Representing complex data using localized principal components with application to astronomical data

Jochen Einbeck[1], Ludger Evers[2], and Coryn Bailer-Jones[3]

[1] Department of Mathematical Sciences, Durham University, Science Laboratories, South Road, Durham DH1 3LE, United Kingdom, jochen.einbeck@durham.ac.uk
[2] Department of Mathematics, University of Bristol, University Walk, BS8 1TW, United Kingdom, l.evers@bris.ac.uk
[3] Max-Planck-Institut für Astronomie, Königstuhl 17, 69117 Heidelberg, Germany, calj@mpia-hd.mpg.de

in *Principal Manifolds for Data Visualization and Dimension Reduction*,
A. Gorban, B. Kegl, D. Wunsch, and A. Zinovyev (eds),
Lecture Notes in Computational Science and Engineering,
Springer, 2007, pp. 180–204

Summary. Often the relation between the variables constituting a multivariate data space might be characterized by one or more of the terms: "nonlinear", "branched", "disconnected", "bended", "curved", "heterogeneous", or, more general, "complex". In these cases, simple principal component analysis (PCA) as a tool for dimension reduction can fail badly. Of the many alternative approaches proposed so far, local approximations of PCA are among the most promising. This paper will give a short review of localized versions of PCA, focusing on local principal curves and local partitioning algorithms. Furthermore we discuss projections other than the local principal components. When performing local dimension reduction for regression or classification problems it is important to focus not only on the manifold structure of the covariates, but also on the response variable(s). Local principal components only achieve the former, whereas localized regression approaches concentrate on the latter. Local projection directions derived from the partial least squares (PLS) algorithm offer an interesting trade-off between these two objectives.

We apply these methods to several real data sets. In particular, we consider simulated astrophysical data from the future Galactic survey mission Gaia.

1 Introduction

Principal component analysis is a well established tool for dimension reduction. For data $\boldsymbol{X} = (\boldsymbol{x}_1', \ldots, \boldsymbol{x}_n')'$ with $\boldsymbol{x}_i \in \mathbb{R}^p$ the principal components provide a sequence of best linear approximations to it. Let $\boldsymbol{\Sigma}$ be the empirical covariance matrix of $\boldsymbol{X}$, then the principal components are given by the eigen decomposition

$$\boldsymbol{\Sigma} = \boldsymbol{\Gamma}\boldsymbol{\Lambda}\boldsymbol{\Gamma}' \tag{1}$$

where $\boldsymbol{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_p)$ is a diagonal matrix containing the ordered eigenvalues of $\boldsymbol{\Sigma}$, with $\lambda_1 \geq \ldots \geq \lambda_p$, and $\boldsymbol{\Gamma}$ is an orthogonal matrix. The columns of $\boldsymbol{\Gamma} = (\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_p)$ are the eigenvectors of $\boldsymbol{\Sigma}$ and are called *principal component loadings*. The first loading $\boldsymbol{\gamma}_1$ maximizes the variance of the *scores* $\boldsymbol{X\gamma}$ over all $\boldsymbol{\gamma} \in \mathbb{R}^p$ with $\|\boldsymbol{\gamma}\| = 1$, the second loading $\boldsymbol{\gamma}_2$ maximizes the variance of $\boldsymbol{X\gamma}$ over all $\boldsymbol{\gamma} \in \mathbb{R}^p$ with $\|\boldsymbol{\gamma}\| = 1$ which are orthogonal to $\boldsymbol{\gamma}_1$, and so on.

As an example, we consider a speed-flow diagram recorded on the freeway 4-W in Contra Costa County, California (figure 1(a)).[4] The displayed points correspond to the average speed and flow over 5 minute intervals. The data were collected on 10th December 2006; this was a Sunday, so the traffic flow was quite low, not exceeding 1300 vehicles per hour. Most cars went at a speed close to the speed limit. One can easily imagine a first principal component line $\boldsymbol{g}(\eta) = \bar{\boldsymbol{x}} + \eta \cdot \boldsymbol{\gamma}_1$ (with overall mean $\bar{\boldsymbol{x}}$) fitted through the data cloud, capturing the main part (here: 99.54%) of the variance in this data set (figure 1(a)). Clearly, the projection indices $\eta_i = (\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T\boldsymbol{\gamma}_1$ of the data projected onto this line are good indicators for the positions of the data points within the data cloud.

However, the application of principal component analysis postulates implicitly some form of linearity. More precisely, one assumes that the data cloud is directed, and that the data points can be well approximated by their projections onto the line $\boldsymbol{g}$ corresponding to the first principal component (or in general the affine hyperplane corresponding to the first d principal components). This implicit assumption of linear PCA can already be violated in very simple situations. As an example, consider the speed-flow diagram recorded three days earlier with the same detector at the same location. This was a Thursday and traffic was busy, leading to occasional congestion. As a consequence, cars often had to reduce their speed at higher flows, resulting in the data cloud shown in figure 1(b). This kind of pattern has frequently been reported in the transportation science literature, see e.g. [27].

These data are somewhat half-moon shaped and it does not seem to make much sense to speak of some principal direction for this data. The projection indices onto any straight line like the first principal component (figure 1(b)) would be uninformative with respect to the position of the data within this bent data cloud. This will certainly get worse if the the data cloud is strongly

[4] The data are taken from the database PemS [45].

twisted, if it has crossings, if it consists of several branches, if the curvature permanently changes, or if there are several disconnected clouds, and so on.

Solutions to problems of this kind are readily available, and fall into two major categories: *Nonlinear principal component analysis* (hereafter: NLPCA) and *principal curves* (or their multivariate extension, *principal manifolds*). The dividing lines between these two approaches are often rather fuzzy, but the following vague rule can be used to distinguish between them. Principal curves is a nonparametric extension of linear PCA, while NLPCA is a parametric (or semi-parametric), but nonlinear version of PCA. Most, but not all algorithms belonging to either of the categories accomplish their task in two steps:

Projection Define a dimension reducing transformation $\boldsymbol{f}(\boldsymbol{x}) : \mathbb{R}^p \to \mathbb{R}^d$
Reconstruction Find a mapping back to the data space $\boldsymbol{g}(\boldsymbol{\eta}) : \mathbb{R}^d \to \mathbb{R}^p$

Here, the vector $\boldsymbol{\eta}$ represents a $d-$ dimensional parameterization of the $p-$ dimensional data space. Summarizing, the reconstructed curve is given by $\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}))$, which is mostly found by (implicitly or explicitly) minimizing the reconstruction error

$$\mathbb{E}\|\boldsymbol{x} - \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}))\|^2, \tag{2}$$

or its empirical counterpart

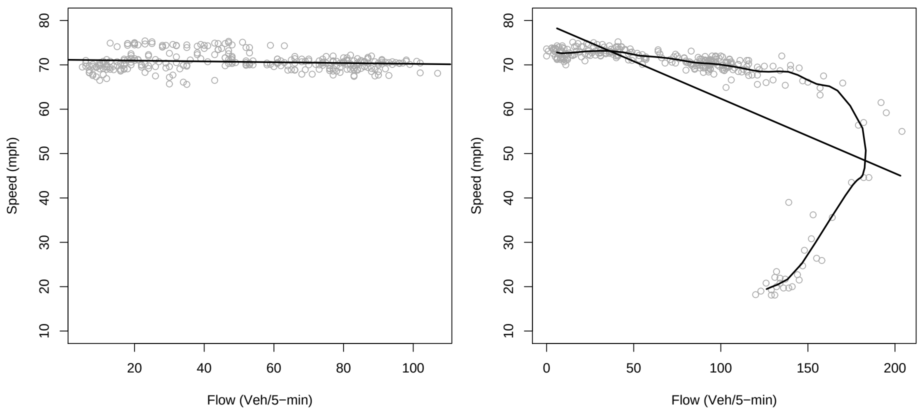
$$\sum_{i=1}^{n} \|\boldsymbol{x}_i - \boldsymbol{g}(\boldsymbol{f}(\boldsymbol{x}_i))\|^2. \tag{3}$$

PCA can be seen as a special case of the above algorithm. It follows directly from the extremal properties of the eigenvectors that the first d principal components define the *linear*, d-dimensional functions minimizing equations (2) or (3). In the special case of PCA $\boldsymbol{f}(\boldsymbol{x}) = (\gamma_1 \cdots \gamma_d)' (\boldsymbol{x} - \boldsymbol{\mu})$, $\boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\mu} + (\gamma_1 \cdots \gamma_d)\, \boldsymbol{\eta}$, and $(\boldsymbol{g} \circ \boldsymbol{f})(\boldsymbol{x}) = \boldsymbol{\mu} + (\gamma_1 \cdots \gamma_d)\, (\gamma_1 \cdots \gamma_d)' (\boldsymbol{x} - \boldsymbol{\mu})$, where $\boldsymbol{\mu} = \mathbb{E}(\boldsymbol{x})$. If all principal components are used (i.e. $d = p$) then it is easy to see that $\boldsymbol{g} \circ \boldsymbol{f}$ is the identity and the reconstruction error vanishes.

A further difference between principal curves and NLPCA is that the projection $\boldsymbol{f}$ in NLPCA has to be continuous whereas the projection in principal curves can be discontinuous [37]. An important concept for NLPCA as well as for principal curves is that of self-consistency, implying that the reconstructed curve is the mean over all data with equal projection, i.e $\boldsymbol{g}(\boldsymbol{\eta}) = \mathbb{E}(\boldsymbol{x}|\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{\eta})$. This idea is the cornerstone of the original principal curve approach by Hastie and Stuetzle [28] (hereafter: HS) as well as for the NLPCA by Bolton et al. [5], who also consider all possible combinations of linear and nonlinear projection and reconstruction. A comparison of NLPCA and HS principal curves can be found in [39].

A special type of principal curves, a so-called *local principal curve* ([16], see Section 2.2), is shown for the second of the two data sets in figure 1.

This article is concerned with extensions of PCA based on *localization.* In Section 2 we review such methods and also present a new approach based on



(a) 10th December 2006 (b) 7th December 2006

Fig. 1. Speed flow diagrams for Freeway 4-W on 10th (a) and 7th (b) of December 2006, with principal component lines (straight lines) and a principal curve through the latter data set (smooth curve).

local partitioning. In Section 3 we discuss how principal component regression can be localized. Furthermore we discuss projection directions other than the principal components. Using the projection directions used in the partial least squares (PLS) algorithm allows for taking both the inherent structure of the covariates and the response variable into account. In Section 4 we apply these methods to simulated data produced in order to develop classification and parametrization methods for the upcoming Gaia astronomical survey mission. This survey will obtain spectroscopic data (i.e. high dimensional vectors) on over one billion (10^9) stars, from which we wish to estimate intrinsic astrophysical parameters (temperature, abundance, surface gravity).

2 Localized principal component analysis

The term "local(ized) PCA" was coined decades ago in the literature on dimension reduction and feature extraction. However, it has not been used in a unique way; there are several ways of interpreting the term *local(ized)*. In this section, we will give a brief review of types of local PCA and illustrate some of them by applying them to the second speed-flow diagram presented in Section 1. In addition, we consider another artificial data set (corresponding to figure 3d in [16]), representing a spiral-like data cloud, with which principal curve / NLPCA algorithms generally tend to have problems.

2.1 Cluster-wise PCA

Historically the term *localized PCA* was used for cluster-wise PCA. Instead of fitting principal components through the whole data set, cluster-wise PCA partitions the data cloud into clusters, within which the principal component analysis is carried out "locally".

Originally cluster-wise PCA was proposed as a tool for exploratory data analysis [24, 6]. It was rediscovered by the neural network community in the 1990s in the context of non-linear signal extraction as an alternative to auto-associative neural networks [33]. Five-layer auto-associative neural networks were successfully employed for (global) nonlinear PCA [40], using input and output layers with p units, and a hidden layer with $d < p$ units. Auto-associative neural networks possess many desirable theoretical properties in terms of signal approximation [30]. However neural networks typically yield a non-convex optimization problem that is burdensome to solve, and the algorithm is — without suitable regularization — prone to getting trapped in poor local optima. Empirical evidence [33] suggests that cluster-wise PCA is at least on a par with auto-associative neural networks.

The cluster-wise PCA algorithm can be seen as a generalization of the Generalized Lloyd algorithm ("k-means") [25]. In the k-means algorithm the cluster centers are points, whereas in localized PCA, the cluster centers are hyperplane segments. The outline of the the cluster-wise PCA algorithm is as follows [18]:

Algorithm 1: Cluster-wise ("local") PCA

1. Choose a target dimension d ($d = 1$ e.g. yields a line), a number of clusters Q, and an initial partitioning of the input space into Q disjoint regions $R^{(1)} \cup \ldots \cup R^{(Q)} = \mathbb{R}^p$.
2. Iterate ...
   i. For each partition $R^{(q)}$ $(q = 1, \ldots, Q)$ compute the "local" covariance matrices
   $$\boldsymbol{\Sigma}^{(q)} = \frac{1}{n^{(q)}} \sum_{i \in R^{(q)}} (\boldsymbol{x}_i - \bar{\boldsymbol{x}}^{(q)})(\boldsymbol{x}_i - \bar{\boldsymbol{x}}^{(q)})'$$
   with $\bar{\boldsymbol{x}}^{(q)} = \frac{1}{n^{(q)}} \sum_{i \in R^{(q)}} \boldsymbol{x}_i$ and $n^{(q)}$ being the number of observations in partition $R^{(q)}$. Obtain the local principal components $\boldsymbol{\gamma}_1^{(q)}, \ldots, \boldsymbol{\gamma}_d^{(q)}$ from the eigen decomposition of $\boldsymbol{\Sigma}^{(q)}$ (with corresponding eigenvalues $\lambda_1^{(q)} \geq \ldots \geq \lambda_d^{(q)} \geq \ldots$).
   ii. Update the partitioning $R^{(1)}, \ldots, R^{(Q)}$ by allocating each observation $\boldsymbol{x}_i$ to the "nearest" partition, i.e. the partition whose hyperplane segment is closest to $\boldsymbol{x}_i$.

Note that it is important in step 2.ii. to only consider the hyperplane segment $\left\{\bar{\boldsymbol{x}}^{(q)} + \sum_{j=1}^{d} \xi_j \boldsymbol{\gamma}_j^{(q)} : \xi_1, \ldots \xi_d \in \mathbb{R}\right\} \cap R^{(q)}$ instead of the whole hyperplane. Originally the partitioning was fixed a priori and thus there was no need to iterate steps i. and ii. This much simpler setup however is typically detrimental to the performance of the algorithm. A variant of algorithm 1 featuring automatic selection of clusters, and of the number of principal components within clusters, was proposed by Liu et al. [36] and used for star/galaxy classification.

It is easy to see that in complete analogy to the k-means algorithm the sum of squared distances (3) is never increased by any step of the above algorithm. However as with k-means or auto-associative neural networks, there is no guarantee that the global optimum is found. The most important drawback of this method is that its results are highly dependent on the initial choice of the partitioning.

This gives rise to the idea of slowly "building up" the partitions recursively akin to classification and regression trees (CARTs) [7]. Starting with a single partition, each partition is recursively "split up" into two partitions if the data can locally be better be approximated by two hyperplane segments instead of a single one.

In order to evaluate whether such a split is necessary, every partition $R^{(q)}$ is split at the mean of the partition $\bar{\boldsymbol{x}}^{(q)}$ orthogonally to the first principal component $\boldsymbol{\gamma}^{(q)}$ of the data in partition q, yielding two partitions $R^{(l)}$ and $R^{(r)}$. Next a small number of "k-segments" steps (steps 2.i. and 2.ii. from algorithm 1) are carried out for the partitions $R^{(l)}$ and $R^{(r)}$, however only using the data initially belonging to the partition $R^{(q)}$. The split is retained if

$$\frac{\lambda_1^{(q)}+\ldots+\lambda_d^{(q)}}{\lambda_1^{(q)}+\ldots+\lambda_p^{(q)}} < C\cdot\left(\frac{n^{(l)}}{n^{(q)}}\cdot\frac{\lambda_1^{(l)}+\ldots+\lambda_d^{(l)}}{\lambda_1^{(l)}+\ldots+\lambda_p^{(l)}}+\frac{n^{(r)}}{n^{(q)}}\cdot\frac{\lambda_1^{(r)}+\ldots+\lambda_d^{(r)}}{\lambda_1^{(r)}+\ldots+\lambda_p^{(r)}}\right), \tag{4}$$

where $n^{(q)}$ is the number of observations in partition $R^{(q)}$, $\lambda_k^{(q)}$ the k-th largest eigenvalue of the covariance of the data in partition $R^{(q)}$, and C is a constant which is typically chosen to be 1. Note that, contrary to CARTs, only a single split is considered for each partition. Considering every possible split of the partition would be computationally wasteful and in most cases not necessary as the split is subsequently optimized using a few "k-segments" steps.

In addition to testing whether partitions should be split, one can test whether neighboring partitions should be joined. This allows the algorithm to get around some of the suboptimal local extrema. In our experience it is beneficial to use a rather loose definition of neighborhood. Two partitions $R^{(l)}$ and $R^{(r)}$ are hereby considered to be neighbors if for at least for one observation $\boldsymbol{x}_i$, both $R^{(l)}$ and $R^{(r)}$ are amongst the $s > 2$ "closest" partitions. If the inequality (4) does not hold for two neighboring partitions $R^{(l)}$ and $R^{(r)}$, these are joined forming a new partition $R^{(q)} = R^{(l)} \cup R^{(r)}$.

This yields the following new algorithm:

Algorithm 2: Recursive local PCA

1. Start with a single partition $R^{(1)}$ containing all the data.

```
2. Iterate ...
    i. Test for each partition $R^{(q)}$ whether it should be split (using the criterion (4)).
   ii. Carry out a fixed number of "k segments" steps (steps 2.i. and 2.ii. of algorithm 1) updating all partitions.
  iii. Test for each neighboring pair of partitions whether the pair should be joined (using the criterion (4)).
   ... until there is no change in the allocation of observations to partitions.
```

Note that this algorithm does not require the choice of an initial partitioning. It can be beneficial to "enforce" a certain number of splits during the first few iterations, i.e. initially carry out every possible split irrespective of whether it meets criterion (4) and initially skip step 2.ii.

A similar idea has been proposed by Verbeek et al. [46]. Their method is however based on splitting off "zero length" segments corresponding to each observation.

Figure 2 illustrates the algorithm using a simple example. The data cannot be summarized by its first principal component (top left panel). Whilst it can be summarized by two line segments, the two line segments found in the second step are no truthful representation of the data (top right panel). The four line segments found in the third step correspond to the structure behind the data (bottom right panel). Finally the adjacent segments are joined leading to two partitions only (bottom right panel).

Note that the algorithm does not yield a principal curve or manifold, but merely disconnected line or hyperplane segments. It can be seen as finding tangent approximations to the principal curve or manifold. Thus the algorithm can by design cope with disconnected or branching principal curves or manifolds. Note that in the case of principal curves ($d = 1$) one can join the line segments to build a polygonal line [46].

2.2 Principal curves

Principal curves were firstly introduced by Hastie and Stuetzle [28]. Their definition is based on the concept of self-consistency. A curve $\boldsymbol{g}$ is called a Hastie-Stuetzle (HS) principal curve if $\boldsymbol{g}(\eta) = \mathbb{E}(\boldsymbol{x}|\eta_{\boldsymbol{g}}(\boldsymbol{x}) = \eta)$, with projection index $\eta_{\boldsymbol{g}}(\boldsymbol{x}) := \arg\min_\eta \|\boldsymbol{x} - \boldsymbol{g}(\eta)\|$. Hastie and Stuetzle show that a HS principal curve is a critical point of the distance function (2). Hastie and Stuetzle's definition however has a number of shortcomings. Whilst the principal curve can be shown to be a critical point of the distance function, one can show under fairly general conditions that it is just a saddle point of the distance function [14]. Further, when the data $\boldsymbol{x}$ are generated by adding noise $\boldsymbol{\epsilon}$ with $\mathbb{E}\boldsymbol{\epsilon} = \boldsymbol{0}$ to a curve $\boldsymbol{g}$, i.e. $\boldsymbol{x} = \boldsymbol{g}(\eta) + \boldsymbol{\epsilon}$, $\boldsymbol{g}$ is typically not the principal curve. Kégl et al. [34] overcome the former problem by considering principal

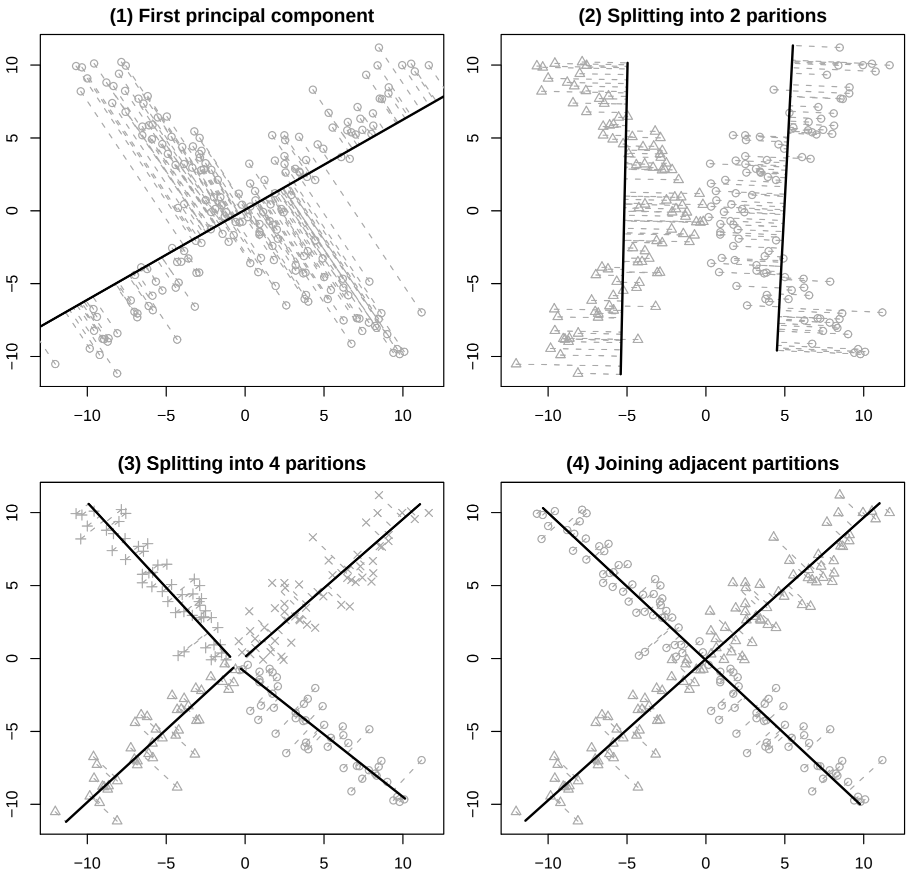


Fig. 2. Example illustrating the recursive local PCA algorithm.

curves of fixed length. Tibshirani's definition of principal curves [43] overcomes the latter shortcoming by defining principal curves using a generative model.

The algorithms proposed by Hastie and Stuetzle, Tibshirani, and Kégl can all be described as "top-down" (or "global") algorithms. All start with the first principal component and then iteratively "bend" it. The algorithm by Hastie and Stuetzle can be summarized as follows.

Algorithm 3: "Top-down" principal curves (HS)

```
1. Initialize the principal curve as the first principal component line $\boldsymbol{g}(\eta) = \bar{\boldsymbol{x}} + \eta\gamma_1$.
2. Iterate between ...
    i. **Projection** Project the data points $\boldsymbol{x}_i$ onto the principal curve $\boldsymbol{g}$ yielding projection indexes $\eta_{\boldsymbol{g}}(\boldsymbol{x}_i) = \eta_i$.
```

ii. **Reconstruction** Fit the data points $\boldsymbol{x}_i$ component-wise against the projection indices η_i using a scatterplot smoother.

...until the change of some measure of goodness-of-fit falls below a certain threshold. The curve $\boldsymbol{g} : \mathbb{R} \longrightarrow \mathbb{R}^p$ reconstructed in the last iteration is the estimated principal curve.

The reconstruction step requires some form of smoothing, otherwise all data points would just be interpolated. It is implemented by using splines or local smoothers such as LOESS [10]. Note that this algorithm entails that the order of the projections η_i is maintained in each iteration. Tibshirani's approach yields a similar algorithm. As Tibshirani's definition does not assume errors that are orthogonal to the principal curve the observations cannot be directly associated to a single point on the curve. Rather the algorithm is based on weights ("responsibilities") that correspond to the likelihood that a certain observation was generated by a certain point on the curve. From this point of view Tibshirani's algorithm is very similar to the EM algorithm for Gaussian mixture models. Kégl et al.'s algorithm uses polygonal lines with an increasing number of nodes.

Though all the three algorithms were shown to give satisfying results in a variety of circumstances, they all suffer from the problem of being highly dependent on the initialization. An unsuitable initialization of the projection indices η_i cannot be corrected later on. This is particularly obvious for spiral-like data, see figure 3 (top).

Hence, instead of starting with a *global* initial line, it often seems more appropriate to construct the principal curve in a "bottom-up" manner by considering in every step only data in a *local* neighborhood of the currently considered point. Delicado [12] proposed the first principal curve approach which can be assigned to this family, the *principal curves of oriented points (PCOP)*. A second approach in this direction was recently made by Einbeck et al. [16] and is known as *local principal curves (LPC)*. These two algorithms differ from all other existing NLPCA / principal curve algorithms in several crucial points:

- They do not start with an initial globally constructed line like the first principal component.
- They do not dissect into the stages of projection and reconstruction. The entire fitting process is carried out in the data space only.
- They do not maximize/minimize a global fitting criterion.

We illustrate this family of methods by providing the LPC algorithm explictly. To fix terms, let $K_{\boldsymbol{H}}(\cdot)$ be a $p-$dimensional kernel function, $\boldsymbol{H}$ a multivariate $(p \times p)$ bandwidth matrix, $w_i^{\boldsymbol{x}} = K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x}) / \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})$, and $\boldsymbol{x}_i' = (x_{i1}, \ldots, x_{ip})$, $i = 1, \ldots, n$.

Algorithm 4: Local principal curves (LPC)

---

1. Select a starting point $\boldsymbol{x}_0$ and a step size t_0. Set $\boldsymbol{x} = \boldsymbol{x}_0$.

2. Calculate the local center of mass $\boldsymbol{\mu}^{\boldsymbol{x}} = \sum_{i=1}^{n} w_i^{\boldsymbol{x}} \boldsymbol{x}_i$ at $\boldsymbol{x}$. Denote by $\mu_j^{\boldsymbol{x}}$ the j-th component of $\boldsymbol{\mu}^{\boldsymbol{x}}$.
3. Estimate the local covariance matrix $\boldsymbol{\Sigma}^{\boldsymbol{x}} = (\sigma_{jk}^{\boldsymbol{x}})$ at $\boldsymbol{x}$ via
$$\sigma_{jk}^{\boldsymbol{x}} = \sum_{i=1}^{n} w_i^{\boldsymbol{x}} (x_{ij} - \mu_j^{\boldsymbol{x}})(x_{ik} - \mu_k^{\boldsymbol{x}}).$$
Let $\boldsymbol{\gamma}^{\boldsymbol{x}}$ be the first column of the loadings matrix $\boldsymbol{\Gamma}^{\boldsymbol{x}}$ computed locally at $\boldsymbol{x}$ in analogy to (1).
4. Setting $\boldsymbol{x} := \boldsymbol{\mu}^{\boldsymbol{x}} + t_0 \boldsymbol{\gamma}^{\boldsymbol{x}}$, one finds the updated value of $\boldsymbol{x}$.
5. Repeat steps 2 to 4 until the sequence of $\boldsymbol{\mu}^{\boldsymbol{x}}$ remains approximately constant (implying that the end of the data cloud is reached). Then set again $\boldsymbol{x} = \boldsymbol{x}_0$, set $\boldsymbol{\gamma}^{\boldsymbol{x}} := -\boldsymbol{\gamma}^{\boldsymbol{x}}$ and continue with step 4.

The step size t_0 is recommended to be set equal to h if $\boldsymbol{H} = \text{diag}(h, \ldots, h)$. The starting point $\boldsymbol{x_0}$ may or may not be a member of the data cloud, and can be selected by hand or at random. Depending on the particular data set, it can have a rather crucial influence on the fitted curve. (In the examples provided in figure 3 (middle), the spiral fit is quite independent of $\boldsymbol{x_0}$, but it may require some attempts to get the fit through the traffic data right. A successful (randomly selected) choice was here e.g. $\boldsymbol{x_0} = (93, 71)$, using $h = t_0 = 7$.) For branched or disconnected data clouds it is often useful to work with multiple initializations [15, 16] and to compose the principal curve from the individual parts resulting from each starting point. In order to improve stability it is beneficial to replace the first local principal component $\boldsymbol{\gamma}^{\boldsymbol{x}}$ by the weighted average between the current local principal component $\boldsymbol{\gamma}^{\boldsymbol{x}}$ and the previous local principal component $\boldsymbol{\gamma}^{\boldsymbol{x}_{\text{old}}}$, i.e. use $\alpha\boldsymbol{\gamma}^{\boldsymbol{x}} + (1-\alpha)\boldsymbol{\gamma}^{\boldsymbol{x}_{\text{old}}}$ for a suitable $\alpha \in (0, 1]$ ("angle penalization", see [16]).

To make the difference to Delicado's algorithm clear, let $\mathcal{H}(\boldsymbol{x}, \boldsymbol{b})$ be the hyperplane which contains $\boldsymbol{x}$ and is orthogonal to a vector $\boldsymbol{b}$. Then, for given $\boldsymbol{x}$, Delicado defines the *principal direction* $\boldsymbol{b}^*(\boldsymbol{x})$ as the vector minimizing the total variance of all observations lying on this hyperplane, and $\boldsymbol{\mu}^*(\boldsymbol{x})$ as the expectation of all points lying on $\mathcal{H}(\boldsymbol{x}, \boldsymbol{b}^*(\boldsymbol{x}))$. The set of fixed points of $\boldsymbol{\mu}^*$ defines the set $\mathcal{P}(\boldsymbol{X})$ of *principal oriented points (POPs)*, and any curve $\alpha : \mathbb{R} \supset I \longrightarrow \mathbb{R}^p$ with $\{\alpha(s) : s \in I\} \in \mathcal{P}(\boldsymbol{X})$ is a PCOP. This is a probabilistic definition, hence estimates are required in practice, and this is where localization enters into Delicado's algorithm in two different ways. Firstly, one considers a certain neighborhood of the current hyperplane $\mathcal{H}$, and data nearer to $\mathcal{H}$ are associated with higher weights. Secondly, a cluster analysis is performed on $\mathcal{H}$, and only data in the local cluster are used for averaging. In contrast to the LPCs, where the local center of mass is found in a one-step approximation, Delicado's analogue to step 2 requires iteration until convergence. Summarizing the essential difference slightly simplified, whereas LPC is based on calculating alternately a local first principal component γ_1^x and a local center of mass $\boldsymbol{\mu}^x$, in Delicado's algorithm these two components are replaced by the estimated principal direction and the estimated fixed points (POPs), respectively. While Delicado's algorithm is based on an elaborated and sound probabilistic theory, it is quite burdensome from a computational point of view, as his principal directions are not obtainable through an eigen decomposition [16], and a large number of cluster analyses has to be run. On the other hand, the computationally faster LPCs are a purely empirical concept, but they can be shown to be an approximation of Delicado's algorithm for small cluster sizes [16]. For either algorithm, extensions to branched principal curves using the notion of *local second principal curves* have been suggested [12, 15] and automatic bandwidth selection routines have been proposed [13, 16].

2.3 Further approaches

The term local PCA has also been used for other techniques and in other contexts, which we briefly summarize here. Firstly, one can extend the algorithms presented in Section 2.1 by allowing for smoothness over clusters using mixture models [38]. This means, rather than allocating an observation $\boldsymbol{x}_i$ to the nearest partition q, one considers the posterior probabilities $\boldsymbol{\pi}_{i,q}$ that $\boldsymbol{x}_i$ is generated by partition q, and defines *for each observation* the weighted (first) local principal component $\boldsymbol{\gamma}_{1,i} = \sum_q \boldsymbol{\pi}_{i,q}\boldsymbol{\gamma}_1^{(q)}$. This method requires knowledge of the mixture density itself and produces eigenvectors which are biased towards adjacent components, which motivated [38] to provide a modified algorithm using Fisher's linear discriminant (FLD) instead of PCA.

Secondly, there are approaches from the NLPCA family using local methods. While most NLPCA algorithms work by applying some nonlinear, but parametric, transformation in the projection and/or the reconstruction step ([47], [8], among others), Bolten et al. [5] allowed for a nonparametric reconstruction $\boldsymbol{g}$. Using in the projection step a nonlinear transformation followed by a linear mapping, i.e. $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{x})$, $\boldsymbol{\phi} : \mathbb{R}^p \rightarrow \mathbb{R}^l, \boldsymbol{W} \in \mathbb{R}^{d\times l}$, the reconstructed curve takes the form $\boldsymbol{g}(\boldsymbol{W}\boldsymbol{\phi}(\boldsymbol{x}))$, which is estimated using projection pursuit regression [23]. The actual smoothing method employed is the so-called supersmoother [21], which is a running line smoother using a local neighborhood of the target point with variable span. We do not consider NLPCA approaches further, as the contribution by U. Kruger in this volume is explicitly dedicated to them.

Thirdly, the term "local principal component analysis" was used by Aluja-Banet and Nonell-Torrent [2] for PCA using contiguity relations based on a non-directed graph expressing binary relations between the individuals. Their motivation is to control for the effect of a latent or third variable (e.g. geographical position) by virtually eliminating its influence on the principal component analysis. This falls somewhat out of the context of the present paper and we do not consider it further.

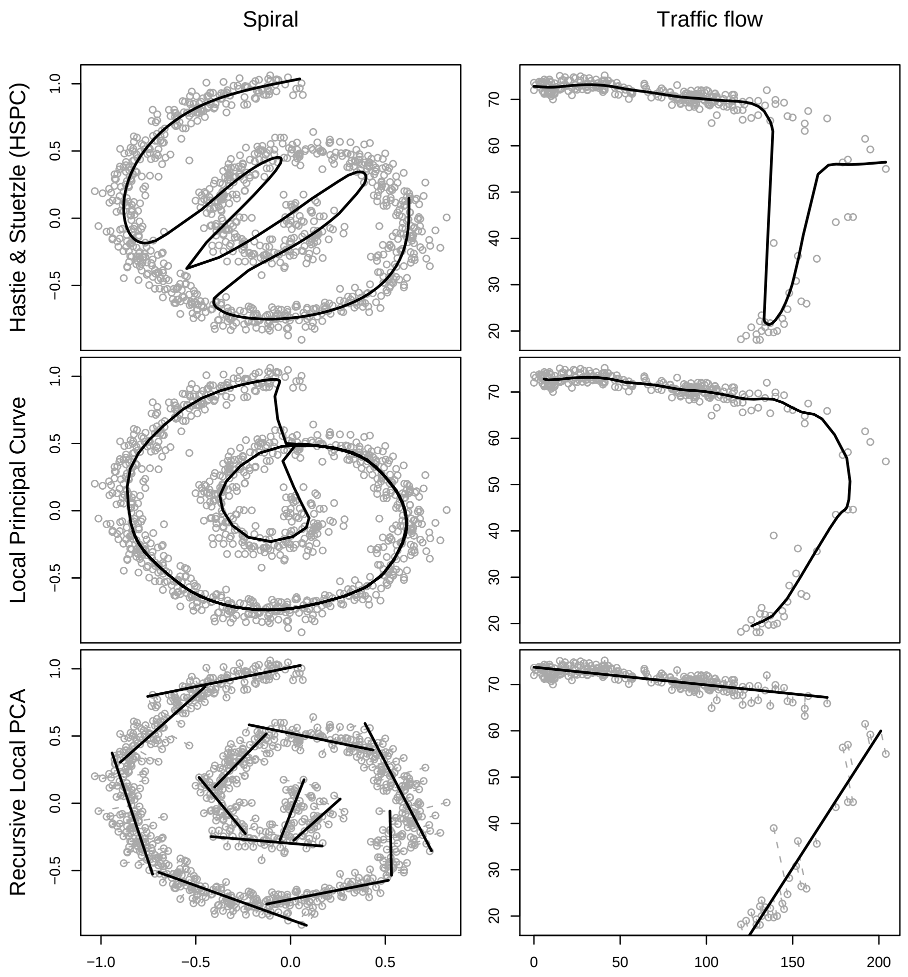


Fig. 3. Principal curves obtained for the spiral data and the non-linear traffic flow data using algorithm 3 (top row), algorithm 4 (middle row), and algorithm 2 (bottom row)

3 Combining principal curves and regression

3.1 Principal component regression and its shortcomings

Using principal components for dimension reduction prior to carrying out further analysis is an old paradigm in statistics. The combination of principal component analysis and linear regression, known as principal component regression (PCR), was proposed already in the 1950s by Kendall [35] and

Hotelling [31]. PCR consists of two steps. As a first step the first d principal components are extracted from the covariates $\boldsymbol{X}$. In the second step, the first d principal component scores $\boldsymbol{T}_d := (\boldsymbol{t}_1 \cdots \boldsymbol{t}_d) = \boldsymbol{X} \cdot \left(\boldsymbol{\gamma}^{(1)} \cdots \boldsymbol{\gamma}^{(d)}\right)$ are used as covariates in a linear regression model. Geometrically, this corresponds to projecting the covariates onto the affine hyperplane spanned by the first d principal components and using the projection indices as covariates in the regression.

The purpose of the projection step is to eliminate directions in which the covariates have little variance. The rationale behind this is that directions with little variance correspond to directions in which there is little information in the covariates and thus are prone to leading to an overfit.

Being built upon principal component analysis, PCR suffers from the same shortcomings as PCA. If the structure of the covariates cannot be suitably approximated by an affine hyperplane of low dimension, PCA and thus PCR are likely to fail. In the following we will propose a new regression-tree like algorithm, based on algorithm 2, which can be used for local dimension reduction of the covariate space of a regression model.

3.2 The generalization to principal curves

Instead of projecting the covariates onto the principal component surface we can project them onto a principal curve or manifold. In complete analogy to PCR we can then use the projection indices in the regression model.

This requires a unique parameterization of the principal manifold (e.g. a suitable unit speed parameterization). Whilst some of the algorithms presented in Section 2 compute the projection indices, other principal curve or manifold algorithms like algorithm 4 only yield a set of points on the curve or manifold.

The problem of having to compute the projection indices onto a non-linear curve or manifold can be circumvented by using tangent approximations to the principal curve or manifold as provided by algorithm 2, because the projections are then onto hyperplane segments and not onto non-linear manifolds. The projections onto the hyperplane segment corresponding to partition $R^{(q)}$ are

$$(t_{i1}^{(q)}, \ldots, t_{id}^{(q)}) = \underset{\substack{t_{i1}^{(q)}, \ldots, t_{id}^{(q)} \in \mathbb{R} \\ \bar{\boldsymbol{x}}^{(q)} + \sum_{j=1}^{d} t_{ij}^{(q)} \boldsymbol{\gamma}_j^{(q)} \in R^{(q)}}}{\arg\min} \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}^{(q)} - \sum_{j=1}^{d} t_{ij}^{(q)} \boldsymbol{\gamma}_j^{(q)} \right\|^2 .$$

The $(t_{i1}^{(q)}, \ldots, t_{id}^{(q)})$ are easy to compute as they are either the orthogonal projection of the covariates onto the hyperplane or a point on the boundary of the hyperplane segment.

In analogy to algorithm 2 one can fit a regression model separately in each partition. This, however, leads to a couple of disadvantages. First of all, the prediction would be discontinuous at the boundaries between the partitions. Furthermore, keeping the "hard partitioning" can lead to a large variance of the predictions, especially if the number of partitions is large. In order to avoid these detrimental effects the partitioning is "softened" by the introduction of weights that reflect how far an observation $\boldsymbol{x}_i$ is from a certain hyperplane segment. In order to achieve this all observations are projected onto all of the hyperplane segments. For each partition $R^{(q)}$ weights $\omega_i^{(q)}$ are associated with each observation $\boldsymbol{x}_i$. The weight $\omega_i^{(q)}$ decreases with increasing distance between the observation and the hyperplane, i.e. $\omega_i^{(q)} := w(\delta_i^{(q)})$, where $\delta_i^{(q)} = \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}^{(q)} - \sum_{j=1}^d t_{ij}^{(q)} \boldsymbol{\gamma}_j^{(q)} \right\|^2$ is the squared distance between the i-th observation and its projection onto the hyperplane segment belonging to the partition $R^{(q)}$, and $w : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ is a decreasing function, such as $w(\delta) = e^{-C \cdot \delta}$. Figure 4 illustrates this idea. The steps of the proposed algorithm are as follows:

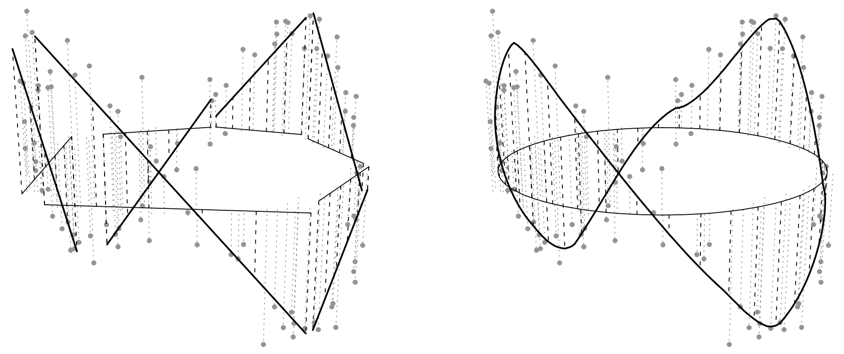


Fig. 4. Local regression lines in the partitions and resulting regression function for covariates lying on a circle in the two-dimensional plain. The vertical dimension corresponds to the response. Left: "hard partitioning"; right: "softened".

Algorithm 5: Projection-based regression trees

1. Carry out algorithm 2 yielding a set of partitions $R^{(1)}, \ldots, R^{(Q)}$ with the corresponding hyperplane segments.
2. Compute the squared distances $\delta_i^{(q)} = \left\| \boldsymbol{x}_i - \bar{\boldsymbol{x}}^{(q)} - \sum_{j=1}^d t_{ij}^{(q)} \boldsymbol{\gamma}_j^{(q)} \right\|^2$ and weights $\omega_i^{(q)} := w(\delta_i^{(q)})$.
3. For each partition $R^{(1)}, \ldots, R^{(Q)}$:

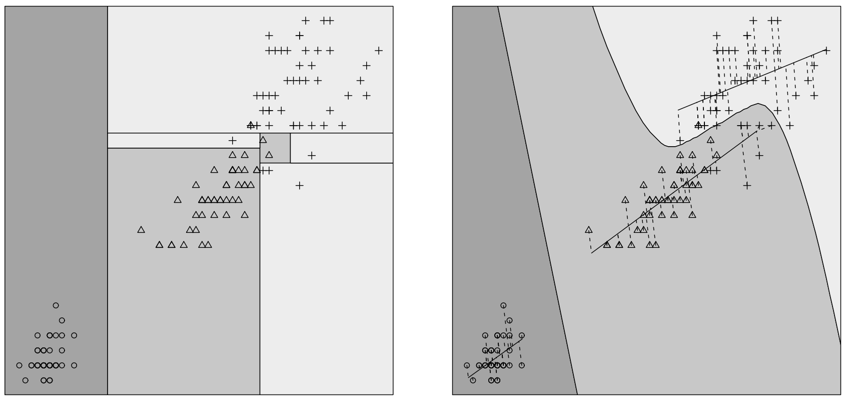
   Fit a regression model using the projections of the data $\boldsymbol{T}^{(q)} := \left(\boldsymbol{t}_1^{(q)} \cdots \boldsymbol{t}_d^{(q)} \right)$ as covariates, $\boldsymbol{y}$ as response, and $\omega_i^{(q)}$ as observation weights.
4. Compute the predictions as weighted means

$$\hat{y}_i = \frac{\sum_{q=1}^{Q} \omega_i^{(q)} \hat{y}_i^{(q)}}{\sum_{q=1}^{Q} \omega_i^{(q)}},$$

where $\hat{y}_i^{(q)}$ is the prediction obtained from the regression model in partition $R^{(q)}$.

Note that a wide variety of regression methods can be used in step 3. If the regression method does not support weights one can alternatively sample from all observations using the $\omega_i^{(q)}$ as sampling weights.

Algorithm 5 is a prediction algorithm that is like CARTs based on recursive partitioning. In contrast to CARTs, which split up the covariate space by a series of (orthogonal) "cuts", the method proposed here partitions the covariate space according to a tessellation defined by the hyperplane segments. Figure 5 illustrates this comparison. As algorithm 5 takes the structure of the covariates into account, it provides some sort of regularization. Thus it tends to have a smaller variance than CARTs, however at the price of a potentially increased bias. In a boosting [20] context, this makes the method proposed here an interesting candidate for a weak learner.



(a) "Classical" CART (b) Projection-based classification tree

Fig. 5. Comparison between a classification tree (CART) and a projection-based classification tree

The algorithm presented here is similar to cluster-wise linear regression [42], however there a few major differences. Most importantly, a dimension reduction step is carried out in each cluster and the clusters are not constructed around a single point, the cluster center, but around hyperplane segments. Furthermore the boundaries between the partitions are softened by the exponential weights. Note that the algorithm considered so far does not take the response variable $\boldsymbol{y}$ into account when constructing the partitions.

3.3 Using directions other than the local principal components

The rationale behind projecting the data onto the principal component plane or a principal manifold was based on equating the variance of the covariates to the information the covariates provide. But this does not necessarily imply that the directions with the largest variance necessarily provide the *relevant* information necessary to predict the response $\boldsymbol{y}$.[5] In other words, it might be beneficial to consider both the structure of the covariates and the response variable.

We propose to modify algorithm 2 to take both these aspects into account. The core step of algorithm 2 were the “k-segments steps” used already in algorithm 1. In step 2.i. the principal components are computed in each partition. The core property of the principal components is that they maximize the variance of the projections $\mathrm{Var}(\boldsymbol{X}^{(q)}\boldsymbol{\gamma})$. Clearly, this does not take the response $\boldsymbol{y}^{(q)}$ into account at all.[6]

If we were to choose the direction that predicts the response $\boldsymbol{y}^{(q)}$ best, we would choose the least squares regression estimate $\hat{\boldsymbol{\beta}}^{(q)} = \left(\boldsymbol{X}^{(q)\prime}\boldsymbol{X}^{(q)}\right)^{-1}\boldsymbol{X}^{(q)\prime}\boldsymbol{y}^{(q)}$, which maximizes the squared correlation coefficient $\rho^2(\boldsymbol{y}^{(q)}, \boldsymbol{X}^{(q)\prime}\boldsymbol{\beta})$. Note that the latter criterion does not consider the variance of the covariates $\boldsymbol{X}^{(q)}$ at all.

It seems to be a suitable compromise to choose a projection direction $\boldsymbol{\gamma}$ that maximizes the product of the two aforementioned criteria, which is equivalent to maximizing the squared covariance between the $\boldsymbol{X}^{(q)}\boldsymbol{\gamma}$ and $\boldsymbol{y}^{(q)}$, i.e.

$$\mathrm{Cov}^2(\boldsymbol{y}^{(q)}, \boldsymbol{X}^{(q)}\boldsymbol{\gamma}) = \rho^2(\boldsymbol{y}^{(q)}, \boldsymbol{X}^{(q)}\boldsymbol{\gamma}) \cdot \mathrm{Var}(\boldsymbol{X}^{(q)}\boldsymbol{\gamma}) \cdot \mathrm{Var}(\boldsymbol{y}^{(q)}).$$

This covariance is maximised by the projection direction of the PLS algorithm [49, 11]. This alternative projection direction can easily be incorporated into algorithms 1 and 2. Step 2.i. of algorithm 1 simply has to be replaced by the computation of the PLS projection direction obtained by carrying out a PLS regression with the data belonging to partition $R^{(q)}$. Using this modification typically improves the predictive performance. The following paragraph gives an example for this.

3.4 A simple example

In the following we will consider a data set taken from [9] to illustrate the algorithm proposed in this section. The data consists of measurements of the radial velocity of a spiral galaxy taken at 323 points in the area of sky which it covers. The different points in the area of the sky are determined by their north-south and east-west coordinates. The data is visualized in figure 6. It is

[5] See [32] for a more detailed discussion.

[6] $\boldsymbol{X}^{(q)}$ contains hereby all covariates $\boldsymbol{x}^{(q)}$ from the partition $R^{(q)}$, i.e. the $\boldsymbol{x}_i \in R^{(q)}$. $\boldsymbol{y}^{(q)}$ is defined analogously.

easy to see that the measurements lie within a small number of slots crossing the origin.

The data set was randomly split into a training set of 162, and a test set of 161 observations. Table 1 gives the L_2 error (and its standard deviation) obtained when predicting the radial velocity using different methods based on 1000 replications. The methods considered were algorithm 5 once with the principal components and once with the PLS projection direction, a linear model, generalized additive models (GAM) [29], multivariate adaptive regression splines (MARS) [22], and projection pursuit regression (PPR) [23]. The results show that the algorithm proposed here clearly outperforms the other methods. This is mostly due to the fact that it can successfully exploit the structure of the covariates. The PLS based variant of the algorithm gives much better results than the principal-component based variant.

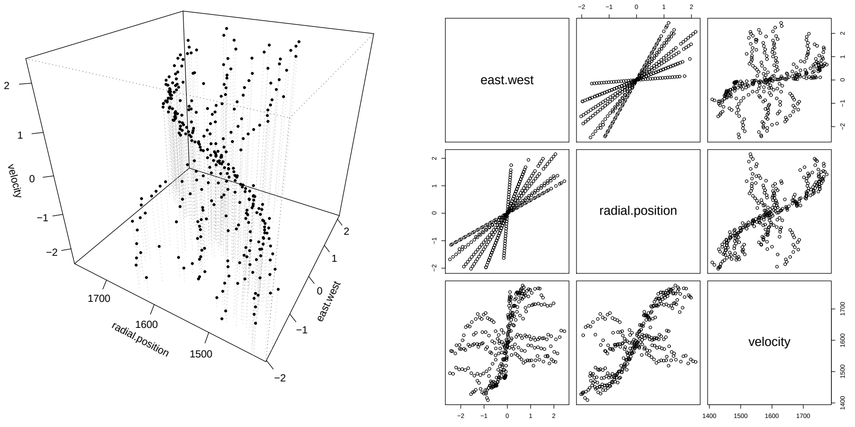


Fig. 6. Visualization of the galaxy data.

4 Application to the Gaia survey mission

Gaia is an astrophysics mission of the European Space Agency (ESA) which will undertake a detailed survey of over 10^9 stars in our Galaxy and extragalactic objects. An important part of the scientific analysis of these data is the classification of all the objects as well as the estimation of stellar astrophysical parameters. Below we give a brief overview of the mission and the data before describing the results obtained using the techniques presented in the preceding section.

Table 1. Average L_2 error (and standard deviation of the error) obtained for the galaxy example for the different methods compared.

	Training set	Test set
Using PC directions	1642.00 (578.4)	1758.44 (615.5)
Using PLS directions	511.42 (79.3)	577.05 (101.9)
Linear model	7420.45 (352.9)	7591.19 (377.5)
MARS	2965.64 (334.7)	3738.76 (494.7)
GAM	3027.14 (321.6)	3554.26 (385.5)
PPR	2207.73 (622.0)	3317.94 (820.7)

4.1 The astrophysical data

After launch in 2011, Gaia will study the composition and origin of our Galaxy by determining the properties of stars in different populations across our entire Galaxy [17, 41]. One of its major contributions will be to measure stellar distances to much higher accuracy than has hitherto been possible (and will do it for a vast numbers of stars). Gaia will also measure the three-dimensional space motions of stars in exquisite detail. These will be used together in dynamical models to map out the distribution of matter, and can be used to answer fundamental questions concerning galaxy formation.[7]

Much of the Gaia astrometric (3D position, 3D velocity) data would be of little value if we did not know the intrinsic properties of the stars observed, quantities such as the temperature, mass, chemical compositions, radius etc. (collectively referred to as *Astrophysical Parameters*, or APs; see [3]). For this reason, Gaia is equipped with a low resolution spectrograph to sample the spectral energy distribution at 96 points across the optical and near-infrared wavelength range (330–1000 nm). The measurements themselves are photon counts (energy flux). Each object can therefore be represented as a point in a 96-dimensional data space. For those objects which are stars, the astrophysical parameters of most interest are the following four: (1) *effective temperature*, which roughly corresponds to the temperature of the observable part of the stellar atmosphere; (2) the *surface gravity*, which is the acceleration due to gravity at the surface of the star; (3) the *metallicity* or *abundance*, a single measurement of the chemical composition of the star relative to that of the Sun; (4) the interstellar extinction, which measures how much of the star's light has been absorbed or scattered by interstellar dust lying between us and the star. In practice there is additional "cosmic variance" due to other APs, but these four are the main ones of interest. (In the rest of this paper we examine a simpler case in which there is no variance due to interstellar extinction.)

After a century of progress in astrophysics, we now have sophisticated models of stellar structure and from these we can generate synthetic spectra

[7] For more information see `http://www.rssd.esa.int/Gaia`

which reproduce real stellar spectra reasonably well. Therefore, we can construct libraries of template spectra with known APs and use these to train supervised regression models in order to estimate stellar APs. This has received quite a lot of attention in the astronomical literature, with methods based on nearest neighbors (e.g. [1]), neural networks (e.g. [4], [48]) and support vector machines (e.g. [44])

In its simplest form, the problem of estimating APs with Gaia is one of finding the optimal mapping between the 96-dimensional data space and the 3 or more dimensional AP space. In theory this can be solved directly with regularized nonlinear regression, with the mapping inferred from simulated data. In practice, however, it is more complicated. For example, the mapping is not guaranteed to be unique, so some kind of partitioning of the data space may be appropriate. Furthermore, the intrinsic dimensionality of the data space is much lower than 96, so we could probably benefit from dimensionality reduction. Standard PCA has been applied to such data (e.g. [4], [19]). It produces more robust models, but possibly at the cost of filtering out low amplitude features which are nonetheless relevant for determining the "weaker" APs (see below). Here we investigate local reduction techniques as an alternative.

4.2 Principal-manifold based approach

In this section we study how the methods discussed in Sections 2 and 3 can be applied to Gaia spectral data. The data comprise several thousand spectra showing variance in the three astrophysical parameters temperature (in Kelvin), metallicity and gravity; the latter two variables are on a logarithmic scale.[8] Temperature is a "strong" parameter, meaning it accounts for most of the variance across the data set. Gravity and metallicity, in contrast, are weak. The parameters have a correlated impact on the data, e.g. at high temperatures, varying the metallicity has a much smaller impact on spectra than it does at low temperatures. The data used here are simulated with no noise added.

The plot of the first three principal components of the covariates, the photon counts, shows clearly that these possess some low-dimensional structure (figure 7), which cannot be *linearly* approximated. This suggests employing principal-manifolds based methods.

The low-dimensional structure of the photon counts can be exploited by the projection-based regression tree algorithm (algorithm 5). Recall that the algorithm is based on the idea that the response is mainly determined by the projection of the covariates onto the (tangent to the) principal manifold. This however does not need to be the case; it might well be that the *relevant* information is not captured by the principal manifold.

[8] In the present example the spectra actually have a dimensionality of just 16, rather than 96 as mentioned above, because when we carried out this work the Gaia instruments were still being developed. Nonetheless, the results we present are illustrative of problems typical in observational astronomy.

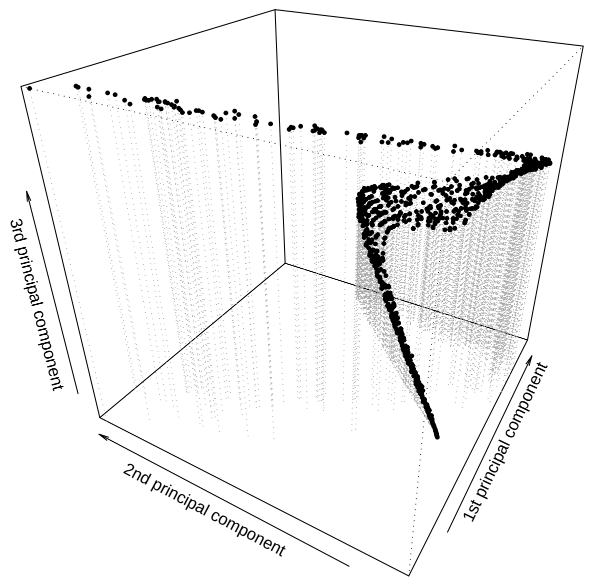


Fig. 7. First three principal components of the photon counts

In the following we will compare whether exploiting the manifold structure of the data allows for obtaining better predictions of the APs. Given the highly nonlinear structure of the data we use support vector regression machines (with a Gaussian kernel) in each partition.[9] The partitions are determined by the PLS-based version of algorithm 5 using 6-dimensional (gravity, metallicity) and 4-dimensional (temperature) hyperplane segments. The results obtained are compared to two further support vector regression machines: one using the original data space of 16 photon counts directly, and one using the first 6 (gravity, metallicity) / 4 (temperature) global principal components as covariates.

For this purpose, a training set of 3,000 observations, a calibration set of 1,000 observations, and a test set of 1,000 observations is repeatedly drawn from the simulated Gaia data (100 replications). Table 2 gives the results. As support vector regression machines minimize the L_1 distance, we use the L_1 error $\sum_i |y_i - \hat{y}_i|/n$. The relative error reported is $\sum_i |y_i - \hat{y}_i| / \sum_i |y_i - \bar{y}|$.

The results obtained for the temperature and the metallicity show that using the manifold structure of the photon counts allows for a significant improvement of the predictive performance. The hyperplanes extracted in algorithm 5 seem to capture the information that is relevant to predicting the

[9] The optimal cost parameter and the optimal kernel width are determined for each partition individually using a calibration set of 1,000 observations.

Table 2. Average L_1 error and relative error (and standard deviation of the error) obtained for the Gaia data.

(a) Results obtained for the temperature

	Training set		Test set	
	L_1 error	Relative err.	L_1 error	Relative err.
Algorithm 5	252.9 (75.1)	0.042 (0.012)	258.9 (75.1)	0.043 (0.013)
SVR (all counts)	408.6 (11.6)	0.068 (0.002)	432.1 (16.0)	0.074 (0.004)
SVR (P.Comps.)	404.3 (19.6)	0.067 (0.003)	412.8 (21.8)	0.070 (0.004)

(b) Results obtained for the gravity

	Training set		Test set	
	L_1 error	Relative err.	L_1 error	Relative err.
Algorithm 5	0.083 (0.003)	0.069 (0.002)	0.109 (0.008)	0.090 (0.005)
SVR (all counts)	0.080 (0.001)	0.067 (0.001)	0.104 (0.005)	0.087 (0.005)
SVR (P.Comps.)	0.091 (0.002)	0.076 (0.001)	0.146 (0.009)	0.118 (0.007)

(c) Results obtained for the metallicity

	Training set		Test set	
	L_1 error	Relative err.	L_1 error	Relative err.
Algorithm 5	0.193 (0.018)	0.134 (0.012)	0.269 (0.016)	0.189 (0.011)
SVR (all counts)	0.279 (0.005)	0.193 (0.003)	0.363 (0.013)	0.253 (0.008)
SVR (P.Comps.)	0.256 (0.007)	0.177 (0.004)	0.389 (0.014)	0.269 (0.011)

temperature and the metallicity whilst some of the noise is discarded in the projection step, which facilitates the prediction. This explains why algorithm 5 outperforms the support vector regression machine using all photon counts. The principal components are less able to capture the relevant information; the performance of the support vector regression machine using the first 6 (or 4, respectively) global principal components is clearly worse.

The results obtained for the gravity, the weakest of the APs, however give a different picture. Using the manifold structure does *not* allow for improved predictions of the gravity. The information relevant for predicting the gravity seems to be "orthogonal" to the extracted hyperplane segments. Thus the support vector regression machine using all the photon counts performs better than the ones based on lower-dimensional projections.

5 Conclusion

We have reviewed several approaches and algorithms for the representation of high-dimensional complex data structures through lower-dimensional curves or manifolds, which share the property of being based — by some means or other — on carrying out principal component analysis "locally". We have focused on localized versions of principal curves (algorithm 4), and on an "intelligent" partitioning algorithm (algorithm 2) avoiding the necessity to specify an initial partitioning as with usual cluster-wise ("local") PCA.

We have demonstrated, using the latter of the two algorithms, how localized principal components can be used to reduce the dimension of the predictor space in a non-linear high-dimensional regression problem. The information relevant to predicting the response variable can be elegantly taken into account by maximizing the covariance between the response variable and the extracted projection directions akin to partial least squares (PLS). We have applied this technique successfully to photon count data from the Gaia mission, where we were able to improve the predictive performance for some of the response variables significantly. The framework presented here however does not guarantee an improved prediction, especially if the information relevant to the prediction problem at hand cannot be captured by the extracted low-dimensional structure — as it was the case for the gravity.

It would be desirable to be able to also apply local principal curves (algorithm 4) to such problems. In contrast to algorithm 2, LPCs have the advantage of representing the covariate space through a proper curve (or manifold) instead of disconnected line (or hyperplane) segments. Currently, algorithm 4 can only extract one-dimensional curves, which it approximates by a sequence of points. Generalizing this idea to higher dimensions, one can approximate a d-dimensional principal manifold by a d-dimensional mesh, very much like the elastic net algorithm proposed by Gorban and Zinovyev [26]. The (basic) elastic net algorithm however is a "top-down" method that iteratively bends a mesh of points, starting with a given topology. The generalization of algorithm 4 would use a "bottom-up" approach, i.e. learn the local topology from the data requiring no initialization.

However, generalizing algorithm 4 to d-dimensional manifolds poses a number of challenges. Firstly, the angle penalization needs to be modified that it can be applied to local principal components of higher order. Further one has to make sure that the different branches meet, forming a proper mesh of points, which will require keeping the distance between subsequent $\boldsymbol{\mu}^{\boldsymbol{x}}$ constant.

References

1. C. Allende Prieto. An automated system to classify stellar spectra – I. *Monthly Notices of the Royal Astronomical Society*, 339:1111–1116, 2003.

2. T. Aluja-Banet and R. Nonell-Torrent. Local principal component analysis. *Qüestioó*, 3:267–278, 1991.
3. C. A. L. Bailer-Jones. Determination of stellar parameters with GAIA. *Astrophysics and Space Science*, 280:21–29, 2002.
4. C. A. L. Bailer-Jones, M. Irwin, and T. von Hippel. Automated classification of stellar spectra. II: Two-dimensional classification with neural networks and principal components analysis. *Monthly Notices of the Royal Astronomical Society*, 298:361–377, 1998.
5. R. J. Bolton, D. J. Hand, and A. R. Webb. Projection techniques for nonlinear principal component analysis. *Statistics and Computing*, 13:267–276, 2003.
6. E. Braverman. Methods for extremal groupings of the variables and the problems of finding important factors. *Automation and Remote Control*, 31:123–132 (English translation from Russian), 1970.
7. L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth and Brooks/Cole, Monterey, 1984.
8. B. Chalmond and S. C. Girard. Nonlinear modelling of scattered data and its application to shape change. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21:422–432, 1999.
9. J. M. Chambers and T. J. Hastie, editors. *Statistical Models in S.* CRC Press, Inc., Boca Raton, 1991.
10. W. S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74:829–836, 1979.
11. S. de Jong. SIMPLS. An alternative approach to partial least squares regression. *Chemometric and Intelligent Laboratory Systems*, 18:251–263, 1993.
12. P. Delicado. Another look at principal curves and surfaces. *Journal of Multivariate Analysis*, 77:84–116, 2001.
13. P. Delicado and M. Huerta. Principal curves of oriented points: Theoretical and computational improvements. *Computational Statistics*, 18:293–315, 2003.
14. T. Duchamps and W. Stuetzle. Extremal properties of principal curves in the plane. *Annals of Statistics*, 24(4):1511–1520, 1996.
15. J. Einbeck, G. Tutz, and L. Evers. Exploring multivariate data structures with local principal curves. In C. Weihs and W. Gaul, editors, *Classification - The Ubiquitous Challenge*, pages 257–263, Heidelberg, 2005. Springer.
16. J. Einbeck, G. Tutz, and L. Evers. Local principal curves. *Statistics and Computing*, 15:301–313, 2005.
17. ESA. Gaia: Composition, formation and evolution of the galaxy. Technical Report ESA-SCI 4, ESA, 2000.
18. E. Diday et Collaborateurs. *Optimisation en Classification Automatique.* INRIA, Le Chesnay, France, 1979.
19. P. Re Fiorentin, C. A. L. Bailer-Jones, Y. S. Lee, T. C. Beers, and T. Sivarani. Estimating stellar atmospheric parameters from SDSS/SEGUE spectra. *Astronomy & Astrophysics*, submitted, 2007.
20. Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.
21. J. H. Friedman. A variable span scatterplot smoother. Technical Report No. 5, Laboratory for Computational Statistics, Stanford University, 1984.
22. J. H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19(1):1–67, 1991.

23. J. H. Friedman and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76(376):817–823, 1981.
24. K. Fukunaga and D. Olsen. An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, 20(2):176–183, 1971.
25. A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Amsterdam, 1992.
26. A. Gorban and A. Zinovyev. Elastic principal graphs and manifolds and their practical applications. *Computing*, 75:359–379, 2005.
27. F. L. Hall, V. F. Hurdle, and J. M. Banks. Synthesis of recent work on the nature of speed-flow and flow-occupancy (or density) relations on freeways. *Transportation Research Record*, 1365:12–17, 1992.
28. T. Hastie and W. Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84:502–516, 1989.
29. T. Hastie and R. Tibshirani. Generalized additive models. *Statistical Science*, 1:297–318, 1986.
30. K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.
31. H. Hotelling. The relations of the newer multivariate statistical methods to factor analysis. *British Journal of Statistical Psychology*, 10:69–79, 1957.
32. I. T. Jolliffe. A note on the use of principal components in regression. *Applied Statistics*, 31(3):300–303, 1982.
33. N. Kambhatla and T. K. Leen. Dimension reduction by local PCA. *Neural Computation*, 9:1493–1516, 1997.
34. B. Kégl, A. Krzyżak, T. Linder, and K. Zeger. Learning and design of principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:281–297, 2000.
35. M. G. Kendall. *A Course in Multivariate Analysis.* Griffin, London, 1957.
36. Z.-Y. Liu, K.-C. Chiu, and L. Xu. Improved system for object detection and star/galaxy classification via local subspace analysis. *Neural Networks*, 16:437–451, 2003.
37. E. C. Malthouse. Limitations of nonlinear PCA as performed with generic neural networks. *IEEE Transactions on Neural Networks*, 9:165–173, 1998.
38. D.J. Marchette and W.L. Poston. Local dimensionality reduction using normal mixtures. *Computational Statistics*, 14:469–489, 1999.
39. A. H. Monaghan. Nonlinear principal component analysis by neural networks: theory and application to the Lorenz system. *Journal of Climate*, 13:821–835, 2000.
40. E. Oja. Data compression, feature extraction and autoassociation in feedforward neural networks. In T. Kohonen, K. Makkisara, O. Simula, and J. Kangas, editors, *Artificial neural networks*, pages 737–745. Elsevier, Amsterdam, 1991.
41. M. A. C. Perryman, K. S. de Boer, G. Gilmore, E. Høg, M. G. Lattanzi, L. Lindegren, X. Luri, F. Mignard, O. Pace, and P. T. de Zeeuw. Gaia: Composition, formation and evolution of the galaxy. *Astronomy and Astrophysics*, 369:339–363, 2001.
42. H. Späth. Algorithm 39: Clusterwise linear regression. *Computing*, 22:367–373, 1979.
43. R. Tibshirani. Principal curves revisited. *Statistics and Computing*, 2:183–190, 1992.

44. P. Tsalmantza, M. Kontizas, C. A. L. Bailer-Jones, B. Rocca-Volmerange, R. Korakitis, E. Kontizas, E. Livanou, A. Dapergolas, I. Bellas-Velidis, A. Vallenari, and M. Fioc. Towards a library of synthetic galaxy spectra and preliminary results of the classification and parametrization of unresolved galaxies from Gaia. *Astronomy and Astrophysics*, page submitted, 2007.
45. P. Varaiya. Freeway performance measurement system (PeMS) version 4. Technical Report UCB-ITS-PRR-2004-31, California Partners for Advanced Transit and Highways (PATH), 2004.
46. J. J. Verbeek, N. Vlassis, and B. Kröse. A k-segments algorithm for finding principal curves. Technical report, Computer Science Institute, University of Amsterdam, 2000. IAS-UVA-00-11.
47. A. R. Webb. An approach to non-linear principal component analysis using radially symmetric basis functions. *Statistics and Computing*, 6:159–168, 1996.
48. P. G. Willemsen, M. Hilker, A. Kayser, and C. A. L. Bailer-Jones. Analysis of medium resolution spectra by automated methods: Application to M 55 and ω Centauri. *Astronomy and Astrophysics*, 436:379–390, 2005.
49. H. Wold. Soft modeling. The basic design and some extensions. In H. Wold and K. G. Jøreskog, editors, *Systems under Indirect Observation. Causality - Structure - Prediction*, volume 1, pages 1–53. North Holland, 1982.