Foundations and Trends® in
Machine Learning
Vol. 2, No. 4 (2009) 275–365
© 2010 C. J. C. Burges
DOI: 10.1561/2200000002

Dimension Reduction: A Guided Tour

By Christopher J. C. Burges

Contents

Foundations and Trends® in
Machine Learning
Vol. 2, No. 4 (2009) 275–365
© 2010 C. J. C. Burges
DOI: 10.1561/2200000002

Dimension Reduction: A Guided Tour

Christopher J. C. Burges

Microsoft Research, One Microsoft Way, Redmond, WA 98052-6399, USA, chris.burges@microsoft.com

Abstract

We give a tutorial overview of several foundational methods for dimension 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, canonical correlation analysis (CCA), kernel CCA, Fisher discriminant analysis, oriented PCA, and several techniques for sufficient dimension reduction. For the manifold methods, we review multidimensional scaling (MDS), landmark MDS, Isomap, locally linear embedding, Laplacian eigenmaps, and spectral clustering. Although this monograph focuses on foundations, we also provide pointers to some more modern techniques. We also describe the correlation dimension as one method for estimating the intrinsic dimension, and we point out that the notion of dimension can be a scale-dependent quantity. The Nyström method, which links several of the manifold algorithms, is also reviewed. We use a publicly available data set to illustrate some of the methods. The goal is to provide a self-contained overview of key concepts underlying many of these algorithms, and to give pointers for further reading.

1
Introduction

Dimension reduction[1] is the mapping of data to a lower dimensional space such that uninformative variance in the data is discarded, or such that a subspace in which the data lives is detected. Dimension reduction has a long history as a method for data visualization, and for extracting key low dimensional features (for example, the two-dimensional orientation of an object, from its high dimensional image representation). In some cases the desired low dimensional features depend on the task at hand. Apart from teaching us about the data, dimension reduction can lead us to better models for inference. The need for dimension reduction also arises for other pressing reasons. Stone [85] showed that, under certain regularity assumptions (including that the samples be IID), the optimal rate of convergence[2] for nonparametric regression varies

[1] We follow both the lead of the statistics community and the spirit of the paper to reduce 'dimensionality reduction' and 'dimensional reduction' to 'dimension reduction'.

[2] The definition of 'optimal rate of convergence' is technical and for completeness we reproduce Stone's definitions here [85]. A 'rate of convergence' is defined as a sequence of numbers, indexed by sample size. Let θ be the unknown regression function, Θ the collection of functions to which θ belongs, $\hat{T}_n$ 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

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. We can get a very rough idea of the impact of sample size on the rate of convergence as follows. Consider a particular point in the sequence of values corresponding to the optimal rate of convergence: $m = 10{,}000$ samples, for $p = 2$ and $d = 10$. Suppose that d is increased to 20; what number of samples in the new sequence gives the same value? The answer is approximately 10 million. If our data lies (approximately) on a low dimensional manifold $\mathcal{L}$ that happens to be embedded in a high dimensional manifold $\mathcal{H}$, then modeling the data directly in $\mathcal{L}$ rather than in $\mathcal{H}$ may turn an infeasible problem into a feasible one.

The purpose of this monograph is to describe the mathematics and key ideas underlying the methods, and to provide some links to the literature for those interested in pursuing a topic further.[3] The subject of dimension reduction is vast, so we use the following criterion to limit the discussion: we restrict our attention to the case where the inferred feature values are continuous. The observables, on the other hand, may be continuous or discrete. Thus this review does not address clustering methods, or, for example, feature selection for discrete data, such as text. This still leaves a very wide field, and so we further limit the scope by choosing not to cover probabilistic topic models (in particular, latent Dirichlet allocation, nonnegative matrix factorization, probabilistic latent semantic analysis, and Gaussian process latent variable models). Furthermore, implementation details, and important theoretical details such as consistency and rates of convergence of sample quantities to their population values, although important, are not discussed. For an alternative, excellent overview of dimension reduction methods, see Lee and Verleysen [62]. This monograph differs from that work in several ways. In particular, while it is common in the literature to see methods applied to artificial, low dimensional data sets such as the famous Swiss Roll, in this monograph we prefer to use higher dimensional data: while low dimensional toy data can be valuable to

$\lim_n \sup_\Theta P(\|\hat{T}_n - \theta\| \geq cb_n) = 0$; $\{b_n\}$ is called an optimal rate of convergence if it is both a lower rate of convergence and an achievable rate of convergence. Here the $\inf_{\hat{T}_n}$ is over all possible estimators $\hat{T}_n$.

[3] This monograph is a revised and extended version of Burges [17].

express ideas and to illustrate strengths and weaknesses of a method, high dimensional data has qualitatively different behavior from two- or three-dimensional data. Here, we use the publicly available KDD Cup [61] training data. This is anonymized breast cancer screening data for 1,712 patients, 118 of whom had a malignant cancer; each feature vector has 117 features, and a total of 102,294 such samples are available. The goal of the Cup was to identify those patients with a malignant tumor from the corresponding feature vectors in a test set. We use the data here because it is relevant to an important real-world problem, it is publicly available, and because the training data has labels (some of the techniques we describe below are for supervised problems).

Regarding notation: we denote the sample space (the $\underline{h}$igh dimensional space in which the data resides) as $\mathcal{H}$, the $\underline{l}$ow dimensional space (to which many of the methods discussed below map the data) as $\mathcal{L}$, and we reserve $\mathcal{F}$ to denote a $\underline{f}$*eature space* (often a high or infinite-dimensional Hilbert space, to which the kernel versions of the methods below map the data as an intermediate step). 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. Random variables are denoted by upper case; we use $E[X|y]$ as shorthand for the function $E[X|Y = y]$, in contrast to the random variable $E[X|Y]$. Following Horn and Johnson [54], the set of p by q matrices is denoted M_{pq}, the set of (square) p by p matrices by M_p, the set of symmetric p by p matrices by S_p, and the set of (symmetric) positive semidefinite 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 place dimension reduction techniques into two broad categories: methods that rely on projections (Section 3) and methods that attempt to model the manifold on which the data lies (Section 4). Section 3 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 here. 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 [33], and PCA ignores correlations in the data that are higher than second order. We end Section 3 with a brief look at projective methods for dimension reduction of *labeled* data: sliced inverse regression, and kernel dimension reduction. Section 4 starts with an overview of the Nyström method, which can be used to extend, and link, several of the algorithms described in this monograph. We then examine some methods for dimension reduction which assume that the data lies on a low dimensional manifold embedded in a high dimensional space, namely locally linear embedding, multidimensional scaling, Isomap, Laplacian eigenmaps, and spectral clustering.

Before we begin our exploration of these methods, however, let's investigate a question that is more fundamental than, and that can be explored independently of, any particular dimension reduction technique: if our data lives on a manifold $\mathcal{M}$ that is embedded in some Euclidean space, how can we estimate the dimension of $\mathcal{M}$?

2

Estimating the Dimension

Consider the data shown schematically in Figure 2.1. Think of the circle as representing the view through a microscope, with magnification increasing from left to right. The data is embedded in $\mathcal{R}^2$, but at different magnifications its intrinsic dimensionality appears to vary: on the left, the data appears to have zero dimensions; at intermediate magnification a one-dimensional structure emerges; and at higher magnifications, the microscope detects a two-dimensional structure, which in this schematic example we are imagining to be due to noise. In order to run any kind of distance-dependent analysis on this data, it would seem advantageous to operate, somehow, at the scale shown in the center panel in the figure, and to ignore variance in the data at much smaller scales.

Now suppose that you are observing through the microscope and you turn down the magnification a little (increase the radius of the circle in Figure 2.1). On the far left, the number of points included in the field of view will not increase; in the center, it will increase linearly; and on the right, as the square of the radius. Thus for the i-th data point, we can compute the number of neighboring data points $C_i(\epsilon)$ that fall in a sphere of radius ϵ around it; if the points are sufficiently

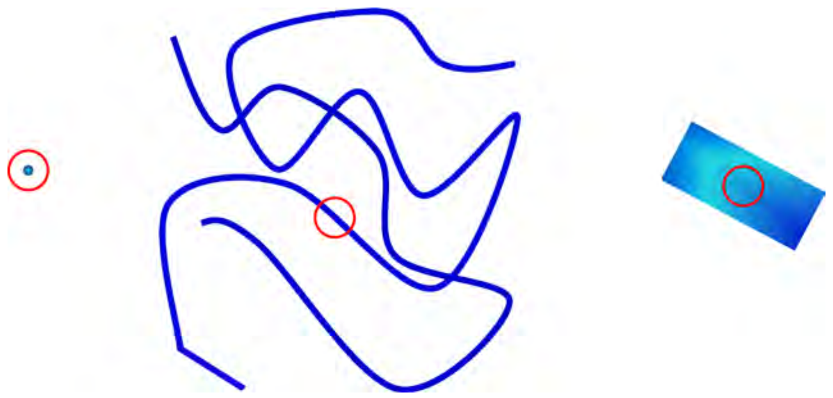

Fig. 2.1 A microscope (circle) examining data (the curves) with added noise (the curves have finite width). The magnification increases from left to right.

dense we expect that $C_i(\epsilon)$ will grow as ϵ^d, where d is the intrinsic dimension. Finally, with limited data, we can improve our estimates by summing: $C(\epsilon) \equiv \sum_{i=1}^{m} C_i(\epsilon)$.

This method for estimating an intrinsic dimension was introduced by Grassberger and Procaccia [43][1] who used it for one- and two-dimensional time series data for which arbitrarily large sample sizes could be artificially generated. They define:

$$C(\epsilon) = \lim_{m\to\infty} \frac{1}{m(m-1)} \times \{\text{number of pairs } \{\mathbf{x}_i, \mathbf{x}_j\} \text{for which } |\mathbf{x}_i - \mathbf{x}_j| < \epsilon\}, \quad (2.1)$$

and estimate the intrinsic dimension ν as the slope of $\log(C(\epsilon))$ as a function of $\log(\epsilon)$ in the limit as ϵ approaches zero.

2.1 A Cautionary Note

High dimensional data behaves qualitatively very differently from low dimensional data. For example, if the data consists of vectors in $\mathcal{R}^d$ and is very sparse (meaning that most components of most vectors are zero), then most vectors will be orthogonal (their inner products will be zero), and so if in addition they have fixed length, then those pairs

[1] Grassberger and Procaccia refer to the quantity as the *correlation exponent* and they note its close relation to the fractal dimension.

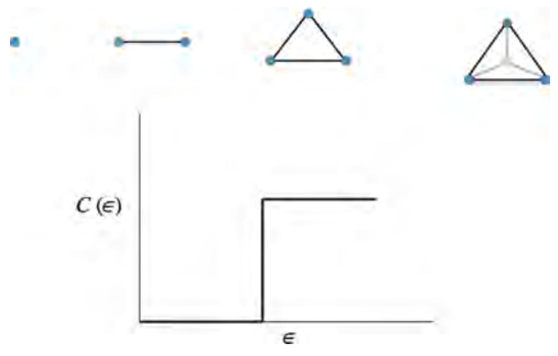

Fig. 2.2 High dimensional data can be equidistant.

of vectors with zero inner product will be equidistant. If the samples are, in general, close to equidistant, then no distance-based dimension reduction technique will work very well. This is illustrated schematically in Figure 2.2, where an artificial data set is built by adding a dimension, and a point, repeatedly, to build a high dimensional regular simplex for which all pairs of points have distance equal to one. In that case, not surprisingly, the dependence of $C(\epsilon)$ on ϵ simply tells us that the data fills the space. As a second example, consider a d-dimensional data set for which the vector components take values in $\{\pm 1\}$ and are IID with zero mean. Then for any pair i, j, we have $E[\|\mathbf{x}_i - \mathbf{x}_j\|^2] = 2d$ and we can apply Hoeffding's bound to give:

$$P(|\|\mathbf{x}_i - \mathbf{x}_j\|^2 - 2d| \geq d\epsilon) = P(|\mathbf{x}_1 \cdot \mathbf{x}_2| \geq d\epsilon/2) \leq 2\exp\left(-\frac{d\epsilon^2}{8}\right). \tag{2.2}$$

Thus for $\epsilon = 0.2$ and $d = 500$, the probability that the pairwise distance squared exceeds its mean is bounded above by 0.164; and for $\epsilon = 0.2$ and $d = 5{,}000$, the probability is bounded above by 2.8×10^{-11}. Happily most real data sets have more structure: the features are not IID and distance-based algorithms such as k-th nearest neighbor can often work well. If k-th nearest neighbor (for supervised classification) often works well, we can hope that distance-based dimension reduction techniques will too. We will encounter another way of looking at this problem in the next section.

2.2 Empirical Investigation

Although Grassberger and Procaccia [43] only considered the limit as $\epsilon \to 0$, it's interesting to extend their argument to consider the effective dimension of the data at different length scales. One way to visualize this is to plot $\log(C(\epsilon))$ versus $\log(\epsilon)$ and examine how the slope varies with $\log(\epsilon)$. Figure 2.3 shows two 2-spheres; on the left, the data is uniformly distributed, while on the right, the data becomes denser toward the poles (the Matlab code to generate this data is given below). Here 20,000 points were sampled. The corresponding plots of $\log(C(\epsilon))$ versus $\log(\epsilon)$ are shown in Figure 2.4, together with straight line fits. A straight line fit to the uniform data gives the slope (estimated dimension) as 1.9924: the data has the same dimension over a wide range of scales. Although the method is clearly invariant to a global scaling of the density of the data, it's interesting to test its sensitivity to local variations; a straight line fit to the "snowy sphere" data gives a slope of 1.76 on all the data, 1.89 for the 10,000 smallest pairwise distances, and 1.92 for just the first 1,000 pairwise distances (the plot shows the straight line fit using all the data). However, note that this sensitivity (of the estimate using all available distances to variations in density) is a different issue from the notion of a scale-dependent dimension,

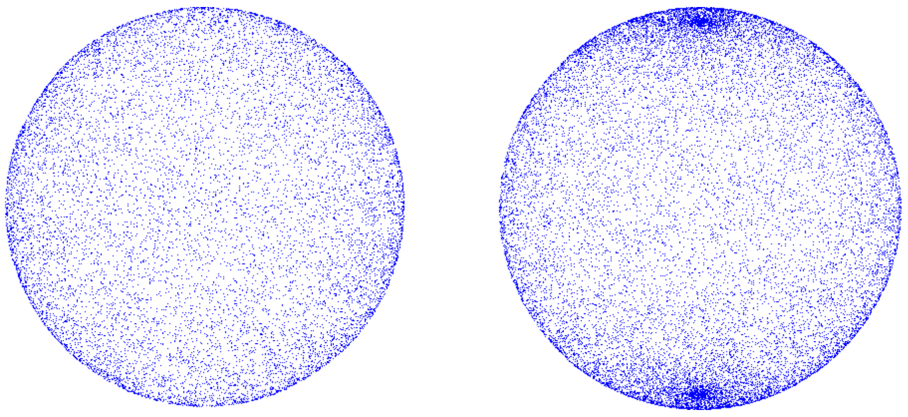
Fig. 2.3 Left: samples uniformly distributed over the 2-sphere. Right: samples whose density varies as the cosine of the azimuth. (Note that the spheres are transparent, so all points are visible.)

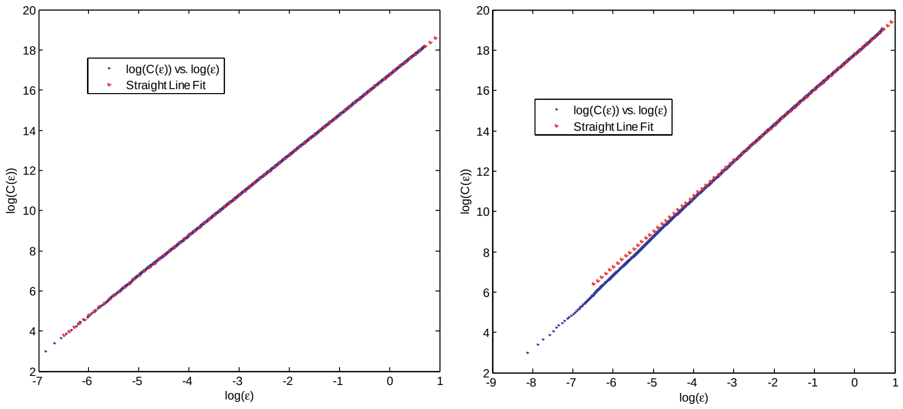

Fig. 2.4 Straight line fits for the pairwise distance sphere data shown in Figure 2.3.

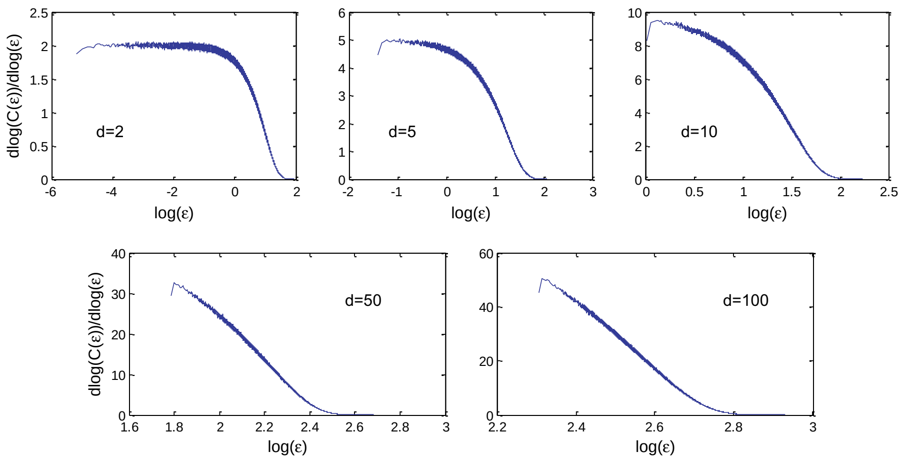

Fig. 2.5 Slope of $\log(C(\epsilon))$ versus $\log(\epsilon)$, versus $\log(\epsilon)$, for spherical Gaussians in 2, 5, 10, 50, and 100 dimensions, 50,000 samples, and using a smoothing window of size 5,000 points to estimate the slope. The modeled dimension underestimates the actual dimension more severely as the dimension increases.

which would be computed as the slope of $\log(C(\epsilon))$ versus $\log(\epsilon)$ at a given ϵ.

Next we examine the sensitivity of the estimates to the dimension of the problem at hand. Figure 2.5 shows plots of estimates of the slopes of the $\log(C(\epsilon))$ versus $\log(\epsilon)$ curves for spherical Gaussians in 2, 5, 10, 50, and 100 dimensions. The figures were generated by sampling

50,000 points from each distribution, and using a smoothing window containing 5,000 points; note that the slopes are generated from graphs with 2.5 billion pairwise distances. We see that the estimates increasingly underestimate the true dimension as that dimension increases.

Finally, we apply the method to the 2008 KDD CUP data. We take the first 500 patients and compute the pairwise distances. The plot on the left in Figure 2.6 is included simply as a "cautionary tail": the densities along the tails of the plot are very low, and a straight line fit along the main body of the curve will fail: most of the data lives on the upper lip of the curve, as the histogram superimposed on the left figure shows. (One can also see this in Figure 2.4.) This effect is an echo of the observation made in the previous section: put simply (but imprecisely), high dimensional data tends to be close to equidistant. A straight line fit in $\log(\epsilon) \in [2,3]$ gives a slope of 2.45 with an error of $\delta = 0.2122$ (by "error" we mean that, making the approximation that the deviations from the fit are independent, normal, and have constant variance, then $y \pm \delta$ contains at least 50% of the predictions). However, as the above examples suggest, although these estimates of the intrinsic dimension are more accurate when they are lower, the estimates are only approximate; but they can nevertheless be useful as guidance for choosing starting points for manifold modeling methods that require

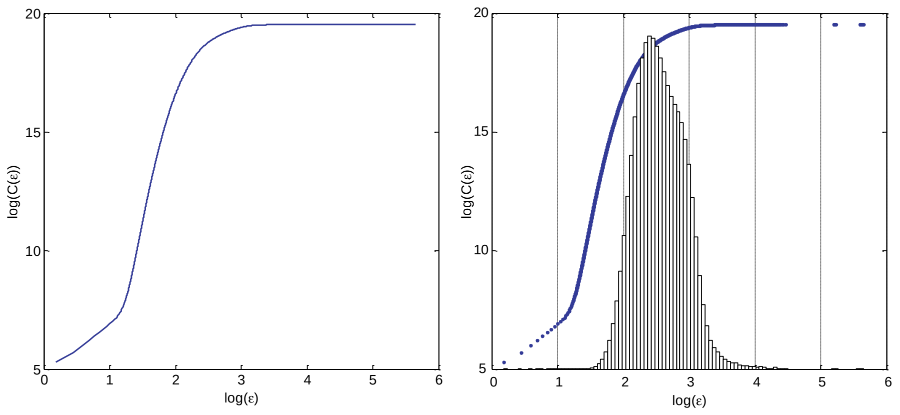

Fig. 2.6 A naïve line plot (left) suggests different dimensions (slopes) at several scales, but plotting the individual points and overlaying the histogram (right) reveals that the leftmost tail is almost empty, and in fact most of the data sits close to the right inflection point.

that one has in hand an estimate of the intrinsic dimension. Note also that this method is just one of many possible techniques for estimating the intrinsic dimension: see for example Lee and Verleysen [62].

Algorithm 1 Generating data on the 2-sphere

```
$n \leftarrow 20000;$
$theta = randn(n, 1) * pi - pi/2;$
$phi = randn(n, 1) * 2 * pi;$
$ctr = 1;$
$snowy = false;$
for $i = 1 : n$ do
  $t = theta(i);$
  $p = phi(i);$
  if $rand \leq abs(cos(t)) \;\vee\; snowy$ then
    $positions(ctr, 1) = cos(t) * cos(p);$
    $positions(ctr, 2) = cos(t) * sin(p);$
    $positions(ctr, 3) = sin(t);$
    $ctr = ctr + 1;$
  end if
end for
```

3

Projective Methods

If dimension 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, [37]). 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[1]). Apart from handling high dimensional data, projection pursuit methods can be robust to noisy or irrelevant features [57], and have been applied to regression [35], 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 [36]. How are the interesting directions found? One approach is to search for projections such that the projected data departs from normality [57]. 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 data set is

[1] See J. H. Friedman’s interesting response to Huber [57] in the same issue.

still approximately normal, then the data set 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 [31] (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 data sets are approximately normal, perhaps projections are not always the best way to find low dimensional representations. Let's review their results in some 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.[2] Then (Diaconis and Freedman [31], Theorem 1.1) the empirical distribution of the projections along any given unit direction tends to $N(0,\sigma^2)$ weakly in probability.[3] 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 [31]).

As a concrete example, consider data uniformly distributed over the unit $n+1$-sphere $\mathcal{S}^{n+1}$ for odd[5] 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. The setup is shown in Figure 3.1. 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

[2] More formally, the conditions are: for σ^2 positive and finite, and for any positive ϵ, $(1/m)\mathrm{card}\{j \le m : |\|\mathbf{x}_j\|^2 - \sigma^2 d| > \epsilon d\} \to 0$ and $(1/m^2)\mathrm{card}\{1 \le j,k \le m : |\mathbf{x}_j \cdot \mathbf{x}_k| > \epsilon d\} \to 0$ [31].

[3] Some authors refer to convergence "weakly in probability" simply as convergence in probability. A sequence X_n of random variables is said to converge in probability to a random variable X if $lim_{n\to\infty} P(|X_n - X| > \epsilon) = 0$ for all $\epsilon > 0$ [45].

[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 formulas are slightly different.

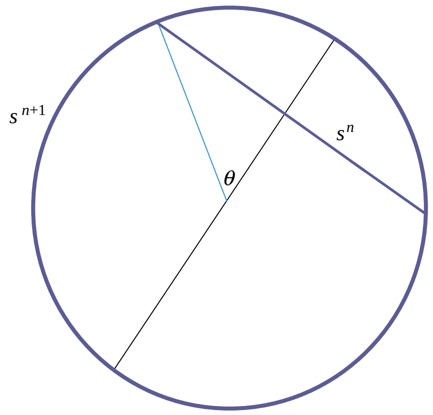

Fig. 3.1 Data distributed over the surface of an $n+1$-sphere. Although the distribution is far from Gaussian (and in fact the data is constrained to lie on a manifold, so the data has zero density in most of its convex hull), projections of the data along one dimension give densities that are close to Gaussian, and those densities become closer to Gaussian as n increases.

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:

$$C = 2^{-\frac{1}{2}(n+1)} \frac{n!!}{(\frac{1}{2}(n-1))!}. \tag{3.1}$$

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 3.2 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.

3.1 Independent Component Analysis

The notion of searching for non-normality, which is at the heart of projection pursuit (the goal of which is dimension reduction), is also a key idea underlying independent component analysis (ICA) [58], so we give a brief description here. ICA views the data as being generated by a

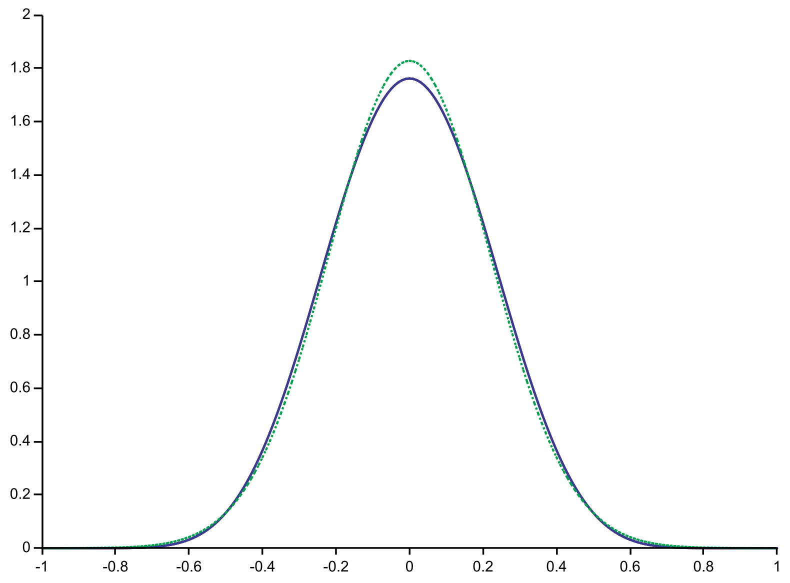

Fig. 3.2 Dotted line: a Gaussian density with zero mean and variance 1/21. Solid line: the density projected from data distributed uniformly over the 20-sphere, to any line passing through the origin.

mixture of unknown latent variables, and although typically the number of latent variables is assumed to equal the dimension of the data, the method has parallels with dimension reduction. ICA searches for projections such that the probability distributions of the data along those projections are statistically independent. Consider for example the case of two speakers speaking into two microphones, where each microphone captures sound from both speakers. The microphone signals may be written $\mathbf{y} = A\mathbf{x},\ \mathbf{x}, \mathbf{y} \in \mathcal{R}^2$, where the components of $\mathbf{x}$ are the (assumed statistically independent and zero mean) signals from each individual speaker, and where A is a fixed two-dimensional mixing matrix. In principle, we could separate out the source signals by finding A and inverting it. However, both A and $\mathbf{x}$ are unknown here, and any invertible scaling of each component of $\mathbf{x}$, followed by any permutation of the components of the rescaled $\mathbf{x}$ (the net result of which is another pair of statistically independent variables) can be compensated for by redefining A. We can remove the scaling degrees of freedom from the problem by whitening the data $\mathbf{y}$ and then assuming that A is a rotation matrix, which amounts to choosing a coordinate system in which $\mathbf{x}$ is white (which, since the x_i are independent and zero mean, is equivalent to just rescaling the x_i). Note that this also means that if $\mathbf{x}$ happens to be normally distributed, then ICA fails, since A can then be any orthogonal matrix (since any orthogonal matrix applied to independent, unit variance Gaussian variables results in independent, unit variance Gaussian variables). To give nontrivial results, ICA therefore requires that the original signals be non-Gaussian (or more precisely, that at most one is Gaussian distributed), and in fact it turns out that finding the maximally non-Gaussian component (under the assumptions that the $\mathbf{x}$ are IID, zero mean, and unit variance) will yield an independent component [58]. ICA components may also be found by searching for components with minimum mutual information, since zero mutual information corresponds to statistical independence. Such functions — whose optimization leads to the desired independent components — are called contrast functions. Bach and Jordon [5] approach ICA by proposing contrast functions based on canonical correlation analysis (CCA) in Reproducing Kernel Hilbert Spaces (RKHSs); we will encounter CCA, and RKHSs used in similar ways, below.

3.2 Principal Component Analysis (PCA)

3.2.1 PCA: Finding an Informative Direction

Given data $\mathbf{x}_i \in \mathcal{R}^d$, $i = 1,\ldots,m$, 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] $\theta_i \in \mathcal{R}$, $\sum_i \theta_i = 0$, and $\mathbf{n} \in \mathcal{R}^d$ has unit length. The sample variance of

[6] Note that if all x_i lie on a given line then so does μ.

the projection along $\mathbf{n}$ is then[7]:

$$v_n \equiv \frac{1}{m}\sum_{i=1}^{m}((\mathbf{x}_i - \boldsymbol{\mu}) \cdot \mathbf{n})^2 = \frac{1}{m}\sum_{i=1}^{m}\theta_i^2, \tag{3.2}$$

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

$$v_n' \equiv \frac{1}{m}\sum_{i=1}^{m}((\mathbf{x}_i - \boldsymbol{\mu}) \cdot \mathbf{n}')^2 = \frac{1}{m}\sum_{i=1}^{m}\theta_i^2(\mathbf{n} \cdot \mathbf{n}')^2. \tag{3.3}$$

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 maximal variance. If instead your data lies in a two (or higher) dimensional subspace, the above argument can be repeated, picking off the highest variance directions in turn. The next section investigates how that works. There, and in the following section, we will follow the intuitive description used in this section, using projections to describe the properties of PCA; in Section 3.2.6, we will describe the same results using more concise matrix-based methods, which will also provide a transition to the matrix methods used in the rest of the review.

[7] When the choice is immaterial to the argument, we use denominator m (sample viewed as the whole population) rather than $m - 1$ (unbiased estimator of population variance).

3.2.2 PCA: Ordering by Variance

We have seen that directions of maximum variance can be interesting, but how can we find them? From here on, unless otherwise stated, we allow the x_i to be arbitrarily distributed. The sample variance along an arbitrary unit vector $\mathbf{n}$ is $\mathbf{n}^T C\mathbf{n}$ where C is the sample covariance matrix. Since C is positive semidefinite, its eigenvalues are positive or zero; let us choose the indexing such that the (unit norm) 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 (or can be so chosen, if several share the same eigenvalue), we can expand any $\mathbf{n}$ in terms of them: $\mathbf{n} = \sum_{a=1}^{d} \alpha_a \mathbf{e}_a$, and we would like to find the α_a that maximize $\mathbf{n}^T C\mathbf{n} = \mathbf{n}^T \sum_a \alpha_a C\mathbf{e}_a = \sum_a \lambda_a \alpha_a^2$, subject to $\sum_a \alpha_a^2 = 1$ (to give unit normed $\mathbf{n}$). This is just a convex combination of the λs, 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 principal eigenvectors, if the principal eigenvalue has geometric multiplicity greater than one), and furthermore, the sample variance of the projection of the data along $\mathbf{n}$ is then 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 $\mathbf{m}^T C\mathbf{m} = \sum_{a=2}^{d} \lambda_a \beta_a^2$, 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 decreasing projected variance. PCA for feature extraction thus 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. Finally, 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, given that one wants to project to a $d' < d$-dimensional space, if one uses the d' principal directions, then the mean-squared error introduced by representing the data by their projections along these directions is minimized.

3.2.3 PCA Decorrelates the Data

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, no assumptions regarding the distribution of X have been made here.

3.2.4 PCA: Reconstruction with Minimum Squared Error

The basis provided by the eigenvectors of the covariance matrix is also optimal for dimension 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 dimension 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 data set 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 indexed in order of non-increasing eigenvalues, then choosing those eigenvectors as basis vectors will give minimal reconstruction error, as we will show. If the data is not centered, then the mean should be subtracted first, the dimension reduction performed, and the mean then added back[8]; 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:

$$\frac{1}{m}\sum_i \|\mathbf{x}_i - \tilde{\mathbf{x}}_i\|^2 = \frac{1}{m}\sum_i \|\mathbf{x}_i\|^2 - \frac{1}{m}\sum_{a=1}^{d'}\sum_i (\mathbf{x}_i \cdot \mathbf{u}_a)^2, \tag{3.4}$$

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

$$-\sum_{a=1}^{d'} \mathbf{u}_a^T C \mathbf{u}_a = -\sum_{a=1}^{d'} (\sum_{p=1}^{d} \beta_{ap}\mathbf{e}_p^T) C (\sum_{q=1}^{d} \beta_{aq}\mathbf{e}_q) = -\sum_{a=1}^{d'}\sum_{p=1}^{d} \lambda_p \beta_{ap}^2. \tag{3.5}$$

Introducing Lagrange multipliers ω_{ab} to enforce the orthogonality constraints [16], in order to minimize the reconstruction error we must maximize:

$$F = \sum_{a=1}^{d'}\sum_{p=1}^{d} \lambda_p \beta_{ap}^2 - \sum_{a,b=1}^{d'} \omega_{ab} \left(\sum_{p=1}^{d} \beta_{ap}\beta_{bp} - \delta_{ab} \right). \tag{3.6}$$

Choosing[9] $\omega_{ab} \equiv \omega_a \delta_{ab}$ 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 $\omega_a = \lambda_a$ and $\beta_{ap} = \delta_{ap}$ for $p \le d'$, $\beta_{ap} = 0$ otherwise. The objective function then simply becomes $\sum_{p=1}^{d'} \lambda_p$, which is maximized by choosing the first d' largest λ_p. Note that this also amounts to a proof that, for projections that give minimal reconstruction error, the "greedy"

[8] 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.

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

approach to PCA dimension 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 observation applies to finding projections that maximally reduce the residual variance. Again, note that this argument is distribution independent.

3.2.5 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 $X \sim \mathcal{N}(0,C)$. Then since the $\mathbf{y}$s are linear combinations of the $\mathbf{x}$s, they are also normally distributed, with zero mean and sample 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 X and that of Y is maximized [6, 32]. Since the mapping W is deterministic, the conditional entropy $H(Y|X)$ vanishes, and the mutual information is just $I(X,Y) = H(Y) - H(Y|X) = H(Y)$. Using a small, fixed bin size, we can approximate this by the differential entropy,

$$H(Y) = -\int p(\mathbf{y})\log_2 p(\mathbf{y})d\mathbf{y} = \frac{1}{2}\log_2(e(2\pi)^{d'}) + \frac{1}{2}\log_2\det(C_y). \tag{3.7}$$

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 [97] for a proof for the general case $1 < d' < d$.

3.2.6 The Matrix View of PCA

Here we revisit the maximal variance projection and decorrelation properties of PCA using the more succinct matrix based approach.[10] If E is the (orthonormal) matrix of column eigenvectors of the covariance matrix C, and Λ the diagonal matrix of (nonnegative) eigenvalues of C, then

$$CE = E\Lambda, \tag{3.8}$$

and we can always choose the ordering of the columns of E so that the $\lambda_i \equiv \Lambda_{ii}$ are ordered: $\lambda_i \leq \lambda_{i+1} \ \forall \ i = 1, \ldots, d-1$. Now for some unit vector $\mathbf{n}_1 \in \mathcal{R}^d$ consider the quantity:

$$\mathbf{n}_1' E^T C E \mathbf{n}_1 = \mathbf{n}_1' \Lambda \mathbf{n}_1. \tag{3.9}$$

The left-hand side is the variance of the projections of the (centered) data along the unit vector $E\mathbf{n}_1$. The right-hand side is $\sum_i n_{1i}^2 \lambda_i$, and since $\sum_i n_{1i}^2 = 1$, this is a convex combination of the λs, which is maximized by choosing the largest λ, i.e., by choosing $n_{1i} = \delta_{i,1}$. For that choice of $\mathbf{n}_1$, $E\mathbf{n}_1$ is the principal eigenvector, and the variance of the data projected along that direction is just λ_1. We can repeat the same argument for the direction that is orthogonal to $E_{\cdot 1}$ by searching for the unit vector $\mathbf{n}_2$ that is orthogonal to $\mathbf{n}_1$ (i.e., for which $n_{21} = 0$) and which maximizes the right-hand side, which is given by $n_{2i} = \delta_{i,2}$, and the corresponding direction (that maximizes the variance of the projections of the centered data in the subspace orthogonal to $\mathbf{n}_1$) is just $E\mathbf{n}_2$, the second principal eigenvector. Applying the same argument iteratively shows that the eigenvectors of C give the desired directions, and the corresponding variances are the λs.

[10] The above vector-based views are useful to facilitate our intuitive understanding of the properties of PCA, but once one has this, matrix-based methods are usually preferred for their brevity and rich mathematical support.

Regarding decorrelating the data, suppose that we replace the data by their projections along the eigenvectors of C, that is, $\mathbf{x} \to E^T\mathbf{x}$. Then if $\boldsymbol{\mu}$ is the mean data vector, we also have $\mathbf{x} - \boldsymbol{\mu} \to E^T(\mathbf{x} - \boldsymbol{\mu})$, and the covariance matrix of the transformed data is:

$$C' = \frac{1}{m}\sum_{i=1}^{m} E^T(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T E = E^T C E = \Lambda, \qquad (3.10)$$

so C' is diagonal (and thus the data in the new coordinate system is decorrelated).

3.2.7 Dimension Reduction with PCA

Given the above, one method for performing dimension reduction is to compute the principal components for the data, and to take projections of the feature vectors along them. For example, if only two eigenvalues are nonzero, this will map the data to a two-dimensional space with no error. Figure 3.3 shows the eigenspectrum, and the results of projecting along the first three principal directions, for the features corresponding to the first 500 patients in the KDD Cup data. The 24,406 points that are labeled negatively are shown in black; the 162 positives are overlaid in yellow. Clearly, the projections have some structure that may be worth investigating with PCA, but that structure does not appear to be useful in predicting the labels.

3.3 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", and how in general would you even find such subsets? These problems could be addressed if we could learn a mixture of generative models for the data, where each model corresponded to its own PCA decomposition. Tipping and Bishop [89, 88] proposed such a model — "Probabilistic PCA" — building on earlier work linking PCA decomposition to factor analysis. The advantages of a probabilistic model are numerous: for example, the

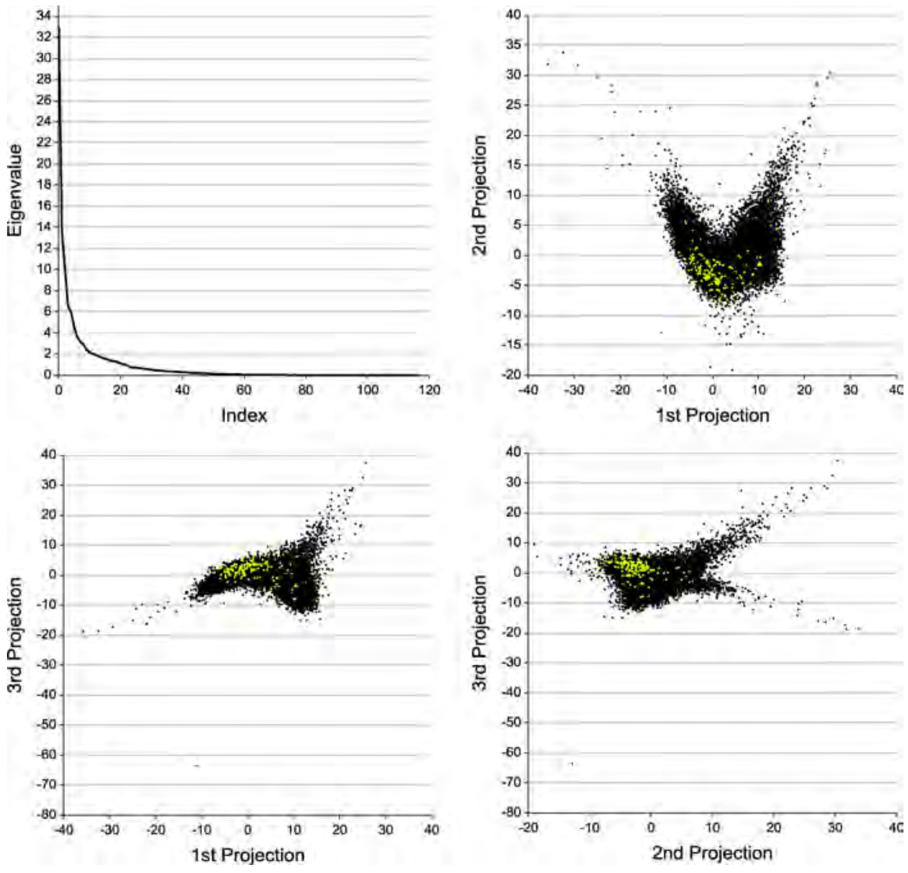

Fig. 3.3 Top left: the eigenspectrum. Top right and bottom: data viewed by its projections along the first three principal components. Samples with positive labels are colored yellow.

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 in fact a form of factor analysis, which itself is a classical dimension reduction technique. Factor analysis first appeared in the behavioral sciences community over a century ago, when Spearman hypothesized that intelligence could be reduced to a single underlying factor [83]. If, given an n-by-n correlation matrix between variables $X_i \in \mathcal{R}$, $i = 1, \ldots, n$, there is a single variable g such that the conditional 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 $\mathbf{z} \in \mathcal{R}^n$ [28]. Modern factor analysis usually considers a model where the underlying factors $X \in \mathcal{R}^{d'}$ are Gaussian, and where a Gaussian noise term $\boldsymbol{\epsilon} \in \mathcal{R}^d$ is added:

$$\begin{aligned} Y &= WX + \boldsymbol{\mu} + \boldsymbol{\epsilon} \\ X &\sim \mathcal{N}(0, \mathbf{1}) \\ \boldsymbol{\epsilon} &\sim \mathcal{N}(0, \Psi). \end{aligned} \tag{3.11}$$

Here $Y \in \mathcal{R}^d$ 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, Y has mean $\boldsymbol{\mu}$ and "model covariance" $WW' + \Psi$. For this model, given X, the vectors $Y - \boldsymbol{\mu}$ become uncorrelated, and ϵ_i captures the variance that is unique to Y_i. Since X and $\boldsymbol{\epsilon}$ are Gaussian distributed, so is Y, and so the maximum likelihood estimate of $\boldsymbol{\mu}$ is just the empirical expectation of the $\mathbf{y}$'s. However, in general, W and Ψ must be estimated iteratively, using for example the EM algorithm [30]. There is an instructive exception to this [7]. Suppose that $\Psi = \sigma^2 \mathbf{1}$, so that the $d - d'$ smallest eigenvalues of the model covariance are the same and are equal to σ^2. Suppose also that S, the sample covariance of the $\mathbf{y}$'s, is equal to the model covariance; we can then read off d' as the multiplicity of the smallest eigenvalue σ^2 of S. Let $\mathbf{e}^{(j)}$ be the j-th orthonormal eigenvector of S with eigenvalue λ_j. Then it is straightforward to check that $W_{ij} = \sqrt{(\lambda_j - \sigma^2)}\mathbf{e}_i^{(j)}$, $i = 1, \ldots, d$, $j = 1, \ldots, d'$ satisfies $WW' + \Psi = S$ 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) assumes a model of the form (Equation (3.11)) 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 [89]:

$$W_{ML} = U(\Lambda - \sigma^2 \mathbf{1})R, \tag{3.12}$$

$$\sigma^2_{ML} = \frac{1}{d - d'} \sum_{i=d'+1}^{d} \lambda_i, \tag{3.13}$$

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 X given Y, in the limit $\sigma \to 0$, in which case the components of X become the PCA projections of Y). This closed form result is rather striking in view of the fact that for general factor analysis (for example, for diagonal but non-isotropic Ψ) 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 has missing components; and d' (which so far we've assumed known) can itself be estimated using Bayesian arguments [11]. 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 [30], thus giving a principled approach to combining several local PCA models [88].

3.4 The Kernel Trick

Before describing our next extension of PCA — Kernel PCA — we outline a mathematical device it shares with many other algorithms (for example, support vector machines (SVMs); for a simple example see Burges [15]). Given samples $\mathbf{x}_i, i = 1, \ldots, m$, suppose you have an algorithm (for example, k-th nearest neighbor) which depends only the inner products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$. Note that here we assume only that such inner products can be defined, and in particular we do not assume that $\mathbf{x}_i \in \mathcal{R}^d$; for example the $\mathbf{x}_i$ could be graphs, or sets of categories. Now suppose we map the data to a (possibly infinite dimensional) vector space $\mathcal{F}$ via the mapping $\mathbf{\Phi} : \mathbf{\Phi}(\mathbf{x}) \in \mathcal{F}$. We further require that $\mathcal{F}$ be complete and come equipped with an inner product (in other words, $\mathcal{F}$ is a Hilbert space). Now consider applying the same algorithm to the transformed data $\mathbf{\Phi}(\mathbf{x}_i)$. Since the algorithm depends only on the inner products between (the representations of the) samples, the algorithm is also well defined in $\mathcal{F}$. Now suppose there exists a (symmetric) "kernel" function $k(\mathbf{x}_i, \mathbf{x}_j)$ such that for all $\mathbf{x}_i$, $\mathbf{x}_j$, $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{\Phi}(\mathbf{x}_i), \mathbf{\Phi}(\mathbf{x}_j) \rangle$. Since your algorithm depends only on inner products, $\mathbf{\Phi}(\mathbf{x})$ need never be explicitly computed; one can always just substitute the value of the kernel, whenever the value of an inner product is needed. This is the "kernel trick": given any algorithm that depends only on inner products between samples, the algorithm can be computed implicitly in any space $\mathcal{F}$ for which k can be defined, which means, for example, that linear algorithms can be mapped to one of a very rich set of possible nonlinear versions by simple choice of the function k.

When does such a k exist for a Hilbert space? Take $\mathcal{F}$ to be a space whose elements are real-valued functions. Consider the set of linear evaluation functionals $\mathcal{I}_{\mathbf{x}} : f \in \mathcal{F} \to f(\mathbf{x}) \in \mathcal{R}$, indexed by $\mathbf{x}$. If every such linear functional is continuous, then there is a special function $k_{\mathbf{x}}$ associated with $\mathcal{F}$, also indexed by $\mathbf{x}$, and called a reproducing kernel, for which $\langle f, k_{\mathbf{x}} \rangle = f(\mathbf{x})$ for every $f \in \mathcal{F}$. Such Hilbert spaces are called Reproducing Kernel Hilbert Spaces (RKHSs) and this particular relation is called the reproducing property. In particular, the function $k_{\mathbf{x}_1}$ evaluated at some other point $\mathbf{x}_2$ is defined as $k(\mathbf{x}_1, \mathbf{x}_2) \equiv k_{\mathbf{x}_1}(\mathbf{x}_2)$, and using the reproducing property on $k_{\mathbf{x}}$ itself yields $\langle k_{\mathbf{x}_1}, k_{\mathbf{x}_2} \rangle = k(\mathbf{x}_1, \mathbf{x}_2)$. It follows from this that the kernels are symmetric in their arguments and are positive definite functions. Mapping the notation back to our description above, $\Phi(\mathbf{x})$ is simply $k_{\mathbf{x}}$. RKHSs were first introduced as a method to work implicitly in high dimensional spaces (in which classifiers are linear separating hyperplanes), by Aizerman et al. [2] in the theory of potential functions (although the formalism in Aizerman et al. [2] was not cast in terms of Hilbert spaces and kernels, the potential functions introduced are kernels in RKHSs); RKHSs gained further traction in the work of Kimeldorf and Wahba [60], who introduced the "Representer Theorem", which shows that under general conditions, the solution to a general regularized optimization problem in an RKHS can be written as an expansion over functions $k_{\mathbf{x}_i}$, where the $\mathbf{x}_i$ are training samples; and RKHSs appeared on the machine learning scene in Boser et al. [13], where they were first applied to support vector machines, to obtain classifiers that, although linear in the RKHS, are nonlinear when viewed as functions over the sample space.

Finally, note that the mapping Φ in general has no inverse: there will exist points $\mathbf{z} \in \mathcal{F}$ for which there exists no $\mathbf{x}$ such that $\mathbf{z} = \Phi(\mathbf{x})$. This means that in practice, the evaluation of a kernel algorithm requires that inner products in $\mathcal{F}$ be computed using the above kernel expansion, which can be computationally expensive. An early and very effective way to reduce this computational load is given in Burges [18].[11]

3.5 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 nonlinearly on the data), and this can severely limit the usefulness of the approach. Several versions of nonlinear PCA have been proposed in the hope of overcoming this problem (see e.g., Diamantaras and Kung [32]). In this section we describe one such algorithm called kernel PCA [81].

Kernel PCA applies the kernel trick to create a nonlinear version of PCA in sample space by performing ordinary 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 (although the hope for those doing dimension reduction is that a number $d' \ll d$ of projections will suffice). It is 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 inner products between the vectors. However, Schölkopf et al. [81] show how the problem can indeed be formulated entirely in terms of inner 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 inner product of each side of the eigenvalue equation with each (centered) mapped data point. Let's see

[11] I am told that this method is used to speed up SVM classifiers that recognize all handwritten addresses, and approximately 20% of machine print addresses, by the United States Postal Service today, as well as in several other countries [M. Parakhin, Private Communication, 2010].

how this works. The covariance matrix of the mapped data in feature space is:

$$C \equiv \frac{1}{m}\sum_{i=1}^{m}(\mathbf{\Phi}_i - \boldsymbol{\mu})(\mathbf{\Phi}_i - \boldsymbol{\mu})^T, \tag{3.14}$$

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

$$C\mathbf{v} = \lambda\mathbf{v}. \tag{3.15}$$

Since this can be written as $\frac{1}{m}\sum_{i=1}^{m}\Psi_i\langle\Psi_i, \mathbf{v}\rangle = \lambda\mathbf{v}$, the eigenvectors $\mathbf{v}$ lie in the span of the Ψ_is, so the k-th eigenvector can be expanded as:

$$\mathbf{v}^k = \sum_i \alpha_i^k \Psi_i \tag{3.16}$$

for some α_i^k. Note that, although the dimension of Ψ_i may be very high (or even infinite), there are only m α_is (for a given eigenvector): we will denote the vector whose i-th component is α_i by $\boldsymbol{\alpha} \in \mathcal{R}^m$. Since the $\mathbf{v}$s lie in the span of the Ψ_is, we can equivalently look for solutions of the m equations:

$$\langle\Psi_i, C\mathbf{v}\rangle = \lambda\langle\Psi_i, \mathbf{v}\rangle. \tag{3.17}$$

Now consider:

$$\langle\Psi_i, \Psi_j\rangle = K_{ij} - \frac{1}{m}\sum_k\langle\Phi_i, \Phi_k\rangle - \frac{1}{m}\sum_k\langle\Phi_k, \Phi_j\rangle + \frac{1}{m^2}\sum_{kl}\langle\Phi_k, \Phi_l\rangle, \tag{3.18}$$

where $K_{ij} \equiv k(\mathbf{x}_i, \mathbf{x}_j)$ is the matrix of inner products[12] in $\mathcal{F}$. Letting $\mathcal{I}$ denote the m-by-m matrix with all entries equal to $\frac{1}{m}$, then the second term on the right-hand side is,[13] for any j,

$$-\frac{1}{m}\sum_k K_{ik} = -\sum_k K_{ik}\mathcal{I}_{kj} = (-K\mathcal{I})_{ij}, \tag{3.19}$$

[12] A matrix of inner products is called a Gram matrix. Any Gram matrix G is necessarily positive semidefinite, as is easily seen in this case from $\mathbf{z}'K\mathbf{z} = \sum_{ij} z_i z_j\langle\Phi_i, \Phi_j\rangle = \|\sum_i z_i\Phi_i\|^2$.

[13] The above derivation emphasizes the relation between kernels and inner products. A more compact derivation for general Gram matrices is given in Section 4.2.

the third term is, for any i,

$$-\frac{1}{m}\sum_k K_{kj} = -\sum_k \mathcal{I}_{ik}K_{kj} = (-\mathcal{I}K)_{ij}, \tag{3.20}$$

and the fourth term is, for any i, j,

$$\frac{1}{m^2}\sum_{kl} K_{kl} = \sum_{lk} \mathcal{I}_{ik}K_{kl}\mathcal{I}_{lj} = (\mathcal{I}K\mathcal{I})_{ij}, \tag{3.21}$$

so

$$\langle \Psi_i, \Psi_j\rangle = K - K\mathcal{I} - \mathcal{I}K + \mathcal{I}K\mathcal{I} = (\mathbf{1}-\mathcal{I})K(\mathbf{1}-\mathcal{I}) \equiv PKP, \tag{3.22}$$

where we have introduced the projection matrix $P \equiv \mathbf{1} - \mathcal{I}$ and where $\mathbf{1}$ is the m-by-m unit matrix. Thus the centered version of the kernel matrix is $\bar{K} \equiv PKP$. Combining Equations (3.14), (3.16), (3.17), and (3.22) gives:

$$\bar{K}\bar{K}\boldsymbol{\alpha} = m\lambda\bar{K}\boldsymbol{\alpha}. \tag{3.23}$$

Now every solution to

$$\bar{K}\boldsymbol{\alpha} = m\lambda\boldsymbol{\alpha} \tag{3.24}$$

is also a solution of Equation (3.23), and it turns out that for our purposes, it is sufficient to solve Equation (3.24). To see this, note that every solution of (3.23) can be written as $\boldsymbol{\alpha}_{\mathcal{N}} + \boldsymbol{\alpha}_{\perp}$, where $\boldsymbol{\alpha}_{\mathcal{N}}$ lies in the null space $\mathcal{N}$ of $\bar{K}$ and where $\boldsymbol{\alpha}_{\perp}$ lies in the orthogonal subspace $\mathcal{N}_{\perp}$; then $\boldsymbol{\alpha}_{\mathcal{N}}$ is also a solution to Equation (3.23), and $(\bar{K}\boldsymbol{\alpha}_{\perp})$ is also a solution to Equation (3.24). Hence the solutions to Equation (3.23) that are in $\mathcal{N}_{\perp}$, and the solutions to Equation (3.24), are in 1–1 correspondence. We can ignore solutions $\boldsymbol{\alpha}_{\mathcal{N}} \in \mathcal{N}$ since to compute the projection of a given mapped sample $\mathbf{x}_j$ we only need to compute:

$$\langle \Psi_j, \mathbf{v}\rangle = \sum_i \alpha_i \langle \Psi_j, \Psi_i\rangle = (\bar{K}(\boldsymbol{\alpha}_{\mathcal{N}} + \boldsymbol{\alpha}_{\perp}))_j = (\bar{K}\boldsymbol{\alpha}_{\perp})_j. \tag{3.25}$$

Thus we can find all relevant solutions to Equation (3.23) by taking all solutions to Equation (3.24) and pre-multiplying by $\bar{K}$. Finally, to compute the projections we need to normalize the eigenvectors in $\mathcal{F}$ to have unit length: that is,

$$\langle \mathbf{v}, \mathbf{v} \rangle = \sum_{ij} \alpha_i \alpha_j \langle \Psi_i, \Psi_j \rangle = m\lambda \sum_i \alpha_i \alpha_i, \tag{3.26}$$

so the $\boldsymbol{\alpha}$s must be normalized to have length $\frac{1}{\sqrt{m\lambda}}$. Since the eigenvalues of $\bar{K}$ are $m\lambda$, we can accomplish this by computing a given eigenvector of $\bar{K}$, normalizing it to have length one, and then dividing by the square root of its eigenvalue. We summarize the kernel PCA algorithm schematically below, for projections of the "in sample" points (the points used to construct K).

Algorithm 2 Kernel Principal Component Analysis (Schematic)

```
Given: $m$ samples $\mathbf{x}_i \in \mathcal{R}^d,\ i = 1, \ldots, m$
Compute the kernel matrix $K \in S_m,\ K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$
Compute the centered kernel matrix $\bar{K} \equiv PKP,\ P_{ij} = \delta_{ij} - \frac{1}{m}$
Compute the eigenvectors $\boldsymbol{\alpha}_i$ and eigenvalues $\eta_i$ of $\bar{K}$ $(i = 1, \ldots, m)$
Choose the $i$-th eigenvector $\boldsymbol{\alpha}_i$ along which you'd like to project
Normalize $\boldsymbol{\alpha}_i$ to have length $\frac{1}{\sqrt{\eta_i}}$
Then for sample $\mathbf{x}_i,\ i \in 1, \ldots, m$, the value of the projection of $\Phi(\mathbf{x}_i) \in \mathcal{F}$ along the $j$-th eigenvector $\mathbf{v}_j$ of the covariance matrix of the samples in $\mathcal{F}$ is just $\langle \Psi_i, \mathbf{v}_j \rangle = \sum_k \alpha_k^j \langle \Psi_i, \Psi_k \rangle = \sum_k \bar{K}_{ik} \alpha_k^j = \eta_j \alpha_i^j$.
```

We have not yet addressed the question of how to extend kernel PCA to an out-of-sample point $\mathbf{x}$. One could certainly just add $\mathbf{x}$ to the given samples and repeat the above computations, but this is computationally very inefficient. One could also just approximate the exact computation by computing $\langle \Psi(\mathbf{x}), \mathbf{v}^j \rangle$. This gives for the projection the value

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

where the last two terms can be dropped since they are additive constants (they don't depend on $\mathbf{x}$). One might worry that this is an ill-controlled approximation because the mean $\boldsymbol{\mu}$ is no longer correct, and even if we assume that we can ignore the change in $\boldsymbol{\mu}$, the above argument is no longer correct either, since there is no reason why $\Psi(\mathbf{x})$ should lie in the span of the $\Psi(\mathbf{x}_i)$ (in fact for RBF kernels, unless $\mathbf{x}$ happens to coincide with one of the $\mathbf{x}_i$, it won't). However, Williams and Seeger [99] show that in fact, this approximation is equivalent to using the well-understood Nyström approximation, which we will describe below.

Kernel PCA may be viewed as a way of putting more effort into the up-front computation of features, rather than 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 [81]. It shares with other 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 [67]. 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 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). Figure 3.4 shows an example of applying kernel PCA to three overlapping two-dimensional Gaussians.

3.6 Canonical Correlation Analysis

Suppose we have two paired data sets $\mathbf{x}_{1i} \in \mathcal{R}^{d_1}, \mathbf{x}_{2i} \in \mathcal{R}^{d_2}$, $i = 1, \ldots, m$. Note that d_1 may not equal d_2. Canonical Correlation Analysis

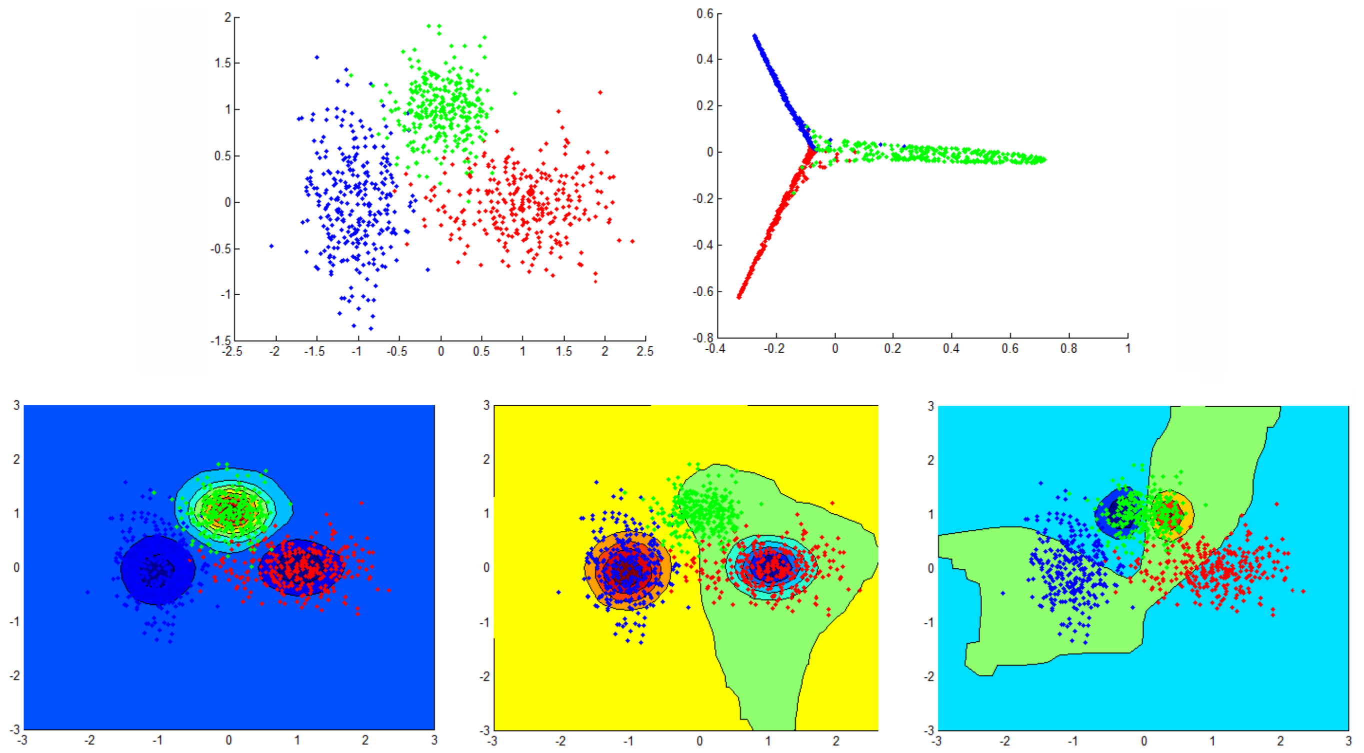

Fig. 3.4 Top left: 900 points sampled from three Gaussians (300 samples each), with covariance matrices [0.3,0;0.0.1] (rightmost cluster), [0.1,0;0,0.3] (leftmost cluster), and [0.1,0;0,0.1] (top cluster). Top right: the data plotted using the first two KPCA projections as coordinates (which are just the rows of the two principal scaled eigenvectors), for the RBF kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$ with $\sigma^2 = 0.05$. Bottom: contour plots for the projections of a grid of points in $\mathcal{L}$ (chosen in equal steps of 0.2 from $x, y = -3$ to 3) using the three principal eigenvectors in $\mathcal{F}$. Note that the first direction separates all three clusters, the second separates two, and the third splits a cluster.

(CCA) [55] finds paired directions $\{\mathbf{w}_{1i}, \mathbf{w}_{2i}\}$, $\mathbf{w}_{1i} \in \mathcal{R}^{d_1}$, $\mathbf{w}_{2i} \in \mathcal{R}^{d_2}$, $i \leq \min(d_1, d_2)$ such that the projection of the first data set along $\mathbf{w}_{1i}$ is maximally correlated with the projection of the second data set along $\mathbf{w}_{2i}$. In addition, for $i \neq j$, the projections of the first data set along the pairs $\{\mathbf{w}_{1i}, \mathbf{w}_{1j}\}$, of the second data set along the pairs $\{\mathbf{w}_{2i}, \mathbf{w}_{2j}\}$, and of the first and second data sets along the pairs $\{\mathbf{w}_{1i}, \mathbf{w}_{2j}\}$, respectively, are all uncorrelated. Furthermore, the values of the $\mathbf{w}.\mathbf{x}$'s themselves are invariant to invertible affine transformations of the data, which gives CCA a coordinate independent meaning, in contrast to ordinary correlation analysis. Hotelling gives the following example, taken from Kelly [59]: 140 seventh-grade school children were tested for their ability in reading and arithmetic. Two measures of efficacy were used for reading (speed and "power") and two for arithmetic (also called speed and "power"). In this case CCA revealed that, according to this data, reading and arithmetic involve one and only one common mental factor, with a p-value of approximately 0.0001. The underlying assumption in CCA is that $\mathbf{x}_{1i}$ and $\mathbf{x}_{2i}$ are different views of the same object (for example, measurements of mathematical ability, and reading ability, for the i-th seventh-grader). For a more recent treatment of CCA, see for example Anderson [4].

CCA may be summarized as follows (in this section, we will reserve the subscripts $\{p, q\}$ to lie in $\{1, 2\}$, and we remind the reader that subscripts $\{i, j\}$ index vectors, and $\{a, b\}$ index vector components). We are given two random vectors $\mathbf{X}_1$, $\mathbf{X}_2$ with ranges in $\mathcal{R}^{d_1}$ and $\mathcal{R}^{d_2}$. We assume that we are able to compute expectations of products of the random variables that are the components of the $\mathbf{X}$'s. To keep the discussion uncluttered we also assume that $E[\mathbf{X}_{1a}] = E[\mathbf{X}_{2b}] = 0$, $a = 1, \ldots, d_1$; $b = 1, \ldots, d_2$. Let us define random variables $U \equiv \mathbf{X}_1 \cdot \mathbf{w}_1$ and $V \equiv \mathbf{X}_2 \cdot \mathbf{w}_2$ for some $\mathbf{w}_1 \in \mathcal{R}^{d_1}$, $\mathbf{w}_2 \in \mathcal{R}^{d_2}$. We wish to find $\mathbf{w}_1$, $\mathbf{w}_2$, such that the correlation:

$$\rho \equiv \frac{E[UV]}{\sqrt{E[U^2]E[V^2]}} = \frac{\mathbf{w}_1' C_{12} \mathbf{w}_2}{\sqrt{(\mathbf{w}_1' C_{11} \mathbf{w}_1)(\mathbf{w}_2' C_{22} \mathbf{w}_2)}} \equiv \frac{A_{12}}{\sqrt{A_{11} A_{22}}} \tag{3.27}$$

is maximized, where $C_{pq} \equiv E[\mathbf{X}_p \mathbf{X}_q']$ is the (matrix) covariance (for $p = q$) or cross-covariance (for $p \neq q$) and where we have introduced scalars $A_{pq} \equiv \mathbf{w}_p' C_{pq} \mathbf{w}_q$. Setting the derivative of ρ^2 with respect to

w_{pa} equal to zero for $p \in \{1,2\}$ gives:

$$C_{11}^{-1} C_{12} \mathbf{w}_2 = \frac{A_{12}}{A_{11}} \mathbf{w}_1, \tag{3.28}$$

$$C_{11}^{-1} C_{12} C_{22}^{-1} C_{21} \mathbf{w}_1 = \rho^2 \mathbf{w}_1, \tag{3.29}$$

(where we have assumed that the covariance matrices C_{11} and C_{22} are nonsingular; note that $A_{12} = A_{21}$ and that $C_{12} = C'_{21}$), and similarly for $\{1 \Leftrightarrow 2\}$.

The matrices left-multiplying the $\mathbf{w}$'s in Equations (3.28) and (3.29) are not necessarily symmetric (note that C_{12} is not necessarily square). Since the eigenvalues of general square matrices need not be real, it would be comforting to check that solving Equations (3.28) and (3.29) will always result in real, positive ρ^2. We can use Cholesky decomposition to write $C_{pp} \equiv R_{pp} R'_{pp}$, where R_{pp} is lower triangular [54]: then writing $\mathbf{z}_1 \equiv R'_{11} \mathbf{w}_1$, Equation (3.29) becomes:

$$R_{11}^{-1} C_{12} C_{22}^{-1} C_{21} R_{11}^{\prime -1} \mathbf{z}_1 = \rho^2 \mathbf{z}_1. \tag{3.30}$$

The left hand multiplicand is now a (symmetric) positive definite matrix, since for any vector $\mathbf{s} \in \mathcal{R}^{d_1}$, we have:

$$\mathbf{s}' R_{11}^{-1} C_{12} C_{22}^{-1} C_{21} R_{11}^{\prime -1} \mathbf{s} = \mathbf{t}' \mathbf{t}, \tag{3.31}$$

where $\mathbf{t} \equiv R_{22}^{-1} C_{21} R_{11}^{-1} \mathbf{s}$, so ρ^2 is indeed real and positive.

While we are on the subject of sanity checks, it is conceivable that Equation (3.29) is necessary but not sufficient: that is, can there exist eigenvalues of the eigenvector Equation (3.29) for which ρ does not take the form $\rho^2 = A_{12}^2/(A_{11}A_{22})$? No, because Equation (3.29) and the $\{1 \Leftrightarrow 2\}$ version of Equation (3.28) gives:

$$\rho^2 A_{11} = \mathbf{w}'_1 C_{12} C_{22}^{-1} C_{21} \mathbf{w}_1 = \mathbf{w}'_1 C_{12} \frac{A_{12}}{A_{22}} \mathbf{w}_2 = \frac{A_{12}^2}{A_{22}}. \tag{3.32}$$

3.6.1 CCA Decorrelates the Data

CCA shares with PCA the property that the projections decorrelate the data. For CCA, the projections decorrelate the individual data sets just as for PCA, but in addition, the cross-correlation of the projected data vanishes, and the directions are conjugate with respect to the cross-covariance matrices. To see this, consider the set of solutions $\mathbf{w}_{1i}$ and corresponding $\mathbf{w}_{2i}$. First note that from Equation (3.29), for $\rho_i \neq \rho_j$,

$$\mathbf{w}'_{1j} C_{12} C_{22}^{-1} C_{21} \mathbf{w}_{1i} = \rho_i^2 \mathbf{w}'_{1j} C_{11} \mathbf{w}_{1i} = \rho_j^2 \mathbf{w}'_{1i} C_{11} \mathbf{w}_{1j} = 0. \tag{3.33}$$

Hence $\mathbf{w}'_{1i} C_{11} \mathbf{w}_{1j} = 0 = \mathbf{w}'_{2i} C_{22} \mathbf{w}_{2j}$. Similarly from Equation (3.28), we have $\mathbf{w}'_{2j} C_{21} \mathbf{w}_{1i} = (A_{12}/A_{22}) \mathbf{w}'_{2j} C_{22} \mathbf{w}_{2i} = 0$, again for distinct eigenvalues. For repeated eigenvalues, the $\mathbf{w}$'s may again be chosen to be conjugate with respect to the covariance matrices. Thus in the new basis, the variables are uncorrelated:

$$E[U_i U'_j] = E[\mathbf{w}_{1i} \cdot \mathbf{X}_1 \mathbf{w}_{1j} \cdot \mathbf{X}_1] = \mathbf{w}'_{1i} C_{11} \mathbf{w}_{1j} = 0 \text{ for } i \neq j, \tag{3.34}$$

and similarly $E[V_i V'_j] = E[U_i V'_j] = 0$ if $i \neq j$.

3.6.2 CCA is Invariant under Invertible Affine Transformations

What happens to the $\mathbf{w} \cdot \mathbf{x}$ projections if we translate, rotate, or scale the data? For example, do the projections change if we whiten the data sets first? One of the strengths of CCA is that this is not necessary: the projected values are invariant under invertible affine transformations $\mathbf{x} \in \mathcal{R}^d \to B\mathbf{x} + b$, $B \in M_d$, $b \in \mathcal{R}^d$, provided the $\mathbf{w}$'s are appropriately transformed.

Invariance with respect to translations follows directly from the definition of ρ, since covariance matrices are functions of the centered data. We can check invariance under the invertible transformation $\bar{\mathbf{x}}_1 \equiv B\mathbf{x}_1$ as follows: in the new coordinate system, Equation (3.29) becomes:

$$\bar{C}_{11}^{-1} \bar{C}_{12} C_{22}^{-1} \bar{C}_{21} \bar{\mathbf{w}}_1 = \rho^2 \bar{\mathbf{w}}_1, \tag{3.35}$$

where $\bar{C}_{11} = BC_{11}B'$, $\bar{C}_{12} = BC_{12}$ and $\bar{C}_{21} = C_{21}B'$, so that

$$C_{11}^{-1} C_{12} C_{22}^{-1} C_{21} B' \bar{\mathbf{w}} = \rho^2 B' \bar{\mathbf{w}}. \tag{3.36}$$

Hence the eigenvalues ρ take the same values. Thus solving in the transformed coordinate system we see that we will find $\bar{\mathbf{w}}_1$ which are related to $\mathbf{w}_1$ by $\mathbf{w}_1 = B'\bar{\mathbf{w}}_1$, so that for any $\mathbf{x}_1 \in S_1$, $\mathbf{w}_1 \cdot \mathbf{x}_1 = (\bar{\mathbf{w}}_1 B) \cdot (B^{-1} \bar{\mathbf{x}}_1) = \bar{\mathbf{w}}_1 \cdot \bar{\mathbf{x}}_1$. Thus the projections themselves remain invariant, and hence the correlations between projections remain invariant. By simply swapping $\{1 \leftrightarrow 2\}$ in the above argument we see that in this sense, CCA is invariant under invertible affine transformations of both S_1 and S_2 independently.

Note that the property of affine invariance is not shared by ordinary correlation analysis, in the sense that the matrix whose ab-th element is $E[x_{1a}x_{2b}]/\sqrt{E[x_{1a}^2]E[x_{2b}^2]}$ can take very different forms in different coordinate systems. For example, given a set of random variables that are distributed as the components of a multivariate Gaussian, one can choose an affine transformation to a new coordinate system in which the data are uncorrelated: correlation alone is a coordinate-dependent concept.

3.6.3 CCA in Practice; Kernel CCA

The expectations in the above analysis require knowledge of the underlying distributions, and this is often not available. In that case one usually uses the empirical distribution:

$$P(X_{pia} = x_{pia},\ X_{qjb} = x_{qjb}) = (1/m)\delta_{ij}, \tag{3.37}$$

giving covariance matrices (for zero mean data):

$$\begin{aligned} C_{pa,qb} &= (1/m)\sum_{i,j=1}^{m} \mathbf{x}_{pia}\mathbf{x}_{qjb}P(X_{pia} = x_{pia},\ X_{qjb} = x_{qjb}) \\ &= (1/m)\sum_{i=1}^{m} \mathbf{x}_{pia}\mathbf{x}_{qib}. \end{aligned} \tag{3.38}$$

Since CCA may be viewed as an extension of PCA to two paired data sets, and since the kernel trick can be applied to PCA, it's reasonable to expect that a kernel version of CCA might be developed. This is indeed the case, as first shown independently in Akaho [3] and in Bach and Jordan [5].

Kernel CCA follows kernel PCA in spirit. The data $\mathbf{x}_1 \in \mathcal{R}^{d_1}$, $\mathbf{x}_2 \in \mathcal{R}^{d_2}$ are mapped to feature spaces $\mathcal{F}_1$ and $\mathcal{F}_2$ by maps Φ_1, Φ_2, respectively (note that $\mathcal{F}_1$ and $\mathcal{F}_2$ may or may not be the same). Since the $\mathbf{w}_{1i} \in \mathcal{F}_1$, $\mathbf{w}_{2i} \in \mathcal{F}_2$ are used only to take projections, we can assume that they lie in the span of the data, so that there exist α_p such that:

$$\mathbf{w}_p = \sum_{i=1}^{m} \alpha_{pi} \Phi_p(\mathbf{x}_{pi}), \tag{3.39}$$

where we have dropped the index enumerating the $\mathbf{w}$'s (and the corresponding index on the αs) for clarity. Thus, for a given solution, $\alpha_p \in \mathcal{R}^m$. Since CCA depends only on inner products the Φ's are never explicitly needed:

$$\mathbf{w}_p \cdot \Phi_p(\mathbf{x}_{pj}) = \sum_{i=1}^{m} \alpha_{pi} \langle \Phi_p(\mathbf{x}_{pi}), \Phi_p(\mathbf{x}_{pj}) \rangle = \sum_{i=1}^{m} \alpha_{pi} K_p(\mathbf{x}_{pi}, \mathbf{x}_{pj}). \tag{3.40}$$

Following the above analysis, but in the spaces $\mathcal{F}_p$, yields:

$$\rho = \max_{\alpha_1, \alpha_2} \frac{\alpha_1' K_1 K_2 \alpha_2}{\sqrt{\alpha_1' K_1^2 \alpha_1 \alpha_2' K_2^2 \alpha_2}}, \tag{3.41}$$

where $K_p \in M_m$. For any data and mapping Φ_p for which the K's are invertible, this can be solved analytically; however, the solutions have perfect correlation (or anticorrelation): $\rho = \pm 1$. An example of such a choice of $\mathcal{F}$ is the space corresponding to radial basis function kernels. Such a mapping clearly gives too much "wiggle room" to the data; we need to regularize. This can be achieved with the same regularization device used in partial least squares, by penalizing the norm of the $\mathbf{w}_p$ vectors. For large data sets, an additional problem must be addressed: a square matrix with number of rows equal to the sample size must be inverted. This can be overcome using approximate techniques such as incomplete Cholesky decomposition. We refer the reader to Bach and Jordan [5] and to Hardoon et al. [47] for details.

While CCA was originally proposed as a kind of factor analysis for paired data sets, the projections can also be used as (heuristic) similarity measures: Hardoon et al. [47] consider the problem of content-based image retrieval from the Web, where vectors in S_1 represent the image content and vectors in S_2 represent the text surrounding the image. At run time, the user enters some text, and an appropriate image is hopefully retrieved. The similarity of a piece of text and an image is defined as the cosine between the vector whose i-th component is $\mathbf{x}_1 \cdot \mathbf{w}_{1i}$ and the vector whose i-th component is $\mathbf{x}_2 \cdot \mathbf{w}_{2i}$; the lookup requires a scan over the image database for each incoming text query.

Kernel CCA also immediately yields a way to use kernels to assess statistical independence, which Bach and Jordan [5] exploited to develop kernel ICA, and which has led to many works on kernel-based assessment of independence: see for example Gretton et al. [44], as well as Fukumizu et al. [38], which is discussed in Section 3.9.4.

3.7 Linear Discriminant Analysis

In the following few sections we will consider the supervised setting, where some form of signal for each data point is available (for example, a class label; or an annotation that defines whether the point is an undistorted signal, or is a distorted version of that signal). Here for completeness, and because Distortion Discriminant Analysis (described in the next section) is closely related, we briefly describe the classical approach to dimension reduction for labeled data: Fisher Linear Discriminant Analysis (LDA) for binary labels, and its multiclass generalization, Multiple Discriminant Analysis (MDA). These methods may be viewed as natural extensions of PCA to the case of labeled data. Consider the task of binary classification, and consider the problem of finding a unit vector $\mathbf{n}$ such that the projections of the two classes along $\mathbf{n}$ are well separated. We can make a first attempt at defining what we mean by "well separated" by asking that the difference of the means of the two classes, $\mu_1 - \mu_2$, have maximum projection along $\mathbf{n}$, that is by maximizing:

$$(\mathbf{n} \cdot (\mu_1 - \mu_2))^2 \equiv \mathbf{n}' S_B \mathbf{n}, \ \|\mathbf{n}\|_2 = 1, \tag{3.42}$$

where $S_B \equiv (\mu_1 - \mu_2)(\mu_1 - \mu_2)'$ is known as the "between-class scatter" [33]. Equation (3.42) is trivially satisfied by setting:

$$\mathbf{n} = \frac{\mu_1 - \mu_2}{\|\mu_1 - \mu_2\|_2}. \tag{3.43}$$

This is clearly a sensible thing to do if one happens to know that each class is spherically distributed and that the classes are linearly separable. However, consider the data shown in the left panel of Figure 3.5.

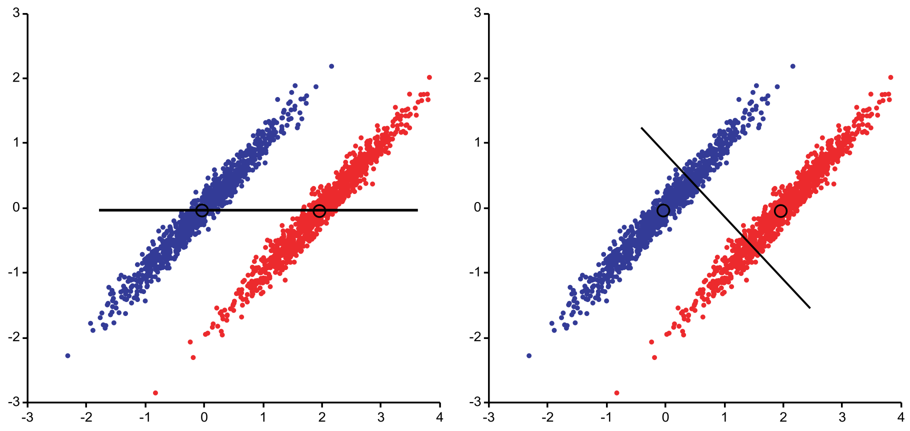

Fig. 3.5 Both panels: Two sets of 1,000 Gaussian distributed points, with one class on the left and the other on the right. Left panel: the difference of the means, the positions of which are denoted by the small circles, can give a poor projection direction for class separation. Right panel: the direction shown is that given by LDA for this data. Note that only the direction matters; the placement of the lines denoting the unit vectors is arbitrary.

There, the above **n** is clearly suboptimal because each class has projections along **n** that strongly overlaps: that is, the value of the projection along **n** is not a good predictor of class membership. Instead, we would like to maximize the inter-class projection along **n** as above, but simultaneously minimize the variance of the intra-class projections along **n**, where we can represent the latter using the pooled variance (recall that for a data set with covariance matrix C, the variance along the unit vector **n** is just $\mathbf{n}'C\mathbf{n}$):

$$\mathbf{n}'\frac{1}{m}\left\{\sum_{i=1}^{m_1}(\mathbf{x}_{1i}-\mu_1)(\mathbf{x}_{1i}-\mu_1)'+\sum_{i=1}^{m_2}(\mathbf{x}_{2i}-\mu_2)(\mathbf{x}_{2i}-\mu_2)'\right\}\mathbf{n}\equiv\mathbf{n}'S_W\mathbf{n}, \tag{3.44}$$

where the number of samples in class $i\in\{1,2\}$ is denoted m_i, where $m\equiv m_1+m_2$, and where S_W is known as the "within-class scatter" [33]. Fisher linear discriminant analysis thus finds directions that maximize the ratio $\frac{S_B}{S_W}$; this ratio is known as a "Rayleigh quotient" and the maximizing directions **n** are found as the eigenvectors of the generalized eigenvalue equation:

$$S_B\mathbf{n}=\lambda S_W\mathbf{n}. \tag{3.45}$$

The right panel of Figure 3.5 shows the direction given by the principal eigenvector of Equation (3.45) for the data shown. Note that Equation (3.45) is equivalent to an ordinary eigenvalue equation if S_W is of full rank.

Extending these ideas to the multiclass case is straightforward. If C is the number of classes, a simple approach is to compute C LDA directions, one for each "one versus rest" problem (i.e., separate class 1 from the rest, then class 2, etc.) (note that these directions in general will not be orthogonal). One can also extend the above argument directly to the multiclass case, resulting again in a problem whose solution maximizes a Rayleigh quotient. There, the mapping is to a space of dimension $C - 1$ (this approach assumes that the original dimension d satisfies $d \geq C - 1$); the within-class scatter S_W becomes the obvious extension of the binary case, where the terms on the left-hand side of Equation (3.44) are replaced by C such terms, one for each class; and the between-class scatter becomes the weighted sum:

$$S_B = \frac{1}{m}\sum_{i=1}^{C} m_i(\mu_i - \mu)(\mu_i - \mu)', \tag{3.46}$$

where μ is the overall mean. Again the directions are the generalized eigenvectors of Equation (3.45), using the new definitions of S_B and S_W. For more details see for example Dusa and Hart [33].

3.8 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 [19, 20]. We first describe the method of oriented PCA (OPCA) [32]. 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 $\mathbf{z}_i^k \equiv \tilde{\mathbf{x}}_i^k - \mathbf{x}_i$. For example, the $\mathbf{x}_i$ could be a spectral representation of a piece of music recorded in a studio, while the $\tilde{\mathbf{x}}_i^k$, could be the same representation of the same music, recorded after applying a noise filter (for example, recording the output of an FM radio playing the piece,

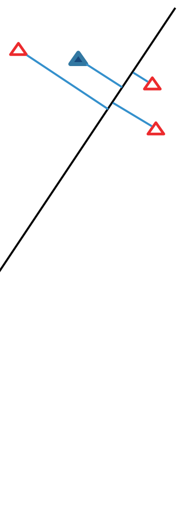

Fig. 3.6 OPCA searches for directions in which the projections of a signal vector (filled triangle), and of its noisy variants (open triangles), are close, while the projections of different signal vectors (star, filled triangle) are far from each other.

or simply the data encoded using a low bit rate). In order to map the noisy data to a representation which is as close as possible (in some metric) to the original, 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: this is illustrated in Figure 3.6.

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 and 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 [32] solves this problem. The first OPCA direction is defined as that direction $\mathbf{n}$ that maximizes the generalized Rayleigh quotient [32, 33] $q_0 = \frac{\mathbf{n}'C_1\mathbf{n}}{\mathbf{n}'C_2\mathbf{n}}$, 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:

$$C \equiv \frac{1}{m}\sum_i (\mathbf{x}_i - E[\mathbf{x}])(\mathbf{x}_i - E[\mathbf{x}])', \tag{3.47}$$

$$R \equiv \frac{1}{mN}\sum_{i,k} \mathbf{z}_i^k (\mathbf{z}_i^k)', \tag{3.48}$$

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 and 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, $\mathbf{n}_i'C\mathbf{n}_j \propto \delta_{ij}$, $\mathbf{n}_i'R\mathbf{n}_j \propto \delta_{ij}$.

OPCA is similar mathematically to multiclass discriminant analysis (MDA) where the number of classes is equal to m [33], but there is a crucial difference: in MDA, there is no notion of a “canonical” (or “signal”) sample for each class; the MDA within-class scatter for a given class is computed as the covariance matrix for that class. In OPCA, for each class, the mean vector used in MDA is replaced by the single canonical (zero noise) point, which can lie far from the sample mean. This is done in both the numerator, where each MDA class mean is replaced by the corresponding signal point, and where the overall mean is replaced by the mean of the signal points; and in the denominator, where the sample covariance for a given class is replaced by sums of squares of differences between noise vectors for that class and the signal vector for that class. This amounts to leveraging additional, valuable information about the problem, and can lead to significantly improved results for problems where such data is available (such as the audio fingerprinting task, where a very clean version of each original clip can be obtained).

"Distortion Discriminant Analysis" [19, 20] 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 and are simply concatenated 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).

3.9 Sufficient Dimension Reduction

In this section we continue the supervisory thread and consider techniques for dimension reduction where the data consists of predictor–response pairs $\{\mathbf{x}_i, y_i\}$, $i = 1, \ldots, m$. There are several reasons one might want to do this: if the $\mathbf{x}$'s appear in the underlying functional dependence only through a small number of projections, then those projections may be used to construct various plots to visualize the data; and smaller, more accurate models of the regression itself can be constructed if one knows that an entire subspace can be ignored.

We follow Cook's characterization of sufficient dimension reduction (SDR) [23]: let X be the random vector and Y the random variable taking the values $\{\mathbf{x}_i, y_i\}$, $\mathbf{x}_i \in \mathcal{R}^d$, $y_i \in \mathcal{R}$, respectively, and assume that the pair (Y, X) has joint distribution P. The goal of sufficient dimension reduction is then to find a map $\Phi : \mathcal{R}^d \to \mathcal{R}^q$, $q < d$, such that $Y|X \sim Y|\Phi(X)$ (that is, Y conditioned on X has the same distribution as Y conditioned on $\Phi(X)$). The following are then equivalent:

$$X|(Y, \Phi(X)) \sim X|\Phi(X) \ \textit{(inverse regression)} \tag{3.49}$$

$$Y|X \sim Y|\Phi(X) \ \textit{(forward regression)} \tag{3.50}$$

$$Y \perp\!\!\!\perp X|\Phi(X) \ \textit{(sufficient reduction)} \tag{3.51}$$

An equivalent formulation is as follows. We consider models of the form:

$$y = f(\mathbf{a}_1'\mathbf{x}, \mathbf{a}_2'\mathbf{x}, \ldots, \mathbf{a}_k'\mathbf{x}, \epsilon), \quad \mathbf{a}_i, \mathbf{x} \in \mathcal{R}^d, \quad \epsilon \in \mathcal{R}, \tag{3.52}$$

where the ϵ's model the noise and are assumed independent of X. The goal is to find the minimal number of vectors $\mathbf{a}_i'$ for which the above relation holds, in which case the $\mathbf{a}_i'$ span a "minimal dimension reduction subspace" (DRS) [22]. Note that this problem is ill-posed in the sense that given any solution, another solution can be constructed by changing f and the $\mathbf{a}_i'$ appropriately. The presence of the y's can drastically change the picture: for example, X could be distributed in such a way that no useful dimension reduction of the X alone is possible, whereas Y might depend only on a single component of X. Now let A denote that matrix whose columns are the $\mathbf{a}_i$. Then the above list of inner products may be written as $A^T\mathbf{x}$ and the DRS is that subspace spanned by the columns of A^T: we will denote this by S_A. Again we can write this as a statement about statistical independence as follows:

$$Y \perp\!\!\!\perp X \mid A^T X. \tag{3.53}$$

The goal of Sufficient Dimension Reduction[14] is to estimate a DRS, when it exists. Let's start by describing one of the earliest approaches to SDR.

3.9.1 Sliced Inverse Regression

Sliced Inverse Regression (SIR) was introduced in a seminal paper by Li [64]. Since this paper sparked a fruitful line of research, we will examine the basic ideas in detail here. Normal (forward) regression estimates $E[Y|\mathbf{x}]$. Inverse regression instead estimates $E[X|y]$, which is a much easier problem since it amounts to solving d one-dimensional regression problems. It is a remarkable fact that a DRS for the above problem (Equation (3.52)) can be estimated, up to degeneracies we will describe below, when the marginal $p(X)$ is elliptic,[15] and assuming that the $\mathbf{x}_i$ are IID. This can be done despite the fact that, as mentioned above, the problem as stated is ill-posed, and despite the fact that we know nothing about f or ϵ directly. As y varies, $E[X|y]$ will trace a curve in $\mathcal{R}^d$. Noting that, given the form (Equation (3.52)), for fixed ϵ, a small change $\mathbf{x} \to \mathbf{x} + \delta\mathbf{x}$ in the subspace orthogonal to S_A leaves y unchanged, one might hope to find conditions under which $E[X|y]$ can be shown to lie in S_A. Li [64] gives us one such sufficient condition, which we explore next.

Theorem 3.1. Given Equation (3.52), further assume that $E[X|\mathbf{a}_1'\mathbf{x}, \mathbf{a}_2'\mathbf{x}, \ldots, \mathbf{a}_k'\mathbf{x}]$ lies in the subspace spanned by $\Sigma_X \mathbf{a}_i$, where Σ_X is the covariance matrix of X. Then the centered inverse regression curve $E[X|y] - E[X]$ lies in that subspace.

Proof. Here we will sacrifice brevity and a little generality in the cause of gaining further insight: we will assume that $p(X)$ is elliptic, and first show that this leads to the condition in the proof. We will denote the

[14] The phrase *Sufficient Dimension Reduction* was introduced to the statistics community by Cook and Lee [25]. The phrase *Sufficient Dimensionality Reduction* was introduced to the machine learning community by Globerson and Tishby [39]. The approaches are quite different; we briefly summarize the latter below.

[15] An elliptic density is one for which the contours of constant density are ellipsoids, such as the Gaussian.

hyperplane defined by $\cap_i^k \{\mathbf{x} : \mathbf{a}_i'\mathbf{x} = \alpha_i\}$, simply by $\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}$. First note that for *any* density $p(X)$, $E[X|\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}]$ must itself lie on the hyperplane $\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}$, since

$$\mathbf{a}_j' E[X|\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}] = E[\mathbf{a}_j' X|\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}] = E[\alpha_j|\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}] = \alpha_j. \tag{3.54}$$

However, this is not quite what we need: while this does mean that the expectation lies in a subspace, that subspace will vary as the α_i vary. We are interested in a stronger characterization of a subspace that depends on the $\mathbf{a}_i$ only.

Let's change coordinates to a basis in which the density is spherical,[16] $\mathbf{z} = \Sigma_X^{-1/2}\mathbf{x}$. Introducing $\mathbf{b}_i = \Sigma_X^{1/2}\mathbf{a}_i$, in this coordinate system the constraints $\mathbf{a}_i'\mathbf{x} = \alpha_i$ become $\mathbf{a}_i'\Sigma_X^{1/2}\mathbf{z} \equiv \mathbf{b}_i'\mathbf{z} = \alpha_i$. Consider the quantity:

$$E[Z|\{\mathbf{b}_i'\mathbf{z} = \alpha_i\}]. \tag{3.55}$$

The $\mathbf{b}_i$ need not be orthogonal: however, we can always introduce an orthonormal set $\mathbf{u}_i$ such that for some β_i, the hyperplane:

$$H \equiv \{\mathbf{a}_i'\mathbf{x} = \alpha_i\} = \{\mathbf{b}_i'\mathbf{z} = \alpha_i\} = \{\mathbf{u}_i'\mathbf{z} = \beta_i\} \tag{3.56}$$

(since any $n-k$ hyperplane can be expressed as the intersection of k $n-1$ hyperplanes with orthogonal normals). Since $p(Z)$ is spherical and is centered at the origin, the induced density on H will also be spherical, and will be centered on the point of closest approach of H to the origin. Now points on H can be written as $\mathbf{z} = \sum_{i=1}^k \mathbf{u}_i\beta_i + \mathbf{u}_\perp$, where $\mathbf{u}_\perp'\mathbf{u}_i = 0$: the $\mathbf{u}_\perp$ lies in H. The nearest point on H to the origin is therefore $\sum_{i=1}^k \mathbf{u}_i\beta_i$, since $\mathbf{u}_\perp = 0$ minimizes $\|\sum_{i=1}^k \mathbf{u}_i\beta_i + \mathbf{u}_\perp\|_2$, and so transforming back to the $\mathbf{b}$'s, there must exist scalars γ_i such that:

$$E[Z|\{\mathbf{u}_i'\mathbf{z} = \beta_i\}] = \sum_{i=1}^k \mathbf{u}_i\beta_i = \sum_{i=1}^k \mathbf{b}_i\gamma_i = \sum_{i=1}^k \Sigma_X^{1/2}\mathbf{a}_i\gamma_i. \tag{3.57}$$

[16] We assume that Σ_X has been regularized if necessary so that Σ_X^{-1} (and the density $p(X)$ itself) exists.

Applying $\Sigma_X^{1/2}$ to both sides gives the result:

$$E[X|\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}] = \sum_{i=1}^{k} \Sigma_X \mathbf{a}_i \gamma_i, \tag{3.58}$$

so $E[X|\{\mathbf{a}_i'\mathbf{x} = \alpha_i\}]$ is spanned by the $\Sigma_X \mathbf{a}_i$.

This geometric argument has brought us quite close to the desired result: we wish to show a similar result for

$$E[X|Y] = E[X|f(\mathbf{a}_1'X, \mathbf{a}_2'X, \ldots, \mathbf{a}_k'X, \epsilon)]. \tag{3.59}$$

By the tower property [74, 73], and by conditional independence, $E[X|Y] = E[E[X|\{\mathbf{a}_iX\}, Y]|Y] = E[E[X|\{\mathbf{a}_iX\}]|Y]$, so by linearity of expectation, for any $\mathbf{s}$ in the space orthogonal to the $\Sigma_X \mathbf{a}_i$, we have that $\mathbf{s}'E[X|Y] = E[\mathbf{s}'E[X|\{\mathbf{a}_iX\}]|Y] = 0$. □

The SIR algorithm is summarized below.

Algorithm 3 Sliced Inverse Regression

```
Choose number of buckets $N_B$
Whiten the data: $\mathbf{x} \to \mathbf{z} \equiv \Sigma_X^{-1/2}(\mathbf{x} - \mu)$
For each bucket $b_i$ compute the mean $\mu_i \equiv \frac{1}{|b_i|}\sum_{j \in b_i} \mathbf{z}_j$
Compute weighted sample covariance: $C = \frac{1}{m}\sum_{i=1}^{N_B} |b_i|\mu_i\mu_i'$
Compute principal eigenvectors $\eta_k$
Output $a_k = \eta_k \Sigma_X^{-1/2}$
```

3.9.2 Sliced Average Variance Estimation

Note that SIR may only estimate a subspace of S_A. Consider the one-dimensional example shown in Figure 3.7. There, the estimated subspace has dimension zero, since $E[X|Y] = 0$.

SIR is a first moment method; using second or higher moments would help solve the symmetry problem, and Cook and Weisberg [26] propose Sliced Average Variance Estimate (SAVE) to this end. In SAVE, $var(X|Y)$ rather than $E[X|Y]$ is expanded in terms of the central subspace directions. Again let $\mathbf{z}_i$ denote the centered, whitened version of the $\mathbf{x}_i$, let P_η be the projection operator to S_A (the DRS for

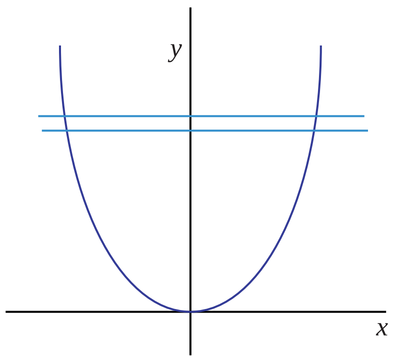

Fig. 3.7 SIR is only guaranteed to find a subspace of a minimum Dimension Reduction Subspace.

the **z**'s), and let $Q_\eta \equiv \mathbf{1} - P_\eta$ be the complementary projection operator that projects to the subspace orthogonal to S_A. Then if the **x**'s are elliptically distributed, we have [26]:

$$cov(\mathbf{z}|y) = w_y Q_\eta + P_\eta cov(\mathbf{z}|y) P_\eta, \tag{3.60}$$

where w_y is just one, if the **x** are Gaussian distributed, or a function of y if not. Rearranging terms we see that:

$$w_y \mathbf{1} - cov(\mathbf{z}|y) = P_\eta (w_y \mathbf{1} - cov(\mathbf{z}|y)) P_\eta \tag{3.61}$$

in other words, the matrix $w_y \mathbf{1} - cov(\mathbf{z}|y)$ is equal to its projection to S_A, which means that its eigenvectors are elements of S_A. Cook and Weisberg [26] thus propose the kernel:

$$\frac{1}{m} \sum_i |b_i| (\mathbf{1} - cov(\mathbf{z}|y)_i)^2, \tag{3.62}$$

using the same notation as above (taking the square results in a positive semidefinite matrix). The algorithm is given below.

3.9.3 SIR and SAVE Compared

Let's compare SIR and SAVE on a simple problem: spherical Gaussian data in ten dimensions (for these experiments we used 100,000 samples). Figure 3.8 shows the eigenvalues and eigenvectors for the two

Algorithm 4 Sliced Average Variance Estimation

```
Choose number of buckets $N_B$
Whiten the data: $\mathbf{x} \to \mathbf{z} \equiv \Sigma_X^{-1/2}(\mathbf{x} - \mu)$
For each bucket $b_i$ compute the covariance $C_i(\mathbf{z}_i), i \in b_i$
Compute the kernel $K \equiv \frac{1}{m}\sum_{i=1}^{N_B} |b_i|(\mathbf{1} - C_i)^2$
Compute principal eigenvectors $\eta_k$
Output $a_k = \eta_k \Sigma_X^{-1/2}$
```

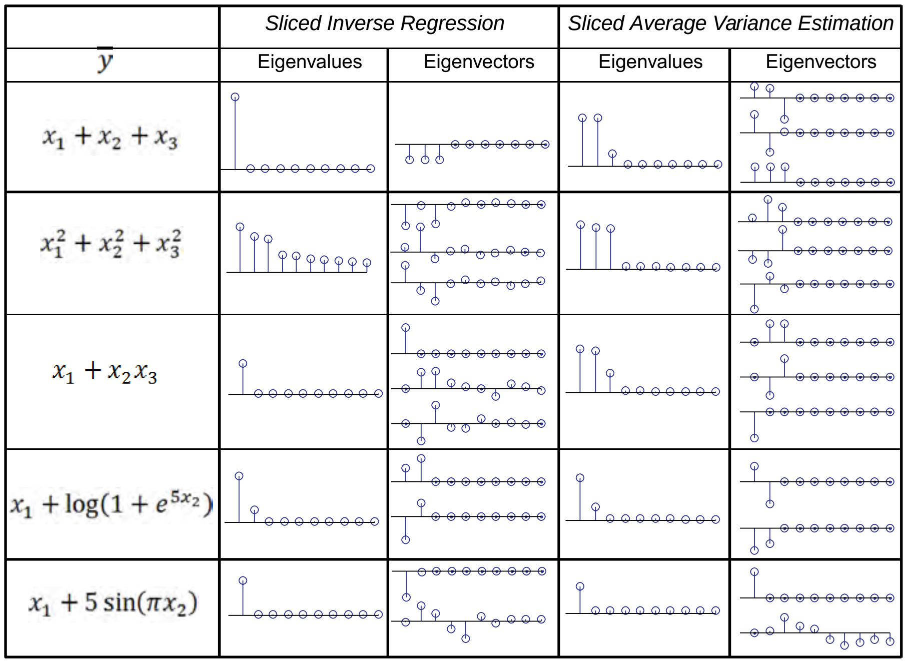

Fig. 3.8 A comparison of SIR versus SAVE for conditional dimension reduction on Gaussian data.

approaches, given the chosen y-dependence shown in the first column. To read this, one would determine which eigenvalues are above some threshold, and estimate S_A as the span of the corresponding eigenvectors.

For $y = x_1 + x_2 + x_3$, we see that the SIR eigenvalues identify a subspace of dimension one (and the corresponding eigenvector lies in that subspace), while the SAVE results correctly identify the three-dimensional subspace (and note that the three principal eigenvectors have the correct span). For $y = x_1^2 + x_2^2 + x_3^2$, SIR fails in a different way: the eigenvalues do not show a clear cutoff at three dimensions. However, as expected (since it uses second moments), SAVE succeeds. For $y = x_1 + x_2 x_3$, SIR again underestimates the dimension but SAVE saves the day. For $y = x_1 + \log(1 + 5e^{x_2})$, both methods succeed in identifying a two-dimensional subspace. Finally, for $y = x_1 + 5\sin(\pi x_2)$, both methods identify the subspace spanned by x_1 but fail to identify the full subspace. (The factor of 5 was chosen to ensure that the oscillation is significant where the data has large support; too small an amplitude would give a y that is well-approximated by a one-dimensional subspace.)

We end this section with a brief tour of some related work. Li [65] also proposes a second-order method, "principal Hessian directions" (pHd), to handle the symmetry problem. The idea is based on the fact that the Hessian of the forward regression function will be degenerated along directions orthogonal to the central subspace. A basis is chosen in which, for the first coordinate, the average curvature of the regression function along that axis is maximal, then this is repeated for each successive coordinate. Those coordinates are then identified as central subspace directions. Li [65] recommends using both SIR and pHd for any given problem, since the former tends to be more stable, but the latter handles symmetric cases. Li et al. [63] propose approaching SDR by estimating contour directions of small variations in the response; the method solves the problem of finding the full set of central subspace directions (unlike SIR and pHd), but it still assumes elliptic X, although robustness to departures from ellipticity is claimed. More recently, Cook and Forzani [24] present a maximum likelihood estimator of the DRS which empirically shows significantly improved performance over SIR and SAVE.

3.9.4 Kernel Dimension Reduction

SIR, where applicable, has the significant advantages, that is, easy to implement and can handle large data sets. However, as noted above it has some limitations. First, SIR, and the above methods it inspired, assumes elliptically distributed data. SIR can miss finding central subspace directions, for example if the data has symmetries under which the inverse regression is invariant, and similarly, pHd can miss such directions if the corresponding coordinates only appear as variances in f. In fact, the dimension of the space that SIR finds is bounded above, for tasks in which y takes one of c discrete values, by $c-1$, which limits its applicability to classification tasks. Kernel dimension reduction (KDR) [38] addresses all of these issues, and the approach is aimed directly at the defining condition for Sufficient Dimension Reduction: $Y \perp\!\!\!\perp X|A^T\mathbf{x}$. Furthermore, the approach is very general. We briefly summarize the ideas here and refer the reader to Fukumizu et al. [38] for details. Associate with the random variables X and Y (where the latter is no longer restricted to be a scalar), Reproducing Kernel Hilbert Spaces (RKHSs) $\mathcal{F}_X$, and $\mathcal{F}_Y$. In addition assign to $\mathcal{F}_X$ and $\mathcal{F}_Y$ the Lebesque measures of the probability spaces over which X and Y are defined (so that, for example, for $f_1, f_2 \in \mathcal{F}_X$, $\langle f_1, f_2\rangle = \int f_1(x)f_2(x)dP(X)$). Then a "cross-covariance" operator $\Sigma_{YX} : \mathcal{F}_X \to \mathcal{F}_Y$ can be defined so that:

$$\langle g, \Sigma_{YX} f\rangle = E_{XY}[(f(X) - E_X[f(X)])(g(Y) - E_Y[g(Y)])]. \quad (3.63)$$

A *conditional covariance operator* $\Sigma_{YY|X} \equiv \Sigma_{YY} - \Sigma_{YX}\Sigma_{XX}^{-1}\Sigma_{XY}$ is then defined.[17] Next, introduce a matrix whose d columns (if X takes values in $\mathcal{R}^d$) are orthonormal, so that BB^T is a projection operator to the subspace spanned by those columns. Let k_X, k_Y be the kernels associated with RKHS's $\mathcal{F}_X$ and $\mathcal{F}_Y$, respectively. Define a corresponding kernel over the subspace by $k^B(\mathbf{x}_1, \mathbf{x}_2) \equiv k(B^T\mathbf{x}_1, B^T\mathbf{x}_2)$. Since the Σ operators can be defined in terms of the kernels, this leads to a cross-covariance operator for the subspace: $\Sigma^B_{YY|X} \equiv \Sigma_{YY} - \Sigma^B_{YX}\Sigma^{B\ -1}_{XX}\Sigma^B_{XY}$. Fukumizu et al. [38] then show that, subject to some weak conditions on $\mathcal{F}_X$, $\mathcal{F}_Y$ and the probability measures, $\Sigma^B_{YY|X} \geq \Sigma_{YY|X}$ (where the inequality refers to an ordering that can be defined for self-adjoint operators), and that $\Sigma^B_{YY|X} = \Sigma_{YY|X} \Leftrightarrow Y \perp\!\!\!\perp X|B^TX$. Thus the conditional covariance operator for the projected space is

[17] This is shorthand for a form in which the last term is written in terms of bounded operators between the Hilbert spaces, and in fact is well defined when Σ_{XX}^{-1} does not exist.

directly related to the conditional independence quantity we are after. Finally, the authors write a sample version of the objective function, using centered Gram matrices G_X^B, G_Y of the data (we will encounter such quantities again below), as:

$$\begin{aligned} &Tr[G_Y(G_X^B + m\epsilon_m I_m)^{-1}] \\ &\qquad \text{subject to } B^T B = \mathbf{1}, \end{aligned} \tag{3.64}$$

where m is the sample size and ϵ a regularization parameter. B is then found using gradient descent.

So far, we have not considered the case in which there is structure in the high dimensional space. For example, suppose that your 100-dimensional data actually lies on a two-dimensional torus, with noise added in the remaining 98 directions. Nilsson et al. [69] solve such structured problems for the supervised (regression) case by combining KDR with the Laplacian eigenmaps approach to manifold modeling. We will cover the latter, which is an unsupervised version of dimension reduction on manifolds, in the next section.

We end by noting that, while KDR is appealing in how it directly solves the SDR problem, and in its freedom from the drawbacks attending the previously mentioned methods, the above optimization problem, as stated, will be intractable for large data sets (the Gram matrices are in M_{mm}; the kernels are required to be universal [84], and will in general have high rank). However, as for kernel PCA, this could be addressed by subsampling, or by using the Nyström method. Recent, parallel work by Hsing and Ren [56] also proposes RKHSs as providing a unified framework for dimension reduction through inverse regression. Here we have not considered the issue of consistency or convergence properties of the methods: we refer the reader to the papers for details.

3.9.5 Sufficient Dimensionality Reduction

Here we briefly describe *Sufficient Dimensionality Reduction* (SDR'), a similarly named but quite different technique [39]. SDR' is not a supervised method. Rather than searching for a subspace that satisfies Equation (3.53), SDR' models the density $p(X)$, parameterized by y, using two-way contingency tables. X and the model parameters Y

are discrete variables (the parameters are also treated as random variables), and SDR' is a dimension reduction method in the sense that the number of parameters needed to describe $p(X)$ is reduced from $|X||Y|$ to $(d+1)(|X|+|Y|)$, where d is the dimension of a feature space to which X is mapped and $|X|$ and $|Y|$ are the cardinalities of the sets X and Y, respectively. The key idea of SDR' is to identify feature mappings $\phi(x)$ such that the y's can be described by a small set of such features. When p is in the exponential family, such sufficient statistics can be constructed, but this is not the case otherwise: SDR' uses an information theoretic max–min framework to quantify the information about the y's that can be gleaned from the $\phi(x)$'s: hence the term "sufficient" in the designation. Although interesting, the method is not dimension reduction in the usual sense and so we will not pursue it here.

4

Manifold Modeling

In Section 3 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.[1] How, then, can such underlying structure be found? This chapter outlines some methods designed to accomplish this. We first describe the Nyström method (hereafter simply abbreviated as "Nyström"), which provides a thread linking several of the algorithms we describe.

4.1 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 approximation is exact. This is a powerful method that can be used to speed up kernel algorithms [99], to efficiently extend some algorithms (described below) to out-of-sample (test) points [9], and in some cases, to make

[1] 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.

an otherwise infeasible algorithm feasible [34]. In this section only, we adopt the notation that matrix indices refer to sizes unless otherwise stated, so that A_{mm} means that $A \in M_m$.

4.1.1 Original Nyström

The Nyström method originated as a method for approximating the solution of Fredholm integral equations of the second kind [71]. 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:

$$\int k(\mathbf{x},\mathbf{y})u(\mathbf{y})p(\mathbf{y})d\mathbf{y} = \lambda u(\mathbf{x}). \tag{4.1}$$

The integral is approximated using the quadrature rule [71]:

$$\lambda u(\mathbf{x}) \approx \frac{1}{m}\sum_{i=1}^{m} k(\mathbf{x},\mathbf{x}_i)u(\mathbf{x}_i), \tag{4.2}$$

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 Equation (4.2), which gives a much better approximation using simple interpolation [71]. 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.

4.1.2 Exact Nyström Eigendecomposition

Suppose that a kernel matrix $\tilde{K}_{mm}$ has rank $r < m$. Since $\tilde{K}_{mm}$ is 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 [54]. 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 a new kernel matrix 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:

$$K_{mm} \equiv \begin{bmatrix} A_{rr} & B_{rn} \\ B'_{nr} & C_{nn} \end{bmatrix}. \tag{4.3}$$

Since A has full rank, the r rows $\begin{bmatrix} A_{rr} & B_{rn} \end{bmatrix}$ are linearly independent, and since K has 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:

$$\begin{bmatrix} B'_{nr} & C_{nn} \end{bmatrix} = H_{nr} \begin{bmatrix} A_{rr} & B_{rn} \end{bmatrix}. \tag{4.4}$$

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[2]:

$$K_{mm} = \begin{bmatrix} A & B \\ B' & B'A^{-1}B \end{bmatrix} = \begin{bmatrix} A \\ B' \end{bmatrix}_{mr} A_{rr}^{-1} \begin{bmatrix} A & B \end{bmatrix}_{rm}. \tag{4.5}$$

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 [34]. 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:

$$K_{mm} = \begin{bmatrix} U \\ B'U\Lambda^{-1} \end{bmatrix}_{mr} \Lambda_{rr} \begin{bmatrix} U' & \Lambda^{-1}U'B \end{bmatrix}_{rm} \equiv D\Lambda D'. \tag{4.6}$$

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

[2] 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 $C = B'A^{-1}B$.

desired matrix of orthogonal column eigenvectors is:

$$V_{mr} \equiv \begin{bmatrix} A \\ B' \end{bmatrix} A^{-1/2} U_Q \Lambda_Q^{-1/2} \tag{4.7}$$

(so that $K_{mm} = V\Lambda_Q V'$ and $V'V = \mathbf{1}_{rr}$) [34].

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.

4.1.3 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 [41].

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 (4.2), sum over m' elements on the right-hand side where $m' \ll m$ and $m' > r$, and approximate the eigenvector of the full kernel matrix K_{mm} by evaluating the left-hand side at all m points [99]. Empirically, it has been observed that choosing m' to be some small integer factor larger than r works well.[3] How does using Equation (4.2) correspond to the expansion in Equation (4.6), in the case where the Schur complement vanishes? Expanding A, B in their definition in Equation (4.3) to $A_{m'm'}$, $B_{m'n}$, so that $U_{m'm'}$ contains the column eigenvectors of A and $U_{mm'}$ contains the approximated (high

[3] J. Platt, Private Communication.

dimensional) column eigenvectors, Equation (4.2) becomes:

$$U_{mm'}\Lambda_{m'm'} \approx K_{mm'}U_{m'm'} = \begin{bmatrix} A \\ B' \end{bmatrix} U_{m'm'} = \begin{bmatrix} U\Lambda_{m'm'} \\ B'U_{m'm'} \end{bmatrix}, \tag{4.8}$$

so multiplying by $\Lambda^{-1}_{m'm'}$ from the right shows that the approximation amounts to taking the matrix D in Equation (4.6) 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 [99]: 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 Equation (4.2). 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.

4.2 Multidimensional Scaling

We begin our look at manifold modeling algorithms with multidimensional scaling (MDS), which arose in the behavioral sciences [12]. MDS starts with a measure of dissimilarity between each pair of data points in the data set (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) pairwise 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 [27]). 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 $P \equiv \mathbf{1} - \frac{1}{m}\mathbf{e}\mathbf{e}' \equiv \mathbf{1} - \mathcal{I}$. We already encountered P (also called a projection operator) in our discussion of kernel PCA, where we found that for any kernel matrix, PKP gives the centered form (the inner product matrix between centered points). Here we will explore centering a little further. Let X be the matrix whose rows are the data points $\mathbf{x} \in \mathcal{R}^d$, $X \in M_{md}$. Since $\mathbf{e}\mathbf{e}' \in M_m$ is the matrix of all ones, PX subtracts the mean vector from each row $\mathbf{x}$ in X (hence the name "centering"), and in addition, $P\mathbf{e} = 0$. In fact $\mathbf{e}$ is the only eigenvector (up to scaling) with eigenvalue zero, for suppose $P\mathbf{f} = 0$ 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 has rank $m - 1$, and P 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 $\mathbf{x}_i,\ \mathbf{x}_j \in \mathcal{R}^d$, where $\|\cdot\|$ is the Euclidean norm. Notice that the elements are squared distances, despite the name. Now P can 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,

$$P[\mathbf{x}_i \cdot \mathbf{x}_j]P = PXX'P = (PX)(PX)' = [(\mathbf{x}_i - \boldsymbol{\mu}) \cdot (\mathbf{x}_j - \boldsymbol{\mu})].$$

In addition, using this result together with $P\mathbf{e} = 0$, we have that:

$$\begin{aligned} P[\|\mathbf{x}_i - \mathbf{x}_j\|^2]P &= P\ [\|\mathbf{x}_i\|^2 e_i e_j + \|\mathbf{x}_j\|^2 e_i e_j - 2\mathbf{x}_i \cdot \mathbf{x}_j]\ P \\ &= -2P\mathbf{x}_i \cdot \mathbf{x}_j P = -2[(\mathbf{x}_i - \boldsymbol{\mu}) \cdot (\mathbf{x}_j - \boldsymbol{\mu})]. \end{aligned}$$

For the following theorem, the earliest form of which is due to Schoenberg [78], we first note that, for any $A \in M_m$, and letting $\mathcal{I} \equiv \frac{1}{m}\mathbf{e}\mathbf{e}'$,

$$PAP = \{(\mathbf{1} - \mathcal{I})A(\mathbf{1} - \mathcal{I})\}_{ij} = A_{ij} - A_{ij}^R - A_{ij}^C + A_{ij}^{RC}, \qquad (4.9)$$

where $A^C \equiv A\mathcal{I}$ is the matrix A with each column replaced by the column mean, $A^R \equiv \mathcal{I}A$ is A with each row replaced by the row mean, and $A^{RC} \equiv \mathcal{I}A\mathcal{I}$ is A with every element replaced by the mean of all the elements. Also we define a set of Gram vectors $\mathbf{x}_i$ for a Gram matrix G to be any vectors for which $G_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j$.

Theorem 4.1. Consider the class of symmetric matrices $A \in S_n$ such that $A_{ij} \geq 0$ and $A_{ii} = 0\ \forall i,j$. Then $\bar{A} \equiv -PAP$ 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 $\mathbf{x}_i \in \mathcal{R}^m$, $i = 1, \ldots, m$ 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 Equation (4.9),

$$\bar{A}_{ij} = (-PAP)_{ij} = \mathbf{x}_i \cdot \mathbf{x}_j = -A_{ij} + A^R_{ij} + A^C_{ij} - A^{RC}_{ij}, \tag{4.10}$$

so

$$\begin{aligned} 2(\mathbf{y}_i - \mathbf{y}_j)^2 &\equiv (\mathbf{x}_i - \mathbf{x}_j)^2 \\ &= A^R_{ii} + A^C_{ii} - A^{RC}_{ii} + \{i \to j\} - 2(-A_{ij} + A^R_{ij} + A^C_{ij} - A^{RC}_{ij}) \\ &= 2A_{ij}, \end{aligned}$$

using $A_{ii} = 0$, $A^R_{ij} = A^R_{jj}$, and $A^C_{ij} = A^C_{ii}$. Thus A is a distance matrix with embedding vectors $\mathbf{y}_i$. Now suppose that $A \in S_n$ 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 sums to zero, we have $\bar{A} = -(PAP) = 2(PY)(PY)'$, 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 [54], 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. □

4.2.1 General Centering

Is P 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 [10]. 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) [10, 79] (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.

4.2.2 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 $\sqrt{\lambda_\alpha}\tilde{\mathbf{e}}_\alpha^{(i)}$, $\alpha = 1, \ldots, p$, 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

$$\sum_{i=1}^{m} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|^2 = \frac{1}{2}\left(\sum_{a=1}^{r} \lambda_a - \sum_{a=1}^{p} \lambda_a\right) = \sum_{a=p+1}^{r} \lambda_a. \tag{4.11}$$

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, to $\bar{A}$, which is $\hat{A} \equiv \sum_{\alpha:\lambda_\alpha > 0} \lambda_\alpha \mathbf{e}^{(\alpha)} \mathbf{e}^{(\alpha)\prime}$ (see the Appendix). 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 [27]; 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.

We end this section with two remarks. First, for classical metric MDS, we are provided with a Euclidean distance matrix and wish to find the lowest dimensional representation of the data points the reproduces the distance matrix. If we had been given the coordinates of the original data, we could perform the same task (find the subspace in which the data lies) using PCA, which would give the same solution. Second, the above analysis shows that one can easily map from the distance matrix to the centered dot product matrix, and vice versa, using projection matrices. This suggests that one might apply the kernel trick to algorithms that are distance-based by first mapping the distances to dot products and then replacing the dot products by kernels. This is exactly the trick used by kernel PCA.

4.2.3 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) [29]. 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. [9] and explained in more detail in Platt [70], 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:

$$D = \begin{bmatrix} E & \mathbf{f} \\ \mathbf{f}' & g \end{bmatrix}, \quad K \equiv -\frac{1}{2} P^c D P'^c = \begin{bmatrix} A & \mathbf{b} \\ \mathbf{b}' & c \end{bmatrix}. \tag{4.12}$$

Then from Equation (4.6) we see that the Nyström method gives the approximate column eigenvectors for the extended system as:

$$\begin{bmatrix} U \\ \mathbf{b}' U \Lambda^{-1} \end{bmatrix}. \tag{4.13}$$

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 [70] 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 $i \le q$ and $c_{q+1} = 0$: in that case, since

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

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 $b_i = (-\frac{1}{2} P^c D P'^c)_{q+1,i}$, $i = 1, \ldots, q$, we find that:

$$b_k = -\frac{1}{2}\left[D_{q+1,k} - \frac{1}{q} \sum_{j=1}^{q} D_{q+1,j} e_k - \frac{1}{q} \sum_{i=1}^{q} D_{ik} + \frac{1}{q^2} \left(\sum_{i,j=1}^{q} D_{ij} \right) e_k \right]. \tag{4.14}$$

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[4]

[4] 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.

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 De Silva and Tenenbaum [29]. 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.

4.3 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 3.2.1, again unbeknownst to you, your data lies on a curve, but in contrast to Section 3.2.1, 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) [87], 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 [87] gives an example of a five-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 [42]; 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 [87] 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 $\mathcal{I} : \mathcal{R}^d \to \mathcal{R}^{d'}$ 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 [29]. Landmark Isomap simply employs landmark MDS [29] 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.

4.4 Locally Linear Embedding

Locally linear embedding (LLE) [75, 76] 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 neighbors 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 neighbors, and then construct the $\mathbf{y}_i$ so that they can be expressed as the same linear combination of their corresponding neighbors (the latter also indexed by $\mathcal{N}(i)$). To simplify the discussion let's assume that the number of the neighbors 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. 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 Ws, 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 [75] 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 4.5.1). 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:

$$F = \frac{1}{2}\sum_i \|\mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j\|^2 - \frac{1}{2}\sum_{\alpha\beta}\lambda_{\alpha\beta}\left(\sum_i \frac{1}{m}Y_{i\alpha}Y_{i\beta} - \delta_{\alpha\beta}\right), \tag{4.15}$$

where for convenience we treat the Ws 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[5] the matrix equation:

$$(\mathbf{1} - W)'(\mathbf{1} - W)Y = \frac{1}{m}Y\Lambda. \tag{4.16}$$

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

[5] Again, we are free to choose any conditions on the $\lambda_{\alpha\beta}$ providing a solution can be found; see Burges [16] for background on Lagrange multipliers.

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 Equation (4.16).

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.

4.5 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 data set 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 neighbors, for N sufficiently large. The normalized Laplacian matrix for any weighted, undirected graph is defined [21] by $\mathcal{L} \equiv D^{-1/2} L D^{-1/2}$, where $L_{ij} \equiv D_{ij} - W_{ij}$ and $D_{ij} \equiv \delta_{ij}(\sum_k W_{ik})$. 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 normalized 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 [21]: 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 if 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.

4.5.1 Laplacian Eigenmaps

The Laplacian eigenmaps algorithm [8] 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:

$$\sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij} = 2 \sum_{i,j,a} (\mathbf{y}_i)_a (\mathbf{y}_j)_a (D_{ii}\delta_{ij} - W_{ij}) = 2\mathrm{Tr}(Y'LY). \tag{4.17}$$

In order to ensure that the target space has dimension d' (minimizing Equation (4.17) 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 $L\mathbf{y} = \lambda D\mathbf{y}$ [8]. 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 $\lambda_{\alpha\beta} \equiv \lambda_\beta \delta_{\alpha\beta}$, 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 unlike MDS, Laplacian eigenmaps must use some criteria other than the sizes of the eigenvalues for deciding 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.

4.5.2 Spectral Clustering

Although spectral clustering is a clustering method, it is very closely related to dimension reduction. In fact, since clusters may be viewed as large-scale structural features of the data, any dimension 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 [82, 66, 68]. 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 [82] 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 min-cut problem is given by:

$$\min_{\mathbf{y}} \frac{\mathbf{y}' L \mathbf{y}}{\mathbf{y}' D \mathbf{y}} \text{ such that } y_i \in \{1, -b\} \text{ and } \mathbf{y}' D \mathbf{e} = 0 \tag{4.18}$$

[82] 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 Equation (10) in Shi and Malik [82]). 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 dimension 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 [82]. Meila and Shi [66] point out 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 analyze the thresholding used in spectral clustering, and a method for learning the weights W_{ij} based on training data with known clusters [66]. The dimension 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 [68]. Zhang and Jordan [100] present a more general approach to the relaxation (of binary values denoting cluster membership, to reals) and rounding (mapping the solution back to binary indicators of cluster membership) problems shared by graph-cut

approaches. Their view of the relaxation problem builds on an observation of Rahimi and Recht [72], namely that the normalized cut problem of Shi and Malik [82] can be interpreted as searching for a hyperplanar gap in the empirical distribution. Zhang and Jordan [100] show that this idea can be naturally extended to handle multiway spectral clustering, and they suggest a Procrustes analysis to solve the rounding problem.

4.6 Pulling the Threads Together

At this point the reader is probably struck by how similar the mathematics underlying all of these approaches is. We've used essentially the same Lagrange multiplier trick to enforce constraints three times; all of the methods in this section (and most in this review) rely heavily on eigendecompositions. 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. [9]). Isomap is often called a "global" dimension 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 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 [8]; and Ham et al. [46] show

how Laplacian eigenmaps, LLE, and Isomap can be viewed as variants of kernel PCA. Platt [70] 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 Equation (4.11). 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 $X'X$ [54], 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 [98] 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., PKP 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 [98] 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 4.1.3), and exactly the same can be done with MDS.

5

Pointers and Conclusions

5.1 Pointers to Further Reading

Dimension reduction is a very active field of research. While this review has focused on the foundations underlying the classical (and related) techniques, here we give pointers to some other well-known methods (in approximate order of appearance); the list below is incomplete, but we hope useful nevertheless. Again we use $\mathcal{H}$ ($\mathcal{L}$) to denote the high and low dimensional space with elements $\mathbf{x} \in \mathcal{R}^d$ and $\mathbf{y} \in \mathcal{R}^{d'}$ ($d' \ll d$), respectively.

In the *Method of Principal Curves*, 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 [48]; the idea has since been extended by applying various regularization schemes (including kernel-based), and to manifolds of higher dimension [80].

The *Information Bottleneck* method [90] may also be viewed as a dimension reduction method. Information Bottleneck aims to distill the information in a random (predictor) variable X that is needed to describe a (response) variable Y, using a model variable Z, by maximizing the (weighted) difference in mutual information between Y and Z, and between X and Z. The use of information theory is intuitively appealing, although the method requires that the joint density $p(X,Y)$ be estimated.

Neighborhood Components Analysis (NCA) [40] applies a global linear transformation $\mathbf{y}_i = A\mathbf{x}_i,\ A \in M_{d'd}$ to the data such that, in the transformed space, k-nearest neighbor performs well. The probability that point $\mathbf{x}_i$ belongs to class k, p_{ik}, is computed using a simple softmax distribution in the transformed space: $p_{ik} = \frac{\sum_{j \in S_k} \exp(-\|A\mathbf{x}_i - A\mathbf{x}_j\|^2)}{\sum_j \exp(-\|A\mathbf{x}_i - A\mathbf{x}_j\|^2)}$, where S_k is the set of indices of points in class k. Given this, NCA applies gradient descent to maximize the expected number of correctly classified points. By simply choosing $d' < d$ in the definition of A, NCA also performs supervised linear dimension reduction. The reduction can significantly speed up the KNN computation, both because the dimension itself is smaller and also because data partitioning schemes such as kd-trees work better in lower dimensions.

Maximum Variance Unfolding (MVU) [96, 86, 77] preserves distances $\|\mathbf{x}_i - \mathbf{x}_j\| = \|\mathbf{y}_i - \mathbf{y}_j\|$, as does Isomap, but it differs from Isomap in that it does so only locally: only distances between neighboring points are so constrained, as opposed to Isomap's striving to preserve geodesic distances between points that may or may not be close. As with Isomap, the idea is that the folding of the manifold upon itself in $\mathcal{H}$ is information that can be usefully discarded in forming the low dimensional representation, but MVU additionally allows local rotations and translations, which allows it to choose from a still larger class of mappings. Let $I_{ij}(k)$ be the indicator variable denoting that $\mathbf{x}_i$ and $\mathbf{x}_j$ are k-nearest neighbors, or that there exists a point $\mathbf{x}_m$ such that $\mathbf{x}_i$ and $\mathbf{x}_m$ are k-nearest neighbors and that $\mathbf{x}_j$ and $\mathbf{x}_m$ are k-nearest neighbors.[1] MVU maximizes $\sum_i \|\mathbf{y}_i\|^2$ subject to constraints $\|\mathbf{x}_i - \mathbf{x}_j\| = \|\mathbf{y}_i - \mathbf{y}_j\|\ \forall i,j : I_{ij}(k) = 1$ and such that the mapped data

[1] This statement is actually imprecise: k-nearest neighbor is not necessarily a symmetric relation. Let $S_i(k)$ be the set of indices of points that lie in the k-nearest neighbors of $\mathbf{x}_i$. One can define a symmetric $I_{ij}(k)$ by setting $I_{ij}(k) = 1 : \{i,\ j : \{i \in S_j(k) \wedge j \in S_i(k)\}\ \vee\ \{\exists m : i \in S_m(k) \wedge j \in S_m(k)\}\}$, 0 otherwise, or a (less sparse but still symmetric) $I_{ij}(k)$ by setting $I_{ij}(k) = 1 : \{i,\ j : \{i \in S_j(k) \vee j \in S_i(k)\}\ \vee\ \{\exists m : i \in S_m(k) \wedge j \in S_m(k)\}\}$, 0 otherwise.

is centered ($\sum_i \mathbf{y}_i = 0$). This is a computationally tractable approximation for minimizing the rank of the Gram matrix of the samples in $\mathcal{L}$ (i.e., to minimize the dimension in $\mathcal{L}$), and it's very striking that it can be rewritten as a convex semidefinite programming problem.

Restricted Boltzmann Autoencoders [53] use a stack of Restricted Boltzmann Machines (RBMs) [1, 50] to create an autoencoder — a neural net that learns to minimize the error between its inputs and outputs. Autoencoders have a central hidden layer whose activations form the low dimensional representation of the data (and so the number of units in that layer is equal to $d' \ll d$, where d is the number of input (and output) units). Fully connected nets trained using gradient descent tend to be poor autoencoders due to the weight vectors getting stuck in local minima: choosing good initial values for the weights (and thresholds) is key, a task which is accomplished by using individually trained RBMs, and stacking them by treating the outputs of the so-far-trained RBMs as inputs for the next layer RBM. Deep Belief Nets [51] have the fascinating property of being able to be run in a generative mode, which gives the researcher a direct window as to what the model has learned; if a DBN (which is also composed of a stack of RBMs) reconstructs convincing "inputs" in generative mode, it's good evidence that the features that the model has learned are informative for the task at hand, so it's perhaps not surprising that autoencoders built in this manner perform so well.

Stochastic Neighbor Embedding (SNE) [52] places a spherical Gaussian distribution $\mathcal{N}_{H,\mathbf{x}_i,\sigma_i}(\|\mathbf{x}-\mathbf{x}_i\|^2)$ over each data point $\mathbf{x}_i \in \mathcal{H} = \mathcal{R}^d$ (with mean $\mathbf{x}_i$ and variance σ_i^2) and thus models the conditional probability that point $\mathbf{x}_i$ would pick any other point $\mathbf{x}_j$ as its neighbor (so that if for some j,k, $\|\mathbf{x}_i-\mathbf{x}_j\| < \|\mathbf{x}_i-\mathbf{x}_k\|$, then $\mathbf{x}_i$ assigns higher probability to $\mathbf{x}_j$ than to $\mathbf{x}_k$). Similarly, a Gaussian $\mathcal{N}_{L,\mathbf{y}_i,\sigma_L}(\|\mathbf{y}-\mathbf{y}_i\|^2$ is attached to each point $\mathbf{y}_i \in \mathcal{L} = \mathcal{R}^{d'}$, $d' \ll d$. Note that while each Gaussian $\mathcal{N}_H(\mathbf{x}_i)$ in $\mathcal{H}$ has a variance σ_i that is learned, the Gaussians in $\mathcal{L}$ are all assigned fixed, equal variances σ_L. For some index i, for the low dimensional representation $\mathbf{y}_i$, one can imagine moving the points $\mathbf{y}_{j\neq i}$ around so as to make $p_{L,\mathbf{y}_i,\sigma_L}(\|\mathbf{y}_i-\mathbf{y}_k\|^2)$ as close as possible (in some sense) to $p_{H,\mathbf{x}_i,\sigma_i}(\|\mathbf{x}_i-\mathbf{x}_k\|^2)$. SNE uses the Kullback–Leilbler divergence and minimizes the sum over all such pairs of points, using gradient descent (for which the variables are the positions of the points $\mathbf{y}_i$; the σ_i are set using an entropy-based criterion that can be thought of as setting the effective number of nearest neighbors). The optimization problem is non-convex, and further tricks (momentum, and a form of simulated annealing) are employed to avoid local minima; see Hinton and Roweis [52].

SNE has recently been extended to *t-distributed SNE* [93]. t-SNE differs from SNE in that it uses a simpler version of the cost function, and it uses Student t-distributions rather than Gaussians in $\mathcal{L}$, to overcome an "overcrowding" problem that is observed in SNE. Matlab code for t-SNE is available from van Der Maaten [91] and we used it to perform t-SNE on data for the first 500 patients in our KDD Cup data set. The results are shown in Figure 5.1, where the negatives are shown in light gray and the positives in red (with larger font). Note that t-SNE finds an interesting double cluster structure, although this does not (by eye) appear to be predictive as to class. Finally, we note that t-SNE has also been extended to a parametric form (which can be

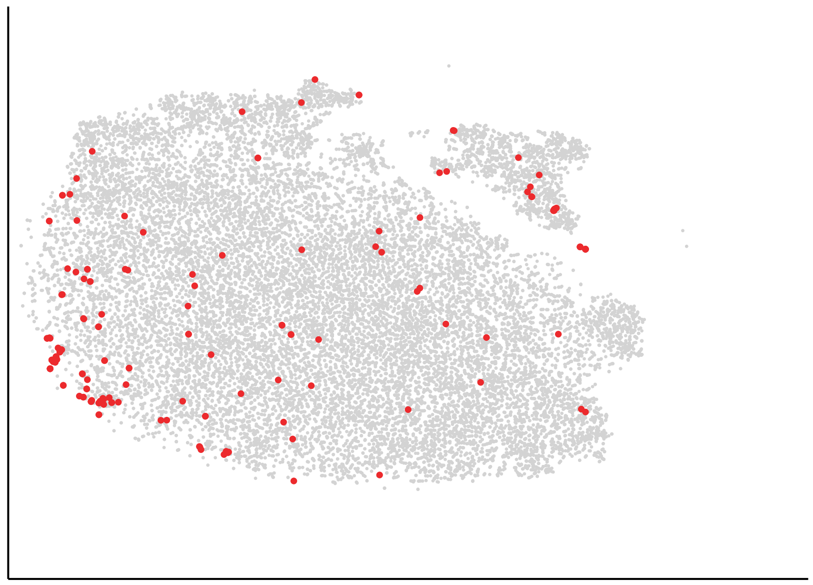

Fig. 5.1 One thousand iterations of t-SNE applied to the KDD Cup data. The points with positive label are shown in larger font in red.

used to easily find the mapping for new test points) by combining it with ideas from autoencoders: see van der Maaten [92].

Finally, we note that this monograph is far from being the first to review methods for dimension reduction. For reviews of spectral methods, see Saul et al. [77]; von Luxurg [95]; and Zhang and Jordan [100]. For a comparison of many different methods, including methods to estimate the intrinsic dimension, see Lee and Verleysen [62] and more recently, van der Maaten [94].

5.2 Conclusions

The student who has read this far may be asking him or herself the following question: "A lot of work has already been done on dimension reduction — what interesting research directions are left?". Since I'd like to try to leave the reader excited about the future of this subject, let me offer a few observations on this question here. One key ingredient that is missing is a solid theoretical foundation, akin, for example, to the learning theory we now have for the classification task. The first step in developing such a theory is to be clear about what the problem is that we wish to solve. For example, some of the methods described above are very intuitive, and effective at discarding variance that is deemed not useful (the shape of the manifold in $\mathcal{H}$ upon which the data approximately lie) but one can imagine situations where that information may instead be important for the task at hand. Visualization itself is usually not the end of the story: a useful visualization should be actionable. It would be advantageous to have precise objective functions which reflect that end utility. As seen above, conditional, or supervised, dimension reduction — the presence of labels — can completely change the story. If the goal is, for example, to map the data to a lower dimensional space where standard classifiers (or regressors, or rankers) can do a better job, one might extend the parameterization to include the dimension reduction mapping itself. NCA is a step in this direction, but it is a global, linear method, for which the target dimension is an input; methods that relax these restrictions (for conditional dimension reduction) would be interesting directions for research. The notion of scale-dependent dimension has not yet been investigated in the machine learning community, to the best of my knowledge. It may be that a level of noise that completely defeats standard classifiers could be overcome if the system could home in on the length scales at which the signal resides. Dimension reduction may also be viewed as a form of regularization when it is used in conjunction with supervised learning: one very common approach to regularization in machine learning is to form the objective function by adding a regularization term to a loss function, but that method, although simple, is ultimately a very coarse approach: one would like much more control over modeling the noise, and new approaches to regularization, including dimension reduction, may prove to be fertile grounds for investigation. Finally, in the longer term, the ease with which humans can extract low dimensional, very informative data from their high dimensional visual and aural inputs is an inspiration for those who would like to build machines that can do the same, for a wider variety of high dimensional data; we are clearly far from this goal today.

A

Appendix: The Nearest Positive Semidefinite Matrix

The following result is generally useful (and was used in Section 4.2.2):

Theorem A.1. Let $B \in S_m^+$ be the closest positive semidefinite matrix, in Frobenius norm, to a real square matrix $A \in M_m$. Let S be the symmetric part of A and let $S = U\Lambda U^T$ be its eigendecomposition, so that U is the orthogonal matrix of column eigenvectors and Λ is the diagonal matrix of eigenvalues of S. Let Λ' be Λ with all negative eigenvalues replaced by zero. Then $B = U\Lambda' U^T$.

The problem is equivalent to:

$$\text{Minimize } \|A - B\|_F^2 \text{ subject to } B \in S_m^+, \tag{A.1}$$

where subscript F denotes the Frobenius norm. This is a convex optimization problem with a strictly convex objective function [14], which therefore has a unique solution (that is, at the solution, both B and the value of the objective function are unique). The problem can be solved as follows.

Proof. Our proof follows Higham [49]. Split A into its symmetric and antisymmetric parts S and T:

$$A = \frac{1}{2}(A + A^T) + \frac{1}{2}(A - A^T) \equiv S + T.$$

The Frobenius norm has an inner product defined: for $C, D \in M_{mn}$, $\langle C, D\rangle = Tr(C^T D) = Tr(CD^T)$. Symmetric and antisymmetric matrices are thus orthogonal in Frobenius norm:

$$Tr(T^T S) = \sum_{ij} T_{ji} S_{ji} = -\sum_{ij} T_{ij} S_{ij} = -Tr(T^T S) = 0,$$

hence $\|A - B\|_F^2 = \|S - B\|_F^2 + \|T\|_F^2$ (since B is symmetric by assumption) and so we only have to consider the minimization over symmetric matrices $S \in S_m$. Now the Frobenius norm is unitarily invariant: for real A and orthogonal U,

$$\begin{aligned}\|UA\|_F^2 &= \sum_{ijkm} U_{ik} A_{kj} U_{im} A_{mj} = \sum_{ijkm} U_{ik} U_{mi}^T A_{kj} A_{mj} \\ &= \sum_{jkm} \delta_{km} A_{kj} A_{mj} = \sum_{jk} A_{kj} A_{kj} = \|A\|_F^2.\end{aligned}$$

Replace S by its eigenvalue decomposition, that is, write $S = U\Lambda U^T$, with U orthogonal. Then,

$$\|S - B\|^2 = \|U\Lambda U^T - B\|^2 = \|\Lambda - U^T B U\|^2 \equiv \|\Lambda - C\|^2.$$

Note that $B \in S_m^+$ implies that $C \in S_m^+$ (since, for arbitrary $\mathbf{z} \in \mathcal{R}^m$, $z^T C z = (Uz)^T B (Uz) \geq 0$), and so $C_{ii} \geq 0$ for $i = 1, \ldots, m$. Let λ_i, $i = 1, \ldots, m$ be the eigenvalues of S. Then,

$$\begin{aligned}\|S - B\|_F^2 = \|\Lambda - C\|_F^2 &= \sum_{i \neq j} C_{ij}^2 + \sum_i (\lambda_i - C_{ii})^2 \\ &\geq \sum_{i:\lambda_i < 0} (\lambda_i - C_{ii})^2 \geq \sum_{i:\lambda_i < 0} \lambda_i^2. \qquad \text{(A.2)}\end{aligned}$$

The last expression provides a lower bound that is independent of C, hence choosing any $C \in S_m^+$ for which the inequalities become equalities minimizes $\|S - B\|_F^2$ and hence minimizes $\|A - B\|_F^2$. By defining B^* as the eigendecomposition of S, but with negative eigenvalues replaced by zero, the inequalities in Equation (A.2) indeed become equalities, and so B^* solves Equation (A.1). □

Acknowledgments

We thank Michael Jordan, Dennis Cook, and the anonymous reviewers for their valuable comments and suggestions. Thanks also to Lawrence Saul, Bernhard Schölkopf, Jay Stokes, and Mike Tipping for commenting on the earlier version of this work. Thanks also to the management team at Microsoft Research for their support of this effort.

References

[1] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines," *Cognitive Science*, vol. 9, 1985.

[2] 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*, vol. 25, pp. 821–837, 1964.

[3] S. Akaho, "A kernel method for canonical correlation analysis," in *Proceedings of the International Meeting of the Psychometric Society (IMPS2001)*, 2001.

[4] T. W. Anderson, *An Introduction to Multivariate Statistical Analysis.* Wiley Series in Probability and Statistics, 2003.

[5] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *Journal of Machine Learning Research*, vol. 3, pp. 1–48, 2002.

[6] P. F. Baldi and K. Hornik, "Learning in linear neural networks: A survey," *IEEE Transactions on Neural Networks*, vol. 6, pp. 837–858, July 1995.

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

[8] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, pp. 1373–1396, 2003.

[9] 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.

[10] C. Berg, J. P. R. Christensen, and P. Ressel, *Harmonic Analysis on Semigroups.* Springer-Verlag, 1984.

[11] C. M. Bishop, "Bayesian PCA," in *Advances in Neural Information Processing Systems 11*, MIT Press, 1999.

[12] I. Borg and P. Groenen, “Modern Multidimensional Scaling: Theory and Applications,” Springer, 1997.

[13] B. E. Boser, I. M. Guyon, and V. Vapnik, “A training algorithm for optimal margin classifiers,” in *Fifth Annual Workshop on Computational Learning Theory*, pp. 144–152, ACM, Pittsburgh, 1992.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[15] C. J. C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998.

[16] C. J. C. Burges, “Some notes on applied mathematics for machine learning,” in *Advanced Lectures on Machine Learning*, (O. Bousquet, U. von Luxburg, and G. Rätsch, eds.), pp. 21–40, Springer Lecture Notes in Artificial Intelligence, 2004.

[17] C. J. C. Burges, “Geometric methods for feature selection and dimensional reduction,” in *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers*, (L. Rokach and O. Maimon, eds.), Kluwer Academic, 2005.

[18] C. J. C. Burges, “Simplified support vector decision rules,” in *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 71–77, 1996.

[19] C. J. C. Burges, J. C. Platt, and S. Jana, “Extracting noise-robust features from audio,” in *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 1021–1024, IEEE Signal Processing Society, 2002.

[20] C. J. C. Burges, J. C. Platt, and S. Jana, “Distortion discriminant analysis for audio fingerprinting,” *IEEE Transactions on Speech and Audio Processing*, vol. 11, pp. 165–174, 2003.

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

[22] R. D. Cook, *Regression Graphics.* Wiley, 1998.

[23] R. D. Cook, “Model based sufficient dimension reduction for regression,” Isaac Newton Institute Lectures on Contemporary Frontiers in High-Dimensional Statistical Data Analysis, http://www.newton.ac.uk/webseminars/pg+ws/2008/sch/schw01/0108/cook/, 2008.

[24] R. D. Cook and L. Forzani, “Likelihood-based sufficient dimension reduction,” *Journal of the American Statistical Association*, vol. 104, pp. 197–208, 2009.

[25] R. D. Cook and H. Lee, “Dimension reduction in binary response regression,” *Journal of the American Statistical Association*, vol. 94, pp. 1187–1200, 1999.

[26] R. D. Cook and S. Weisberg, “Sliced inverse regression for dimension reduction: Comment,” *Journal of the American Statistical Association*, vol. 86, pp. 328–332, 1991.

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

[28] R. B. Darlington, “Factor analysis,” Technical report, Cornell University, http://comp9.psych.cornell.edu/Darlington/factor.htm, 1997.

[29] V. De Silva and J. B. Tenenbaum, “Global versus local methods in nonlinear dimensionality reduction,” *Advances in Neural Information Processing Systems*, vol. 15, pp. 705–712, MIT Press, 2002.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society B*, vol. 39, pp. 1–22, 1977.

[31] P. Diaconis and D. Freedman, “Asymptotics of graphical projection pursuit,” *Annals of Statistics*, vol. 12, pp. 793–815, 1984.

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

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

[34] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, “Spectral grouping using the Nyström method,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, 2004.

[35] J. H. Friedman and W. Stuetzle, “Projection pursuit regression,” *Journal of the American Statistical Association*, vol. 76, pp. 817–823, 1981.

[36] J. H. Friedman, W. Stuetzle, and A. Schroeder, “Projection Pursuit density estimation,” *Journal of the American Statistical Association*, vol. 79, pp. 599–608, 1984.

[37] J. H. Friedman and J. W. Tukey, “A projection pursuit algorithm for exploratory data analysis,” *IEEE Transactions on Computers*, vol. 23, pp. 881–890, 1974.

[38] K. Fukumizu, F. R. Bach, and M. I. Jordan, “Kernel dimension reduction in regression,” *Annals of Statistics*, vol. 37, pp. 1871–1905, 2009.

[39] A. Globerson and N. Tishby, “Sufficient dimensionality reduction,” *Journal of Machine Learning Research*, vol. 3, 2003.

[40] J. Goldberger, S. T. Roweis, G. E. Hinton, and R. Salakhutdinov, “Neighbourhood components analysis,” *Advances in Neural Information Processing Systems*, vol. 17, 2005.

[41] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Johns Hopkins, 3rd ed., 1996.

[42] M. Gondran and M. Minoux, *Graphs and Algorithms*. Wiley, 1984.

[43] P. Grassberger and I. Procaccia, “Measuring the strangeness of strange attractors,” *Physica*, vol. 9D, pp. 189–208, 1983.

[44] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, “Measuring statistical dependence with Hilbert-Schmidt norms,” in *Algorithmic Learning Theory, Springer Lecture Notes in Computer Science*, vol. 3734, pp. 63–77, 2005.

[45] G. Grimmet and D. Stirzaker, *Probability and Random Processes*. Oxford University Press, 3rd ed., 2001.

[46] 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.

[47] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, “Canonical correlation analysis: An overview with application to learning methods,” *Neural Computation*, vol. 12, pp. 2639–2664, 2004.

[48] T. J. Hastie and W. Stuetzle, "Principal curves," *Journal of the American Statistical Association*, vol. 84, pp. 502–516, 1989.

[49] N. J. Higham, "Computing the nearest symmetric positive semidefinite matrix," *Linear Algebra and its Applications*, vol. 103, pp. 103–118, 1988.

[50] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, pp. 1771–1800, 2002.

[51] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 7, pp. 1527–1554, 2006.

[52] G. E. Hinton and S. E. Roweis, "Stochastic neighbor embedding," *Advances in Neural Information Processing Systems*, vol. 14, pp. 833–840, 2002.

[53] G. E. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, pp. 504–507, 2007.

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

[55] H. Hotelling, "Relations between two sets of variates," *Biometrika*, vol. 28, pp. 321–377, 1936.

[56] T. Hsing and H. Ren, "An RKHS formulation of the inverse regression dimension-reduction problem," *Annals of Statistics*, vol. 37, pp. 726–755, 2009.

[57] P. J. Huber, "Projection pursuit," *Annals of Statistics*, vol. 13, pp. 435–475, 1985.

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

[59] T. L. Kelley, *Crossroads in the Mind of Man: A study of Differentiable Mental Abilities.* Stanford University Press, 1928.

[60] G. S. Kimeldorf and G. Wahba, "Some results on Tchebycheffian spline functions," *Journal of Mathematical Analysis and Applications*, vol. 33, pp. 82–95, 1971.

[61] Knowledge Discovery and Data Mining Cup, http://www.kddcup2008.com/index.html, 2008.

[62] J. Lee and M. Verleysen, *Nonlinear Dimensionality Reduction.* Springer, 2007.

[63] B. Li, H. Zha, and F. Chiaromonte, "Contour regression: A general approach to dimension reduction," *The Annals of Statistics*, vol. 33, pp. 1580–1616, 2005.

[64] C.-K. Li, "Sliced Inverse Regression for dimension reduction," *Journal of the American Statistical Association*, vol. 86, pp. 316–327, 1991.

[65] C.-K. Li, "On Principal Hessian Directions for data visualization and dimension reduction: Another application of Stein's lemma," *Journal of the American Statistical Association*, vol. 87, pp. 1025–1039, 1992.

[66] M. Meila and J. Shi, "Learning segmentation by random walks," *Advances in Neural Information Processing Systems*, vol. 12, pp. 873–879, 2000.

[67] S. Mika, B. Schölkopf, A. J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, "Kernel PCA and de-noising in feature spaces," *Advances in Neural Information Processing Systems*, vol. 11, MIT Press, 1999.

[68] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *Advances in Neural Information Processing Systems*, vol. 14, MIT Press, 2002.

[69] J. Nilsson, F. Sha, and M. I. Jordan, "Regression on manifolds using kernel dimension reduction," in *Proceedings of the 24th International Conference on Machine Learning*, 2007.

[70] J. C. Platt, "Fastmap, MetricMap, and landmark MDS are all Nyström algorithms," in *Proceedings of the 10th International Conference on Artificial Intelligence and Statistics*, 2005.

[71] 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 ed., 1992.

[72] A. Rahimi and B. Recht, "Clustering with normalized cuts is clustering with a hyperplane," Workshop on Statistical Learning in Computer Vision, Prague, 2004.

[73] S. M. Ross, *Introduction to Probability Models.* Academic Press, 10th ed., 2010.

[74] S. M. Ross and E. A. Peköz, *A Second Course in Probability.* www.Probability Bookstore.com, Boston, MA, 2007.

[75] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.

[76] L. K. Saul and S. T. Roweis, "Think globally, fit locally: Unsupervised learning of low dimensional manifolds," *Journal of Machine Learning Research*, vol. 4, pp. 119–155.

[77] L. K. Saul, K. Q. Weinberger, J. H. Ham, F. Sha, and D. D. Lee, "Spectral methods for dimensionality reduction," in *Semisupervised Learning*, (O. Chapelle, B. Schölkopf, and A. Zien, eds.), pp. 293–308, MIT Press, 2006.

[78] 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*, vol. 36, pp. 724–732, 1935.

[79] B. Schölkopf, "The kernel trick for distances," *Advances in Neural Information Processing Systems*, vol. 13, pp. 301–307, MIT Press, 2001.

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

[81] B. Schölkopf, A. Smola, and K.-R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, pp. 1299–1319, 1998.

[82] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905, 2000.

[83] C. E. Spearman, "'General intelligence' objectively determined and measured," *American Journal of Psychology*, vol. 5, pp. 201–293, 1904.

[84] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," *Journal of Machine Learning Research*, vol. 37, pp. 726–755, 2001.

[85] C. J. Stone, "Optimal global rates of convergence for nonparametric regression," *Annals of Statistics*, vol. 10, pp. 1040–1053, 1982.

[86] J. Sun, S. Boyd, L. Xiao, and P. Diaconis, "The fastest mixing Markov process on a graph and a connection to a maximum variance unfolding problem,"

Society for Industrial and Applied Mathematics (SIAM) Review, vol. 48, pp. 681–699, 2006.

[87] J. B. Tenenbaum, “Mapping a manifold of perceptual observations,” *Advances in Neural Information Processing Systems*, vol. 10, MIT Press, 1998.

[88] M. E. Tipping and C. M. Bishop, “Mixtures of probabilistic principal component analyzers,” *Neural Computation*, vol. 11, pp. 443–482, 1999.

[89] M. E. Tipping and C. M. Bishop, “Probabilistic principal component analysis,” *Journal of the Royal Statistical Society*, vol. 61, p. 611, 1999.

[90] N. Tishby, F. C. Pereira, and W. Bialek, “The information bottleneck method,” in *Proceedings of the 37th Annual Allerton Conference on Communication, Control and Computing*, pp. 368–377, 1999.

[91] L. van der Maaten, “t-Distributed Stochastic Neighbor Embedding,” http://homepage.tudelft.nl/19j49/t-SNE.html.

[92] L. van der Maaten, “Learning a parametric embedding by preserving local structure,” in *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2009.

[93] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE,” *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[94] L. van der Maaten, E. Postma, and J. van den Herik, “Dimensionality reduction: a comparative review,” Tilburg University Technical Report TiCC-TR 2009–005, 2009.

[95] U. von Luxburg, “A tutorial on spectral clustering,” *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.

[96] K. Q. Weinberger and L. K. Saul, “An introduction to nonlinear dimensionality reduction by maximum variance unfolding,” in *Proceedings of the Twenty First National Conference on Artificial Intelligence (AAAI-06)*, pp. 1683–1686, 2006.

[97] S. Wilks, *Mathematical Statistics.* Wiley, 1962.

[98] C. K. I. Williams, “On a connection between kernel PCA and metric multidimensional scaling,” *Advances in Neural Information Processing Systems*, vol. 13, pp. 675–681, MIT Press, 2001.

[99] C. K. I. Williams and M. Seeger, “Using the Nyström method to speed up kernel machines,” *Advances in Neural Information Processing Systems*, vol. 13, pp. 682–688, MIT Press, 2001.

[100] Z. Zhang and M. I. Jordan, “Multiway spectral clustering: A margin-based perspective,” *Statistical Science*, vol. 23, pp. 383–403, 2008.