compute principal components in $\mathcal{F}$, we need $\mathbf{v}$ to have unit length, that is, $\mathbf{v}\cdot\mathbf{v} = 1 = \bar{\lambda}\boldsymbol{\alpha}\cdot\boldsymbol{\alpha}$, so the $\boldsymbol{\alpha}$ must be normalized to have length $1/\sqrt{\bar{\lambda}}$.

The recipe for extracting the i'th principal component in $\mathcal{F}$ using kernel PCA is therefore:

1 Compute the i'th principal eigenvector of K^C, with eigenvalue $\bar{\lambda}$.

2 Normalize the corresponding eigenvector, $\boldsymbol{\alpha}$, to have length $1/\sqrt{\bar{\lambda}}$.

3 For a training point $\mathbf{x}_k$, the principal component is then just

$$(\mathbf{\Phi}(\mathbf{x}_k) - \boldsymbol{\mu}) \cdot \mathbf{v} = \bar{\lambda}\alpha_k$$

4 For a general test point $\mathbf{x}$, the principal component is

$$\begin{aligned}(\mathbf{\Phi}(\mathbf{x}) - \boldsymbol{\mu}) \cdot \mathbf{v} &= \sum_i \alpha_i k(\mathbf{x}, \mathbf{x}_i) - \frac{1}{m} \sum_{i,j} \alpha_i k(\mathbf{x}, \mathbf{x}_j) \\ &- \frac{1}{m} \sum_{i,j} \alpha_i k(\mathbf{x}_i, \mathbf{x}_j) + \frac{1}{m^2} \sum_{i,j,n} \alpha_i k(\mathbf{x}_j, \mathbf{x}_n)\end{aligned}$$

where the last two terms can be dropped since they don't depend on $\mathbf{x}$.

Kernel PCA may be viewed as a way of putting more effort into the up-front computation of features, rather than putting the onus on the classifier or regression algorithm. Kernel PCA followed by a linear SVM on a pattern recognition problem has been shown to give similar results to using a nonlinear SVM using the same kernel (Schölkopf et al., 1998). It shares with other Mercer kernel methods the attractive property of mathematical tractability and of having a clear geometrical interpretation: for example, this has led to using kernel PCA for de-noising data, by finding that vector $\mathbf{z} \in \mathcal{R}^d$ such that the Euclidean distance between $\Phi(\mathbf{z})$ and the vector computed from the first few PCA components in $\mathcal{F}$ is minimized (Mika et al., 1999). Classical PCA has the significant limitation that it depends only on first and second moments of the data, whereas kernel PCA does not (for example, a polynomial kernel $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + b)^p$ contains powers up to order $2p$, which is particularly useful for e.g. image classification, where one expects that products of several pixel values will be informative as to the class). Kernel PCA has the computational limitation of having to compute eigenvectors for square matrices of side m, but again this can be addressed, for example by using a subset of the training data, or by using the Nyström method for approximating the eigenvectors of a large Gram matrix (see below).

1.4 Oriented PCA and Distortion Discriminant Analysis

Before leaving projective methods, we describe another extension of PCA, which has proven very effective at extracting robust features from audio (Burges et al., 2002; Burges et al., 2003). We first describe the method of oriented PCA (OPCA) (Diamantaras and Kung, 1996). Suppose we are given a set of 'signal' vectors $\mathbf{x}_i \in \mathcal{R}^d$, $i = 1, \ldots, m$, where each $\mathbf{x}_i$ represents an undistorted data point, and suppose that for each $\mathbf{x}_i$, we have a set of N distorted versions $\tilde{\mathbf{x}}_i^k,\ k = 1, \ldots, N$. Define the corresponding 'noise' difference vectors to be $\mathbf{z}_i^k \equiv \tilde{\mathbf{x}}_i^k - \mathbf{x}_i$. Roughly speaking, we wish to find linear projections which are as orthogonal as possible to the difference vectors, but along which the variance of the signal data is simultaneously maximized. Denote the unit vectors defining the desired projections by $\mathbf{n}_i,\ i = 1, \ldots, d',\ \mathbf{n}_i \in \mathcal{R}^d$, where d' will be chosen by the user. By analogy with PCA, we could construct a

feature extractor $\mathbf{n}$ which minimizes the mean squared reconstruction error $\frac{1}{mN}\sum_{i,k}(\mathbf{x}_i - \hat{\mathbf{x}}_i^k)^2$, where $\hat{\mathbf{x}}_i^k \equiv (\tilde{\mathbf{x}}_i^k \cdot \mathbf{n})\mathbf{n}$. The $\mathbf{n}$ that solves this problem is that eigenvector of $R_1 - R_2$ with largest eigenvalue, where R_1, R_2 are the correlation matrices of the $\mathbf{x}_i$ and $\mathbf{z}_i$ respectively. However this feature extractor has the undesirable property that the direction $\mathbf{n}$ will change if the noise and signal vectors are globally scaled with two different scale factors. OPCA (Diamantaras and Kung, 1996) solves this problem. The first OPCA direction is defined as that direction $\mathbf{n}$ that maximizes the generalized Rayleigh quotient (Duda and Hart, 1973; Diamantaras and Kung, 1996) $q_0 = \frac{\mathbf{n}'C_1\mathbf{n}}{\mathbf{n}'C_2\mathbf{n}}$, where C_1 is the covariance matrix of the signal and C_2 that of the noise. For d' directions collected into a column matrix $\mathcal{N} \in M_{dd'}$, we instead maximize $\frac{\det(\mathcal{N}'C_1\mathcal{N})}{\det(\mathcal{N}'C_2\mathcal{N})}$. For Gaussian data, this amounts to maximizing the ratio of the volume of the ellipsoid containing the data, to the volume of the ellipsoid containing the noise, where the volume is that lying inside an ellipsoidal surface of constant probability density. We in fact use the correlation matrix of the noise rather than the covariance matrix, since we wish to penalize the mean noise signal as well as its variance (consider the extreme case of noise that has zero variance but nonzero mean). Explicitly, we take

$$C \equiv \frac{1}{m}\sum_i (\mathbf{x}_i - E[\mathbf{x}])(\mathbf{x}_i - E[\mathbf{x}])' \tag{1.16}$$

$$R \equiv \frac{1}{mN}\sum_{i,k} \mathbf{z}_i^k (\mathbf{z}_i^k)' \tag{1.17}$$

and maximize $q = \frac{\mathbf{n}'C\mathbf{n}}{\mathbf{n}'R\mathbf{n}}$, whose numerator is the variance of the projection of the signal data along the unit vector $\mathbf{n}$, and whose denominator is the projected mean squared "error" (the mean squared modulus of all noise vectors $\mathbf{z}_i^k$ projected along $\mathbf{n}$). We can find the directions $\mathbf{n}_j$ by setting $\nabla q = 0$, which gives the generalized eigenvalue problem $C\mathbf{n} = qR\mathbf{n}$; those solutions are also the solutions to the problem of maximizing $\frac{\det(\mathcal{N}'C\mathcal{N})}{\det(\mathcal{N}'R\mathcal{N})}$. If R is not of full rank, it must be regularized for the problem to be well-posed. It is straightforward to show that, for positive semidefinite C, R, the generalized eigenvalues are positive, and that scaling either the signal or the noise leaves the OPCA directions unchanged, although the eigenvalues will change. Furthermore the $\mathbf{n}_i$ are, or may be chosen to be, linearly independent, and although the $\mathbf{n}_i$ are not necessarily orthogonal, they are conjugate with respect to both matrices C and R, that is, $\mathbf{n}_i'C\mathbf{n}_j \propto \delta_{ij}$, $\mathbf{n}_i'R\mathbf{n}_j \propto \delta_{ij}$. Finally, OPCA is similar to linear discriminant analysis (Duda and Hart, 1973), but where each signal point $\mathbf{x}_i$ is assigned its own class.

'Distortion discriminant analysis' (Burges et al., 2002; Burges et al., 2003) uses layers of OPCA projectors both to reduce dimensionality (a high priority for audio or video data) and to make the features more robust. The above

features, computed by taking projections along the $\mathbf{n}$'s, are first translated and normalized so that the signal data has zero mean and the noise data has unit variance. For the audio application, for example, the OPCA features are collected over several audio frames into new 'signal' vectors, the corresponding 'noise' vectors are measured, and the OPCA directions for the next layer found. This has the further advantage of allowing different types of distortion to be penalized at different layers, since each layer corresponds to a different time scale in the original data (for example, a distortion that results from comparing audio whose frames are shifted in time to features extracted from the original data - 'alignment noise' - can be penalized at larger time scales).

2. Manifold Modeling

In Section 1 we gave an example of data with a particular geometric structure which would not be immediately revealed by examining one dimensional projections in input space[10]. How, then, can such underlying structure be found? This section outlines some methods designed to accomplish this. However we first describe the Nyström method (hereafter simply abbreviated 'Nyström'), which provides a thread linking several of the algorithms described in this review.

2.1 The Nyström method

Suppose that $K \in M_n$ and that the rank of K is $r \ll n$. Nyström gives a way of approximating the eigenvectors and eigenvalues of K using those of a small submatrix A. If A has rank r, then the decomposition is exact. This is a powerful method that can be used to speed up kernel algorithms (Williams and Seeger, 2001), to efficiently extend some algorithms (described below) to out-of-sample test points (Bengio et al., 2004), and in some cases, to make an otherwise infeasible algorithm feasible (Fowlkes et al., 2004). In this section only, we adopt the notation that matrix indices refer to sizes unless otherwise stated, so that e.g. A_{mm} means that $A \in M_m$.

2.1.1 Original Nyström. The Nyström method originated as a method for approximating the solution of Fredholm integral equations of the second kind (Press et al., 1992). Let's consider the homogeneous d-dimensional form with density $p(\mathbf{x}),\ \mathbf{x} \in \mathcal{R}^d$. This family of equations has the form

$$\int k(\mathbf{x},\mathbf{y})u(\mathbf{y})p(\mathbf{y})d\mathbf{y} = \lambda u(\mathbf{x}) \tag{1.18}$$

The integral is approximated using the quadrature rule (Press et al., 1992)

$$\lambda u(\mathbf{x}) \approx \frac{1}{m}\sum_{i=1}^{m} k(\mathbf{x},\mathbf{x}_i)u(\mathbf{x}_i) \tag{1.19}$$

which when applied to the sample points becomes a matrix equation K_{mm} $\mathbf{u}_m = m\lambda\mathbf{u}_m$ (with components $K_{ij} \equiv k(\mathbf{x}_i, \mathbf{x}_j)$ and $u_i \equiv u(\mathbf{x}_i)$). This eigensystem is solved, and the value of the integral at a new point $\mathbf{x}$ is approximated by using (1.19), which gives a much better approximation that using simple interpolation (Press et al., 1992). Thus, the original Nyström method provides a way to smoothly approximate an eigenfunction u, given its values on a sample set of points. If a different number m' of elements in the sum are used to approximate the same eigenfunction, the matrix equation becomes $K_{m'm'}\mathbf{u}_{m'} = m'\lambda\mathbf{u}_{m'}$ so the corresponding eigenvalues approximately scale with the number of points chosen. Note that we have not assumed that K is symmetric or positive semidefinite; however from now on we will assume that K is positive semidefinite.

2.1.2 Exact Nyström Eigendecomposition. Suppose that $\tilde{K}_{mm}$ has rank $r < m$. Since it's positive semidefinite it is a Gram matrix and can be written as $\tilde{K} = ZZ'$ where $Z \in M_{mr}$ and Z is also of rank r (Horn and Johnson, 1985). Order the row vectors in Z so that the first r are linearly independent: this just reorders rows and columns in $\tilde{K}$ to give K, but in such a way that K is still a (symmetric) Gram matrix. Then the principal submatrix $A \in S_r$ of K (which itself is the Gram matrix of the first r rows of Z) has full rank. Now letting $n \equiv m - r$, write the matrix K as

$$K_{mm} \equiv \begin{bmatrix} A_{rr} & B_{rn} \\ B'_{nr} & C_{nn} \end{bmatrix} \tag{1.20}$$

Since A is of full rank, the r rows $\begin{bmatrix} A_{rr} & B_{rn} \end{bmatrix}$ are linearly independent, and since K is of rank r, the n rows $\begin{bmatrix} B'_{nr} & C_{nn} \end{bmatrix}$ can be expanded in terms of them, that is, there exists H_{nr} such that

$$\begin{bmatrix} B'_{nr} & C_{nn} \end{bmatrix} = H_{nr} \begin{bmatrix} A_{rr} & B_{rn} \end{bmatrix} \tag{1.21}$$

The first r columns give $H = B'A^{-1}$, and the last n columns then give $C = B'A^{-1}B$. Thus K must be of the form[11]

$$K_{mm} = \begin{bmatrix} A & B \\ B' & B'A^{-1}B \end{bmatrix} = \begin{bmatrix} A \\ B' \end{bmatrix}_{mr} A_{rr}^{-1} \begin{bmatrix} A & B \end{bmatrix}_{rm} \tag{1.22}$$

The fact that we've been able to write K in this 'bottleneck' form suggests that it may be possible to construct the *exact* eigendecomposition of K_{mm} (for its nonvanishing eigenvalues) using the eigendecomposition of a (possibly much smaller) matrix in M_r, and this is indeed the case (Fowlkes et al., 2004). First use the eigendecomposition of A, $A = U\Lambda U'$, where U is the matrix of column eigenvectors of A and Λ the corresponding diagonal matrix of eigenvalues, to rewrite this in the form

$$K_{mm} = \begin{bmatrix} U \\ B'U\Lambda^{-1} \end{bmatrix}_{mr} \Lambda_{rr} \begin{bmatrix} U & \Lambda^{-1}U'B \end{bmatrix}_{rm} \equiv D\Lambda D' \tag{1.23}$$

This would be exactly what we want (dropping all eigenvectors whose eigenvalues vanish), if the columns of D were orthogonal, but in general they are not. It is straightforward to show that, if instead of diagonalizing A we diagonalize $Q_{rr} \equiv A + A^{-1/2}BB'A^{-1/2} \equiv U_Q \Lambda_Q U_Q'$, then the desired matrix of orthogonal column eigenvectors is

$$V_{mr} \equiv \begin{bmatrix} A \\ B' \end{bmatrix} A^{-1/2} U_Q \Lambda_Q^{-1/2} \tag{1.24}$$

(so that $K_{mm} = V \Lambda_Q V'$ and $V'V = \mathbf{1}_{rr}$) (Fowlkes et al., 2004).

Although this decomposition is exact, this last step comes at a price: to obtain the correct eigenvectors, we had to perform an eigendecomposition of the matrix Q which depends on B. If our intent is to use this decomposition in an algorithm in which B changes when new data is encountered (for example, an algorithm which requires the eigendecomposition of a kernel matrix constructed from both train and test data), then we must recompute the decomposition each time new test data is presented. If instead we'd like to compute the eigendecomposition just once, we must approximate.

2.1.3 Approximate Nyström Eigendecomposition. Two kinds of approximation naturally arise. The first occurs if K is only approximately low rank, that is, its spectrum decays rapidly, but not to exactly zero. In this case, $B'A^{-1}B$ will only approximately equal C above, and the approximation can be quantified as $\left\|C - B'A^{-1}B\right\|$ for some matrix norm $\|\cdot\|$, where the difference is known as the Schur complement of A for the matrix K (Golub and Van Loan, 1996).

The second kind of approximation addresses the need to compute the eigendecomposition just once, to speed up test phase. The idea is simply to take Equation (1.19), sum over d elements on the right hand side where $d \ll m$ and $d > r$, and approximate the eigenvector of the full kernel matrix K_{mm} by evaluating the left hand side at all m points (Williams and Seeger, 2001). Empirically it has been observed that choosing d to be some small integer factor larger than r works well (Platt). How does using (1.19) correspond to the expansion in (1.23), in the case where the Schur complement vanishes? Expanding A, B in their definition in Eq. (1.20) to A_{dd}, B_{dn}, so that U_{dd} contains the column eigenvectors of A and U_{md} contains the approximated (high dimensional) column eigenvectors, (1.19) becomes

$$U_{md}\Lambda_{dd} \approx K_{md}U_{dd} = \begin{bmatrix} A \\ B' \end{bmatrix} U_{dd} = \begin{bmatrix} U\Lambda_{dd} \\ B'U_{dd} \end{bmatrix} \tag{1.25}$$

so multiplying by Λ_{dd}^{-1} from the right shows that the approximation amounts to taking the matrix D in (1.23) as the approximate column eigenvectors: in

this sense, the approximation amounts to dropping the requirement that the eigenvectors be exactly orthogonal.

We end with the following observation (Williams and Seeger, 2001): the expression for computing the projections of a mapped test point along principal components in a kernel feature space is, apart from proportionality constants, exactly the expression for the approximate eigenfunctions evaluated at the new point, computed according to (1.19). Thus the computation of the kernel PCA features for a set of points can be viewed as using the Nyström method to approximate the full eigenfunctions at those points.

2.2 Multidimensional Scaling

We begin our look at manifold modeling algorithms with multidimensional scaling (MDS), which arose in the behavioral sciences (Borg and Groenen, 1997). MDS starts with a measure of dissimilarity between each pair of data points in the dataset (note that this measure can be very general, and in particular can allow for non-vectorial data). Given this, MDS searches for a mapping of the (possibly further transformed) dissimilarities to a low dimensional Euclidean space such that the (transformed) pair-wise dissimilarities become squared distances. The low dimensional data can then be used for visualization, or as low dimensional features.

We start with the fundamental theorem upon which 'classical MDS' is built (in classical MDS, the dissimilarities are taken to be squared distances and no further transformation is applied (Cox and Cox, 2001)). We give a detailed proof because it will serve to illustrate a recurring theme. Let $\mathbf{e}$ be the column vector of m ones. Consider the 'centering' matrix $P^e \equiv \mathbf{1} - \frac{1}{m}\mathbf{e}\mathbf{e}'$. Let X be the matrix whose rows are the datapoints $\mathbf{x} \in \mathcal{R}^n$, $X \in M_{mn}$. Since $\mathbf{e}\mathbf{e}' \in M_m$ is the matrix of all ones, $P^e X$ subtracts the mean vector from each row $\mathbf{x}$ in X (hence the name 'centering'), and in addition, $P^e\mathbf{e} = 0$. In fact $\mathbf{e}$ is the only eigenvector (up to scaling) with eigenvalue zero, for suppose $P^e\mathbf{f} = 0$ for some $\mathbf{f} \in \mathcal{R}^m$. Then each component of $\mathbf{f}$ must be equal to the mean of all the components of $\mathbf{f}$, so all components of $\mathbf{f}$ are equal. Hence P^e has rank $m-1$, and P^e projects onto the subspace $\mathcal{R}^{m-1}$ orthogonal to $\mathbf{e}$.

By a 'distance matrix' we will mean a matrix whose ij'th element is $\|\mathbf{x}_i - \mathbf{x}_j\|^2$ for some $\mathbf{x}_i,\ \mathbf{x}_j \in \mathcal{R}^d$, for some d, where $\|\cdot\|$ is the Euclidean norm. Notice that the elements are squared distances, despite the name. P^e can also be used to center both Gram matrices and distance matrices. We can see this as follows. Let $[C(i,j)]$ be that matrix whose ij'th element is $C(i,j)$. Then $P^e[\mathbf{x}_i \cdot \mathbf{x}_j]P^e = P^e X X' P^e = (P^e X)(P^e X)' = [(\mathbf{x}_i - \boldsymbol{\mu}) \cdot (\mathbf{x}_j - \boldsymbol{\mu})]$. In addition, using this result, $P^e[\|\mathbf{x}_i - \mathbf{x}_j\|^2]P^e = P^e[\|\mathbf{x}_i\|^2 e_i e_j + \|\mathbf{x}_j\|^2 e_i e_j - 2\mathbf{x}_i \cdot \mathbf{x}_j]P^e = -2P^e\mathbf{x}_i \cdot \mathbf{x}_j P^e = -2[(\mathbf{x}_i - \boldsymbol{\mu}) \cdot (\mathbf{x}_j - \boldsymbol{\mu})]$.

For the following theorem, the earliest form of which is due to Schoenberg (Schoenberg, 1935), we first note that, for any $A \in M_m$, and letting $Q \equiv \frac{1}{m}\mathbf{e}\mathbf{e}'$,

$$P^e A P^e = \{(\mathbf{1} - Q)A(\mathbf{1} - Q)\}_{ij} = A_{ij} - A^R_{ij} - A^C_{ij} + A^{RC}_{ij} \tag{1.26}$$

where $A^C \equiv AQ$ is the matrix A with each column replaced by the column mean, $A^R \equiv QA$ is A with each row replaced by the row mean, and $A^{RC} \equiv QAQ$ is A with every element replaced by the mean of all the elements.

Theorem: Consider the class of symmetric matrices $A \in S_n$ such that $A_{ij} \geq 0$ and $A_{ii} = 0 \ \forall i, j$. Then $\bar{A} \equiv -P^e A P^e$ is positive semidefinite if and only if A is a distance matrix (with embedding space $\mathcal{R}^d$ for some d). Given that A is a distance matrix, the minimal embedding dimension d is the rank of $\bar{A}$, and the embedding vectors are any set of Gram vectors of $\bar{A}$, scaled by a factor of $\frac{1}{\sqrt{2}}$.

Proof: Assume that $A \in S_m$, $A_{ij} \geq 0$ and $A_{ii} = 0 \ \forall i$, and that $\bar{A}$ is positive semidefinite. Since $\bar{A}$ is positive semidefinite it is also a Gram matrix, that is, there exist vectors $\mathbf{x}_i \in \mathcal{R}^m$, $i = 1, \cdots, m$ such that $\bar{A}_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j$. Introduce $\mathbf{y}_i = \frac{1}{\sqrt{2}}\mathbf{x}_i$. Then from Eq. (1.26),

$$\bar{A}_{ij} = (-P^e A P^e)_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j = -A_{ij} + A^R_{ij} + A^C_{ij} - A^{RC}_{ij} \tag{1.27}$$

so that

$$\begin{aligned} 2(\mathbf{y}_i - \mathbf{y}_j)^2 \equiv (\mathbf{x}_i - \mathbf{x}_j)^2 &= A^R_{ii} + A^C_{ii} - A^{RC}_{ii} + A^R_{jj} + A^C_{jj} - A^{RC}_{jj} \\ &\quad -2(-A_{ij} + A^R_{ij} + A^C_{ij} - A^{RC}_{ij}) \\ &= 2A_{ij} \end{aligned} \tag{1.28}$$

using $A_{ii} = 0$, $A^R_{ij} = A^R_{jj}$, $A^C_{ij} = A^C_{ii}$, and from the symmetry of A, $A^R_{ij} = A^C_{ji}$. Thus A is a distance matrix with embedding vectors $\mathbf{y}_i$. Now consider a matrix $A \in S_n$ that is a distance matrix, so that $A_{ij} = (\mathbf{y}_i - \mathbf{y}_j)^2$ for some $\mathbf{y}_i \in \mathcal{R}^d$ for some d, and let Y be the matrix whose rows are the $\mathbf{y}_i$. Then since each row and column of P^e sums to zero, we have $\bar{A} = -(P^e A P^e) = 2(P^e Y)(P^e Y)'$, hence $\bar{A}$ is positive semidefinite. Finally, given a distance matrix $A_{ij} = (\mathbf{y}_i - \mathbf{y}_j)^2$, we wish to find the dimension of the minimal embedding Euclidean space. First note that we can assume that the $\mathbf{y}_i$ have zero mean ($\sum_i \mathbf{y}_i = 0$), since otherwise we can subtract the mean from each $\mathbf{y}_i$ without changing A. Then $\bar{A}_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j$, again introducing $\mathbf{x}_i \equiv \sqrt{2}\mathbf{y}_i$, so the embedding vectors $\mathbf{y}_i$ are a set of Gram vectors of $\bar{A}$, scaled by a factor of $\frac{1}{\sqrt{2}}$. Now let r be the rank of $\bar{A}$. Since $\bar{A} = XX'$, and since *rank*$(XX') =$ *rank*(X) for any real matrix X (Horn and Johnson, 1985), and since *rank*(X) is the number of linearly independent $\mathbf{x}_i$, the minimal embedding space for the $\mathbf{x}_i$ (and hence for the $\mathbf{y}_i$) has dimension r. □

2.2.1 General Centering. Is P^e the most general matrix that will convert a distance matrix into a matrix of dot products? Since the embedding vectors are not unique (given a set of Gram vectors, any global orthogonal matrix applied to that set gives another set that generates the same positive semidefinite matrix), it's perhaps not surprising that the answer is no. A distance matrix is an example of a conditionally negative definite (CND) matrix. A CND matrix $D \in S_m$ is a symmetric matrix that satisfies $\sum_{i,j} a_i a_j D_{ij} \leq 0 \; \forall \{a_i \in \mathcal{R} : \sum_i a_i = 0\}$; the class of CND matrices is a superset of the class of negative semidefinite matrices (Berg et al., 1984). Defining the projection matrix $P^c \equiv (\mathbf{1} - \mathbf{e}\mathbf{c}')$, for any $\mathbf{c} \in \mathcal{R}^m$ such that $\mathbf{e}'\mathbf{c} = 1$, then for any CND matrix D, the matrix $-P^c D P'^c$ is positive semidefinite (and hence a dot product matrix) (Schölkopf, 2001; Berg et al., 1984) (note that P^c is not necessarily symmetric). This is straightforward to prove: for any $\mathbf{z} \in \mathcal{R}^m$, $P'^c\mathbf{z} = (\mathbf{1} - \mathbf{c}\mathbf{e}')\mathbf{z} = \mathbf{z} - \mathbf{c}(\sum_a z_a)$, so $\sum_i (P'^c\mathbf{z})_i = 0$, hence $(P'^c\mathbf{z})' D (P'^c\mathbf{z}) \leq 0$ from the definition of CND. Hence we can map a distance matrix D to a dot product matrix K by using P^c in the above manner for any set of numbers c_i that sum to unity.

2.2.2 Constructing the Embedding. To actually find the embedding vectors for a given distance matrix, we need to know how to find a set of Gram vectors for a positive semidefinite matrix $\bar{A}$. Let E be the matrix of column eigenvectors $\mathbf{e}^{(\alpha)}$ (labeled by α), ordered by eigenvalue λ_α, so that the first column is the principal eigenvector, and $\bar{A}E = E\Lambda$, where Λ is the diagonal matrix of eigenvalues. Then $\bar{A}_{ij} = \sum_\alpha \lambda_\alpha e_i^{(\alpha)} e_j^{(\alpha)}$. The rows of E form the dual (orthonormal) basis to $e_i^{(\alpha)}$, which we denote $\tilde{e}_\alpha^{(i)}$. Then we can write $\bar{A}_{ij} = \sum_\alpha (\sqrt{\lambda_\alpha}\tilde{e}_\alpha^{(i)})(\sqrt{\lambda_\alpha}\tilde{e}_\alpha^{(i)})$. Hence the Gram vectors are just the dual eigenvectors with each component scaled by $\sqrt{\lambda_\alpha}$. Defining the matrix $\tilde{E} \equiv E\Lambda^{1/2}$, we see that the Gram vectors are just the rows of $\tilde{E}$.

If $\bar{A} \in S_n$ has rank $r \leq n$, then the final $n-r$ columns of $\tilde{E}$ will be zero, and we have directly found the r-dimensional embedding vectors that we are looking for. If $\bar{A} \in S_n$ is full rank, but the last $n - p$ eigenvalues are much smaller than the first p, then it's reasonable to approximate the i'th Gram vector by its first p components $\sqrt{\lambda_\alpha}\tilde{\mathbf{e}}_\alpha^{(i)}, \; \alpha = 1, \cdots, p$, and we have found a low dimensional approximation to the $\mathbf{y}$'s. This device - projecting to lower dimensions by lopping off the last few components of the dual vectors corresponding to the (possibly scaled) eigenvectors - is shared by MDS, Laplacian eigenmaps, and spectral clustering (see below). Just as for PCA, where the quality of the approximation can be characterized by the unexplained variance, we can characterize the quality of the approximation here by the squared residuals. Let $\bar{A}$ have rank r, and suppose we only keep the first $p \leq r$ components to form the approximate embedding vectors. Then denoting the approximation with a hat,

the summed squared residuals are

$$\begin{aligned}\sum_{i=1}^{m} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|^2 &= \frac{1}{2}\sum_{i=1}^{m} \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|^2 \\ &= \frac{1}{2}\sum_{i=1}^{m}\sum_{a=1}^{p} \lambda_a \tilde{e}_a^{(i)2} + \frac{1}{2}\sum_{i=1}^{m}\sum_{a=1}^{r} \lambda_a \tilde{e}_a^{(i)2} - \sum_{i=1}^{m}\sum_{a=1}^{p} \lambda_a \tilde{e}_a^{(i)2}\end{aligned}$$

but $\sum_{i=1}^{m} \tilde{e}_a^{(i)2} = \sum_{i=1}^{m} e_i^{(a)2} = 1$, so

$$\sum_{i=1}^{m} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|^2 = \frac{1}{2}\left(\sum_{a=1}^{r} \lambda_a - \sum_{a=1}^{p} \lambda_a\right) = \sum_{a=p+1}^{r} \lambda_a \qquad (1.29)$$

Thus the fraction of 'unexplained residuals' is $\sum_{a=p+1}^{r} \lambda_a / \sum_{a=1}^{r} \lambda_a$, in analogy to the fraction of 'unexplained variance' in PCA.

If the original symmetric matrix A is such that $\bar{A}$ is not positive semidefinite, then by the above theorem there exist no embedding points such that the dissimilarities are distances between points in some Euclidean space. In that case, we can proceed by adding a sufficiently large positive constant to the diagonal of $\bar{A}$, or by using the closest positive semidefinite matrix, in Frobenius norm[12], to $\bar{A}$, which is $\hat{A} \equiv \sum_{\alpha:\lambda_\alpha>0} \lambda_\alpha \mathbf{e}^{(\alpha)}\mathbf{e}^{(\alpha)'}$. Methods such as classical MDS, that treat the dissimilarities themselves as (approximate) squared distances, are called metric scaling methods. A more general approach - 'non-metric scaling' - is to minimize a suitable cost function of the difference between the embedded squared distances, and some monotonic function of the dissimilarities (Cox and Cox, 2001); this allows for dissimilarities which do not arise from a metric space; the monotonic function, and other weights which are solved for, are used to allow the dissimilarities to nevertheless be represented approximately by low dimensional squared distances. An example of non-metric scaling is ordinal MDS, whose goal is to find points in the low dimensional space so that the distances there correctly reflect a given rank ordering of the original data points.

2.2.3 Landmark MDS. MDS is computationally expensive: since the distances matrix is not sparse, the computational complexity of the eigendecomposition is $O(m^3)$. This can be significantly reduced by using a method called Landmark MDS (LMDS) (Silva and Tenenbaum, 2002). In LMDS the idea is to choose q points, called 'landmarks', where $q > r$ (where r is the rank of the distance matrix), but $q \ll m$, and to perform MDS on landmarks, mapping them to $\mathcal{R}^d$. The remaining points are then mapped to $\mathcal{R}^d$ using only their distances to the landmark points (so in LMDS, the only distances considered are those to the set of landmark points). As first pointed out in (Bengio et al.,

2004) and explained in more detail in (Platt, 2005), LMDS combines MDS with the Nyström algorithm. Let $E \in S_q$ be the matrix of landmark distances and U (Λ) the matrix of eigenvectors (eigenvalues) of the corresponding kernel matrix $A \equiv -\frac{1}{2}P^c E P'^c$, so that the embedding vectors of the landmark points are the first d elements of the rows of $U\Lambda^{1/2}$. Now, extending E by an extra column and row to accommodate the squared distances from the landmark points to a test point, we write the extended distance matrix and corresponding kernel as

$$D = \begin{bmatrix} E & \mathbf{f} \\ \mathbf{f}' & g \end{bmatrix}, \quad K \equiv -\frac{1}{2}P^c D P'^c = \begin{bmatrix} A & \mathbf{b} \\ \mathbf{b}' & c \end{bmatrix} \tag{1.30}$$

Then from Eq. (1.23) we see that the Nyström method gives the approximate column eigenvectors for the extended system as

$$\begin{bmatrix} U \\ \mathbf{b}' U \Lambda^{-1} \end{bmatrix} \tag{1.31}$$

Thus the embedding coordinates of the test point are given by the first d elements of the row vector $\mathbf{b}' U \Lambda^{-1/2}$. However, we only want to compute U and Λ once - they must not depend on the test point. (Platt, 2005) has pointed out that this can be accomplished by choosing the centering coefficients c_i in $P^c \equiv \mathbf{1} - \mathbf{e}\mathbf{c}'$ such that $c_i = 1/q$ for $i \leq q$ and $c_{q+1} = 0$: in that case, since

$$K_{ij} = -\frac{1}{2}\left(D_{ij} - e_i(\sum_{k=1}^{q+1} c_k D_{kj}) - e_j(\sum_{k=1}^{q+1} D_{ik}c_k) + e_i e_j(\sum_{k,m=1}^{q+1} c_k D_{km} c_m) \right)$$

the matrix A (found by limiting i, j to $1, \ldots, q$ above) depends only on the matrix E above. Finally, we need to relate $\mathbf{b}$ back to the measured quantities - the vector of squared distances from the test point to the landmark points. Using $b_i = (-\frac{1}{2}P^c D P'^c)_{q+1,i}$, $i = 1, \cdots, q$, we find that

$$b_k = -\frac{1}{2}\left[D_{q+1,k} - \frac{1}{q}\sum_{j=1}^{q} D_{q+1,j}e_k - \frac{1}{q}\sum_{i=1}^{q} D_{ik} + \frac{1}{q^2}\left(\sum_{i,j=1}^{q} D_{ij}\right) e_k \right]$$

The first term in the square brackets is the vector of squared distances from the test point to the landmarks, $\mathbf{f}$. The third term is the row mean of the landmark distance squared matrix, $\bar{E}$. The second and fourth terms are proportional to the vector of all ones $\mathbf{e}$, and can be dropped[13] since $U'\mathbf{e} = 0$. Hence, modulo terms which vanish when constructing the embedding coordinates, we have $\mathbf{b} \simeq -\frac{1}{2}(\mathbf{f} - \bar{E})$, and the coordinates of the embedded test point are $\frac{1}{2}\Lambda^{-1/2}U'(\bar{E} - \mathbf{f})$; this reproduces the form given in (Silva and Tenenbaum, 2002). Landmark MDS has two significant advantages: first, it reduces the computational complexity from $O(m^3)$ to $O(q^3 + q^2(m-q) = q^2 m)$; and second, it can be applied to any non-landmark point, and so gives a method of extending MDS (using Nyström) to out-of-sample data.

2.3 Isomap

MDS is valuable for extracting low dimensional representations for some kinds of data, but it does not attempt to explicitly model the underlying manifold. Two methods that do directly model the manifold are Isomap and Locally Linear Embedding. Suppose that as in Section 1.1.1, again unbeknownst to you, your data lies on a curve, but in contrast to Section 1.1.1, the curve is not a straight line; in fact it is sufficiently complex that the minimal embedding space $\mathcal{R}^d$ that can contain it has high dimension d. PCA will fail to discover the one dimensional structure of your data; MDS will also, since it attempts to faithfully preserve all distances. Isomap (isometric feature map) (Tenenbaum, 1998), on the other hand, will succeed. The key assumption made by Isomap is that the quantity of interest, when comparing two points, is the distance along the curve between the two points; if that distance is large, it is to be taken, even if in fact the two points are close in $\mathcal{R}^d$ (this example also shows that noise must be handled carefully). The low dimensional space can have more than one dimension: (Tenenbaum, 1998) gives an example of a 5 dimensional manifold embedded in a 50 dimensional space. The basic idea is to construct a graph whose nodes are the data points, where a pair of nodes are adjacent only if the two points are close in $\mathcal{R}^d$, and then to approximate the geodesic distance along the manifold between any two points as the shortest path in the graph, computed using the Floyd algorithm (Gondran and Minoux, 1984); and finally to use MDS to extract the low dimensional representation (as vectors in $\mathcal{R}^{d'}$, $d' \ll d$) from the resulting matrix of squared distances (Tenenbaum (Tenenbaum, 1998) suggests using ordinal MDS, rather than metric MDS, for robustness).

Isomap shares with the other manifold mapping techniques we describe the property that it does not provide a direct functional form for the mapping $\mathcal{I} : \mathcal{R}^d \rightarrow \mathcal{R}^{d'}$ that can simply be applied to new data, so computational complexity of the algorithm is an issue in test phase. The eigenvector computation is $O(m^3)$, and the Floyd algorithm also $O(m^3)$, although the latter can be reduced to $O(hm^2 \log m)$ where h is a heap size (Silva and Tenenbaum, 2002). Landmark Isomap simply employs landmark MDS (Silva and Tenenbaum, 2002) to addresses this problem, computing all distances as geodesic distances to the landmarks. This reduces the computational complexity to $O(q^2 m)$ for the LMDS step, and to $O(hqm \log m)$ for the shortest path step.

2.4 Locally Linear Embedding

Locally linear embedding (LLE) (Roweis and Saul, 2000) models the manifold by treating it as a union of linear patches, in analogy to using coordinate charts to parameterize a manifold in differential geometry. Suppose that each point $\mathbf{x}_i \in \mathcal{R}^d$ has a small number of close neighbours indexed by the set $\mathcal{N}(i)$,

and let $\mathbf{y}_i \in \mathcal{R}^{d'}$ be the low dimensional representation of $\mathbf{x}_i$. The idea is to express each $\mathbf{x}_i$ as a linear combination of its neighbours, and then construct the $\mathbf{y}_i$ so that they can be expressed as the same linear combination of their corresponding neighbours (the latter also indexed by $\mathcal{N}(i)$). To simplify the discussion let's assume that the number of the neighbours is fixed to n for all i. The condition on the $\mathbf{x}$'s can be expressed as finding that $W \in M_{mn}$ that minimizes the sum of the reconstruction errors, $\sum_i \|\mathbf{x}_i - \sum_{j\in\mathcal{N}(i)} W_{ij}\mathbf{x}_j\|^2$. Each reconstruction error $E_i \equiv \|\mathbf{x}_i - \sum_{j\in\mathcal{N}(i)} W_{ij}\mathbf{x}_j\|^2$ should be unaffected by any global translation $\mathbf{x}_i \to \mathbf{x}_i + \boldsymbol{\delta},\ \boldsymbol{\delta} \in \mathcal{R}^d$, which gives the condition $\sum_{j\in\mathcal{N}(i)} W_{ij} = 1\ \forall i$. Note that each E_i is also invariant to global rotations and reflections of the coordinates. Thus the objective function we wish to minimize is

$$F \equiv \sum_i F_i \equiv \sum_i \left(\frac{1}{2}\|\mathbf{x}_i - \sum_{j\in\mathcal{N}(i)} W_{ij}\mathbf{x}_j\|^2 - \lambda_i \left(\sum_{j\in\mathcal{N}(i)} W_{ij} - 1 \right) \right)$$

where the constraints are enforced with Lagrange multipliers λ_i (Burges, 2004). Since the sum splits into independent terms we can minimize each F_i separately. Thus fixing i and letting $\mathbf{x} \equiv \mathbf{x}_i$, $\mathbf{v} \in \mathcal{R}^n$, $v_j \equiv W_{ij}$, and $\lambda \equiv \lambda_i$, and introducing the matrix $C \in S_n$, $C_{jk} \equiv \mathbf{x}_j \cdot \mathbf{x}_k$, $j,k \in \mathcal{N}(i)$, and the vector $\mathbf{b} \in \mathcal{R}^n$, $b_j \equiv \mathbf{x}\cdot\mathbf{x}_j$, $j \in \mathcal{N}(i)$, then requiring that the derivative of F_i with respect to v_j vanishes gives $\mathbf{v} = C^{-1}(\lambda\mathbf{e} + \mathbf{b})$. Imposing the constraint $\mathbf{e}'\mathbf{v} = 1$ then gives $\lambda = (1 - \mathbf{e}'C^{-1}\mathbf{b})/(\mathbf{e}'C^{-1}\mathbf{e})$. Thus W can be found by applying this for each i.

Given the W's, the second step is to find a set of $\mathbf{y}_i \in \mathcal{R}^{d'}$ that can be expressed in terms of each other in the same manner. Again no exact solution may exist and so $\sum_i \|\mathbf{y}_i - \sum_{j\in\mathcal{N}(i)} W_{ij}\mathbf{y}_j\|^2$ is minimized with respect to the $\mathbf{y}$'s, keeping the W's fixed. Let $Y \in M_{md'}$ be the matrix of row vectors of the points $\mathbf{y}$. (Roweis and Saul, 2000) enforce the condition that the $\mathbf{y}$'s span a space of dimension d' by requiring that $(1/m)Y'Y = \mathbf{1}$, although any condition of the form $Y'PY = Z$ where $P \in S_m$ and $Z \in S_{d'}$ is of full rank would suffice (see Section 2.5.1). The origin is arbitrary; the corresponding degree of freedom can be removed by requiring that the $\mathbf{y}$'s have zero mean, although in fact this need not be explicitly imposed as a constraint on the optimization, since the set of solutions can easily be chosen to have this property. The rank constraint requires that the $\mathbf{y}$'s have unit covariance; this links the variables so that the optimization no longer decomposes into m separate optimizations: introducing Lagrange multipliers $\lambda_{\alpha\beta}$ to enforce the constraints, the objective function to be minimized is

$$F = \frac{1}{2}\sum_i \|\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j\|^2 - \frac{1}{2}\sum_{\alpha\beta} \lambda_{\alpha\beta} \left(\sum_i \frac{1}{m} Y_{i\alpha}Y_{i\beta} - \delta_{\alpha\beta} \right) \quad (1.32)$$

where for convenience we treat the W's as matrices in M_m, where $W_{ij} \equiv 0$ for $j \notin \mathcal{N}(i)$. Taking the derivative with respect to $Y_{k\delta}$ and choosing $\lambda_{\alpha\beta} = \lambda_\alpha \delta_{\alpha\beta} \equiv \Lambda_{\alpha\beta}$ gives[8] the matrix equation

$$(\mathbf{1} - W)'(\mathbf{1} - W)Y = \frac{1}{m} Y\Lambda \tag{1.33}$$

Since $(\mathbf{1} - W)'(\mathbf{1} - W) \in S_m$, its eigenvectors are, or can be chosen to be, orthogonal; and since $(\mathbf{1} - W)'(\mathbf{1} - W)\mathbf{e} = 0$, choosing the columns of Y to be the next d' eigenvectors of $(1 - W)'(1 - W)$ with the smallest eigenvalues guarantees that the $\mathbf{y}$ are zero mean (since they are orthogonal to $\mathbf{e}$). We can also scale the $\mathbf{y}$ so that the columns of Y are orthonormal, thus satisfying the covariance constraint $Y'Y = \mathbf{1}$. Finally, the lowest-but-one weight eigenvectors are chosen because their corresponding eigenvalues sum to $m \sum_i \|\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j\|^2$, as can be seen by applying Y' to the left of (1.33).

Thus, LLE requires a two-step procedure. The first step (finding the W's) has $O(n^3 m)$ computational complexity; the second requires eigendecomposing the product of two sparse matrices in M_m. LLE has the desirable property that it will result in the same weights W if the data is scaled, rotated, translated and / or reflected.

2.5 Graphical Methods

In this section we review two interesting methods that connect with spectral graph theory. Let's start by defining a simple mapping from a dataset to an undirected graph G by forming a one-to-one correspondence between nodes in the graph and data points. If two nodes i, j are connected by an arc, associate with it a positive arc weight W_{ij}, $W \in S_m$, where W_{ij} is a similarity measure between points $\mathbf{x}_i$ and $\mathbf{x}_j$. The arcs can be defined, for example, by the minimum spanning tree, or by forming the N nearest neighbours, for N sufficiently large. The Laplacian matrix for any weighted, undirected graph is defined (Chung, 1997) by $\mathcal{L} \equiv D^{-1/2} L D^{-1/2}$, where $L_{ij} \equiv D_{ij} - W_{ij}$ and where $D_{ij} \equiv \delta_{ij}(\sum_k W_{ik})$. We can see that $\mathcal{L}$ is positive semidefinite as follows: for any vector $\mathbf{z} \in \mathcal{R}^m$, since $W_{ij} \geq 0$,

$$0 \leq \frac{1}{2} \sum_{i,j} (z_i - z_j)^2 W_{ij} = \sum_i z_i^2 D_{ii} - \sum_{i,j} z_i W_{ij} z_j = \mathbf{z}' L \mathbf{z}$$

and since L is positive semidefinite, so is the Laplacian. Note that L is never positive definite since the vector of all ones, $\mathbf{e}$, is always an eigenvector with eigenvalue zero (and similarly $\mathcal{L} D^{1/2}\mathbf{e} = 0$).

Let G be a graph and m its number of nodes. For $W_{ij} \in \{0, 1\}$, the spectrum of G (defined as the set of eigenvalues of its Laplacian) characterizes its global properties (Chung, 1997): for example, a complete graph (that is,

one for which every node is adjacent to every other node) has a single zero eigenvalue, and all other eigenvalues are equal to $\frac{m}{m-1}$; if G is connected but not complete, its smallest nonzero eigenvalue is bounded above by unity; the number of zero eigenvalues is equal to the number of connected components in the graph, and in fact the spectrum of a graph is the union of the spectra of its connected components; and the sum of the eigenvalues is bounded above by m, with equality iff G has no isolated nodes. In light of these results, it seems reasonable to expect that global properties of the data - how it clusters, or what dimension manifold it lies on - might be captured by properties of the Laplacian. The following two approaches leverage this idea. We note that using similarities in this manner results in local algorithms: since each node is only adjacent to a small set of similar nodes, the resulting matrices are sparse and can therefore be eigendecomposed efficiently.

2.5.1 Laplacian Eigenmaps. The Laplacian eigenmaps algorithm (Belkin and Niyogi, 2003) uses $W_{ij} = \exp^{-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2}$. Let $\mathbf{y}(\mathbf{x}) \in \mathcal{R}^{d'}$ be the embedding of sample vector $\mathbf{x} \in \mathcal{R}^d$, and let $Y_{ij} \in M_{md'} \equiv (\mathbf{y}_i)_j$. We would like to find $\mathbf{y}$'s that minimize $\sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij}$, since then if two points are similar, their $\mathbf{y}$'s will be close, whereas if $W \approx 0$, no restriction is put on their $\mathbf{y}$'s. We have:

$$\sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij} = 2 \sum_{i,j,a} (\mathbf{y}_i)_a (\mathbf{y}_j)_a (D_{ii}\delta_{ij} - W_{ij}) = 2\mathrm{Tr}(Y'LY) \quad (1.34)$$

In order to ensure that the target space has dimension d' (minimizing (1.34) alone has solution $Y = 0$), we require that Y have rank d. Any constraint of the form $Y'PY = Z$, where $P \in S_m$ and $m \geq d'$, will suffice, provided that $Z \in S_{d'}$ is of full rank. This can be seen as follows: since the rank of Z is d' and since the rank of a product of matrices is bounded above by the rank of each, we have that $d' = rank(Z) = rank(Y'PY) \leq \min(rank((Y'), rank(P), rank(Y))$, and so $rank(Y) \geq d'$; but since $Y \in M_{md'}$ and $d' \leq m$, the rank of Y is at most d'; hence $rank(Y) = d'$. However, minimizing $\mathrm{Tr}(Y'LY)$ subject to the constraint $Y'DY = \mathbf{1}$ results in the simple generalized eigenvalue problem $L\mathbf{y} = \lambda D\mathbf{y}$ (Belkin and Niyogi, 2003). It's useful to see how this arises: we wish to minimize $\mathrm{Tr}(Y'LY)$ subject to the $d'(d'+1)/2$ constraints $Y'DY = \mathbf{1}$. Let $a, b = 1, \ldots, d$ and $i, j = 1, \ldots, m$. Introducing (symmetric) Lagrange multipliers λ_{ab} leads to the objective function $\sum_{i,j,a} y_{ia} L_{ij} y_{ja} - \sum_{i,j,a,b} \lambda_{ab}(y_{ia} D_{ij} y_{jb} - \delta_{ab})$, with extrema at $\sum_j L_{kj} y_{j\beta} = \sum_{\alpha,i} \lambda_{\alpha\beta} D_{ki} y_{i\alpha}$. We choose[8] $\lambda_{\alpha\beta} \equiv \lambda_\beta \delta_{\alpha\beta}$, giving $\sum_j L_{kj} y_{j\alpha} = \sum_i \lambda_\alpha D_{ki} y_{i\alpha}$. This is a generalized eigenvector problem with eigenvectors the columns of Y. Hence once again the low dimensional vectors are constructed from the first few components of the dual eigenvectors, except that in this case, the eigenvectors with lowest eigenvalues are chosen (omitting the

eigenvector $\mathbf{e}$), and in contrast to MDS, they are not weighted by the square roots of the eigenvalues. Thus Laplacian eigenmaps must use some other criterion for deciding on what d' should be. Finally, note that the $\mathbf{y}$'s are conjugate with respect to D (as well as L), so we can scale them so that the constraints $Y'DY = \mathbf{1}$ are indeed met, and our drastic simplification of the Lagrange multipliers did no damage; and left multiplying the eigenvalue equation by $\mathbf{y}'_\alpha$ shows that $\lambda_\alpha = \mathbf{y}'_\alpha L \mathbf{y}_\alpha$, so choosing the smallest eigenvalues indeed gives the lowest values of the objective function, subject to the constraints.

2.5.2 Spectral Clustering. Although spectral clustering is a clustering method, it is very closely related to dimensional reduction. In fact, since clusters may be viewed as large scale structural features of the data, any dimensional reduction technique that maintains these structural features will be a good preprocessing step prior to clustering, to the point where very simple clustering algorithms (such as K-means) on the preprocessed data can work well (Shi and Malik, 2000; Meila and Shi, 2000; Ng et al., 2002). If a graph is partitioned into two disjoint sets by removing a set of arcs, the *cut* is defined as the sum of the weights of the removed arcs. Given the mapping of data to graph defined above, a cut defines a split of the data into two clusters, and the minimum cut encapsulates the notion of maximum dissimilarity between two clusters. However finding a minimum cut tends to just lop off outliers, so (Shi and Malik, 2000) define a normalized cut, which is now a function of all the weights in the graph, but which penalizes cuts which result in a subgraph g such that the cut divided by the sum of weights from g to G is large; this solves the outlier problem. Now suppose we wish to divide the data into two clusters. Define a scalar on each node, $z_i,\ i = 1, \ldots, m$, such that $z_i = 1$ for nodes in one cluster and $z_i = -1$ for nodes in the other. The solution to the normalized mincut problem is given by (Shi and Malik, 2000)

$$\min_{\mathbf{y}} \frac{\mathbf{y}' L \mathbf{y}}{\mathbf{y}' D \mathbf{y}} \text{ such that } y_i \in \{1, -b\} \text{ and } \mathbf{y}' D \mathbf{e} = 0 \tag{1.35}$$

where $\mathbf{y} \equiv (\mathbf{e} + \mathbf{z}) + b(\mathbf{e} - \mathbf{z})$, and b is a constant that depends on the partition. This problem is solved by relaxing $\mathbf{y}$ to take real values: the problem then becomes finding the second smallest eigenvector of the generalized eigenvalue problem $L\mathbf{y} = \lambda D\mathbf{y}$ (the constraint $\mathbf{y}' D\mathbf{e} = 0$ is automatically satisfied by the solutions), which is exactly the same problem found by Laplacian eigenmaps (in fact the objective function used by Laplacian eigenmaps was proposed as Eq. (10) in (Shi and Malik, 2000)). The algorithms differ in what they do next. The clustering is achieved by thresholding the element y_i so that the nodes are split into two disjoint sets. The dimensional reduction is achieved by treating the element y_i as the first component of a reduced dimension representation of the sample $\mathbf{x}_i$. There is also an interesting equivalent physical interpretation,

where the arcs are springs, the nodes are masses, and the $\mathbf{y}$ are the fundamental modes of the resulting vibrating system (Shi and Malik, 2000). Meila and Shi (Meila and Shi, 2000) point out that that matrix $P \equiv D^{-1}L$ is stochastic, which motivates the interpretation of spectral clustering as the stationary distribution of a Markov random field: the intuition is that a random walk, once in one of the mincut clusters, tends to stay in it. The stochastic interpretation also provides tools to analyse the thresholding used in spectral clustering, and a method for learning the weights W_{ij} based on training data with known clusters (Meila and Shi, 2000). The dimensional reduction view also motivates a different approach to clustering, where instead of simply clustering by thresholding a single eigenvector, simple clustering algorithms are applied to the low dimensional representation of the data (Ng et al., 2002).

3. Pulling the Threads Together

At this point the reader is probably struck by how similar the mathematics underlying all these approaches is. We've used essentially the same Lagrange multiplier trick to enforce constraints three times; all of the methods in this review rely on an eigendecomposition. Isomap, LLE, Laplacian eigenmaps, and spectral clustering all share the property that in their original forms, they do not provide a direct functional form for the dimension-reducing mapping, so the extension to new data requires re-training. Landmark Isomap solves this problem; the other algorithms could also use Nyström to solve it (as pointed out by (Bengio et al., 2004)). Isomap is often called a 'global' dimensionality reduction algorithm, because it attempts to preserve all geodesic distances; by contrast, LLE, spectral clustering and Laplacian eigenmaps are local (for example, LLE attempts to preserve local translations, rotations and scalings of the data). Landmark Isomap is still global in this sense, but the landmark device brings the computational cost more in line with the other algorithms. Although they start from quite different geometrical considerations, LLE, Laplacian eigenmaps, spectral clustering and MDS all look quite similar under the hood: the first three use the dual eigenvectors of a symmetric matrix as their low dimensional representation, and MDS uses the dual eigenvectors with components scaled by square roots of eigenvalues. In light of this it's perhaps not surprising that relations linking these algorithms can be found: for example, given certain assumptions on the smoothness of the eigenfunctions and on the distribution of the data, the eigendecomposition performed by LLE can be shown to coincide with the eigendecomposition of the squared Laplacian (Belkin and Niyogi, 2003); and (Ham et al., 2004) show how Laplacian eigenmaps, LLE and Isomap can be viewed as variants of kernel PCA. (Platt, 2005) links several flavors of MDS by showing how landmark MDS and two other MDS algorithms (not described here) are in fact all Nyström algorithms. Despite the

mathematical similarities of LLE, Isomap and Laplacian Eigenmaps, their different geometrical roots result in different properties: for example, for data which lies on a manifold of dimension d embedded in a higher dimensional space, the eigenvalue spectrum of the LLE and Laplacian Eigenmaps algorithms do not reveal anything about d, whereas the spectrum for Isomap (and MDS) does.

The connection between MDS and PCA goes further than the form taken by the 'unexplained residuals' in Eq. (1.29). If $X \in M_{md}$ is the matrix of m (zero-mean) sample vectors, then PCA diagonalizes the covariance matrix $X'X$, whereas MDS diagonalizes the kernel matrix XX'; but XX' has the same eigenvalues as $X'X$ (Horn and Johnson, 1985), and $m - d$ additional zero eigenvalues (if $m > d$). In fact if $\mathbf{v}$ is an eigenvector of the kernel matrix so that $XX'\mathbf{v} = \lambda\mathbf{v}$, then clearly $X'X(X'\mathbf{v}) = \lambda(X'\mathbf{v})$, so $X'\mathbf{v}$ is an eigenvector of the covariance matrix, and similarly if $\mathbf{u}$ is an eigenvector of the covariance matrix, then $X\mathbf{u}$ is an eigenvector of the kernel matrix. This provides one way to view how kernel PCA computes the eigenvectors of the (possibly infinite dimensional) covariance matrix in feature space in terms of the eigenvectors of the kernel matrix. There's a useful lesson here: given a covariance matrix (Gram matrix) for which you wish to compute those eigenvectors with nonvanishing eigenvalues, and if the corresponding Gram matrix (covariance matrix) is both available, and more easily eigendecomposed (has fewer elements), then compute the eigenvectors for the latter, and map to the eigenvectors of the former using the data matrix as above. Along these lines, Williams (Williams, 2001) has pointed out that kernel PCA can itself be viewed as performing MDS in feature space. Before kernel PCA is performed, the kernel is centered (i.e. P^eKP^e is computed), and for kernels that depend on the data only through functions of squared distances between points (such as radial basis function kernels), this centering is equivalent to centering a distance matrix in feature space. (Williams, 2001) further points out that for these kernels, classical MDS in feature space is equivalent to a form of metric MDS in input space. Although ostensibly kernel PCA gives a function that can be applied to test points, while MDS does not, kernel PCA does so by using the Nyström approximation (see Section 2.1.3), and exactly the same can be done with MDS.

The subject of feature extraction and dimensional reduction is vast. In this review I've limited the discussion to mostly geometric methods, and even with that restriction it's far from complete, so I'd like to alert the reader to three other interesting leads. The first is the method of principal curves, where the idea is to find that smooth curve that passes through the data in such a way that the sum of shortest distances from each point to the curve is minimized, thus providing a nonlinear, one-dimensional summary of the data (Hastie and Stuetzle, 1989); the idea has since been extended by applying various regular-

ization schemes (including kernel-based), and to manifolds of higher dimension (Schölkopf and Smola, 2002). Second, competitions have been held at recent NIPS workshops on feature extraction, and the reader can find a wealth of information there (Guyon, 2003). Finally, recent work on object detection has shown that boosting, where each weak learner uses a single feature, can be a very effective method for finding a small set of good (and mutually complementary) features from a large pool of possible features (Viola and Jones, 2001).

Acknowledgments

I thank John Platt for valuable discussions. Thanks also to Lawrence Saul, Bernhard Schölkopf, Jay Stokes and Mike Tipping for commenting on the manuscript.

Notes

1. For convenience we reproduce Stone's definitions (Stone, 1982). Let θ be the unknown regression function, $\hat{T}_n$ an estimator of θ using n samples, and $\{b_n\}$ a sequence of positive constants. Then $\{b_n\}$ is called a lower rate of convergence if there exists $c > 0$ such that $\lim_n \inf_{\hat{T}_n} \sup_\theta P(\|\hat{T}_n - \theta\| \geq cb_n) = 1$, and it is called an achievable rate of convergence if there is a sequence of estimators $\{\hat{T}_n\}$ and $c > 0$ such that $\lim_n \sup_\theta P(\|\hat{T}_n - \theta\| \geq cb_n) = 0$; $\{b_n\}$ is called an optimal rate of convergence if it is both a lower rate of convergence and an achievable rate of convergence.

2. See J.H. Friedman's interesting response to (Huber, 1985) in the same issue.

3. More formally, the conditions are: for σ^2 positive and finite, and for any positive ϵ, $(1/m)\text{card}\{j \leq m : |\|\mathbf{x}_j\|^2 - \sigma^2 d| > \epsilon d\} \to 0$ and $(1/m^2)\text{card}\{1 \leq j, k \leq m : |\mathbf{x}_j \cdot \mathbf{x}_k| > \epsilon d\} \to 0$ (Diaconis and Freedman, 1984).

4. The Cauchy distribution in one dimension has density $c/(c^2 + x^2)$ for constant c.

5. The story for even n is similar but the formulae are slightly different

6. Note that if all x_i lie along a given line then so does μ.

7. The principal eigenvectors are not necessarily the directions that give minimal reconstruction error if the data is not centered: imagine data whose mean is both orthogonal to the principal eigenvector and far from the origin. The single direction that gives minimal reconstruction error will be close to the mean.

8. Recall that Lagrange multipliers can be chosen in any way that results in a solution satisfying the constraints.

9. In fact the method is more general: $\mathcal{F}$ can be any complete, normed vector space with inner product (i.e. any Hilbert space), in which case the dot product in the above argument is replaced by the inner product.

10. Although in that simple example, the astute investigator would notice that all her data vectors have the same length, and conclude from the fact that the projected density is independent of projection direction that the data must be uniformly distributed on the sphere.

11. It's interesting that this can be used to perform 'kernel completion', that is, reconstruction of a kernel with missing values; for example, suppose K has rank 2 and that its first two rows (and hence columns) are linearly independent, and suppose that K has met with an unfortunate accident that has resulted in all of its elements, except those in the first two rows or columns, being set equal to zero. Then the original K is easily regrown using $C = B'A^{-1}B$.

12. The only proof I have seen for this assertion is due to Frank McSherry, Microsoft Research.

13. The last term can also be viewed as an unimportant shift in origin; in the case of a single test point, so can the second term, but we cannot rely on this argument for multiple test points, since the summand in the second term depends on the test point.

References

M.A. Aizerman, E.M. Braverman, and L.I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

P.F. Baldi and K. Hornik. Learning in linear neural networks: A survey. *IEEE Transactions on Neural Networks*, 6(4):837–858, July 1995.

A. Basilevsky. *Statistical Factor Analysis and Related Methods*. Wiley, New York, 1994.

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

Y. Bengio, J. Paiement, and P. Vincent. Out-of-sample extensions for LLE, Isomap, MDS, Eigenmaps and spectral clustering. In *Advances in Neural Information Processing Systems 16*. MIT Press, 2004.

C. Berg, J.P.R. Christensen, and P. Ressel. *Harmonic Analysys on Semigroups*. Springer-Verlag, 1984.

C. M. Bishop. Bayesian PCA. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 382–388, Cambridge, MA, 1999. The MIT Press.

I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 1997.

B. E. Boser, I. M. Guyon, and V .Vapnik. A training algorithm for optimal margin classifiers. In *Fifth Annual Workshop on Computational Learning Theory*, pages 144–152, Pittsburgh, 1992. ACM.

C.J.C. Burges. Some Notes on Applied Mathematics for Machine Learning. In O. Bousquet, U. von Luxburg, and G. Rätsch, editors, *Advanced Lectures on Machine Learning*, pages 21–40. Springer Lecture Notes in Aritificial Intelligence, 2004.

C.J.C. Burges, J.C. Platt, and S. Jana. Extracting noise-robust features from audio. In *Proc. IEEE Conference on Acoustics, Speech and Signal Processing*, pages 1021–1024. IEEE Signal Processing Society, 2002.

C.J.C. Burges, J.C. Platt, and S. Jana. Distortion discriminant analysis for audio fingerprinting. *IEEE Transactions on Speech and Audio Processing*, 11(3):165–174, 2003.

F.R.K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

T.F. Cox and M.A.A. Cox. *Multidimensional Scaling*. Chapman and Hall, 2001.

R.B. Darlington. Factor analysis. Technical report, Cornell University, http://comp9.psych.cornell.edu/Darlington/factor.htm.

V. de Silva and J.B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 705–712. MIT Press, 2002.

P. Diaconis and D. Freedman. Asymptotics of graphical projection pursuit. *Annals of Statistics*, 12:793–815, 1984.

K.I. Diamantaras and S.Y. Kung. *Principal Component Neural Networks*. John Wiley, 1996.

R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. John Wiley, 1973.

C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 26(2), 2004.

J.H. Friedman and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76(376):817–823, 1981.

J.H. Friedman, W. Stuetzle, and A. Schroeder. Projection pursuit density estimation. *J. Amer. Statistical Assoc.*, 79:599–608, 1984.

J.H. Friedman and J.W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, c-23(9):881–890, 1974.

G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins, third edition, 1996.

M. Gondran and M. Minoux. *Graphs and Algorithms*. John Wiley and Sons, 1984.

I. Guyon. *NIPS 2003 workshop on feature extraction:* http://clopinet.com/isabelle/Projects/NIPS2003/.

J. Ham, D.D. Lee, S. Mika, and B. Schölkopf. A kernel view of dimensionality reduction of manifolds. In *Proceedings of the International Conference on Machine Learning*, 2004.

T.J. Hastie and W. Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84(406):502–516, 1989.

R.A. Horn and C.R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

P.J. Huber. Projection pursuit. *Annals of Statistics*, 13(2):435–475, 1985.

A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, 2001.

Y. LeCun and Y. Bengio. Convolutional networks for images, speech and time-series. In M. Arbib, editor, *The Handbook of Brain Theory and Neural Networks*. MIT Press, 1995.

M. Meila and J. Shi. Learning segmentation by random walks. In *Advances in Neural Information Processing Systems*, pages 873–879, 2000.

S. Mika, B. Schölkopf, A. J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch. Kernel PCA and de–noising in feature spaces. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems 11*. MIT Press, 1999.

A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

J. Platt. *Private Communication*.

J. Platt. Fastmap, MetricMap, and Landmark MDS are all Nyström algorithms. In Z. Ghahramani and R. Cowell, editors, *Proc. 10th International Conference on Artificial Intelligence and Statistics*, 2005.

W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vettering. *Numerical recipes in C: the art of scientific computing*. Cambridge University Press, 2nd edition, 1992.

S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(22):2323–2326, 2000.

I.J. Schoenberg. Remarks to maurice frechet's article *sur la définition axiomatique d'une classe d'espace distanciés vectoriellement applicable sur l'espace de hilbert*. *Annals of Mathematics*, 36:724–732, 1935.

B. Schölkopf. The kernel trick for distances. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 301–307. MIT Press, 2001.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

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

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

C.E. Spearman. 'General intelligence' objectively determined and measured. *American Journal of Psychology*, 5:201–293, 1904.

C.J. Stone. Optimal global rates of convergence for nonparametric regression. *Annals of Statistics*, 10(4):1040–1053, 1982.

J.B. Tenenbaum. Mapping a manifold of perceptual observations. In Michael I. Jordan, Michael J. Kearns, and Sara A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

M.E. Tipping and C.M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society*, 61(3):611, 1999A.

M.E. Tipping and C.M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):443–482, 1999B.

P. Viola and M. Jones. Robust real-time object detection. In *Second international workshop on statistical and computational theories of vision - modeling, learning, computing, and sampling*, 2001.

S. Wilks. *Mathematical Statistics*. John Wiley, 1962.

C.K.I. Williams. On a Connection between Kernel PCA and Metric Multidimensional Scaling. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 675–681. MIT Press, 2001.

C.K.I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In Leen, Dietterich, and Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, 2001.