Curve Estimation Based on Localised Principal Components - Theory and Applications

Mohammad Abd-Allateef Zayed

Department of Mathematical Sciences, Durham University

A thesis submitted for the degree of PhD

2011

Abstract

In this work, basic theory and some proposed developments to localised principal components and curves are introduced. In addition, some areas of application for local principal curves are explored.

Only relatively recently, localised principal components utilising kernel-type weights have found their way into the statistical literature. In this study, the asymptotic behaviour of the method is investigated and extended to the context of local principal curves, where the characteristics of the points at which the curve stops at the edges are identified. This is used to develop a method that lets the curve 'delay' convergence if desired, gaining more access to boundary regions of the data. Also, a method for automatic choice of the starting point to be one of the local modes within the data cloud is originated.

The modified local principal curves' algorithm is then used for fitting multi-dimensional econometric data. Special attention is given to the role of the curve parametrisation, which serves as a feature extractor and also as a prediction tool when properly linked to time as a probable underlying latent variable. Local principal curves provide a good dimensionality reduction and feature extraction tool for insurance industry key indicators and consumer price indices. Also, through 'calibrating' it with time, curve parametrisation is used for the purpose of predicting unemployment and inflation rates.

Curve Estimation Based on Localised Principal Components - Theory and Applications

Mohammad Abd-Allateef Zayed

Department of Mathematical Sciences

Durham University

A thesis submitted for the degree of

Doctor of Philosophy

2011

Contents

List of Tables

List of Figures

Declaration

"I hereby declare that no portion of the work that appears in this study has been used in support of an application for another degree or qualification of this or any other University or institution of learning."

Statement of Copyright

Acknowledgements

All praise be to God, the Almighty, for giving me patience, strength and courage to finish this work.

My sincere gratitude and appreciation goes to Dr. Jochen Einbeck, my supervisor, who deserves all respect, honour and loyalty, for his endless encouragement and guidance during all stages of my PhD study. His kindness, help and support was indeed one of the few factors that made me survive my days in Durham.

I am grateful to my family, especially my wife, for their patience and support, sharing with me good and bad moments.

Also, I would like to thank my beloved country Egypt, my sponsor, for giving me this opportunity to complete my PhD studies in one of the UK's leading universities.

Lastly, my regards and blessings to all of those who have supported me in any respect during the preparation and completion of this work.

M. A. Zayed
Durham, UK
November 2011
mohammad.a.zayed@gmail.com

To my loving parents ...

To my caring wife ...

To my dear kids, Mohammad and Fatemah ...

To my sister and brother ...

To all who care for me only for the sake of God ...

Chapter 1

Introduction

1.1 Fitting Multi-dimensional Data

In many real-time situations, economical, demographic, geological and even social studies deal with complex multidimensional data sets. With this type of data, one of the key roles of statistics is to analyse and describe data providing useful summaries that can help developing the current knowledge about this data and also help extracting useful measures that may be of some importance in predicting and analysing the expected future behaviour of the phenomenon under study.

Different approaches of learning from data can be applied. The more suitable is the learning process, the more useful are the summaries extracted. In this context, statistical learning can play an important role in many areas of science, especially applied sciences [40]. A famous approach of statistical learning, namely, *supervised learning*, is of interest in situations where data can be organised in the form of two blocks, input and output, that are thought to be causally related. Supervised learning is mainly about developing a good *learner*, or prediction

model, that enables predicting the future behaviour of the output (outcomes) set based upon studying that of the input (features) set. Another alternative learning approach is that of '*unsupervised learning*'. The latter is used mainly in the cases where we have a set of data that cannot be easily organised in the form of inputs and outputs. In this case, data is thought of as the joint probability distribution of some underlying variables and statistical methods based upon the unsupervised approach of learning are mostly about gaining knowledge on the structure of data and the way it is clustered as well as recognising its main patterns and extracting its main features [40, 51].

It is well expected that data in many areas like actuarial and econometric sciences are of the multidimensional complex structure where unsupervised statistical learning methods can play a main role in reducing data dimensionality and extracting its main features. This is because data does not always imply specific asymmetric relationships between variables under consideration that provide a basis to classify these variables into dependent and independent. Whenever symmetry is assumed, the shape of the relations between variables does not change by interchanging variables' roles.

In such cases, it could be beneficial to know the shapes of the existing relationships as this may give useful interpretations of certain aspects related to the variables under study. When dealing with high-dimensional data, identifying the main landmarks of data and studying its underlying structure and relationships is not that straightforward. The more the dimensions of data the more difficult summarising and visualising the data becomes.

Now, we provide a simple example comparing some traditional supervised and unsupervised learning techniques. We use two-dimensional speed-flow data recorded

for a freeway in California-US on July 2007[(1)]. Figure 1.1 shows different fits for the traffic data. Three supervised methods were used, linear regression, quadratic form, linear interpolation using R function '*approx()*'. The solid line in the figure is a typical representative of unsupervised learning techniques, and is known as 'the first principal component line'. Using this figure, a basic comparison between the alternative ways to fit such data can be done.

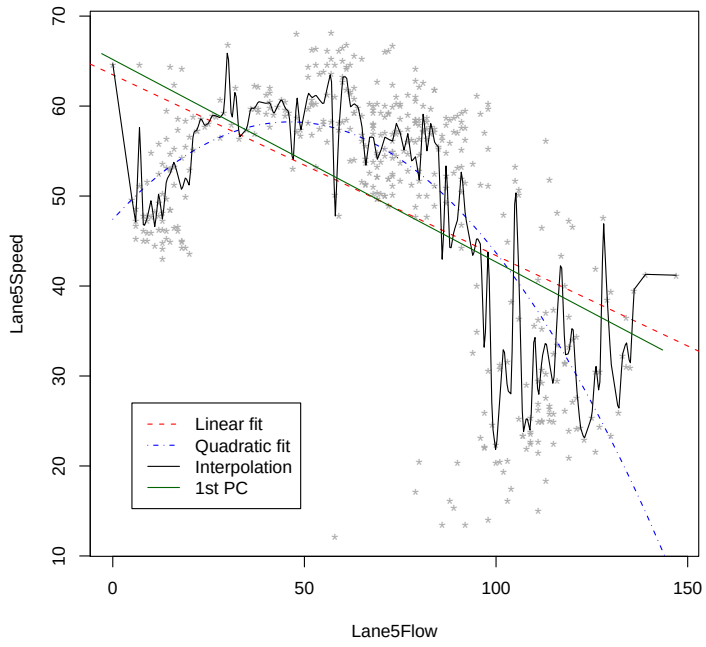

Figure 1.1: Fits for traffic data

This example indicates how the principal components fit can be useful when we need to represent data in smaller dimensions. The two-dimensional data example can be extended to more complicated multi-dimensional data sets, but visualising the data and principal components will be much more difficult.

[(1)] observations of speed and flow are recorded from 9th of July 2007, 9am, to 10th of July 2007, 10pm, on Line 5 of the Californian Freeway SR57-N, VDS number 1202263. The data were originally measured in intervals of thirty seconds, and then aggregated over intervals of 5 minutes length.

In many practical situations, data cannot be well represented by traditional methods such as regression line and interpolation. From Figure 1.1, it is clear that none of the traditional methods used to represent traffic data example gives an adequate fit for the data, and this problem is expected to be more complicated in more complex data sets. The basic problem of the three supervised methods illustrated in Figure 1.1 is that they use an asymmetric view on the variables, implying that each x (flow) will be associated with exactly one estimated y (speed), which is clearly inadequate here.

Although principal components can detect basic patterns in the data set, it is desirable to find a good graphical representation for the data. A group of non-traditional approaches that proves to be more efficient in the majority of these complex situations are methods based on what is called "Principal Curves", an extension to principal component analysis. Principal curves can provide a flexible and not too complicated way to live with high-dimensionality without being a serious obstacle to effectively explore and analyse complex data structures.

One of the main advantages of principal curves based algorithms is the ability to extract information from multidimensional data in one dimension only through the fitted curve. Figure 1.2 displays an example of a principal curve fitted to a three-dimensional life-insurance industry main indicators data. The data consists of three variables; gross claims paid (GCP), number of employees (Emp), total capital and reserves (TCR). The curve is clearly passing through the middle of the data and can provide a good summary for the three variables.

Principal curves shall be introduced and further illustrated in section 2.2 of the coming chapter.

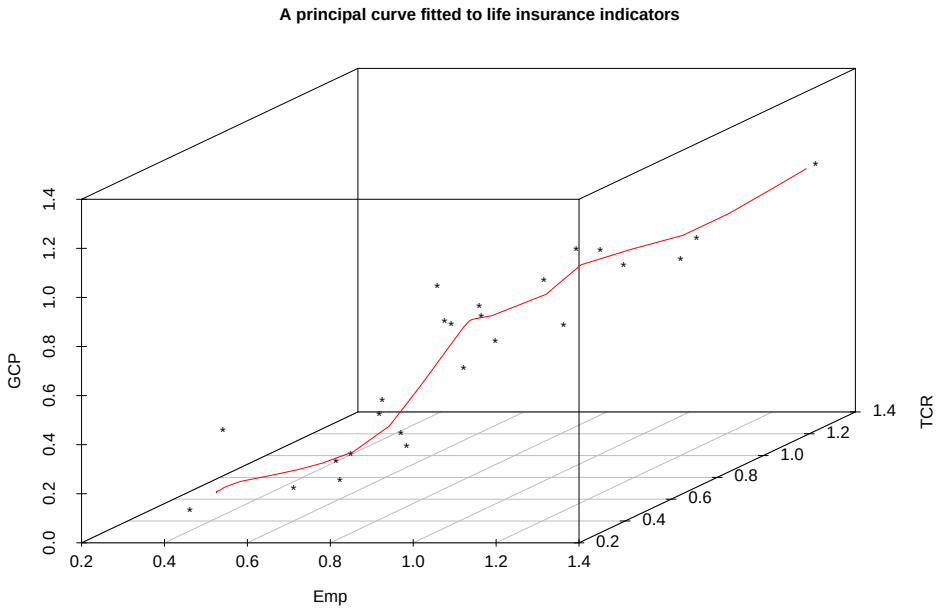

Figure 1.2: A Principal Curve fitted to multi-dimensional data

1.2 Outline of the Thesis

The flow of the topics presented in the work is as follows:

In Chapter 2, we provide a short introduction to principal components and curves viewing the main properties of both. We specially highlight the characteristics of principal components related to explaining the total variance of the data as a whole and of single variables. Principal curves are introduced as a non-parametric alternative to principal components for summarising and extracting features from symmetric data. Also, some alternative approaches and advances for principal curves are briefly reviewed.

The localised versions of linear principal components and principal curves are introduced in Chapter 3. Contributing to the literature, in that chapter, we also

explore the asymptotic behaviour of localised principal components (for high dimensions and large samples) highlighting some of the asymptotics for the method. Local principal curves, a 'bottom-up' strategy for fitting principal curves are introduced. Some of the main features and technicalities for the latter are presented. We also point out some possible improvements to the local principal curves' algorithm.

In Chapter 4, the concept of mean shift, as a mode detection tool, is introduced and the convergence of the mean shift algorithm is discussed. Building upon the asymptotics of localised principal components, some asymptotics for the method are introduced. We study the asymptotics of local principal curves which gives a more clear idea about the curve path around boundary areas. Most importantly, some developments for local principal curves based upon the mean shift algorithm are proposed. In particular, we show how the mean shift algorithm can work as an automatic starting point selection tool. Last, based upon the asymptotic expected behaviour of local principal curve at boundaries, we propose a way of extending the reach of the fitted curve into boundary areas.

In Chapter 5, we introduce some possible applications for local principal curves in the fields of insurance and econometrics, specially for data of time series character. We first explore the application of principal components and local principal curves as a summary performance or efficiency measure for insurance markets. A second example of such applications is what is well known in economy as 'Phillips Curves', where the comparison between fitted curves is highlighted as well as the possible link between time as a latent variable and the curve parametrisation which is a key block for predictions in data of time series character. Another

application for local principal curves was about estimating a curve to model the relationship between gold and currency exchange rates, a case in which time can be integrated as a third dimension to improve the fit. Last, the possible use of the curve parametrisation as a summary index is proposed. Local principal curves are used to construct a global price index using two or more sub-indices.

Main findings and conclusions as well as highlights of possible future research are outlined in Chapter 6.

At the end of this work, some important mathematical results and justifications related to the theory developed in this thesis are separately displayed in the Appendix.

Chapter 2

Principal Components and Curves

2.1 Principal Components

2.1.1 Introduction

The term 'principal components' was first introduced in statistics literature in 1901 by Karl Pearson [58]. It is a procedure that involves linearly transforming a number of possibly correlated variables to a group of uncorrelated variables (principal components) less in number. In this sense, principal component analysis is a way of restructuring data by reducing its dimensionality retaining most of its variation (information) and also it is a means to identify new underlying, and possibly meaningful, variables.

Principal component analysis can be mathematically defined as an orthogonal

linear transformation that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on [42].

In practice, principal components can provide a good representation of data in much less dimensions without being significantly affected by any loss of information, that is by capturing most of the variation in the data. Principal components in that sense provide information about the main directions of variance in the data. Extracting principal components typically starts with computing the covariance (or correlation) matrix and then finding the eigenvectors and eigenvalues of the covariance matrix and sorting them according to decreasing eigenvalue to get the principal components in order from largest to smallest.

Let $\boldsymbol{X} = (X_1, \cdots, X_d) \sim (\boldsymbol{\mu}, \boldsymbol{\Sigma}) : S \longrightarrow T \in \mathbb{R}^d$ be a multivariate random vector, with mean $\boldsymbol{\mu}$ and variance matrix $\boldsymbol{\Sigma}$, which maps elements from a sample space S into a subset T of $\mathbb{R}^d$. (The sample space S may be considered as latent and does not play a role henceforth). Suppose that we have a random sample of n independent replicates for each variable $(X_j)_{j=1,\ldots,d}$ in the random vector $\boldsymbol{X}$, then we have a data set that can be represented as an $(n \times d)$ matrix $\boldsymbol{X}^* = (x_{ij})_{1 \leq i \leq n, \, , 1 \leq j \leq d}$.

Typically, obtaining the sample principal components is done through the following steps:

1. First, subtract the mean from each of the data dimensions; this gives an $(n \times d)$ matrix, $\boldsymbol{U}$, with $(i, j)th$ element $(x_{ij} - \bar{X}_j)$.

2. Calculate the sample covariance matrix of $\boldsymbol{U}$, $\boldsymbol{C} = \frac{1}{n-1}\boldsymbol{U}^T\boldsymbol{U}$.

3. Calculate the eigenvalues and eigenvectors of $\boldsymbol{C}$ (let $\boldsymbol{V}$ denote the $(d \times d)$ matrix of eigenvectors).

4. Order the eigenvectors by eigenvalues highest to lowest and choose a set of significant eigenvectors. By doing this, the dimension of the eigenvectors matrix' is reduced to $(d \times v)$ instead of a $(d \times d)$, where v is the number of eigenvectors chosen. Denote the matrix containing the chosen set of eigenvectors by $\boldsymbol{V}^*$.

5. Finally, the required principal components' scores, $\boldsymbol{Z}^*$ are computed as:

$$\boldsymbol{Z}^* = \boldsymbol{V}^{*T}\boldsymbol{U}^T$$

The eigenvalues of $\boldsymbol{\Sigma}$, $\lambda = \lambda_1 \geq \cdots \geq \lambda_d$, are the roots of

$$|\boldsymbol{\Sigma} - \lambda \boldsymbol{I}| = 0$$

where $\quad \sum_{i=1}^{n} \lambda_i = \mathrm{tr}(\boldsymbol{\Sigma}) \quad , \quad \prod_{i=1}^{n} \lambda_i = \det(\boldsymbol{\Sigma})$

and the eigenvectors of $\boldsymbol{\Sigma}$, $\boldsymbol{\gamma} = \boldsymbol{\gamma}_1 \geq \cdots \geq \boldsymbol{\gamma}_d$, are the normalised eigenvectors satisfying

$$\boldsymbol{\Sigma}\,\boldsymbol{\gamma}_i = \lambda_i\,\boldsymbol{\gamma}_i$$

$$\boldsymbol{\gamma}_i^T\boldsymbol{\gamma}_j = \begin{cases} 1 & : i = j \\ 0 & : i \neq j \end{cases}$$

Principal components (principal component lines) are computed in a way such that the sum of squared *orthogonal* distances between data and their projections

on the line is a minimum. In other words, the principal component line minimises the sum of squared errors in all the variables [37].

Now, we shall use a distance-minimisation-based approach to derive principal components. This is not the standard way of deriving principal components, which is done through maximising the total variance, but we are using this approach here to be consistent with the related material that will be introduced in the coming chapter.

Consider any linear combination of the random vector $\boldsymbol{X}$, say $\boldsymbol{g}(t) = \boldsymbol{m}+t\boldsymbol{\gamma} \in \mathbb{R}^d$, with $t \in \mathbb{R}$ and suitable vectors $\boldsymbol{m}$ and $\boldsymbol{\gamma}$, denote the coordinate of $\boldsymbol{X}$ projected orthogonally onto $\boldsymbol{g}$ by $\boldsymbol{X}^{\boldsymbol{g}}$, where

$$\begin{aligned} \boldsymbol{X}^{\boldsymbol{g}} &= \boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T(\boldsymbol{X} - \boldsymbol{m}) = (\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T)\boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{X} \\ &\equiv \boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{X} \end{aligned} \tag{2.1}$$

The matrix $\boldsymbol{A}_{\boldsymbol{\gamma}} = (\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T)$ is positive semi-definite, which is evident by noting that $\boldsymbol{A}_{\boldsymbol{\gamma}}^T\boldsymbol{A}_{\boldsymbol{\gamma}} = \boldsymbol{A}_{\boldsymbol{\gamma}}$, and hence $||\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{u}||^2 = \boldsymbol{u}^T\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{u}$, for $\boldsymbol{u} \in \mathbb{R}^d$ (see Appendix for details).

Now, for all data points $\boldsymbol{x}_i$, we find $\boldsymbol{m}$ and $\boldsymbol{\gamma}$ such that the line $\boldsymbol{g}$ minimises the squared distances between the data and their projections $\boldsymbol{x}_i^{\boldsymbol{g}} = \boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{x}_i$. Normalising the linear combination [2] by setting $\boldsymbol{\gamma}^T\boldsymbol{\gamma} = 1$, the expression to minimise is

$$\begin{aligned} Q(\boldsymbol{m}, \boldsymbol{\gamma}) &= \sum_{i=1}^{n} ||\boldsymbol{x}_i - \boldsymbol{x}_i^g||^2 - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \\ &= \sum_{i=1}^{n} ||\boldsymbol{x}_i - \boldsymbol{A}_{\boldsymbol{\gamma}}m - \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{x}_i||^2 - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \end{aligned} \tag{2.2}$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} ||\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i - \boldsymbol{m})||^2 - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \\
&= \sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{m})^T \boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i - \boldsymbol{m}) - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \qquad (2.3)
\end{aligned}
$$

where λ is a Lagrange multiplier.

Minimising $Q(\boldsymbol{m}, \boldsymbol{\gamma})$ for $\boldsymbol{\gamma}$, we get (Note that $\frac{\partial}{\partial \boldsymbol{\gamma}} \boldsymbol{u}^T \boldsymbol{A}_{\boldsymbol{\gamma}} \boldsymbol{u} = -2(\boldsymbol{u}\boldsymbol{u}^T)\boldsymbol{\gamma}$)

$$
\begin{aligned}
\frac{\partial Q(\boldsymbol{m}, \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}} &= \sum_{i=1}^{n} \frac{\partial}{\partial \boldsymbol{\gamma}} (\boldsymbol{x}_i - \boldsymbol{m})^T \boldsymbol{A}_{\boldsymbol{\gamma}} (\boldsymbol{x}_i - \boldsymbol{m}) - \lambda \frac{\partial}{\partial \boldsymbol{\gamma}} (\boldsymbol{\gamma}^T \boldsymbol{\gamma}) \\
&= \sum_{i=1}^{n} \left[-2 \, (\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^T \boldsymbol{\gamma} \right] - 2\lambda\boldsymbol{\gamma} \\
&= -2 \left[\sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^T \boldsymbol{\gamma} \; + \; \lambda\boldsymbol{\gamma} \right] \qquad (2.4)
\end{aligned}
$$

and setting this equal to zero yields

$$
\begin{aligned}
\left[\sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^T \right] \boldsymbol{\gamma} &= -\lambda\boldsymbol{\gamma}; \\
\hat{\boldsymbol{\Sigma}}\boldsymbol{\gamma} &= -\lambda\boldsymbol{\gamma}; \qquad (2.5)
\end{aligned}
$$

so that $\boldsymbol{\gamma}$ needs to be an eigenvector of $\hat{\boldsymbol{\Sigma}}$, which is the unbiased estimator of the covariance matrix $\boldsymbol{\Sigma}$.

Multiplying both sides of (2.5) by $\boldsymbol{\gamma}^T$, we get

$$
\boldsymbol{\gamma}^T \boldsymbol{\Sigma} \boldsymbol{\gamma} = \lambda \boldsymbol{\gamma}^T \boldsymbol{\gamma} = \lambda, \qquad (2.6)
$$

so that

$$
\mathrm{Var}(\boldsymbol{\gamma}^T \boldsymbol{X}) = \boldsymbol{\gamma}^T \mathrm{Var}(\boldsymbol{X}) \boldsymbol{\gamma} = \boldsymbol{\gamma}^T \boldsymbol{\Sigma} \boldsymbol{\gamma}. \qquad (2.7)
$$

Hence $\boldsymbol{\gamma}$ needs to be the eigenvector corresponding to the largest eigenvalue of $\boldsymbol{\Sigma}$ such that the total variance accounted for is maximised.

Now, minimising $Q(\boldsymbol{m}, \boldsymbol{\gamma})$ for $\boldsymbol{m}$ yields

$$\begin{aligned}\frac{\partial Q(\boldsymbol{m}, \boldsymbol{\gamma})}{\partial \boldsymbol{m}} &= \sum_{i=1}^{n} \frac{\partial}{\partial \boldsymbol{m}} (\boldsymbol{x}_i - \boldsymbol{m})^T \boldsymbol{A}_{\boldsymbol{\gamma}} (\boldsymbol{x}_i - \boldsymbol{m}) - \lambda \frac{\partial}{\partial \boldsymbol{m}} (\boldsymbol{\gamma}^T \boldsymbol{\gamma}) \\ &= -2 \sum_{i=1}^{n} \boldsymbol{A}_{\boldsymbol{\gamma}} (\boldsymbol{x}_i - \boldsymbol{m}) \end{aligned} \tag{2.8}$$

which, when equated to zero, leads to

$$\begin{aligned} -2 \sum_{i=1}^{n} \boldsymbol{A}_{\boldsymbol{\gamma}} (\boldsymbol{x}_i - \boldsymbol{m}) &= 0 \\ \sum_{i=1}^{n} (\boldsymbol{A}_{\boldsymbol{\gamma}} \boldsymbol{x}_i - \boldsymbol{A}_{\boldsymbol{\gamma}} \boldsymbol{m}) &= 0 \\ \boldsymbol{A}_{\boldsymbol{\gamma}} \sum_{i=1}^{n} \boldsymbol{x}_i &= \boldsymbol{A}_{\boldsymbol{\gamma}} \sum_{i=1}^{n} \boldsymbol{m} \end{aligned}$$

The previous expression will generally have more than one solution. One of them is such

$$\sum_{i=1}^{n} \boldsymbol{x}_i = n\boldsymbol{m}$$

Then,

$$\boldsymbol{m} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \tag{2.9}$$

Hence, the "best" value of $\boldsymbol{m}$ is an estimate of the global mean ($\boldsymbol{\mu}$).

Denoting the largest eigenvector of $\boldsymbol{\Sigma}$ by $\boldsymbol{\gamma}_{(1)}$, we can summarise that the line that minimises the weighted squared distances between data points and their

projected counterparts is given by

$$\boldsymbol{g}(t) = \boldsymbol{\mu} + t\boldsymbol{\gamma}_{(1)},$$

i.e. a line through the global mean in the direction of the first eigenvector of the covariance matrix, which is the global first principal component line. In other words, the *global* first principal component line would be that line through the data cloud which minimises the expected squared distances between data and their projections onto the line.

In general, the j^{th} principal component line is

$$\boldsymbol{g}_{(j)}(t) = \boldsymbol{\mu} + t\boldsymbol{\gamma}_{(j)} \tag{2.10}$$

where $\boldsymbol{\gamma}_{(j)}$ is the j^{th} eigenvector of $\boldsymbol{\Sigma}$.

2.1.2 Properties of Principal Components

Let $\boldsymbol{U}$ be the data matrix after subtracting the means and let $\boldsymbol{V}$ be the matrix of eigenvectors of $\boldsymbol{\Sigma}$. Denote by $\boldsymbol{Z}$ the principal components' scores, then the principal components can be represented as $\boldsymbol{Z} = \boldsymbol{V}^T\boldsymbol{U}^T$ and the k^{th} principal component is

$$\boldsymbol{z}_k = \boldsymbol{\gamma}_k\boldsymbol{U}^T \tag{2.11}$$

where $\boldsymbol{\gamma}_k$ is the k^{th} row of the matrix $\boldsymbol{V}$ (the k^{th} eigenvector), which can be referred to as the vector of coefficients $\gamma_{kj}(j = 1, \cdots, d)$ for the k^{th} principal component. The coefficient γ_{kj} measures the contribution of the j^{th} variable

towards the k^{th} principal component.

Those coefficients represent an important quantity of interest that helps interpreting principal components analysis (hereafter: **PCA**) results. This quantity can be considered a measure of the association between a component and a variable which is in some form an estimate for the information they share. This is called '*loading*'. Loadings (coefficients of association) provide very useful interpretations for the PCA in many applications.

When the eigenvectors are standardised to have length 1 (i.e. $\boldsymbol{\gamma}_k^T \boldsymbol{\gamma}_k = 1$), which is usually the case, the sum of squared loadings over each vector $\boldsymbol{\gamma}_k$ is equal to one. In this case, it follows directly from (2.7) that the variance of the k^{th} principal component is given by:

$$Var(\boldsymbol{z}_k) = \lambda_k \tag{2.12}$$

and the variance of a principal component is greater than or equal to the variance of any proceeding component, i.e.

$$\text{Var}(\boldsymbol{z}_i) \geq \text{Var}(\boldsymbol{z}_j), \quad i < j$$

The percentage of variance accounted for by the first m principal components is

$$100 \times \frac{\sum_{k=1}^{m} \lambda_k}{\sum_{j=1}^{d} \lambda_j} \tag{2.13}$$

where $\sum_{j=1}^{d} \lambda_j$ is the total variance.

Furthermore, the variance of any variable X_j can be expressed in terms of the eigenvalues and eigenvectors as follows

$$Var(X_j) = \sum_{k=1}^{d} \lambda_k \boldsymbol{\gamma}_{kj}^2 \tag{2.14}$$

Other properties of principal components include [47, 50]

- A principal component is centred around the origin, i.e. $E(\boldsymbol{z}_k) = 0$
- Principal components are orthogonal. This implies that

$$\text{Cov}(\boldsymbol{z}_i, \boldsymbol{z}_j) = 0, \quad i \neq j$$

- The variance of the first principal component, $\text{Var}(\boldsymbol{z}_1)$, is greater than or equal to the variance of any standardised linear combination of the data.
- Principal components are not scale-invariant. In other words, changing the scale of data would lead to different principal components' scores $\boldsymbol{Z}$.
- The number of principal components needed to entirely explain the total variation of data is equal to the rank of the covariance matrix $\boldsymbol{\Sigma}$. Hence, if $\text{rank}(\boldsymbol{\Sigma}) = r < d$, then the first r components can entirely explain the total variance.
- For any p-dimensional subspace of the data, S_p , the subspace of the first p principal components has a smaller mean square deviation from the data than S_p.

2.1.3 Visualising Principal Components - an example

It could be useful to see an example of how principal components can give a good representation of the data set in the sense that it reduces dimensionality retaining most of the information (variation) within the data. For illustration purposes, we choose a two-dimensional data set (this can be extended to more than two dimensions, but would require more advanced graphical tools). The data used are the speed-flow data referred to earlier in this chapter. To try visualising the first two principal components for this data, it was loaded and processed in R (refer to Appendix for sample R code to do this), yielding the graph that appears in Figure 2.1.

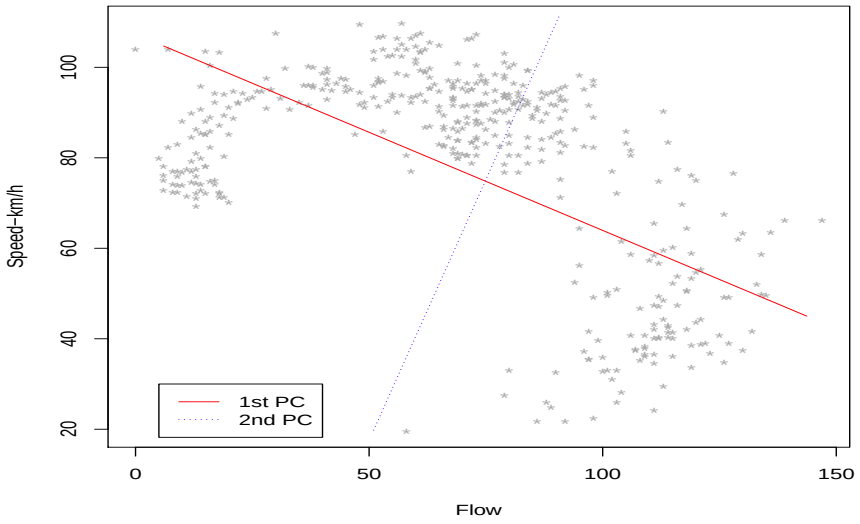

Figure 2.1: Principal Components Graph - an example

It can be seen from Figure 2.1 that the first principal component captures the basic pattern (direction of largest variation) in the data set. Using a simple calculation (see Appendix), the first principal component accounts for 90.7% of the total variance. The second principal component shows another possible,

but not basic, direction that data points spread around. This simple example indicates how useful can the principal components be in summarising data using fewer dimensions. Also, in this sense, principal components can provide a basis for finding intrinsic dimensionality of data [34].

Although principal components can give a basic indication of main patterns in the data set, in many practical situations the linearised way of summarising data through principal components or other traditional methods such as regression and interpolation may not be ideal. There exist non-linear equivalents for principal components. Those are basically methods based upon principal curves.

2.2 Principal Curves

The concept of Principal Curves (hereafter: **PCs**) was firstly brought into the Statistics literature in 1989 by Hastie and Stuetzle [38] (hereafter: HS). As representing multi-dimensional data in fewer dimensions through principal components is an appealing idea, HS suggested a summary technique that goes further and represent the data by a smooth one-dimensional curve that passes through the middle of the data cloud. In fact, their work was based on the idea of capturing the basic and most significant pattern in the data through the largest (first) principal component, then try to modify the straight line representing the first principal component and produce a smooth curve representation instead of a straight line. This implies that principal curves, like linear principal components, focus on the orthogonal or shortest distances to the points, and is fit in a way that minimises these orthogonal distances. In this sense, a principal curve can be considered a non-linear equivalent to a globally fitted principal component line.

2.2.1 The HS Approach

According to HS, a principal curve is defined as 'a smooth one-dimensional curve that passes through the middle of a multi-dimensional data set, providing a non-linear summary of the data'. This one-dimensional curve can be thought as a curve that is parametrised over some parameter τ.

HS stated that, for a given data set X, a one-dimensional curve, $\boldsymbol{\nu}$, is said to be a principal curve if the following hold:

(i) The curve does not intersect itself.

(ii) The curve has finite length inside any bounded subset of $\mathbb{R}^d$.

(iii) The curve is self consistent.

Denote by $\boldsymbol{\nu}$ a smooth (C^∞) unit-speed curve[(1)] in $\mathbb{R}^d$ that does not intersect itself and that is parametrised over a closed interval $T \subseteq \mathbb{R}^1$ and has finite length inside any finite ball in $\mathbb{R}^d$. HS have defined a projection index $\tau_{\boldsymbol{\nu}} : \mathbb{R}^d \to \mathbb{R}^1$ as follows:

$$\tau_{\boldsymbol{\nu}}(\boldsymbol{x}) = \sup_{\tau}\{\tau : \|\boldsymbol{x} - \boldsymbol{\nu}(\tau)\| = \inf_{\mu} \|\boldsymbol{x} - \boldsymbol{\nu}(\mu)\|\} \tag{2.15}$$

meaning that the projection index of any point, $\boldsymbol{x}$ is the largest value of τ which minimises the distance between this point and $\boldsymbol{\nu}(\tau)$.

The main property of principal curves is that they are self-consistent [73] for a particular distribution or data set. This means that any point on the curve is the average of all points that project there. A curve, $\boldsymbol{\nu}$, is said to be self-consistent or a principal curve if the following hold for a.e. τ:

$$E(X|\tau_{\boldsymbol{\nu}}(X) = \tau) = \boldsymbol{\nu}(\tau) \tag{2.16}$$

[(1)] A curve $\boldsymbol{\nu}$ is said to be a unit speed parametrised curve if $\|\boldsymbol{\nu}'\| \equiv 1$.

HS have also showed that principal curves, analogue to principal components, are critical points of the expected squared distance from the points to their projections on the curve. HS defined a distance function that is based upon the Euclidean distance from a point $\boldsymbol{x}$ to its projection on the curve $\boldsymbol{\nu}$ and this function is used to optimise the curve.

Let $d(\boldsymbol{x}, \boldsymbol{\nu})$ be the Euclidean distance from a point $\boldsymbol{x}$ to its projection on the curve

$$d(\boldsymbol{x}, \boldsymbol{\nu}) \equiv \|\boldsymbol{x} - \boldsymbol{\nu}(\tau_{\boldsymbol{\nu}}(\boldsymbol{x})\| \tag{2.17}$$

Define the expected squared distance function $D^2(\boldsymbol{\nu}) \equiv E(d^2(X, \boldsymbol{\nu}))$, if we consider any two straight lines $\boldsymbol{\nu}$ and $\boldsymbol{g}$, then $\boldsymbol{\nu}$ is said to be a critical value of D^2 iff

$$\left.\frac{\partial D^2(\boldsymbol{\nu} + \epsilon \boldsymbol{g})}{\partial \epsilon}\right|_{\epsilon=0} = 0$$

If we assume $\boldsymbol{\nu} \in \iota$ and $\boldsymbol{g} \in \iota$, where ι is the class of differentiable one-dimensional curves in $\mathbb{R}^d$ parametrised by τ, then principal curves have the property of being critical points of the distance function as well [38].

Based on this, the algorithm for finding principal curves usually starts with the straight line representing the largest principal component, then it ensures that the curve is self-consistent by projecting and averaging at each point, and this process is repeated until the value of the expected distance function reaches a specific threshold. In other words, the principal curve is fit in a way such that the average squared distance of the data points and the curve is minimised.

The HS algorithm for finding a principal curve is as follows[(1)]:

(i) Perform a principal component analysis and extract the first principal com-

[(1)] The HS algorithm was originally implemented in SPlus.

ponent. Let $\boldsymbol{\nu}_0(\tau)$ be the first principal component line for the data X. Set $j = 0$.

(ii) Find the data projections on $\boldsymbol{\nu}_0(\tau)$. The projections set is defined as $\tau_{\boldsymbol{\nu}_j}(\boldsymbol{x}) = \max\{\tau : \|\boldsymbol{x} - \boldsymbol{\nu}(\tau)\| = \min_{.} \|\boldsymbol{x} - \boldsymbol{\nu}(.)\|\} \ \forall \boldsymbol{x} \in \mathbb{R}^d$.

(iii) Compute the average of data points and check for self-consistency. Define $\boldsymbol{\nu}_{(j+1)}(\tau) = E[X|\tau_{\boldsymbol{\nu}_{(j)}}(X) = \tau]$.

(iv) Let ϱ denote a pre-defined threshold for the algorithm to stop. If the quantity $\left|1 - \frac{\Delta(\boldsymbol{\nu}_{(j+1)})}{\Delta(\boldsymbol{\nu}_{(j)})}\right|$ falls below ϱ stop the iteration, otherwise, let $j = j + 1$ and go to step (ii).

HS have not explicitly shown the convergence of the algorithm, though they suggested that there is evidence that the state of convergence is expected to be reached. Factors supporting this latter assumption were [46]:

1. HS principal curves are, by definition, fixed points of the algorithm.
2. The expected squared distance converges, as long as each iteration is well defined and produces a differentiable curve.
3. If the fitted principal curve happens to be a straight line, then it is a principal component [38]. Moreover, Kégl [46] has mentioned that, if the second step in the algorithm (projection) is replaced by the fitted least squares straight line, the procedure converges to the largest principal component. We think that the latter statement can only be true if the regression line replaces the third, not the second, step of the algorithm (calculating expectations).

The convergence of the distance function does not necessarily lead to that the fitted principal curve converges [46]. All principal curves are saddle points of

the distance function [20]. Furthermore, the largest principal component minimises the distance function and the smallest principal component maximises it. Without restricting the set of admissible curves, the distance function will fail to converge to a stable solution [46].

Principal curves can be considered as a nonparametric extension to linear principal components which is of interest mainly in the cases where the variables under consideration, the values of which formulate the data cloud, are considered to be symmetric, rather than one or more variable being dependent upon or generated from the remaining ones.

Figure 2.2 shows the HS principal curve fitted to the two-dimensional flow-speed data introduced earlier in this current chapter[(1)]. Of course, this is not likely to be the best non-linear fit for this data, but it is still expected to be better than all linear-based fits.

Principal curves have found their way to a variety of statistical applications and practical multi-dimensional data situations. Some areas of current applications of principal curves include [46]:

- Image processing and feature extraction [4, 44].
- Clustering [71].
- Speech processing [63, 64].
- Process monitoring [19, 82]

[(1)]This HS principal curve is obtained through the *princurve* package in R [39] by using the function *principal.curve()* with its default settings. For more details, please refer to the 'princurve' package help via http://cran.r-project.org/web/packages/princurve/princurve.pdf

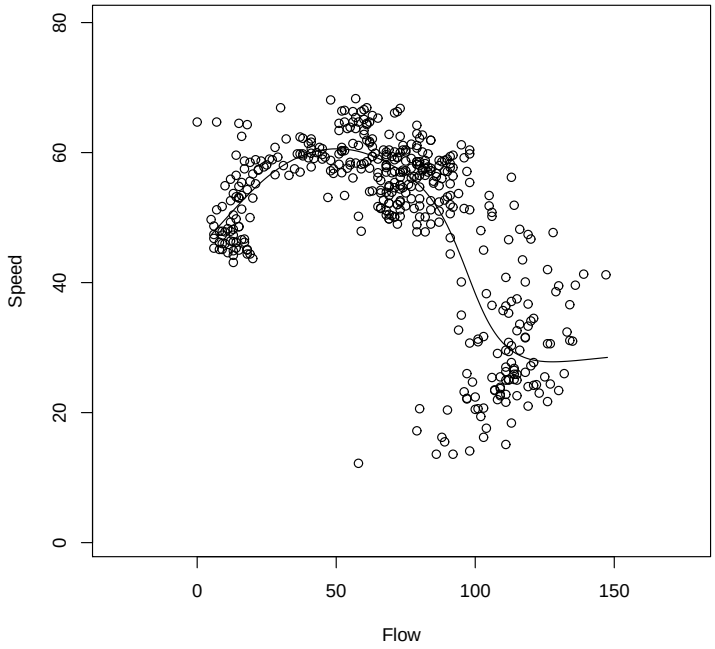

Figure 2.2: HS Principal Curve for traffic data

2.2.2 Alternative Principal Curve Algorithms

Since 1989, there have been several developments to the original principal curve algorithm suggested by Hastie and Stuetzle. The HS approach was mainly based on the self-consistency concept. It also uses two different smoothing techniques to avoid over-fitting [38].

Tibshirani [75] introduced an alternative definition for principal curves, based on a mixture model. Estimation was carried out focusing on one-dimensional curves through an EM algorithm. Banfield et al. [4] have extended the HS algorithm to closed curves and suggested that the new extension reduces both bias and variance of the estimation. Kégl et al. [45] have represented principal curves as polygonal lines. They have shown that the suggested polygonal line algorithm, due to its adaptive way of smoothing, is more robust than the HS algorithm.

Also, the former approach of fitting principal curves reduces the estimation bias by minimising the average distance from the curve rather than from the vertices of the curve. Verbeek et al. [77] proposed an incremental method of constructing principal curves. The method depends on forming polygonal lines by connecting line segments (basically representing local principal component lines) and then checking the quality of the resulting line. Although there is no smoothing contained within the algorithm, smoothing can be done through any regression method using the fitted polygonal line as a basis to assign a latent variable value to each datum (implying an ordering on the data).

Further to the global approaches mentioned above, there exist local approaches for fitting principal curves. One of the main privileges of locally-based approaches is accounting for the local topology of the data. Delicado [17] has suggested an alternative method for defining and constructing principal curves based upon an iterative locally-oriented algorithm to find a set points composing the curve. The latter is what is called 'Principal Oriented Points' (POPs). Einbeck et al. [27] proposed an algorithm for fitting principal curves through iteratively performing localised principal component analysis and applying mean-shifts to construct the set of points defining the curve. Ozertem and Erdogmus [56] have proposed an alternative definition for principal curves and surfaces that characterises the curve or surface in terms of the gradient and the Hessian of the density estimate. It was pointed out that, compared to the traditional methods for manifold learning, the approach adapted defines the underlying manifold from a more differential geometric point of view.

In the next chapter, one of the local approaches for fitting principal curves, in particular, the local principal curve approach introduced by Einbeck et al. [27] shall be further illustrated.

Last, it is worth pointing to another important development for the idea of principal curves which is known as 'principal surfaces' or manifolds. Mostly, those manifolds are represented using multidimensional base functions. Different approaches vary according to the set of base functions applied and the approach of optimising the parameters.

LeBlanc and Tibshirani [48] have introduced an adaptive way of constructing a principal surface of the data optimising linear base functions through multivariate adaptive regression splines. Dong and McAvoy [19] have integrated the HS principal curve algorithm and neural networks using the conjugate gradient method for optimising the parameters. Smola et al. [70] have used a constrained quantisation approach in the context of unsupervised learning and showed that the proposed work can be closely linked to length-constrained principal curves. They have used a minimisation-oriented iteration to optimise Gaussian kernels. The authors suggested that the used approach can be looked at as a link between principal curves and surfaces and generative topographic mapping [5]. Der et al. [18] have applied the self-organising map algorithm to extract principal curves and manifolds from data and shown experimentally the applicability of the suggested approach using noisy data. Chang and Ghosh [9, 10] have introduced the concept of 'Probabilistic Principal Surfaces' (PPS). Extending the basic concept that uses manifold oriented covariance noise model, they have also proposed a parametric model based on minimising PPS-reconstruction-error. Einbeck and Evers [22] have presented local principal manifolds as a nonparametric data reduction approach for modelling data with low-dimensional non-linear latent structure. This latent structure is used to define new data-dependent topologies. The proposed approach was exploited for regression problems. The authors also suggested that the method can be applied for classification or density estimation on the manifold.

Chapter 3

Local Principal Components and Curves

3.1 Local Principal Components

3.1.1 Introduction

Let $\boldsymbol{X} \sim (\boldsymbol{\mu}, \boldsymbol{\Sigma}) : A \longrightarrow S \in \mathbb{R}^d$ be a multivariate random vector, with mean $\boldsymbol{\mu}$ and variance matrix $\boldsymbol{\Sigma}$, which maps elements from a sample space A into a subset S of $\mathbb{R}^d$. (The sample space A may be considered as latent and does not play a role henceforth.) The *global* first principal component line would be that line through the data cloud which minimises the expected squared distances between data and their projections onto the line.

It is well known that the solution to this problem is the line through $\boldsymbol{\mu}$ which points into the direction of the eigenvector $\boldsymbol{\gamma}_1$ of $\boldsymbol{\Sigma}$ corresponding to the largest eigenvalue λ_1 of $\boldsymbol{\Sigma}$. Turning from the probabilistic to the empirical setting, i.e.

given n independent replicates of $\boldsymbol{X}$, say $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in S$, then $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ need to be replaced by consistent estimators, for instance the ML estimators $\hat{\mu} = \bar{\boldsymbol{x}}$ and $\hat{\boldsymbol{\Sigma}} = \frac{1}{n-1}(\boldsymbol{x}_i - \bar{\boldsymbol{x}})(\boldsymbol{x}_i - \bar{\boldsymbol{x}})^T$.

This concept is straightforwardly extended to a scenario in which, given a (non-random) vector $\boldsymbol{x} \in \mathbb{R}^d$, and weights $w^{\boldsymbol{x}}(\boldsymbol{x}_i)$ centred at $\boldsymbol{x}$, we aim to minimise the *weighted* squared distances between data and their projections onto the line. If the weights are of bell–shaped and symmetric shape, their role is effectively to *localise* the estimation problem at $\boldsymbol{x}$. Weight functions of this type are known as kernels, with the prominent example of the Gaussian kernel. As we will verify later, it turns out that, unsurprisingly, the solution to this problem is the line through the locally weighted mean, or short, *local mean*[(1)]

$$\boldsymbol{\mu}^{\boldsymbol{x}} = \frac{\sum_{i=1}^{n} w^{\boldsymbol{x}}(\boldsymbol{x}_i)\boldsymbol{x}_i}{\sum_{i=1}^{n} w^{\boldsymbol{x}}(\boldsymbol{x}_i)} \tag{3.1}$$

which points into the direction of the first eigenvector[(2)] of the *local covariance matrix*

$$\boldsymbol{\Sigma}^{\boldsymbol{x}} = \frac{\sum_{i=1}^{n} w^{\boldsymbol{x}}(\boldsymbol{x}_i)(\boldsymbol{x}_i - \boldsymbol{\mu}^{\boldsymbol{x}})(\boldsymbol{x}_i - \boldsymbol{\mu}^{\boldsymbol{x}})^T}{\sum_{i=1}^{n} w^{\boldsymbol{x}}(\boldsymbol{x}_i)}. \tag{3.2}$$

That is, the "locally weighted" first principal component is given by a vector pointing into the direction which explains most of the "local variance" around $\boldsymbol{x}$, or, in simpler terms, which locally gives the best fit.

[(1)] For denotational convenience, we will from now on omit all 'hats' on symbols denoting estimators – it is clear that $\boldsymbol{\mu}^{\boldsymbol{x}}$ etc. are empirical and not theoretical quantities.

[(2)] When using the term 'first eigenvector', we mean the eigenvector corresponding to the largest eigenvalue.

Let $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d)^T \in \mathbb{R}^d$ and $\kappa(\cdot)$ be a bounded symmetric uni-variate function which integrates to 1 (we do not strictly require it to be non-negative, but usually this will be the case).

A d- variate kernel function K can be defined by either:

- A product kernel function, $K(\boldsymbol{x}) = \kappa(\boldsymbol{x}_1) \times \ldots \times \kappa(\boldsymbol{x}_d)$, or
- A radial kernel function, $K(\boldsymbol{x}) = \kappa(||\boldsymbol{x}||)$.

The two formulations for $K(\boldsymbol{x})$ are equivalent if the base kernel κ is the Gaussian probability density function,

$$\kappa(\boldsymbol{x}) = \frac{1}{\sqrt{2\pi}} e^{-\boldsymbol{x}^2/2},$$

and the following applies to either construction of K.

Now, let $\boldsymbol{H} \in \mathbb{R}^{d \times d}$ denote a positive definite bandwidth matrix. If we localise only in the directions of the coordinate axes, then $\boldsymbol{H} = \text{diag}(h_1^2, \ldots, h_d^2)$, where $h_j, j = 1, \ldots, d$, are the individual bandwidths; and if we smooth equally strong in all directions, then $\boldsymbol{H} = h^2\boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix [78]. Then we can define

$$K_{\boldsymbol{H}}(\cdot) = |\boldsymbol{H}|^{-1/2} K\left(\boldsymbol{H}^{-1/2}\,\cdot\right) \tag{3.3}$$

which is a d-variate probability density function in itself.

Given any line in $\mathbb{R}^d$, say $\boldsymbol{g}(t) = \boldsymbol{m} + t\boldsymbol{\gamma} \in \mathbb{R}^d$, with $t \in \mathbb{R}$ and suitable vectors $\boldsymbol{m}$ and $\boldsymbol{\gamma}$, denote the coordinate of $\boldsymbol{X}$ projected orthogonally onto $\boldsymbol{g}$ by $\boldsymbol{X}^{\boldsymbol{g}}$, where

$$\boldsymbol{X}^{\boldsymbol{g}} = \boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T(\boldsymbol{X} - \boldsymbol{m}) = (\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T)\boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{X}$$

$$\equiv \boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{X}$$

The matrix $\boldsymbol{A}_{\boldsymbol{\gamma}} = (\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T)$ is positive semi-definite, which is evident by noting that $\boldsymbol{A}_{\boldsymbol{\gamma}}^T\boldsymbol{A}_{\boldsymbol{\gamma}} = \boldsymbol{A}_{\boldsymbol{\gamma}}$, and hence $||\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{u}||^2 = \boldsymbol{u}^T\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{u}$, for $\boldsymbol{u} \in \mathbb{R}^d$.

Now, at point $\boldsymbol{x}$, we seek to find $\boldsymbol{m}$ and $\boldsymbol{\gamma}$ such that the line $\boldsymbol{g}$ locally minimises the weighted squared distances between the data and their projected counterparts $\boldsymbol{x}_i^{\boldsymbol{g}} = \boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{m} + \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{x}_i$. Restricting $||\boldsymbol{\gamma}|| = 1$, the expression to minimise is

$$\begin{aligned} Q(\boldsymbol{m}, \boldsymbol{\gamma}) &= \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})||\boldsymbol{x}_i - \boldsymbol{x}_i^g||^2 - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \qquad (3.4) \\ &= \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})||\boldsymbol{x}_i - \boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{m} - \gamma\gamma^T\boldsymbol{x}_i||^2 - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \\ &= \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})||\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i - \boldsymbol{m})||^2 - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \\ &= \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})(\boldsymbol{x}_i - \boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i - \boldsymbol{m}) - \lambda(\boldsymbol{\gamma}^T\boldsymbol{\gamma} - 1) \qquad (3.5) \end{aligned}$$

First, we minimise $Q(\boldsymbol{m}, \boldsymbol{\gamma})$ for $\boldsymbol{\gamma}$.

$$\frac{\partial Q(\boldsymbol{m}, \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}} = \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})\frac{\partial}{\partial \boldsymbol{\gamma}}(\boldsymbol{x}_i - \boldsymbol{m})^T(\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T)(\boldsymbol{x}_i - \boldsymbol{m}) - \lambda\frac{\partial}{\partial \boldsymbol{\gamma}}(\boldsymbol{\gamma}^T\boldsymbol{\gamma})$$

Using the fact that $\frac{\partial}{\partial \boldsymbol{\gamma}}\boldsymbol{u}^T\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{u} = -2(\boldsymbol{u}\boldsymbol{u}^T)\boldsymbol{\gamma}$,

$$\begin{aligned} \frac{\partial Q(\boldsymbol{m}, \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}} &= \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})\left[-2\,(\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^T\boldsymbol{\gamma}\right] - 2\lambda\boldsymbol{\gamma} \\ &= -2\left[\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x})(\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^T\boldsymbol{\gamma} \;+\; \lambda\boldsymbol{\gamma}\right] \qquad (3.6) \end{aligned}$$

and setting this equal to zero yields

$$\begin{aligned}\left[\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{m})(\boldsymbol{x}_i-\boldsymbol{m})^T\right]\boldsymbol{\gamma} &= -\lambda\boldsymbol{\gamma};\\ \boldsymbol{\Sigma}^{\boldsymbol{x}}\boldsymbol{\gamma} &= -\lambda\boldsymbol{\gamma};\end{aligned} \tag{3.7}$$

so that $\boldsymbol{\gamma}$ needs to be an eigenvector, typically the largest, of the local variance matrix $\boldsymbol{\Sigma}^{\boldsymbol{x}}$ as defined in (3.2).

Now, minimising $Q(\boldsymbol{m},\boldsymbol{\gamma})$ for $\boldsymbol{m}$, we get

$$\frac{\partial Q(\boldsymbol{m},\boldsymbol{\gamma})}{\partial \boldsymbol{m}} = -2\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m}) \tag{3.8}$$

which, when equated to zero, yields

$$\begin{aligned}-2\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m}) &= 0\\ \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{x}_i-\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{m}) &= 0\\ \boldsymbol{A}_{\boldsymbol{\gamma}}\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})\boldsymbol{x}_i &= \boldsymbol{A}_{\boldsymbol{\gamma}}\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})\boldsymbol{m}.\end{aligned} \tag{3.9}$$

Since $\det(\boldsymbol{A}_{\boldsymbol{\gamma}}) = 0$, the solution to this equation is not unique. In any case, a solution is

$$\boldsymbol{m} = \frac{\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})\boldsymbol{x}_i}{\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})} = \boldsymbol{\mu}^{\boldsymbol{x}}. \tag{3.10}$$

Hence, the "best" value of $\boldsymbol{m}$ is the local mean at $\boldsymbol{x}$ as defined in (3.1).

Denoting the largest eigenvector of $\boldsymbol{\Sigma}^{\boldsymbol{x}}$ by $\boldsymbol{\gamma}^{\boldsymbol{x}}$, we can summarise that, at any given point $\boldsymbol{x}$, the line that locally minimises the weighted squared distances

between the data and their projections is given by

$$\boldsymbol{g}^{\boldsymbol{x}}(t) = \boldsymbol{\mu}^{\boldsymbol{x}} + t\boldsymbol{\gamma}^{\boldsymbol{x}},$$

i.e. a line through the local mean in the direction of the first eigenvector of the local covariance matrix, which is the localised first principal component line.

Localised principal components, in this kernel-weighted sense, have found their way into the statistical literature only relatively recently. Examples of areas of application for locally weighted principal component analysis include:

- Local dimensionality reduction [43, 66].
- Principal curve estimation using kernel-based approaches [27].
- Adaptive tracing of curvilinear structures [81].
- The implementation of geographically weighted principal components [11].
- Tracking the contribution of sub-indices to a summary index over time [86].

3.1.2 Some Asymptotics for Localised Principal Components

When kernels are used as weights, it is of statistical importance to study the 'asymptotic' behaviour of localised principal components. For relatively large samples and small window sizes (bandwidths), this type of analysis may enable producing approximated estimates or expected values for key parameters and quantities of interest in many multivariate data applications.

Although there has been considerable research on kernel-based asymptotics [8]

in a variety of statistical applications, such as curve estimation, dimensionality reduction, tracking changes and landmarks in multi-dimensional data structures and nonparametric regression [28, 55, 83, 85], the asymptotics for kernel-based localised principal components have not been deeply investigated yet. In this section, we will provide some useful approximations of localised principal components which shall be later extended for local principal curves as well.

Recalling the function $Q(\boldsymbol{m}, \boldsymbol{\gamma})$ (3.4), let f denote the density function of $\boldsymbol{X}$ with support $\text{supp}(f) \subset S$. We assume that the following hold [65, 79]:

(A1) The kernel K is a bounded and compactly supported probability density function such that $\int \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u})\, d\boldsymbol{u} = \mu_2(K)\boldsymbol{I}$, with $\mu_2(K) \in \mathbb{R}$, $\mu_2(K) \neq 0$.

(A2) At $\boldsymbol{x} \in \text{supp}(f)$, f is continuously differentiable and $f(\boldsymbol{x}) > 0$.

(A3) The sequence of bandwidth matrices $\boldsymbol{H}$ is such that $n^{-1}|\boldsymbol{H}|^{-1/2}$ and each entry of $\boldsymbol{H}$ tending to zero as $n \longrightarrow \infty$, with $\boldsymbol{H}$ remaining symmetric and positive definite.

Let $o_p(1)$ denotes a sequence which tends to zero in probability as $n \longrightarrow \infty$ [6] and let $\mathbf{1}$ denote a generic matrix of corresponding dimensions having each entry equal to 1.

We now provide the asymptotic versions for the local covariance matrix $\boldsymbol{\Sigma}^{\boldsymbol{x}}$ and its eigenvectors.

In order to derive an asymptotic version of (3.7), first, we try to find an asymptotic version of $Q(\boldsymbol{m}, \boldsymbol{\gamma})$.

Let's consider the expected value of the first term in (3.5),

$$E\left(\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right)$$

For large values of n and small values of $\boldsymbol{H}$ [(1)]

$$\begin{aligned}
&E\left(\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right)\\
&\overset{(3.3)}{=} E\left(|\boldsymbol{H}|^{-1/2}\sum_{i=1}^{n} K(\boldsymbol{H}^{-1/2}(\boldsymbol{x}_i-\boldsymbol{x}))(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right)\\
&= |\boldsymbol{H}|^{-1/2}\sum_{i=1}^{n} E\left(K(\boldsymbol{H}^{-1/2}(\boldsymbol{x}_i-\boldsymbol{x}))(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right)\\
&\overset{\boldsymbol{x}_i\,iid}{=} n|\boldsymbol{H}|^{-1/2}\int K\left(\boldsymbol{H}^{-1/2}(\boldsymbol{s}-\boldsymbol{x})\right)(\boldsymbol{s}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{s}-\boldsymbol{m})f(\boldsymbol{s})\,d\boldsymbol{s}\\
&= n\int K(\boldsymbol{u})(\boldsymbol{H}^{1/2}\boldsymbol{u}+\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{H}^{1/2}\boldsymbol{u}+\boldsymbol{x}-\boldsymbol{m})f(\boldsymbol{x}+\boldsymbol{H}^{1/2}\boldsymbol{u})\,d\boldsymbol{u}\\
&= n\int K(\boldsymbol{u})\left\{(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})+O(\mathbf{1}^T\boldsymbol{H}^{1/2}\boldsymbol{u})\right\}\left\{f(\boldsymbol{x})+O(\mathbf{1}^T\boldsymbol{H}^{1/2}\boldsymbol{u})\right\}\,d\boldsymbol{u}\\
&= n\left[\,f(\boldsymbol{x})(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})+o(1)\,\right], \qquad (3.11)
\end{aligned}$$

where $\int K(\boldsymbol{u})\,d\boldsymbol{u}=1$, and $\mathbf{1}$ is a vector which only consists of 1's.

Similarly, it can be shown that (see Appendix)

$$\mathrm{Var}\left(\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right)=o(n^2),$$

so that, in summary, the following holds

$$\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})=nf(\boldsymbol{x})(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})+o_P(n)$$

[(1)] see Appendix for further illustration.

We arrive at the penalised asymptotic minimisation problem

$$\tilde{Q}(\boldsymbol{m}, \boldsymbol{\gamma}) = nf(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{m})^T \boldsymbol{A}_\gamma (\boldsymbol{x} - \boldsymbol{m}) - \lambda(\boldsymbol{\gamma}^T \boldsymbol{\gamma} - 1). \tag{3.12}$$

Taking the derivative w.r.t. $\boldsymbol{\gamma}$ yields,

$$\frac{\partial Q(\boldsymbol{m}, \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}} = -2 \left[nf(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{m})(\boldsymbol{x} - \boldsymbol{m})^T \boldsymbol{\gamma} + \lambda \boldsymbol{\gamma} \right], \tag{3.13}$$

and equating this to zero,

$$\begin{aligned} nf(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{m})(\boldsymbol{x} - \boldsymbol{m})^T \boldsymbol{\gamma} &= -\lambda \boldsymbol{\gamma} \\ \tilde{\boldsymbol{\Sigma}}^{\boldsymbol{x}} \boldsymbol{\gamma} &= -\lambda \boldsymbol{\gamma}; \end{aligned}$$

i.e. $\boldsymbol{\gamma}$ is eigenvector of $\tilde{\boldsymbol{\Sigma}}^{\boldsymbol{x}} = nf(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{m})(\boldsymbol{x} - \boldsymbol{m})^T$.

Although the matrix $\tilde{\boldsymbol{\Sigma}}^{\boldsymbol{x}}$ is not a typical variance matrix as it is singular, this has no effect as we are only interested in its eigenvectors.

Noting that $\tilde{\boldsymbol{\Sigma}}^{\boldsymbol{x}}$ can be considered as a matrix of type $\boldsymbol{\Sigma} = c\boldsymbol{\psi}\boldsymbol{\psi}^T$ (see Appendix), the only eigenvector of $\tilde{\boldsymbol{\Sigma}}^{\boldsymbol{x}}$ is given by

$$\frac{\boldsymbol{x} - \boldsymbol{m}}{||\boldsymbol{x} - \boldsymbol{m}||}.$$

Taking the derivative of (3.12) w.r.t. $\boldsymbol{m}$ will give the less useful result of $\boldsymbol{m} = \boldsymbol{x}$.

However, using that the local estimate of $\boldsymbol{m}$ is $\boldsymbol{\mu}^{\boldsymbol{x}}$, we can replace $\boldsymbol{x} - \boldsymbol{m}$ by the asymptotic version of $\boldsymbol{x} - \boldsymbol{\mu}^{\boldsymbol{x}}$ as will be shown in Section 4.1.3, where $\boldsymbol{x} - \boldsymbol{\mu}^{\boldsymbol{x}}$ can

be asymptotically approximated by

$$\mu_2(K)\boldsymbol{H}\frac{\nabla f(\boldsymbol{x})}{f(\boldsymbol{x})},$$

yielding the asymptotic version of $\boldsymbol{\gamma}^{\boldsymbol{x}}$,

$$\tilde{\boldsymbol{\gamma}}^{\boldsymbol{x}} \stackrel{a}{=} \frac{-\mu_2(K)\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x})/f(\boldsymbol{x})}{\mu_2(K)||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x})||/f(\boldsymbol{x})} = -\frac{\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x})}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x})||}, \tag{3.14}$$

where the notation $\stackrel{a}{=}$ means that in the expression succeeding this symbol all terms of an asymptotically higher order than the leading term are omitted. This shows that $\tilde{\boldsymbol{\gamma}}^{\boldsymbol{x}}$ always follows the gradient of the density function. This is a useful result that shall be helpful later on in this context.

3.2 Local Principal Curves

3.2.1 The LPC Algorithm

3.2.1.1 Introduction

Based upon the foundations of 'Principal Curves' and the idea of local modelling [29], and considering the situations of multi-dimensional data with symmetric components, Einbeck et al. [27] presented the idea of "Local Principal Curves" (hereafter: **LPCs**), a flexible technique to model the complex data patterns arising in such situations.

Being based on principal component analysis, both local principal curves (LPCs) and principal curves (PCs) are constructed in a way such that the squared orthogonal distances between points and their projections onto the fitted curve are

minimised. The main difference between LPCs and PCs is that the former defines the fitted curves through a set of points which is obtained by iteratively running principal component analysis locally along the data cloud, rather than running the analysis globally then iteratively averaging and projecting.

Among the features of the LPC algorithm that makes it relatively flexible compared to other PC-based algorithms is that there is no distributional (or other) form assumed for the data. This relaxes the assumption of the un-intersected curves of the original HS algorithm, so that the fitted LPC can be of any shape including closed and branched curves. Also, being classified as a 'bottom-up' strategy of fitting principal curves, the local topology of data is well accounted for making it easier to fit more complex data structures. Another example of desirable LPCs features is that the LPC algorithm is not computationally expensive compared to other peer approaches [27].

Some selected computational and technical details for the LPC algorithm are discussed in the coming subsection(s).

3.2.1.2 The Algorithm

The LPC is defined through a series of points which represent local centres of the mass of the data. Connecting those points, we get a smooth curve that passes through the middle of the data cloud.

If we have a d-dimensional data cloud $X = (X_1, \cdots, X_d)^T \in \mathbb{R}^d, i = 1, ..., n$, where $\boldsymbol{x}_i = (x_{i1}, ..., x_{id})^T$, the LPC algorithm works as follows:

Step 1 Selecting a starting point $\boldsymbol{x}_{(o)}$. Set $\boldsymbol{x} = \boldsymbol{x}_{(o)}$.

Step 2 Computing $\boldsymbol{\mu}^{\boldsymbol{x}}$, the local mean (centre of mass) around $\boldsymbol{x}$.

Step 3 Performing a principal component analysis locally at $\boldsymbol{x}$, and storing the first local eigenvector, $\boldsymbol{\gamma}^{\boldsymbol{x}}$.

Step 4 Finding a new value for $\boldsymbol{x}$ by stepping from $\boldsymbol{\mu}^{\boldsymbol{x}}$ into the direction of $\boldsymbol{\gamma}^{\boldsymbol{x}}$.

Step 5 Repeating steps 2 to 4 until the value of $\boldsymbol{\mu}^{\boldsymbol{x}}$ remains approximately constant (convergence).

Step 6 Having reached convergence, the local principal curve is then constructed through smoothly connecting the values of the $\boldsymbol{\mu}^{\boldsymbol{x}}$ series.

In *Step 4* above, the step size needs to be specified by the data analyst, and is usually set equal to h if $\boldsymbol{H} = h^2\boldsymbol{I}$ (please refer to Einbeck et. al [27] for more details).

The LPC algorithm was originally applied using the R software through the function "*lpc(.)*"[(1)].

Figure 3.1 shows a local principal curve fitted to the traffic (flow-speed) data example introduced earlier[(2)]. It can be seen from the figure that the fitted LPC closely follows the data topology providing 'a curve that passes through the middle of the data cloud'.

3.2.2 Curve Parametrisation

An important characteristic of local principal curves is that they can be parametrised. The parametrisation of the curve plays a key role in calculating data projections onto the curve as well as in extracting data features.

[(1)] In November 2010, a complete R package has been published for the method [21]. For more details, please refer to http://cran.r-project.org/web/packages/LPCM/index.html.

[(2)] The figure was generated using the LPCM package ver.0.41-6.

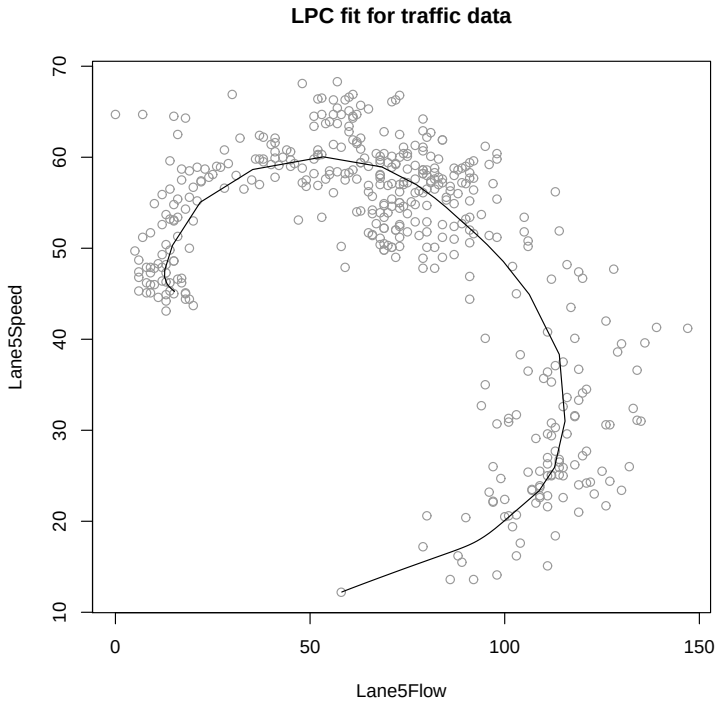

Figure 3.1: LPC fit for traffic data

Here, we briefly illustrate the steps followed by the LPC algorithm for parametrising the curve [25].

Denote by $\boldsymbol{\mu}^\ell = (\boldsymbol{\mu}_1^\ell, \cdots, \boldsymbol{\mu}_d^\ell)^T$ the series of the local centres of mass, $\boldsymbol{\mu}^{\boldsymbol{x}}$'s defining the curve. Let L be the length of the series $\boldsymbol{\mu}^\ell$ (i.e. $\ell = 1, \cdots, L$).

The parametrisation (projection index) τ is constructed such that the curve can be expressed as a function

$$f : I_f \to \mathbb{R}^d, \ \tau \mapsto (f_1(\tau), \cdots, f_d(\tau))^T$$

where $I_f \subset \mathbb{R}$ is the domain of f.

Now, the parametrisation process goes as follows:

- Setting $\tau = 0$ at one of the end points of the curve (this point is consid-

ered as the origin). The parametrisation value τ is assumed to always be increasing in the direction of $\boldsymbol{\gamma}^{\boldsymbol{x}_{(o)}}$.

- Computing a discrete, preliminary parametrisation $(\rho_\ell)_{1\leq\ell\leq L}$, with the same origin as τ. This is by adding up the Euclidean distances between subsequent $\boldsymbol{\mu}^\ell$ values.

- For each dimension $j = 1, \cdots, d$, a cubic spline is fit to interpolate the points $(\rho_\ell, \boldsymbol{\mu}_j^\ell)_{1\leq\ell\leq L}$. This results in a set of pairs of graphs $(\rho, \boldsymbol{\mu}_j(\rho))$, which when put together yields a continuous differentiable spline function $(\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_d)^T(\rho) \equiv \boldsymbol{\mu}(\rho)$. The interpolation performed in this step does not involve any smoothing. $\boldsymbol{\mu}^\ell$ points are just interpolated through a string of cubic polynomials.

- Recalculating the parameter as the arc length of the spline function $\boldsymbol{\mu}(\rho)$.

$$\tau = \int_0^\rho \sqrt{[\boldsymbol{\mu}_1'(u)]^2 + \cdots + [\boldsymbol{\mu}_d'(u)]^2}\ du \tag{3.15}$$

Having completed all the steps above, a final parametrisation vector τ is obtained and is then used for both projecting and feature extraction. It should be noted that calculating the parameter via the arc length along the fitted curve makes it 'unit-speed', so that the distances in parameter space correspond to distances in data space along the fitted LPC [24].

The projection of any data point $\boldsymbol{x}_i, i = 1, \cdots, n$, onto the fitted LPC is the nearest point on the curve to that point $\boldsymbol{x}_i$ (in terms of Euclidean distances). Doing this for all points in the data space yields the projection index τ_i. Figure 3.2 shows the projections of all data points onto the fitted local principal curve for the traffic data example. Each data point has a projection on the curve based upon minimising the squared orthogonal distance.

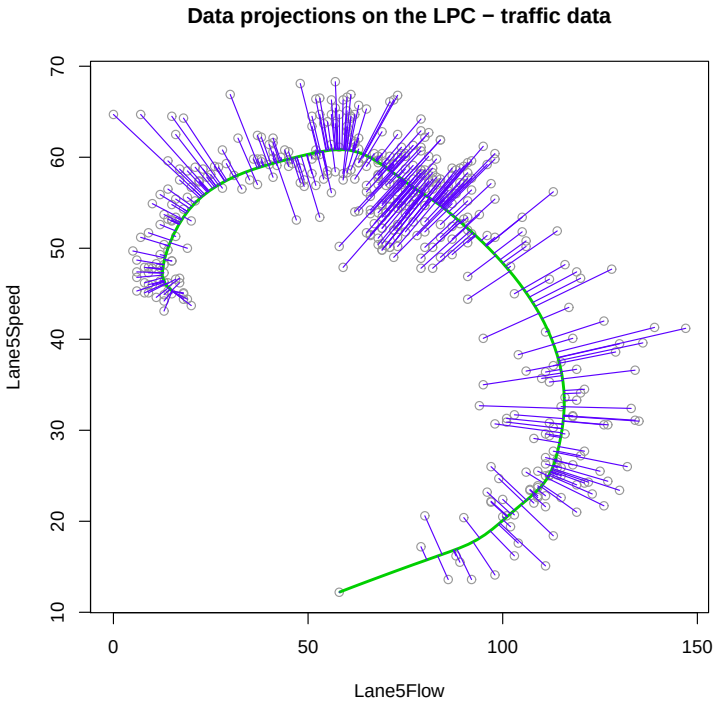

Figure 3.2: LPC projections for traffic data

Now, for any given parameter value τ_i, predictions can be made by evaluating the fitted LPC through the multi-dimensional spline function. This feature can be of importance, especially for multi-dimensional time series data when time can be thought of as an underlying variable that can be related to the curve parametrisation. This shall be further discussed in more detail in Chapter 5.

In general, for any parametrised curve $\boldsymbol{\nu}$, the projection index (dimension reducing mapping) is defined as

$$\tau_{\boldsymbol{\nu}}(\boldsymbol{x}) = \sup_{\tau \in I_{\boldsymbol{\nu}}} \{\tau : \|\boldsymbol{x} - \boldsymbol{\nu}(\tau)\| = \inf_{\eta \in I_{\boldsymbol{\nu}}} \|\boldsymbol{x} - \boldsymbol{\nu}(\eta)\|\} \tag{3.16}$$

The previous projection index function $\tau_{\boldsymbol{\nu}}()$ can be used as a feature extractor for the data as well [53]. For any point $\boldsymbol{x}$, the feature extractor can be expressed

in terms of the arc length as follows:

$$F_{\boldsymbol{\nu}}(\boldsymbol{x}) = \begin{cases} l(\boldsymbol{\nu}, \tau_{\boldsymbol{\nu}}(\boldsymbol{x}_{(o)}), \tau_{\boldsymbol{\nu}}(\boldsymbol{x})), & \tau_{\boldsymbol{\nu}}(\boldsymbol{x}) \geq \tau_{\boldsymbol{\nu}}(\boldsymbol{x}_{(o)}) \\ -l(\boldsymbol{\nu}, \tau_{\boldsymbol{\nu}}(\boldsymbol{x}), \tau_{\boldsymbol{\nu}}(\boldsymbol{x}_{(o)})), & \tau_{\boldsymbol{\nu}}(\boldsymbol{x}) < \tau_{\boldsymbol{\nu}}(\boldsymbol{x}_{(o)}) \end{cases} \tag{3.17}$$

i.e. the arc length between the projections of $\boldsymbol{x}$ and $\boldsymbol{x}_{(o)}$ (the starting point) onto the curve.

A final remark about dimensionality reduction mappings obtained through principal curves is that, though they are more interpretable, they are not considered topology-preserving. Projections calculated through topology-preserving mappings (like the inverse-weighted K-means [59]), have the property that the topology of parameter space reflects that of the data space, which is not necessarily the case with principal-curves-based algorithms.

3.2.3 Other LPC Algorithm Details

3.2.3.1 Choice of Parameters

The LPC algorithm involves several technicalities, some of which can be considered as directly related to the calculation detail (such as controlling the curve direction and angle penalisation [27]) while others can be considered as more relevant to the applicability of the LPC algorithm in different data situations. Some of the latter are introduced hereby.

Starting point selection

There are two basic approaches for setting a value for the starting point $\boldsymbol{x}_{(o)}$, the first is to be chosen at random within the multidimensional data range, and the

second is to choose the point with the highest estimated density as the initial starting point. The first approach is relatively easy, but can result in an outlier which can affect the fitting process. The second approach is more complicated as it needs first a good density estimator, but it results in a more reliable choice of the starting point. Alternatively, a starting point out of the set of observations can be selected manually.

The default setting for the LPC algorithm (the source *lpc()* function) is to choose the starting point at random[(1)].

Bandwidth selection

To apply the LPC algorithm, a bandwidth matrix, $\boldsymbol{H}$, needs to be determined. This matrix contains a set of bandwidths (or squared bandwidths), $h_1, h_2, ..., h_d$, that corresponds to the number of variables (data dimensions), d. Each bandwidth determines the size of local neighbourhood around each point in a certain direction. The optimal choice of the bandwidth matrix depends to a great extent upon the nature of the data set under consideration and becomes more complicated for noisy data.

The choice of the bandwidth(s) is also important for the smoothness of the fitted curve. Over estimating the suitable value(s) of h can result in an over-smoothed curve with relatively small coverage of the data and vice versa. Also too small bandwidths are likely to lead the curve to stick (stop) near the starting point.

A suggested optimal bandwidth selection method specially designed for unsupervised learning techniques is based on what is called 'self coverage' [26], which leads to the bandwidth being set so that tubes centred at the fitted curve cover

[(1)] This was the case in the original *lpc()* function code. In the latest version of the LPCM package (ver.0.44-5 published 28-09-2011), the default choice is to select the starting point automatically in form of a local density mode.

as much of the data as possible.

Although not explicitly designed for principal curves, there also exist several automatic bandwidth selection tools in R which can provide a useful guidance regarding the choice of $\boldsymbol{H}$. Some methods are related to nonparametric smoothing [7], while others are specially designed for kernel density and density derivative estimation [67, 68, 79]. Both methods, especially the latter, can be of some sense in the LPC context since the way of fitting the curve makes it follow the density ridges, which makes the density estimation based tools relevant.

The default initial setting for the LPC algorithm is to set the bandwidth equal to 10% of the range in each direction.

Kernel function

To compute the local centre of mass around a point, a multidimensional kernel function is needed to produce some weighting around the chosen point. There are several types of kernel functions that are normally used. While reaching convergence is faster when using some kernel forms (like the Uniform kernel), other forms may be preferred in terms of smoothness.

The LPC algorithm uses a multidimensional Gaussian kernel, $K_{\boldsymbol{H}}(.)$. The one-dimensional Gaussian kernel usually takes the form: $k(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}$, and $K_{\boldsymbol{H}}$ is obtained from this as a product kernel.

$$k_h(u) = \frac{1}{h} k\left(\frac{u}{h}\right), \quad K_{\boldsymbol{H}}(u) = k_{h_1}(u) \times k_{h_2}(u) \times ... \times k_{h_d}(u) \tag{3.18}$$

After choosing a starting point, $\boldsymbol{x}_{(o)}$, and a d-dimensional kernel, $K_{\boldsymbol{H}}(.)$, the LPC algorithm computes the local centre of mass around the starting point then

around each other point chosen in successive computations. The local centre of mass around a point, $\boldsymbol{x}$, is given by:

$$\boldsymbol{\mu}^{\boldsymbol{x}} = \frac{\sum_{i=1}^{n} K_{\boldsymbol{H}}(X_i - \boldsymbol{x})X_i}{\sum_{i=1}^{n} K_{\boldsymbol{H}}(X_i - \boldsymbol{x})} \tag{3.19}$$

Step size

The next step after calculating the local centre of mass $\boldsymbol{\mu}^{\boldsymbol{x}}$ is to perform a principal component analysis around $\boldsymbol{x}$. Denote by $\boldsymbol{\Sigma}^{\boldsymbol{x}}$ the local covariance matrix of $\boldsymbol{x}$, and let $\boldsymbol{\gamma}^{\boldsymbol{x}}$ be the first eigenvector of $\boldsymbol{\Sigma}^{\boldsymbol{x}}$, we then obtain an updated value of $\boldsymbol{x}$, $\boldsymbol{\mu}^{\boldsymbol{x}} + t_o\boldsymbol{\gamma}^{\boldsymbol{x}}$, where a suitable value of the step size t_o is to be chosen.

When scaling the data and fixing the bandwidth for all directions, the default setting for the LPC algorithm is to set the step size equal to the bandwidth. The bandwidth/step-size factor h/t_o plays a key role in the convergence of the algorithm, especially at data boundaries. This will be further investigated in Chapter 4.

In fact, achieving the required 'optimality' in the choices of the parameters of the *lpc(.)* function is not straightforward specially with high-dimensional data, and it is one of the topics that has a potential to give other ideas of possible improvements to the LPC algorithm.

3.2.3.2 Goodness of Fit in the LPC Context

Einbeck, et al. [27] also suggested a criterion to evaluate the performance of a principal curve similar to the expected squared distances between data X and

the curve $\boldsymbol{\nu}$. The expected square distance can be written in the following form:

$$\Delta(\boldsymbol{\nu}) = E(\inf_{\tau} \|X - \boldsymbol{\nu}(\tau)\|^2) \tag{3.20}$$

Empirically, instead of finding any critical values of (3.20) or minimising it over a class of curves assuming some underlying stochastic model, Einbeck, et al. [27] define the *coverage* of a principal curve by the fraction of all data points which are situated in a certain neighbourhood of the principal curve. Denote the set of points forming a principal curve $\boldsymbol{\nu}$ by $P_{\boldsymbol{\nu}}$, then the coverage of the curve with parameter τ, $C_{\boldsymbol{\nu}}(\tau)$ is:

$$C_{\boldsymbol{\nu}}(\tau) = \#\{\boldsymbol{x} \in X | \ \exists \ p \in P_{\boldsymbol{\nu}} \text{ with } \|\boldsymbol{x} - p\| \leq \tau\}/n \tag{3.21}$$

The coverage function can be interpreted as the empirical distribution function of residuals. It is also a monotone increasing function of τ that will reach 1 for τ tending to infinity [27].

3.2.4 Methodological Improvements to LPC Algorithm

Local principal curves are best used for modelling data which feature a low-dimensional non-linear latent structure. For such a curve fitting algorithm which often deals with noisy multidimensional data structures, it is expected that there arise situations which suggest the need for enhancing the algorithm to best suit and accommodate more complex data sets.

An issue of special interest in our current context is the application of the LPC algorithm to multidimensional time series data sets, especially real-life econometric (and actuarial) data. One of the main things which makes the LPC algorithm

relatively flexible compared to other PC-based algorithms is that, through re-evaluating its parameters (by changing its calculation method or initial values), it can be adapted to provide a better fit in cases of more complex or noisy data.

A natural basic step for exploring possible issues that may arise when applying the LPC algorithm would be to visualise the fitted curve(s) for different combinations of the parameters.

In the next chapter, we will simply show that applying the LPC algorithm could result in several significantly different curves basically because of the different possible choices of both the starting point $\boldsymbol{x}_{(o)}$ and the bandwidth(s) $\boldsymbol{H}$. The more complex is the data, the more difficult becomes reaching a suitable combination of parameters which gives an acceptable result.

An improved version of the LPC algorithm function *lpc(.)* is produced to overcome two main possible issues with applying the algorithm; one with the choice of the starting point and the other with the curve behaviour near data boundaries.

Chapter 4

Mean-Shift and Boundary Extension

4.1 Mean-Shift Algorithm

Consider a set of data points as if they are sampled from some underlying probability density function. In this sense, the areas in the data cloud where points are dense can be thought of as possible clusters whose centres correspond to the modes (local maxima) of the underlying density function.

Let $\boldsymbol{X}$ be a d-variate random vector with density function $f(.)$, mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$. Let $X = (\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n)^T$, $\boldsymbol{x}_i \in \mathbb{R}^d$, be a random sample from $\boldsymbol{X}$.

We define a d-dimensional kernel function $K(.)$ as a radially symmetric function for which a kernel profile $k(.)$ exists, such that:

$$K(u) = c_{k,d} k(\|u\|^2) \tag{4.1}$$

where $c_{k,d}$ is normalisation constant assumed to be 'strictly positive' so that $K(u)$ integrates to one. We assume the kernel profile k is:

- Non-negative ($k(.) \geq 0$).
- Non-increasing ($\forall\ a < b$)

$$k(b) \leq k(a) \ \Rightarrow\ k'(.) \leq 0 \tag{4.2}$$

- Piecewise continuous except for a finite subset and $\int_0^\infty k(u)du < \infty$.

The local mean at a given point $\boldsymbol{x}$, using a d-dimensional kernel function K (weights) and a bandwidth matrix $\boldsymbol{H}$, can be written as:

$$\boldsymbol{\mu}^{\boldsymbol{x}} = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i\, K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i)}{\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i)} \tag{4.3}$$

where $K_{\boldsymbol{H}}(\boldsymbol{x}) = |\boldsymbol{H}|^{-1/2} K(\boldsymbol{H}^{-1/2}\boldsymbol{x})$.

The difference $\boldsymbol{\mu}^{\boldsymbol{x}} - \boldsymbol{x} = \boldsymbol{m}_{H,K}(\boldsymbol{x})$ is called the mean shift[(1)]

$$\boldsymbol{m}_{H,K}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{x})\, K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i)}{\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i)} \tag{4.4}$$

[(1)]Note that both $\boldsymbol{x}$ and $\boldsymbol{\mu}^{\boldsymbol{x}}$ are vector-valued, so $\boldsymbol{\mu}^{\boldsymbol{x}} - \boldsymbol{x}$ is vector-valued as well.

which in the special case $\boldsymbol{H} = h^2\boldsymbol{I}$ simplifies to:

$$\boldsymbol{m}_{h,K}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n}(\boldsymbol{x}_i - \boldsymbol{x})\, K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right)}. \tag{4.5}$$

The 'Mean Shift Algorithm' (hereafter: **MS**) was firstly introduced in computer science literature in 1975 by Fukunaga and Hostetler [33]. It iteratively shifts a data point to the local centre of mass in its neighbourhood. MS is a powerful technique used in a range of nonparametric unsupervised learning methods, including clustering, mode seeking [13], density estimation, tracking, feature space analysis [15], etc.

For each data point, the MS algorithm defines a 'window' (through a bandwidth matrix) which determines the local neighbourhood around this point, and then computes a weighted mean of the data points, then the centre of the current window is shifted to the computed mean and the same procedure is repeated until convergence. The window shift in each iteration is a step towards a more dense region of the data set, which will eventually lead to a local mode.

In other words, the MS algorithm iteratively uses (4.4) to shift a given point $\boldsymbol{x}$ to other locations until $\boldsymbol{m}_{H,K}(\boldsymbol{x})$ becomes nearly negligible. This will be further illustrated in the next subsection.

4.1.1 Convergence of MS Algorithm

Comaniciu and Meer [15] have introduced a proof of the convergence for the mean shift algorithm assuming that the basic condition needed for the algorithm to converge is that the kernel function used has a profile which is convex and

monotonically decreasing.

Denote by $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$ the sequence of successive locations of the mean shift procedure and by $\{\hat{f}(\boldsymbol{y}_j)\}_{j=1,2,\ldots}$ the series of density estimates of the mean shift locations.

Comaniciu and Meer's strategy for the proof of convergence of the MS algorithm is as follows [15]: If $\{\hat{f}(\boldsymbol{y}_j)\}$ is bounded (which is the case as n is finite) and monotonically increasing, then it converges and is a 'Cauchy sequence'. Accordingly, the proof begins with showing that $\{\hat{f}(\boldsymbol{y}_j)\}$ monotonically increases. Then, they argued that this is sufficient for the sequence $\{\boldsymbol{y}_j\}$ to be a Cauchy sequence in a Euclidean space (i.e., in a complete metric space [31]), and hence $\{\boldsymbol{y}_j\}$ also converges.

Li et al. [49] have argued that the latter approach for proving the convergence of the mean shift is mathematically incorrect. In particular, they have shown that the sequence $\{\hat{f}(\boldsymbol{y}_j)\}$ being a Cauchy sequence does not necessarily imply that $\{\boldsymbol{y}_j\}$ is a Cauchy sequence too. Using a convex kernel, the necessary conditions for convergence were that $\{\hat{f}(\boldsymbol{y}_j)\}$ converges and monotonically increases and that for a finite number of critical points, the iterative sequence of the mean shift locations $\{\boldsymbol{y}_j\}$ also converges. We think that the latter approach is more convincing and mathematically correct.

Based upon the above, we hereby outline the convergence of the mean shift algorithm. In particular, we briefly explore the following:

(I) The series of density estimates at successive locations of the mean shift procedure is monotonically increasing (implying that when $\{\hat{f}(\boldsymbol{y}_j)\}$ is bounded,

it is also a Cauchy sequence) [15].

(II) For a finite number of critical points, the sequence of successive locations of the mean shift procedure $\{\boldsymbol{y}_j\}$ converges [49].

(III) At convergence, the MS algorithm always reaches a fixed point which is a local mode. This is an additional result.

(I) Given a d-dimensional kernel function $K(.)$, and a symmetric positive definite $d \times d$ bandwidth matrix $\boldsymbol{H}$, the multivariate kernel density estimator at $\boldsymbol{x}$ is given by [69]

$$\hat{f}_{\boldsymbol{H}}(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i) \tag{4.6}$$

Setting $\boldsymbol{H} = h^2 \boldsymbol{I}$, for simplicity, the kernel density estimator (4.6) becomes

$$\hat{f}(\boldsymbol{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right) \tag{4.7}$$

Using the kernel profile notation, the above equation can be re-written as follows:

$$\hat{f}(\boldsymbol{x}) = \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|^2\right) \tag{4.8}$$

For convergence, it is firstly needed to show that, for any two successive locations of the mean shift procedure $(\boldsymbol{y}_j, \boldsymbol{y}_{j+1})$:

$$[\hat{f}(\boldsymbol{y}_j) < \hat{f}(\boldsymbol{y}_{j+1})] \equiv [\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) > 0]$$

Re-expressing (4.3) using the kernel profile notation, a point $\boldsymbol{y}_{j+1}$ (the local centre

of mass computed at iteration j) can be written as:

$$\boldsymbol{y}_{j+1} = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i k\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} k\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)}$$

For any couple of successive locations of the mean shift, $\boldsymbol{y}_j, \boldsymbol{y}_{j+1}$, we need to show that $f(\boldsymbol{y}_{j+1}) \geq f(\boldsymbol{y}_j)$, i.e.

$$\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) = \frac{c_{k,d}}{nh^d}\left[\sum_{i=1}^{n} k\left(\left\|\frac{\boldsymbol{y}_{j+1} - \boldsymbol{x}_i}{h}\right\|^2\right) - \sum_{i=1}^{n} k\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)\right] \geq 0 \tag{4.9}$$

The convexity of the kernel profile k implies that:

$$k(\boldsymbol{y}_{j+1}) - k(\boldsymbol{y}_j) \geq k'(\boldsymbol{y}_j)(\boldsymbol{y}_{j+1} - \boldsymbol{y}_j)$$

so (4.9) becomes

$$\begin{aligned}
\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) &\geq \frac{c_{k,d}}{nh^d}\left[\sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)\left(\left\|\frac{\boldsymbol{y}_{j+1} - \boldsymbol{x}_i}{h}\right\|^2 - \left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)\right] \\
&= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)\left((\boldsymbol{y}_{j+1}^2 - 2\boldsymbol{y}_{j+1}\boldsymbol{x}_i + \boldsymbol{x}_i^2) - (\boldsymbol{y}_j^2 - 2\boldsymbol{y}_j\boldsymbol{x}_i + \boldsymbol{x}_i^2)\right) \\
&= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)(\boldsymbol{y}_{j+1}^2 - \boldsymbol{y}_j^2 - 2(\boldsymbol{y}_{j+1} - \boldsymbol{y}_j)^T\boldsymbol{x}_i) \\
&= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)(\boldsymbol{y}_{j+1}^2 - \boldsymbol{y}_j^2 - 2(\boldsymbol{y}_{j+1} - \boldsymbol{y}_j)^T\boldsymbol{y}_{j+1}) \\
&= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)(\boldsymbol{y}_{j+1}^2 - \boldsymbol{y}_j^2 - 2(\boldsymbol{y}_{j+1}^2 - \boldsymbol{y}_j\boldsymbol{y}_{j+1})) \\
&= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)(\boldsymbol{y}_{j+1}^2 - \boldsymbol{y}_j^2 - 2\boldsymbol{y}_{j+1}^2 + 2\boldsymbol{y}_j\boldsymbol{y}_{j+1})
\end{aligned}$$

$$= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)(-\boldsymbol{y}_{j+1}^2 - \boldsymbol{y}_j^2 + 2\boldsymbol{y}_j\boldsymbol{y}_{j+1})$$

$$= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)(-1)(\boldsymbol{y}_{j+1}^2 + \boldsymbol{y}_j^2 - 2\boldsymbol{y}_j\boldsymbol{y}_{j+1})$$

$$= \frac{c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} -k'\left(\left\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\right\|^2\right)(||\boldsymbol{y}_{j+1} - \boldsymbol{y}_j||^2)$$

$$\Rightarrow \hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) \geq 0 \tag{4.10}$$

The last result (4.10) is true assuming that the derivative of the kernel profile k exists for all $\boldsymbol{x} \in [0, \infty)$, and that, by definition, $k'(.) \leq 0$ (4.2).

It follows that the sequence $\{\hat{f}(\boldsymbol{y}_j)\}_{j=1,2,\ldots}$ is monotonically increasing and that $(\boldsymbol{y}_{j+1} - \boldsymbol{y}_j) \to 0\,(j \to \infty)$.

(II) Now, we shall explore the convergence of the sequence of MS locations $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$.

Based on the kernel density estimator, to find the local optimum (mode), we need to estimate the density gradient, which is given by the gradient of the density estimate. Using (4.7),

$$\hat{\nabla} f(\boldsymbol{x}) = \nabla \hat{f}(\boldsymbol{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} \nabla K\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right)$$

Using a Gaussian kernel $G(.) = (2\pi)^{-d/2} \exp(-\frac{1}{2}\|.\|^2)$, the density estimate is given by:

$$\hat{f}(\boldsymbol{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} G\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right)$$

$$= \frac{1}{nh^d} \sum_{i=1}^{n} (2\pi)^{-d/2} \exp\left(-\frac{1}{2}\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|^2\right)$$

$$= \frac{1}{nh^d} \sum_{i=1}^{n} g\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right)$$

And the gradient of the density estimate is:

$$\begin{aligned} \nabla \hat{f}(\boldsymbol{x}) &= \frac{1}{nh^d} \sum_{i=1}^{n} g' \left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \\ &= \frac{1}{nh^d} \sum_{i=1}^{n} G' \left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right) \end{aligned}$$

$$\begin{aligned} G' \left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right) &= \nabla G \left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right) \\ &= \nabla \left[(2\pi)^{-d/2} \exp\left(-\frac{1}{2} \left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \right] \\ &= (2\pi)^{-d/2} \exp\left(-\frac{1}{2} \left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \cdot \nabla \left[-\frac{1}{2} \left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right] \\ &= (2\pi)^{-d/2} \exp\left(-\frac{1}{2} \left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \cdot \left[-\frac{1}{2} \frac{1}{h^2} (2)(\boldsymbol{x} - \boldsymbol{x}_i)(1) \right] \\ &= \frac{1}{h^2} (\boldsymbol{x}_i - \boldsymbol{x}) (2\pi)^{-d/2} \exp\left(-\frac{1}{2} \left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \end{aligned}$$

Then, the gradient of the density estimate can be expressed as follows [15]:

$$\begin{aligned} \Rightarrow \nabla \hat{f}(\boldsymbol{x}) &= \frac{1}{nh^d} \sum_{i=1}^{n} \frac{1}{h^2} (\boldsymbol{x}_i - \boldsymbol{x}) (2\pi)^{-d/2} \exp\left(-\frac{1}{2} \left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \\ &= \frac{1}{nh^{d+2}} \sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{x}) (2\pi)^{-d/2} \exp\left(-\frac{1}{2} \left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \\ &= \frac{1}{nh^{d+2}} \sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{x}) G \left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right) \\ &= \frac{1}{nh^{d+2}} \sum_{i=1}^{n} \left[\boldsymbol{x}_i \, G \left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right) - \boldsymbol{x} \, G \left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right) \right] \end{aligned}$$

$$\nabla \hat{f}(\boldsymbol{x}) = \frac{1}{nh^{d+2}} \left[\sum_{i=1}^{n} G\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right) \right] \left[\frac{\sum_{i=1}^{n} \boldsymbol{x}_i \, G\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right)}{\sum_{i=1}^{n} G\left(\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right)} - \boldsymbol{x} \right] \tag{4.11}$$

Using the kernel profile $g(.)$, for $\boldsymbol{x} = \boldsymbol{y}_j$, (4.11) can be re-written as follows:

$$\nabla \hat{f}(\boldsymbol{y}_j) = \frac{c_{g,d}}{nh^{d+2}} \left[\sum_{i=1}^{n} g\left(\left\| \frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h} \right\|^2 \right) \right] [\boldsymbol{y}_{j+1} - \boldsymbol{y}_j] \tag{4.12}$$

Assuming that the number of critical points of $\hat{f}(\boldsymbol{x})$ is finite on the set $A_o = \{y | \hat{f}(y) \geq \hat{f}(\boldsymbol{y}_1)\}$, then those critical points will usually represent the modes or classes in real applications [49].

Without loss of generality, assume there are m_c critical points $\{\boldsymbol{x}_\ell^{(c)}, \ 1 \leq \ell \leq m_c\}$ such that [49]

$$\nabla \hat{f}(\boldsymbol{x}) = 0, \ \ \boldsymbol{x} \in \{\boldsymbol{x}_\ell^{(c)}, 1 \leq \ell \leq m_c\}$$

and

$$\nabla \hat{f}(\boldsymbol{x}) \neq 0, \ \ \boldsymbol{x} \in A_o \ \ \textit{but} \ \ \boldsymbol{x} \notin \{\boldsymbol{x}_\ell^{(c)}, 1 \leq \ell \leq m_c\}$$

Define

$$d_o = \min\{\|\boldsymbol{x}_r^{(c)} - \boldsymbol{x}_s^{(c)}\|, \ \ 1 \leq r \neq s \leq m_c\},$$

let

$$A_{\epsilon,\ell} = \{\boldsymbol{x} \mid \|\boldsymbol{x} - \boldsymbol{x}_\ell^{(c)}\| < \epsilon, \ \ \boldsymbol{x} \in A_o\}, \ \ 1 \leq \ell \leq m_c$$

where $0 \leq \epsilon \leq d_o/3$.

Then, on the bounded closed set $\{\bigcup_{\ell=1}^{m_c} A_o - A_{\epsilon,\ell}\}$, the following hold:

- $\nabla \hat{f}(\boldsymbol{x})$ is continuous and $\nabla \hat{f}(\boldsymbol{x}) \neq 0$.
- $\min \|\nabla \hat{f}(\boldsymbol{x})\| \neq 0$.
- There exists $c_\epsilon > 0$ satisfying $\|\nabla \hat{f}(\boldsymbol{x})\| > c_\epsilon$.

Since $(\boldsymbol{y}_{j+1} - \boldsymbol{y}_j) \to 0\,(j \to \infty)$ and $\nabla \hat{f}(\boldsymbol{y}_j) \to 0\,(j \to \infty)$ (4.12), there exists $N_\epsilon > 0$ satisfying

$$\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| < \epsilon \quad , j \geq N_\epsilon \tag{4.13}$$

$$\|\nabla \hat{f}(\boldsymbol{y}_j)\| < c_\epsilon \quad , j \geq N_\epsilon$$

so

$$\{\boldsymbol{y}_j,\ \ j \geq N_\epsilon\} \subset \bigcup_{\ell=1}^{m_c} A_{\epsilon,\ell}$$

Now, consider the quantity $\|\boldsymbol{x}_1^* - \boldsymbol{x}_2^*\|$, where

$$\boldsymbol{x}_1^* \in A_{\epsilon,i_1},\ \ \boldsymbol{x}_2^* \in A_{\epsilon,i_2},\ \ 1 \leq i_1 \neq i_2 \leq m_c$$

$$\begin{aligned}
\|\boldsymbol{x}_1^* - \boldsymbol{x}_2^*\| &= \|\boldsymbol{x}_1^* - \boldsymbol{x}_{i_1}^{(c)} + \boldsymbol{x}_{i_1}^{(c)} - \boldsymbol{x}_{i_2}^{(c)} + \boldsymbol{x}_{i_2}^{(c)} - \boldsymbol{x}_2^*\| \\
&\geq \|\boldsymbol{x}_{i_1}^{(c)} - \boldsymbol{x}_{i_2}^{(c)}\| - \|\boldsymbol{x}_1^* - \boldsymbol{x}_{i_1}^{(c)}\| - \|\boldsymbol{x}_{i_2}^{(c)} - \boldsymbol{x}_2^*\| \\
&\geq d_o - \epsilon - \epsilon \\
&= 3\epsilon - \epsilon - \epsilon \\
&= \epsilon
\end{aligned}$$

Therefore, from (4.13), the set of points $\{\boldsymbol{y}_j,\ \ j \geq N_\epsilon\}$ can only be those data points neighbouring each local mode, which means that the sequence $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$ converges [49].

(III) Now, we shall try to provide an answer, in a more explicit form, to the question: The MS algorithm converges, but does it always converge to a local mode?

The second term in (4.11) is the mean shift

$$\boldsymbol{m}_{h,K}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i \, K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)} - \boldsymbol{x}, \tag{4.14}$$

so that the gradient is proportional to the mean shift

$$\boldsymbol{m}_{h,K}(\boldsymbol{x}) = c.\nabla \hat{f}_{h,K}(\boldsymbol{x}).$$

So, the direction of the mean shift and the gradient vector is the same. And since the gradient vector is always directed to the maximum increase of the density, so is the mean shift. It follows that the mean shift converges to a local maximum.

Furthermore, at convergence we have $\nabla \hat{f}(\boldsymbol{x}) = 0$. Using this in (4.11), we get:

$$\boldsymbol{x} = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i \, G\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)}{\sum_{i=1}^{n} G\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)}$$

which shows that the MS algorithm always converges to some point representing the local centre of mass within the MS window. Since the mean shift algorithm converges and since $\hat{f}(\boldsymbol{y}_{j+1}) > \hat{f}(\boldsymbol{y}_j) > 0 \;\; \forall j$, and knowing that the direction of the mean shift vector is identical to that of the gradient, the point reached at

convergence is a local mode.

4.1.2 Mean Shift Properties

Using (4.7), the gradient of the density estimator (4.11), employing a kernel $K(.)$ and a bandwidth matrix $\boldsymbol{H} = h^2\boldsymbol{I}$, can be expressed as follows

$$\begin{aligned}\nabla\hat{f}(\boldsymbol{x}) &= \frac{1}{h^2} \times \left[\frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)\right] \times [\boldsymbol{m}_{h,K}(\boldsymbol{x})] \\ &= \frac{1}{h^2} \times \hat{f}(\boldsymbol{x}) \times \boldsymbol{m}_{h,K}(\boldsymbol{x}) \\ \Rightarrow \boldsymbol{m}_{h,K}(\boldsymbol{x}) &= (h^2)\frac{\nabla\hat{f}(\boldsymbol{x})}{\hat{f}(\boldsymbol{x})}\end{aligned}$$

Generalising this result to the case of a general bandwidth matrix as in (4.4), we get the following interesting property of the mean shift [26]

$$\boldsymbol{m}_{H,K}(\boldsymbol{x}) \propto \boldsymbol{H}\frac{\nabla\hat{f}(\boldsymbol{x})}{\hat{f}(\boldsymbol{x})} \tag{4.15}$$

Thus, the mean shift at a data point $\boldsymbol{x}$ is proportional to the normalised density gradient estimate. Also, there exists an asymptotic version of (4.15), which shall be shown in Section 4.2.

This property of the mean shifts implies that, at a critical point $\boldsymbol{x}_m^{(c)}$ that corresponds to a local mode of the density $\hat{f}(\boldsymbol{x})$, the mean shift becomes zero, because $\nabla\hat{f}(\boldsymbol{x}) = 0$, and the local centre of mass around this local mode becomes a fixed point identical to that local mode.

$$\mu^{\boldsymbol{x}_m^{(c)}} = \boldsymbol{x}_m^{(c)} \tag{4.16}$$

Einbeck [26] referred to the previous property (4.16) as *local self consistency* and to all points satisfying this property as *local principal points* (LPPs).

Cheng [13] introduced the term 'shadow of a kernel'. A kernel $K^{(s)}(.)$ is said to be a shadow of kernel $K(.)$ if the mean shift using K is in the gradient direction of the density estimate using $K^{(s)}$. In this case, the mean shift at $\boldsymbol{x}$ is

$$\boldsymbol{m}_{H,K}(\boldsymbol{x}) = \frac{\nabla \hat{f}_{H,K^{(s)}}(\boldsymbol{x})}{2c\hat{f}_{H,K}(\boldsymbol{x})} \tag{4.17}$$

where $c > 0$ is a constant.

A special case of (4.17) is when $K(.)$ is a Gaussian or a truncated Gaussian function. Only in this case the kernel itself is its own shadow, and the mean shift becomes

$$\boldsymbol{m}_{H,K}(\boldsymbol{x}) = \frac{1}{2c}\nabla \log(\hat{f}_{H,K^{(s)}}(\boldsymbol{x})) \tag{4.18}$$

The features of the MS algorithm make it a tool suitable for real data analysis. It does not assume any prior knowledge of the density. The main thing to take care with is the choice of the window size h. Although the adaptive magnitude of the algorithm guarantees convergence regardless of the step size, inappropriate window size can, in some cases, cause modes to be mis-detected. A common problem when choosing a fixed window size is the slow shifts on plateaus of the surface, which is further magnified through taking logs [13] as can be seen from (4.18). In such cases, adaptive window size approach is preferred [12, 14, 16, 87].

The choice of the kernel function only affects the speed of the algorithm until reaching convergence. Comaniciu and Meer [15] have shown that when a Uni-

form kernel is employed, the number of steps to convergence is finite. They also illustrated that the Gaussian kernel, although it is relatively slower to converge, appears to be the optimal one for the mean shift procedure, as it satisfies what they define as the *smooth trajectory* property. That is, when a normal kernel is employed, the path of the MS procedure towards the mode follows a smooth trajectory, and the angle between two consecutive mean shift vectors is always less than 90° (the cosine is strictly positive). In the special case of $\boldsymbol{H} = h^2\boldsymbol{I}$,

$$\frac{\boldsymbol{m}_{h,G}(\boldsymbol{y}_j)^T \boldsymbol{m}_{h,G}(\boldsymbol{y}_{j+1})}{\|\boldsymbol{m}_{h,G}(\boldsymbol{y}_j)\| \|\boldsymbol{m}_{h,G}(\boldsymbol{y}_{j+1})\|} > 0. \tag{4.19}$$

4.1.3 Mean Shift Asymptotics

As mentioned earlier in Chapter 3 (Section 3.1), weighted local principal component analysis, using kernels as weights, has recently been introduced in a variety of statistical applications, but without the asymptotics of it having been addressed. Same applies to the mean shift procedure. Now, we shall provide an approximation of the mean shift (4.4).

Recalling the function $Q(\boldsymbol{m}, \boldsymbol{\gamma})$ (3.4), which represents the locally weighted squared distances between data $\boldsymbol{x}_i$ and their projections $\boldsymbol{x}_i^g$, let f denote the density function of $\boldsymbol{X}$ with support $\text{supp}(f) \subset T$. We assume that the following hold [65, 79]:

(A1) The kernel K is a bounded and compactly supported probability density function such that $\int \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u})\, d\boldsymbol{u} = \mu_2(K)\boldsymbol{I}$, with $\mu_2(K) \in \mathbb{R}$, $\mu_2(K) \neq 0$.

(A2) At $\boldsymbol{x} \in \text{supp}(f)$, f is continuously differentiable and $f(\boldsymbol{x}) > 0$.

(A3) The sequence of bandwidth matrices $\boldsymbol{H}$ is such that $n^{-1}|\boldsymbol{H}|^{-1/2}$ and each entry of $\boldsymbol{H}$ tending to zero as $n \longrightarrow \infty$, with $\boldsymbol{H}$ remaining symmetric and positive definite.

The first assumption (A1) is preliminary for the density estimation and asymptotics theory as introduced by Parzen [57] (Theorem 1A). Assumption (A3) implies that n always tends to infinity faster than $\boldsymbol{H}$ tends to zero, which means that there will be 'sufficiently enough' data within every neighbourhood.

Let $o_p(1)$ denote a sequence which tends to zero in probability as $n \longrightarrow \infty$ and let $\mathbf{1}$ denote a generic matrix having each entry equal to 1. Adapting the results in [65](p.1352) for the current context, we get

$$\begin{aligned} \sum_{i=1}^{n}(\boldsymbol{x}_i - \boldsymbol{x})K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x}) &= n[\mu_2(K)\boldsymbol{H}\nabla f(\boldsymbol{x}) + o_p(\boldsymbol{H}\mathbf{1})] \\ \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i - \boldsymbol{x}) &= n[f(\boldsymbol{x}) + o_p(1)] \end{aligned}$$

Recalling the mean shift (4.4), $\boldsymbol{m}_{H,K}(\boldsymbol{x}) = \dfrac{\sum_{i=1}^{n}(\boldsymbol{x}_i - \boldsymbol{x})K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i)}{\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i)}$, an asymptotic version of the mean shift can be given by the quotient of the two previous expressions

$$\tilde{\boldsymbol{m}}_{H,K}(\boldsymbol{x}) = \mu_2(K)\boldsymbol{H}\frac{\nabla f(\boldsymbol{x})}{f(\boldsymbol{x})} + o_p(\boldsymbol{H}\mathbf{1}) \tag{4.20}$$

Once again, knowing that the mean shift vector always shifts a given point $\boldsymbol{x}$ into a direction in which data are more dense, this means that the larger the shift (step size) the less dense are the data at $\boldsymbol{x}$. In other words, using the previous approximation for the mean shift (4.20), and since $\mu_2(K) \neq 0$, this further confirms that when the mean shift value is zero, the gradient value will be zero as well, which is why the mean shift is being used for density mode detection. This result was used in Section 3.1.2.

4.1.4 A MS-Based Methodological Improvement to the LPC Algorithm

As described in Section 3.2, a local principal curve is fully determined by the series of local centres of mass $\boldsymbol{\mu}^{\boldsymbol{x}}$ computed throughout successive iterations. Apart from the window size (bandwidth) and the kernel applied, the first local centre of mass $\boldsymbol{\mu}^{\boldsymbol{x}_{(o)}}$ is highly sensitive to the choice of the starting point $\boldsymbol{x}_{(o)}$.

Being chosen at random, by default, an extreme choice of the starting point may result in some undesirable situations like forcing the curve to stick at some point (area) that is relatively far from the dense areas of the data cloud, or causing some bumps (un-smoothness) in the fitted curve. In such occasions, the fitted local principal curve will not be considered as a good representation of the data.

The idea that the mean shift algorithm always converges to a fixed point (the nearest local mode) can be very useful in this context. Using the MS algorithm, an additional step has been added to the LPC algorithm as follows:

1. Choose a starting point $\boldsymbol{x}_{(o)}$.
2. Apply the MS algorithm at $\boldsymbol{x}_{(o)}$ until reaching convergence, based upon some pre-determined threshold. At convergence, $\boldsymbol{x}_{(o)}$ has been shifted to a new location, $\boldsymbol{x}$.
3. Calculate $\boldsymbol{\mu}^{\boldsymbol{x}}$, the local centre of mass around $\boldsymbol{x}$.
4. Perform a principal component analysis locally at $\boldsymbol{x}$.
5. Find a new value for $\boldsymbol{x}$ by following the first local principal component starting at $\boldsymbol{\mu}^{\boldsymbol{x}}$.
6. Repeat steps 3 to 5 until $\boldsymbol{\mu}^{\boldsymbol{x}}$ remains (approximately) constant.

Step (2) above, the mean shift step, shifts the starting point $\boldsymbol{x}_{(o)}$ to the nearest local mode, $\boldsymbol{x}$, so that $\gamma^{\boldsymbol{x}}$ (the direction of the first principal component at $\boldsymbol{x}$) does not depend on the starting point any more. This ensures that the LPC algorithm is less sensitive and unaffected by any random choice of $\boldsymbol{x}_{(o)}$.

In order to graphically illustrate the idea of using the MS algorithm for enhancing the random choice of the starting point, the 'traffic' sample data referred to in Chapter 1 (the Introduction) shall be used.

First, it is expected that applying the LPC algorithm could result in several significantly different curves, basically because of the different possible choices of both $\boldsymbol{x}_{(o)}$ and $\boldsymbol{H}$. Using the 'traffic' data, applying the function *lpc(.)* with its defaults (the starting point is chosen at random and the bandwidth in each direction = $1/10^{th}$ of the data range in this direction) may not always produce a good fit. In this case, one should reach a suitable combination of $\boldsymbol{x}_{(o)}$ and H that gives an acceptable result. Usually, when trying to produce well fitted LPCs for some complex data sets, a suitable bandwidth matrix is to be used and then the algorithm is run several times, using the same $\boldsymbol{H}$ in each trial, until reaching a reasonable fit.

Still the choice of $\boldsymbol{x}_{(o)}$ is important. Figure 4.1 shows two different fitted curves that result from using the *lpc(.)* function with fixed bandwidths and a different, manually selected, starting point each time[(1)]. The two specific starting points shown in the plot were intentionally selected as examples of possible inadequate outcomes for the starting point being poorly chosen. The area near each of those

[(1)] This figure was generated using the original source code for the function *lpc()* of the LPCM package (ver.0.32-1).

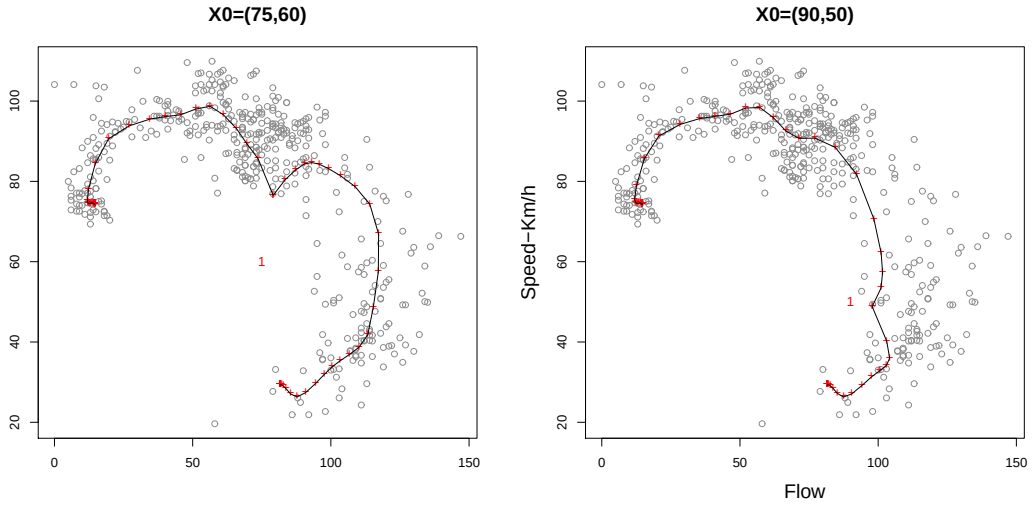

Figure 4.1: Different LPCs using the same bandwidth and different starting points

starting points is somehow far from the majority of data points in the high density areas.

It is easily noticed that, fixing the bandwidth, the shape of the curve is affected by the choice of the starting point, which is labelled with the "1" that appears on each graph. This situation gives an indication that there is some 'sensitivity to the choice of the starting point' problem existing.

Now, we apply the mean shift procedure. Using a Gaussian Kernel, G, the formula to compute the mean shift at any point $\boldsymbol{x}$ is (4.14):

$$\boldsymbol{m}_{h,g}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i \, g\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right\|^2\right)} - \boldsymbol{x} \tag{4.21}$$

where:

- $\boldsymbol{x}_i$: data points in a d-dimensional space $\mathbb{R}^d$, $i = 1, ..., n$
- $g(.)$: the profile of the kernel function.

- h: the selected bandwidth for the kernel function used.

Using the previous formula, the iterative mean shift process is executed several times until reaching an approximately constant value, which represents the nearest local mode to the starting point selected. This was done using R, and the code for the mean shift was added consistently to the original LPC code. Figure 4.2 illustrates graphically how the mean shift procedure works[(1)]. The series of black boxes shows the successive shifts until convergence, and hence reaching a reliable starting point. After applying the MS algorithm, the fit has significantly improved and it is clear that the fitted LPC becomes less sensitive to the choice of the starting point.

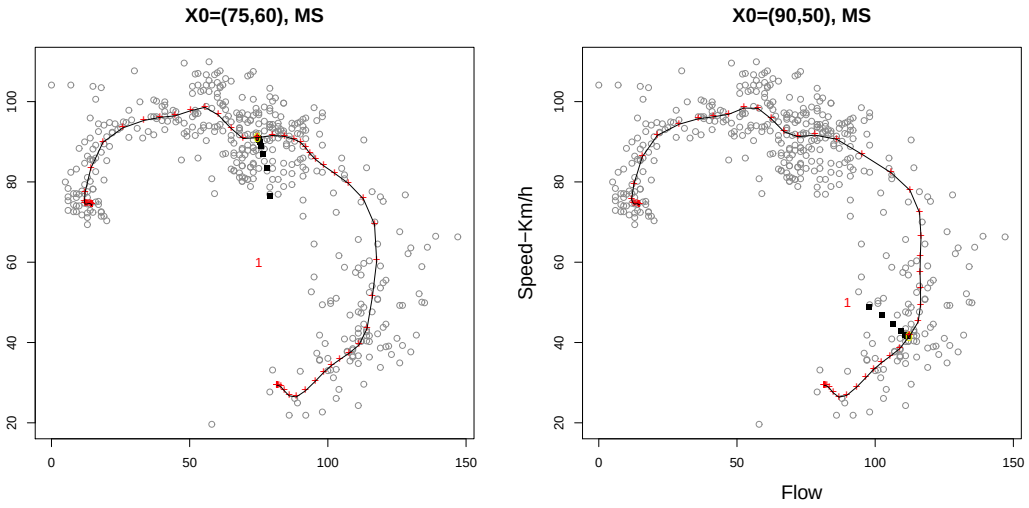

Figure 4.2: Trials shown in Figure 4.1 with mean shift enabled

An interesting property worth mentioning is that for a given bandwidth matrix $\boldsymbol{H}$ and a kernel function $K_H(.)$, when using iterative mean shift to find a starting point, there is a finite set of starting points that result from applying the mean shift function, and the number of points in this set is relative to the number of

[(1)] This figure was generated using a modified version of the LPCM package ver.0.32-1).

local modes that exist in the data set. In other words, the number of possible LPCs is bounded by the number of local maxima of the density function.

To graphically illustrate the previous property, a hundred LPCs were fit twice for the same data, once using the defaults for the *lpc*() function (Figure 4.3) and then once again using the modified code with the MS algorithm integrated (Figure 4.4).

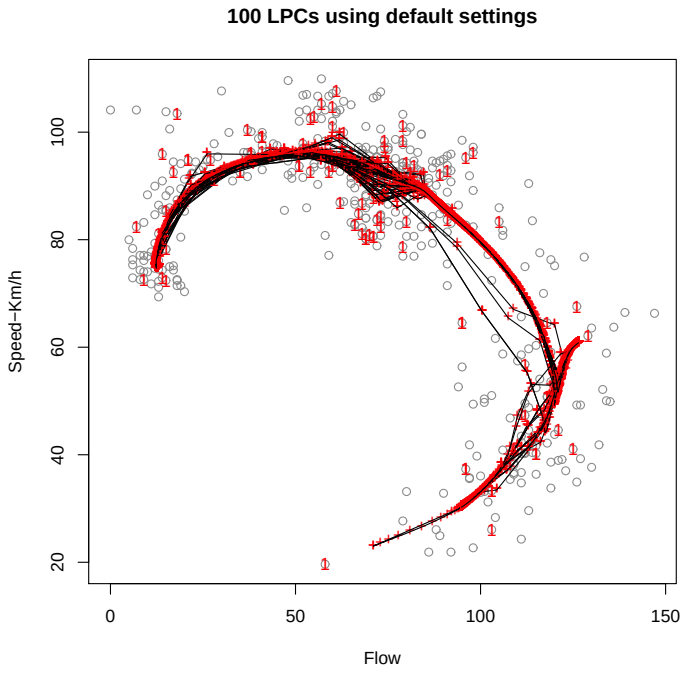

Figure 4.3: A hundred fitted LPCs using lpc() default options

Interestingly, the number of local principal curves when using the MS algorithm for enhancing the choice of the starting point is much less than a hundred and it is bounded by the number of local modes (dense areas) in the data cloud.

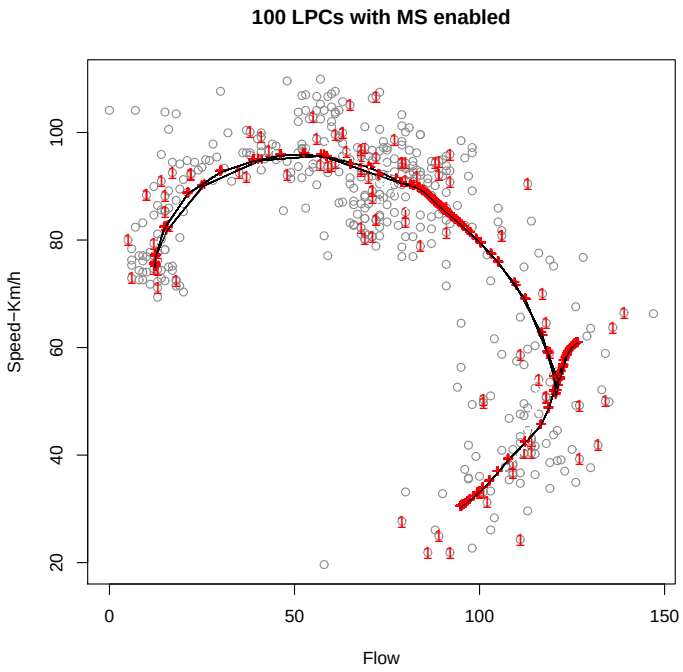

Figure 4.4: A hundred fitted LPCs with mean shift for $\boldsymbol{x}_0$ enabled

4.2 Asymptotics for Local Principal Curves

Building upon the developments in Section 3.1.2, in this section the asymptotic local behaviour when fitting local principal curves for large sample sizes and small neighbourhoods shall be investigated. We first recall that, when fitting a LPC, at any iteration j, the LPC first moves from the current location $\boldsymbol{x}_{(j)}$ to the local centre of mass around that location $\boldsymbol{\mu}_{(j)}$, then steps to a new location $\boldsymbol{x}_{(j+1)}$. Typically, at each iteration, the size of the step is predetermined to be a fixed distance t and the direction of the step is identical to that of the first local eigenvector at $\boldsymbol{x}_{(j)}$, $\boldsymbol{\gamma}_{(j)}$. Considering this, the new location $\boldsymbol{x}_{(j+1)}$ can be represented as

$$\boldsymbol{x}_{(j+1)} = \boldsymbol{\mu}_{(j)} \pm t\, \boldsymbol{\gamma}_{(j)} \tag{4.22}$$

where the sign in '$\pm$' is given by $\operatorname{sign}(\boldsymbol{\gamma}_{(j)} \circ \boldsymbol{\gamma}_{(j-1)})$ (this 'signum flipping' ensures

that the curve maintains its direction).

Asymptotically, recalling the approximated version of $\boldsymbol{\gamma}^{\boldsymbol{x}}$(3.14), $\tilde{\boldsymbol{\gamma}}_{(j)} = -\frac{\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}$, this shows that the LPC always follows the gradient of the density function, which means in practice that it will closely follow the density ridge [23].

Now, considering the difference between two neighbouring points $\boldsymbol{x}_{(j)}$ and $\boldsymbol{x}_{(j+1)}$, from (4.22), one has

$$\begin{aligned}
\boldsymbol{x}_{(j+1)} - \boldsymbol{x}_{(j)} &= \boldsymbol{\mu}_{(j)} - \boldsymbol{x}_{(j)} \pm t\boldsymbol{\gamma}_{(j)} \\
&\overset{a}{=} \mu_2(K)\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})/f(\boldsymbol{x}_{(j)}) \pm t\frac{\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||} \\
&= \left(\frac{\mu_2(K)}{f(\boldsymbol{x}_{(j)})} \pm \frac{t}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}\right)\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)}).
\end{aligned}$$

where for all terms succeeding the $\overset{a}{=}$ symbol, any terms of an asymptotically higher order than the leading term are omitted.

Let $\theta(\boldsymbol{x}) \equiv \boldsymbol{\nabla} f(\boldsymbol{x})/f(\boldsymbol{x})$, then the Taylor expansion of θ at $\boldsymbol{x}$ is given by

$$\theta(\boldsymbol{x} \pm \boldsymbol{\delta}) = \theta(\boldsymbol{x}) \pm \left[\frac{\mathcal{H}(\boldsymbol{x})}{f(\boldsymbol{x})} - \theta(\boldsymbol{x})\theta(\boldsymbol{x})^T\right]\boldsymbol{\delta} + O(\boldsymbol{\delta}^2)$$

where $\boldsymbol{\delta} \longrightarrow 0$ (component-wise), and $\mathcal{H}(\boldsymbol{x})$ is the Hessian of f at $\boldsymbol{x}$.

This implies that the difference between two neighbouring local centres of mass $\boldsymbol{\mu}_{(j)}$ and $\boldsymbol{\mu}_{(j+1)}$ is first-order approximated by

$$\begin{aligned}
&\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)} \\
&= (\boldsymbol{\mu}_{(j+1)} - \boldsymbol{x}_{(j+1)}) + (\boldsymbol{x}_{(j+1)} - \boldsymbol{\mu}_{(j)}) \\
&\overset{a}{=} \mu_2(K)\boldsymbol{H}\frac{\boldsymbol{\nabla} f(\boldsymbol{x}_{(j+1)})}{f(\boldsymbol{x}_{(j+1)})} \pm t\,\tilde{\boldsymbol{\gamma}}_{(j)} \\
&= \mu_2(K)\boldsymbol{H}\,\theta(\boldsymbol{x}_{(j)} + (\boldsymbol{x}_{(j+1)} - \boldsymbol{x}_{(j)})) \pm t\frac{\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}
\end{aligned}$$

$$
\begin{aligned}
&= \mu_2(K)\boldsymbol{H}\left\{\theta(\boldsymbol{x}_{(j)}) + \left[\frac{\mathcal{H}(\boldsymbol{x}_{(j)})}{f(\boldsymbol{x}_{(j)})} - \frac{\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})^T}{f(\boldsymbol{x}_{(j)})^2}\right]\underbrace{(\boldsymbol{x}_{(j+1)} - \boldsymbol{x}_{(j)})}_{O(\boldsymbol{H})}\right\} \\
&\quad \pm t\frac{\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||} \\
&\overset{a}{=} \mu_2(K)\boldsymbol{H}\frac{\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})}{f(\boldsymbol{x}_{(j)})} \pm t\frac{\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||} \\
&= \left[\frac{\mu_2(K)}{f(\boldsymbol{x}_{(j)})} \pm \frac{t}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}\right]\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)}) \qquad (4.23)
\end{aligned}
$$

The first term in (4.23) reflects the contribution of the mean shift towards the LPC step at each iteration, whereas the second term reflects that of the local principal component analysis. In other words, the two-step character of the LPC algorithm is still visible through the previous result (4.23).

For analytical purposes, until the end of this section, the recommended default setting for the LPC algorithm that $\boldsymbol{H} = h^2\boldsymbol{I}$ [27] shall be used. Applying this to (4.23) we get

$$
\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)} \overset{a}{=} \left[\frac{h^2\ \mu_2(K)}{f(\boldsymbol{x}_{(j)})} \pm \frac{t}{||\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}\right]\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)}) \qquad (4.24)
$$

It is noticeable from (4.24) that the mean shift step will always pull the curve towards higher densities. This implies that if the LPC is moving towards higher densities in the data cloud, the effect of both the mean shift and the local PCA steps will steer the curve in the same direction. On the other hand, if the curve is moving towards less dense areas, the two effects will be working in different directions leading the term $\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)}$ to be relatively small. When the two step effects are equal, i.e. $\frac{h^2\mu_2(K)}{f(\boldsymbol{x}_{(j)})} = \frac{t}{||\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}$, then the LPC can get stuck. Denote

by $\boldsymbol{x}_s$ the point at which the curve stops, then at this point

$$f(\boldsymbol{x}_s) = \frac{h^2}{t}\mu_2(K)||\boldsymbol{\nabla} f(\boldsymbol{x}_s)|| \tag{4.25}$$

This means that, for any kernel K, the position at which the curve stops depends on the bandwidth h, the PCA step size t, the kernel function K and the density of the random vector $\boldsymbol{X}$. Furthermore, if a Gaussian kernel is used, then $\mu_2(K) = 1$ and (4.25) becomes

$$f(\boldsymbol{x}_s) = \frac{h^2}{t}||\boldsymbol{\nabla} f(\boldsymbol{x}_s)|| \tag{4.26}$$

which indicates that, when using a Gaussian kernel the LPC stopping point only depends on the bandwidth h, the PCA step size t and the density of $\boldsymbol{X}$.

A more specific case of (4.26) is when we use the default setting of the LPC algorithm that $t = h$, hence (4.26) becomes

$$f(\boldsymbol{x}_s) = h\,||\boldsymbol{\nabla} f(\boldsymbol{x}_s)|| \tag{4.27}$$

So, when all the default settings for the LPC algorithm are employed (i.e. $\boldsymbol{H} = h^2\boldsymbol{I}$, K is Gaussian and $t = h$), the position at which the curve gets stuck only depends on the bandwidth h and the underlying density of the random vector X.

Now, in order to verify the previous conclusion (4.27) by experiment, let us assume that the random vector under consideration is given by

$$\boldsymbol{X} \sim N(\boldsymbol{0}, \sigma^2\boldsymbol{I}) \tag{4.28}$$

with $\boldsymbol{0} \in \mathbb{R}^d$ being a vector of $0's$, and $\boldsymbol{I} \in \mathbb{R}^{d\times d}$ being the identity matrix. The

density function can be expressed as

$$f(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}\sigma^d} \exp\left\{-\frac{1}{2\sigma^2}\boldsymbol{x}^T\boldsymbol{x}\right\}$$

and

$$||\boldsymbol{\nabla} f(\boldsymbol{x})|| = \frac{1}{(2\pi)^{d/2}\sigma^{d+2}} \exp\left\{-\frac{1}{2\sigma^2}\boldsymbol{x}^T\boldsymbol{x}\right\} ||\boldsymbol{x}||$$

so that (4.27) boils down to

$$||\boldsymbol{x}_s|| = \frac{\sigma^2}{h} \tag{4.29}$$

The previous result (4.29) gives an indication where the points satisfying the property (4.27) are situated. To clarify this by experiment [23], we assume that $\boldsymbol{X}$ is bivariate normal, i.e. of type (4.28) with $d = 2$. First, we simulate $n = 10000$ replicates from $\boldsymbol{X}$ assuming $\sigma^2 = 2$ and then we do the same assuming $\sigma^2 = 3$. Next, we fit 20 local principal curves (using a Gaussian base kernel $G(\cdot)$ and setting $t = h = 1$) to each of both data clouds, where the starting points are randomly chosen among all those observations $\boldsymbol{x}_i$ satisfying the property $||\boldsymbol{x}_i|| \leq 1$. Without this affecting the simulation, to cope with the basic assumption for kernels in the context of asymptotic LPCA [65] (compactly supported), the kernel $G(.)$ is truncated at ± 5.

The resulting curves are displayed in Figure 4.5 (top row). In these plots, the dashed and solid circle symbolise the radii $||\boldsymbol{x}|| = 1$ and $||\boldsymbol{x}|| = \sigma^2$, respectively, so according to the theory the curves are expected to get stuck close to the solid circle satisfying (4.29), which is always the case (notice that for both $\sigma^2 = 2$ (left) and $\sigma^2 = 3$ (right), all principal curves converge to endpoints which are very close to the solid circle).

According to (4.29), it is expected that the smaller is h the larger is the area

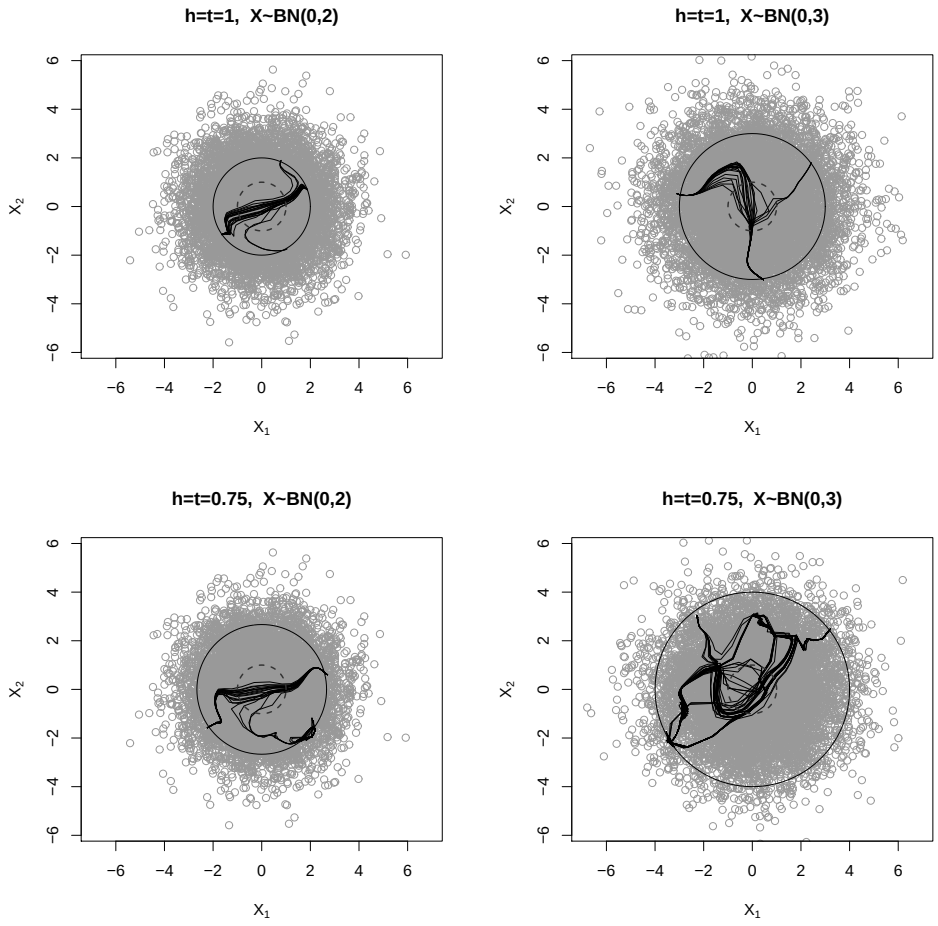

Figure 4.5: 20 local principal curves with bandwidths $h = t = 1$ (top) and $h = t = 0.75$ (bottom) through multivariate Gaussian data with $\sigma^2 = 2$ (left) and $\sigma^2 = 3$ (right). The dashed circle indicates the radius $||\boldsymbol{x}|| = 1$, while the radius of the solid circle is equal to $||\boldsymbol{x}|| = \sigma^2/h$ according to (4.29).

of the data visited by the curves and vise versa. The simulation was repeated setting $h = t = 0.75$ (the bottom panels of the figure) and it is clear that the previous statement is true by experiment.

4.3 LPC Boundary Extension

Until now, we observed that by reducing the bandwidth one obtains curves which proceed further into the boundary region of the data. Access to these boundary regions can be of a special importance, for instance for time series data where the endpoints correspond to the most current observations. Furthermore, curves which are "too short" in the boundaries will result in projections clustered at the endpoints, which impacts negatively on the usability of the curve as a data compression tool, a problem which was observed by [24] in the context of nonlinear compression of high-dimensional spectral data.

In practice, specifically for econometric data, it is desired to try extending the local principal curve beyond its natural endpoint in order to reach more data points at boundaries and enhance the type of summary obtained by fitting the curve specially for short-term predictions. Obviously, decreasing the bandwidth arbitrarily will not be the solution, as this will result in a curve which gets stuck even sooner. Also, the way the LPC algorithm works makes it very difficult to simply decrease h arbitrarily in order to get better access to the boundary regions, as this will impact detrimentally onto other parts of the curve.

Starting from (4.24), one can find a more accurate and practicable way of dealing with this problem. The difference between two successive centres of mass can be written as

$$\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)} \overset{a}{=} h \left[\frac{h\,\mu_2(K)}{f(\boldsymbol{x}_{(j)})} \pm \frac{t}{h} \frac{1}{||\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||} \right] \boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})$$

Having re-expressed (4.24) in the above form, it can be noticed that the term corresponding to the principal component step is multiplied by t/h. Hence, if t

is increased relative to h, the PCA contribution increases relative to the mean shift contribution, and the principal curve will proceed *beyond* the limit given by (4.29).

We illustrate this effect again through simulation. Instead of running the simulation using $t = h$, we now allow these two parameters to decouple. Using Gaussian data $\boldsymbol{X}$ with $\sigma^2 = 3$, 20 local principal curves have been fitted with different ratios of t and h. The resulting curves are displayed in the first three panels of Figure 4.6, and one observes that, for $t/h < 1$, the curve will stop inside the circle defined by (4.29), while for $t/h > 1$, it will stop outside (in fact, the radius at which the curves converge is now $\sigma^2 t/h^2$). However, in practice it is impractical to increase t beyond h, as this would impact detrimentally onto large parts of the curve, and cause erratic behavior especially in the boundary region. Therefore, it is recommended to generally keep the default setting $t = h$, which has proven to work generally well, for the non-boundary part of the principal curve, but reduce h adaptively relative to t as soon as the curve begins to converge to its endpoint.

In the implementation of the LPC algorithm [21], this is achieved by defining a threshold, say T_1, such that when the difference between two successive local centres of mass falls below the threshold

$$\frac{\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)}}{\boldsymbol{\mu}_{(j+1)} + \boldsymbol{\mu}_{(j)}} \leq T_1 \tag{4.30}$$

we start reducing the bandwidth adaptively setting $h_{(j+1)} = (1 - \delta)h_{(j)}$, for some small constant $\delta > 0$.

Having done this, a second threshold, $0 < T_2 < T_1$, is needed to be defined in order to determine when the state of convergence is reached and the algorithm is stopped. The performance of this technique is demonstrated in the bottom

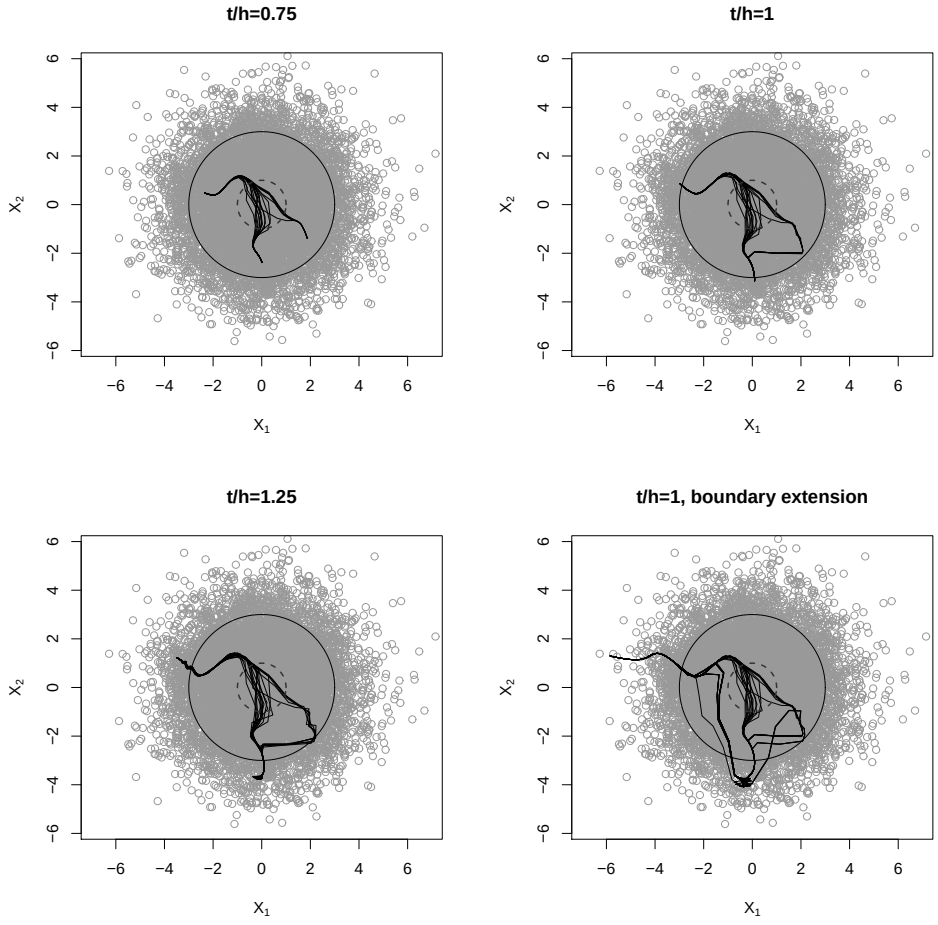

Figure 4.6: 20 local principal curves, all with $h = 1$, and $t = 0.75$ (top left), $t = 1$ (right), and $t = 1.25$ (bottom left) through a multivariate Gaussian sample of size $n = 10000$ with $\sigma^2 = 3$. The bottom right plot uses the boundary extension proposed in Section 4.3. The outer (solid) circles have radius σ^2, and the inner (dashed) circles radius 1.

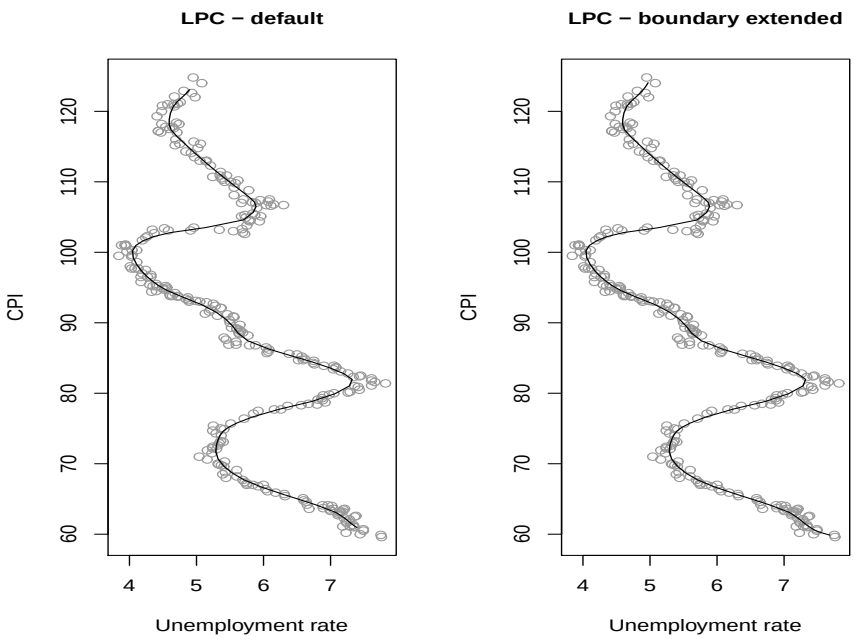

Figure 4.7: Local principal curve - the effect of the boundary extension

right panel of Figure 4.6. Compared to the non-extended fit, it is clear that, after applying the boundary extension, the local principal curves reach further into the boundary region of the data cloud.

As a second real data example, Figure 4.7 shows a local principal curve fitted to time series data for monthly unemployment and inflation rates in the US from March 1984 until April 2008[(1)]. The left panel shows the LPC fitted using default settings, while the right panel shows the fitted curve using the LPC algorithm adjusted for boundary extension. It is apparent that, in the latter case, the curve reaches more data at both boundaries.

[(1)]For more information about this data, please refer to the 'Phillips Curves' application in Chapter 5.

Chapter 5

Applications

5.1 Introduction

Local principal curves can provide a very useful tool in a wide range of high-dimensional data studies, specially when aiming to visualise and analyse this data with a significantly reduced number of dimensions. On one hand, the idea of graphically representing multidimensional data in a simple one dimensional plot for the fitted curve will make it easier to gain some initial information about the basic shape of the data set and also helps detecting basic patterns, which is useful for doing further statistical inference and analyses. On the other hand, the fact that the curve is parametrised over some parameter τ is of great importance if a link is recognised between the curve parametrisation and some real variable(s) which are thought to be related to the data set under study. This link can be used in predicting real multidimensional data points with information about the link variable(s) only.

In this chapter, we present some possible econometric applications for localised

principal components and curves, namely; insurance market key indicators, Phillips Curves, the relationship between gold and the dollar and price index construction. This does not mean that LPCs are limited to econometric applications only. In fact, the LPC algorithm has the flexibility to deal with a wide variety of multidimensional data, but it is just our current field of interest.

As introduced earlier in Chapter 4, the performance of the LPC algorithm is expected to be better after applying the methodological improvements of the boundary extension and the automatic selection of the starting point. Extending the curve at data boundaries can be desirable specially for data of time series character and using the MS algorithm for choosing a suitable starting point considerably increases the robustness of the method.

If not stated otherwise, the enhanced LPC algorithm is used in all the applications introduced in this chapter.

5.2 Insurance Market - Key Indicators

Insurance industry is considered one of the main branches of financial services business all over the world. The analysis of aggregate insurance data, specially multidimensional, can be troublesome because of the large number of variables included and also the expected high correlation between those variables. It could be of interest and benefit for analysing such data to try expressing the data space in fewer dimensions which makes extracting useful summaries and information from the data relatively less problematic.

In this section, we shall provide an example that involves applying both PCA and LPC-based techniques for the purpose of analysing aggregate insurance data.

The data sample consists of seven key indicators for the insurance market in EU member states, Iceland and Norway (25 countries) for the year 2006 [1, 35]. The variables of interest are as follows:

```
BST: Balance sheet total. (EUR million)
Emp: Number of persons employed.
GCP: Gross claims payments (EUR million):
     Actual claims paid as indemnities to policyholders.
GOE: Gross operating expenses (EUR million):
     Mainly, administrative expenses and acquisition costs.
GPW: Turnover or gross premiums written (EUR million):
     Total premiums for current (valid) insurance policies
     (before deductions for reinsurance and commissions).
NoE: Number of enterprises.
TCR: Total capital and reserves. (EUR million)
```

All the key indicators above are extracted for both Life and Non-Life insurance industries in the EU countries in 2006[(1)].

Descriptive statistics (means and standard deviations) and correlations for the EU life insurance data are displayed in Tables 5.1 and 5.2 respectively. Tables 5.3 and 5.4 show descriptives and correlations for non-life insurance data.

It is apparent from Tables 5.1 and 5.3 that there exists large variation in the data set meaning that there is a considerable amount of information contained within the data. Looking into correlations (Tables 5.2 and 5.4), we observe that most of the variables (key indicators) are highly correlated, which suggests that there is

[(1)]Life insurance includes life insurance and life reinsurance with or without a substantial savings element. Non-life insurance includes insurance and reinsurance of non-life insurance business (accident; fire; health; property; motor, marine, aviation, transport; pecuniary loss and liability insurance) [35].

	Mean	St.Dev.
NoE	34.96	38.41
GPW	18794.74	43672.74
Emp	7750.72	17188.20
GCP	15389.42	41120.71
GOE	1590.13	3523.48
BST	160113.34	380199.73
TCR	60854.68	271520.48

Table 5.1: Mean and standard deviation for the EU life insurance business key indicators

	NoE	GPW	Emp	GCP	GOE	BST	TCR
NoE	1.00	0.80	0.79	0.76	0.82	0.81	0.59
GPW	0.80	1.00	0.91	0.99	0.96	0.98	0.90
Emp	0.79	0.91	1.00	0.90	0.95	0.93	0.77
GCP	0.76	0.99	0.90	1.00	0.95	0.99	0.95
GOE	0.82	0.96	0.95	0.95	1.00	0.98	0.80
BST	0.81	0.98	0.93	0.99	0.98	1.00	0.90
TCR	0.59	0.90	0.77	0.95	0.80	0.90	1.00

Table 5.2: Correlations for the EU life insurance business key indicators

a problem of information redundancy due to these powerful correlations.

The fact that principal components represent standardised linear combinations of the original variables and that they are 'orthogonal' (uncorrelated) implies that a primary purpose of PCA is to eliminate information redundancy, along with dimensionality reduction [3].

Since the basic summary statistics for the data show large variability and strong correlations among variables, this suggests that using PCA-based techniques could be useful in terms of summarising such data through a less-dimensional uncorrelated set of components (whether linear in the form of lines or nonlinear in the form of curves).

As a pre-analysis step, to adjust the data set for the clear differences in the measurement scales of the variables, all variables are log-transformed (Figures

	Mean	St.Dev
NoE	77.88	90.43
GPW	11844.19	23457.41
Emp	21588.24	37101.46
GCP	7642.28	14421.03
GOE	2298.67	4425.05
BST	38510.25	81488.13
TCR	6681.52	11415.17

Table 5.3: Mean and standard deviation for the EU non-life insurance business key indicators

	NoE	GPW	Emp	GCP	GOE	BST	TCR
NoE	1.00	0.85	0.86	0.89	0.87	0.81	0.89
GPW	0.85	1.00	0.98	0.99	0.96	0.99	0.94
Emp	0.86	0.98	1.00	0.98	0.94	0.96	0.94
GCP	0.89	0.99	0.98	1.00	0.98	0.97	0.93
GOE	0.87	0.96	0.94	0.98	1.00	0.93	0.86
BST	0.81	0.99	0.96	0.97	0.93	1.00	0.95
TCR	0.89	0.94	0.94	0.93	0.86	0.95	1.00

Table 5.4: Correlations for the EU non-life insurance business key indicators

5.1 and 5.2). The feature of strong linear correlations between variables is still easily noticed.

5.2.1 Classical Principal Component Analysis

Life Insurance Data

Now, to analyse the EU life insurance market aggregate data, principal component analysis shall be carried out for the EU life insurance data in order to try solving the redundancy problem and represent the multidimensional highly-correlated data by a group of uncorrelated components taking into consideration that most variation in the data set is to be retained. Summary PCA results are displayed in Table 5.5.

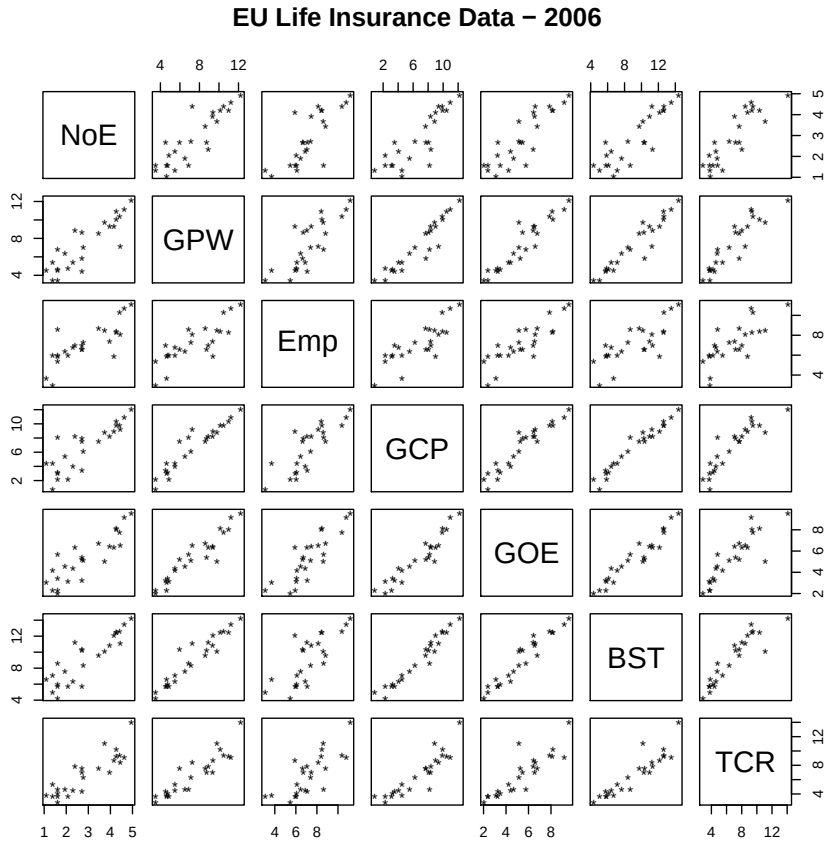

Figure 5.1: EU life insurance market key indicators 2006 (log transformed)

According to the Kaiser-Guttman criterion [60, 84], the results shown in Table 5.5 suggest that we keep the first three principal components as their standard deviations are greater than one. The proportion of variance accounted for by the first three components is 97.56%, which makes the presence of the rest of the components nearly negligible. In other words, using PCA the seven-dimensional data has been transformed into a three-dimensional set of components losing only 2.44% of the total variability in the original data space. The first PC by itself extracts 92.16% of the total variance (information) contained within the data.

Capturing more than 90% of the variation in the data, the first PC can provide a reliable classification of the 25 countries in the data set. This type of analysis

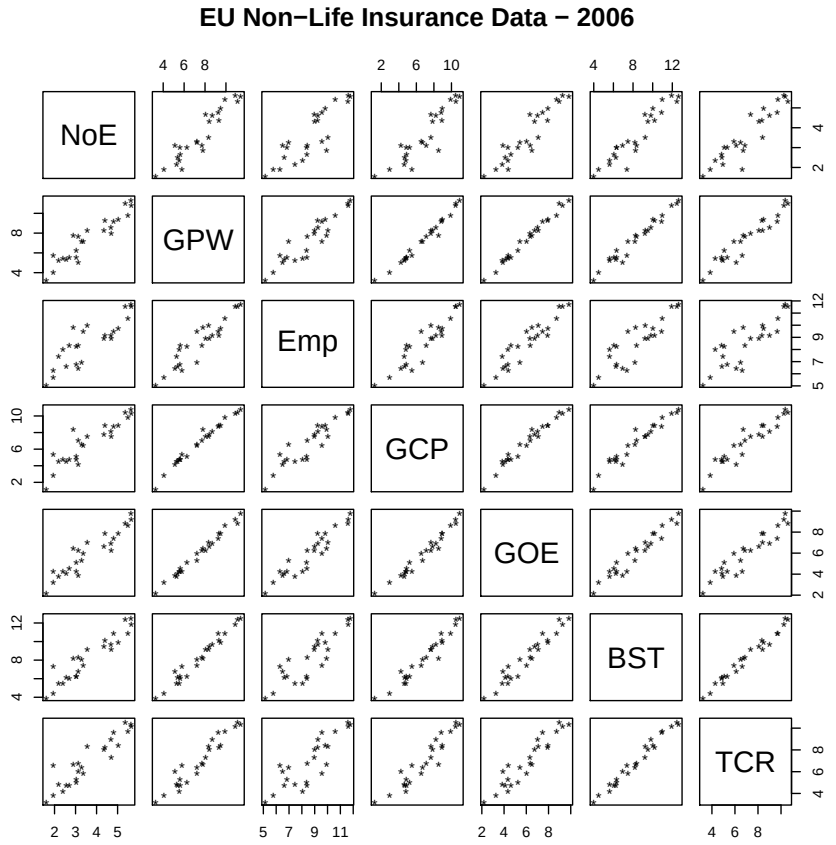

Figure 5.2: EU non-life insurance market key indicators 2006 (log transformed)

can be done through the values ('scores') of the first PC along all the rows in data matrix (the 25 countries), which are plotted as in Figure 5.3.

It can be concluded from Figure 5.3 that the leading five countries in the EU life insurance business in 2006, among the 25 countries included in the analysis were UK, Germany, France, Italy and Netherlands respectively. The countries with smaller scores are less influential, like Iceland and Latvia which came at the end of the list. The scores of the first three PCs with country labels are shown in Figure 5.4.

One of the main and important results of the PCA is the correlation between

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Standard deviation	6.3271	1.1544	1.0064	0.6700	0.5785	0.4545	0.2634
Proportion of Variance	0.9216	0.0307	0.0233	0.0103	0.0077	0.0048	0.0016
Cumulative Proportion	0.9216	0.9523	0.9756	0.9859	0.9937	0.9984	1.0000

Table 5.5: Summary of PCA results for EU life insurance data

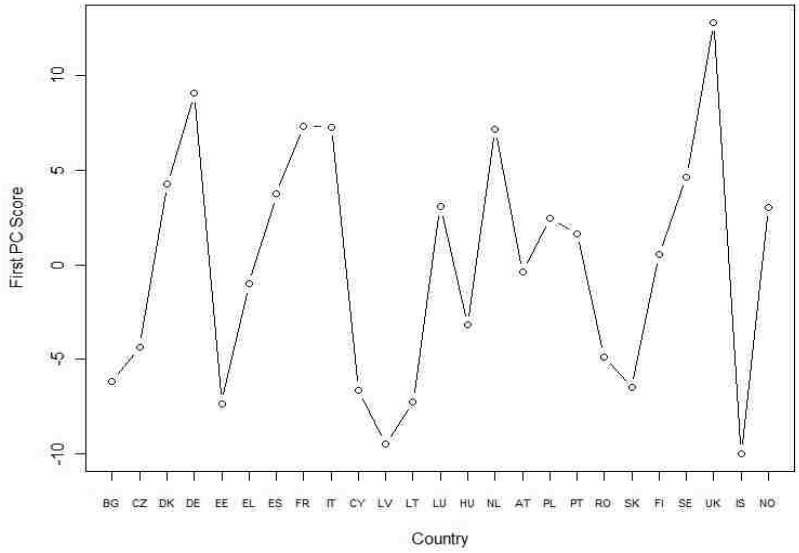

Figure 5.3: Country scores for the first principal component - life insurance

variables and factors, which is usually referred to as 'loadings'. Loadings provide useful interpretations in terms of understanding the components' structure and the common effects for the variables. Table 5.6 displays the loadings' values for the first three principal components.

Taking into consideration the highest loading in each row in Table 5.6, we can say that the first principal component represents the mutual effect of four variables: Gross premiums written, gross claims paid, gross operating expenses and balance sheet total. Those variables may very well represent the overall operating efficiency and outcomes for the insurance market. Considering only relatively large loadings [72], with absolute value greater than 0.4, would lead to a similar interpretation. The second principal component is mainly representing the firm

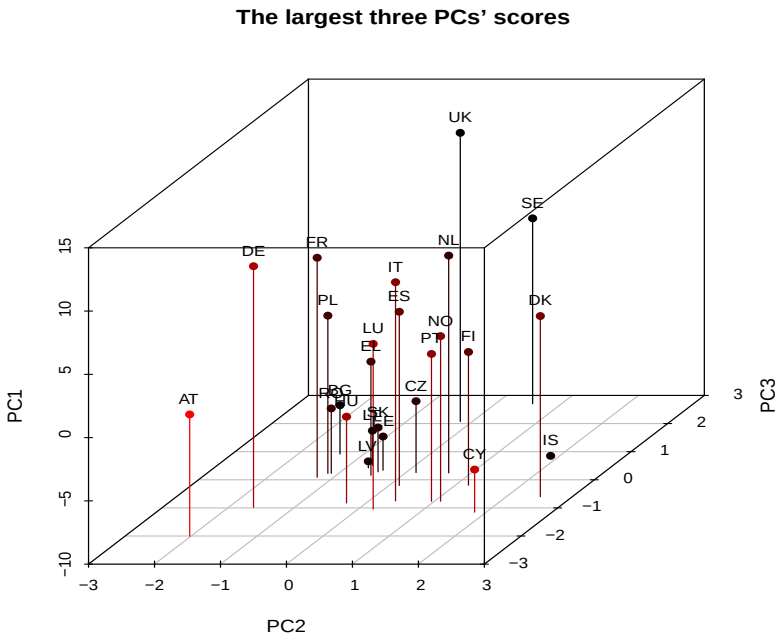

Figure 5.4: Scores for the largest three principal components - life insurance

size in terms of the number of persons employed. The third principal component can be interpreted as the firm solvency in terms of the total capital and reserves.

Non-Life Insurance Data

A similar analysis of that performed using life insurance data is carried out using the data of key indicators for the EU non-life insurance business.

Running the principal component analysis on the EU non-life insurance data (the data are log-transformed), the summary results were as shown in Table 5.7. Only the first PC is to be retained. The first component accounts for 96% of the total variation, which means that we are able to express the seven-dimensional data by only one component losing only 4% of the information contained within the data.

	PC1	PC2	PC3
NoE	0.17	-0.04	0.18
GPW	0.40	0.01	-0.02
Emp	0.26	-0.88	0.22
GCP	0.49	0.03	-0.39
GOE	0.33	-0.17	-0.25
BST	0.46	0.29	-0.31
TCR	0.43	0.32	0.78

Table 5.6: Loadings for the first three PCs - life insurance

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Standard deviation	5.4741	0.8715	0.4734	0.3945	0.2508	0.1912	0.1041
Proportion of Variance	0.9600	0.0243	0.0072	0.0050	0.0020	0.0012	0.0003
Cumulative Proportion	0.9600	0.9843	0.9915	0.9965	0.9985	0.9997	1.0000

Table 5.7: Summary of PCA results for EU non-life insurance data

The scores of the first principal component were as shown in Figure 5.5. It can be concluded from the graph that the non-life insurance business results for Germany, UK and France are the most influential among all the EU states in 2006.

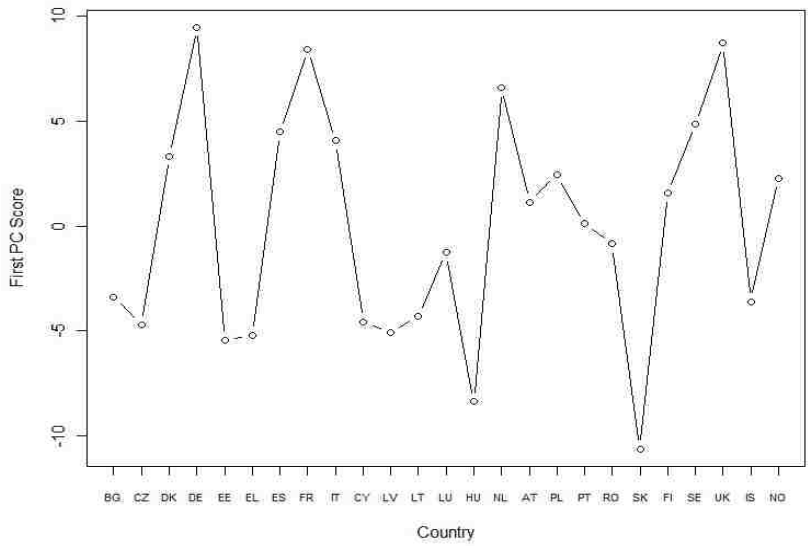

Figure 5.5: Country scores for the first principal component - non-life insurance

The loadings for the first principal component (displayed in Table 5.8) give an indication of the contribution of each variable towards the one-dimensional data summary obtained through the first PC. The largest values were that of: Gross claims paid (GCP), balance sheet total (BST) and gross premiums written (GPW) respectively.

	NoE	GPW	Emp	GCP	GOE	BST	TCR
Loadings	0.21870	0.40590	0.31227	0.45467	0.36929	0.44212	0.38902
Sq. Loadings	0.04783	0.16475	0.09751	0.20672	0.13638	0.19547	0.15134

Table 5.8: Loadings and squared loadings for the first PC - non-life insurance

5.2.2 LPC-Based Analysis

In this section, the insurance business key indicators will be analysed using *localised* principal components and curves. Unlike the traditional *global* principal component analysis, the localised version takes into consideration the local topology of the data. This is important, specially in situations when there exist significant clusters in the data space.

This type of analysis will involve using kernels and bandwidth (window size) as introduced in the preceding chapters. This is to choose the type of weighting (kernel functions) for localised principal component analysis and to define the size of the local neighbourhood (the entries of the bandwidth matrix $\boldsymbol{H}$).

Adapting a localised approach for analysing the data through fitting a local principal curve will result in the data dimensions being reduced only to one represented by the LPC. One can conclude from the PCA results that this should be acceptable knowing that the first principal component accounted for more than 90% of the total variation for both life and non-life insurance data sets. In fact, this

proportion of variance explained by the first principal component line is expected to get higher when we replace that line by a curve which captures more of the variation due to its locally-based method of fit.

Figure 5.6 displays the percentage of variance accounted for by the local principal curve fitted to life-insurance data compared to that of the first global principal component. The solid curve is a truncated smoothed version of the percentage of variance captured through the first local eigenvectors along the fitted LPC. The percentage of variance accounted for by local eigenvectors is computed as in (2.13). We have truncated the values at boundaries, in particular when the difference between two successive centres of mass falls below the pre-determined threshold for applying the boundary extension (4.30). These values near boundary regions were almost constant and very close to 100% (This is expected as, at boundaries, the number of data points usually gets smaller and the bandwidth nearly covers all the data points in the neighbourhoods close to boundaries, and hence the local variance is almost 'totally captured' by the LPC).

Since the data are not perfectly linear, the non-linear summary for this data (LPC) is expected to be better than the linear summary (first principal component). Although the latter conclusion is *statistically* generally acceptable and correct, it is not expected that, *numerically*, this will always be the case, specially for complex data structures. This is mainly because, in each local neighbourhood, we only have a short scene of the whole data such that the direction of the first local eigenvector may not always be identical to that of the global first principal component which is fitted based upon the whole scene of the data.

The average (mean) percentage of variance captured through the first local eigenvectors along the fitted LPC for life insurance data is approximately 86.84%, which is, as expected, slightly less than the variance explained by the largest

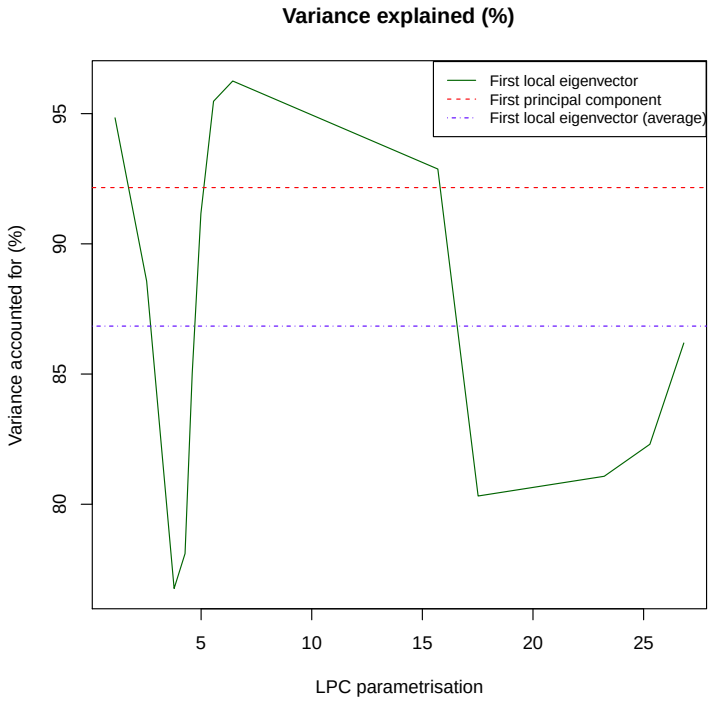

Figure 5.6: Variance accounted for by the LPC - life insurance

global principal component (92.16%). However, in the middle part of the curve, the first local eigenvector was outperforming the first global principal component. Furthermore, the average cumulative percentage of variance captured through the first three *local* eigenvectors along the fitted curve for life insurance was 99.22% which is greater than 97.56%, the cumulative proportion of variance for the first three *global* principal components (Table 5.5). For non-life insurance, the first local eigenvectors along the fitted curve captured around 94.99% on average of the variance (also truncated at boundaries).

Another important quantity of interest to compare the LPC fit with the first principal component line is the sum of squared Euclidean distances between the data points and their projections onto the line (curve). Applying (2.1), we can

compute the latter for the first principal component (see Appendix for R computations). For the LPC, we use the built-in *lpc.spline()* function in the LPCM package [21]. The sum of squared distances for the LPC fit for life insurance, data was 7590.09 compared to 7608.18 for the first principal component, and for non-life insurance 7380.86 and 7647.67 respectively.

The symmetric kernel function used for the LPC fit for insurance data is a Gaussian kernel. Also, if needed, the initial default choice of the bandwidth vector $(\text{diag}(\boldsymbol{H}^{1/2}\boldsymbol{I}) = h_1, \cdots, h_7)$ can be checked and improved using any automatic bandwidth choice function in R (for this purpose, the R package *KernSmooth* was used as a direct plug-in methodology to select the bandwidth suitable for kernel density estimate for each variable [79, 80]).

The LPC boundary-extended algorithm was used to fit both life and non-life insurance data. It is quite hard to plot the fitted curve as we need a seven-dimensional graphics tool which is not available yet in R. The LPC algorithm automatically produces the 2D plots with the fitted LPC for all pairs of variables(see Figure 5.7). For illustration purpose only, a 3D projection plots for the fitted seven-dimensional LPCs for life and non-life insurance data are shown in Figures 5.8 and 5.9 respectively.

Similar to the traditional PCA, we are interested in the local contributions of the variables (insurance business key indicators), i.e. the loadings in terms of localised eigenvectors. For the standardised version of eigenvectors, a useful feature for examining the total variance explained by the curve is that at every point on the curve, the sum of squared loadings of the first eigenvector should be equal to one ($\|\boldsymbol{\gamma}^{\boldsymbol{x}}\|^2 = 1$). Figure 5.11 displays the 'local' loadings for each variable for the non-life insurance data. Figures 5.10 and 5.12 show the cumulative squared

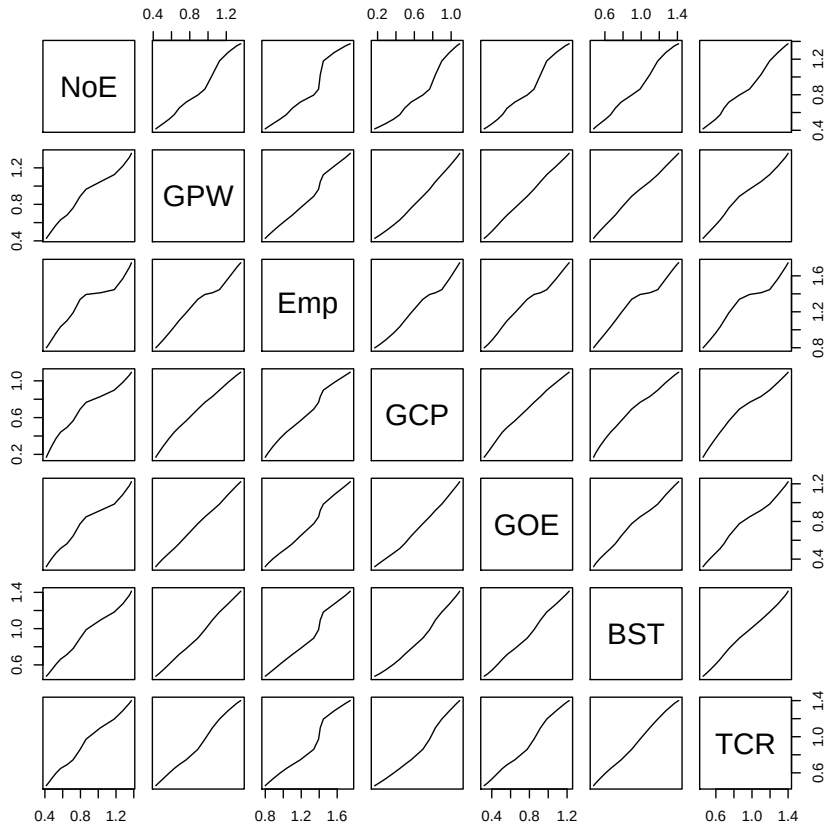

Figure 5.7: The fitted LPC - non-life insurance

loadings of the first eigenvectors for life and non-life insurance data respectively.

The loadings' plots show the contribution of each variable along the fitted curve. These plots can be quite informative in some sense. For instance, if the curve parametrisation can be thought of as some measure of scale (size), loadings in this sense can provide a tool to observe changes of variables' effects with respect to the insurance market size. For example, we can notice that the effect of the number of enterprises (NoE) is almost negligible for small life insurance markets and that the influence of total capital and reserves (TCR) generally increases for big life insurance markets (Figure 5.10). Also, it can be observed that claims paid (GCP) has a larger effect for smaller non-life insurance markets(Figure 5.12).

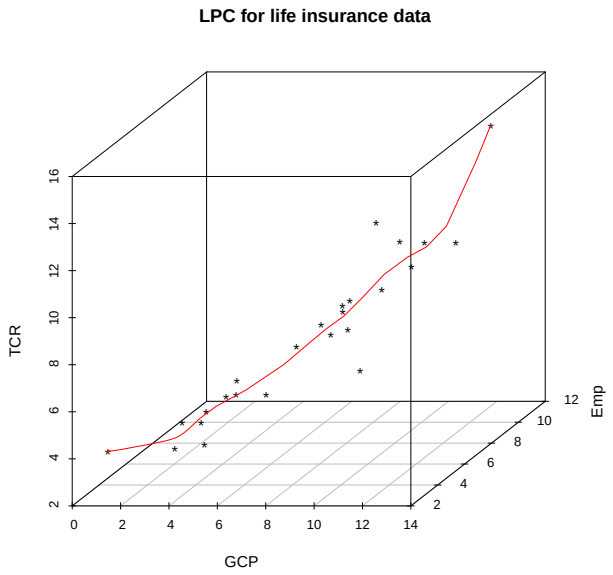

Figure 5.8: A 3D plot for the fitted LPC - life insurance

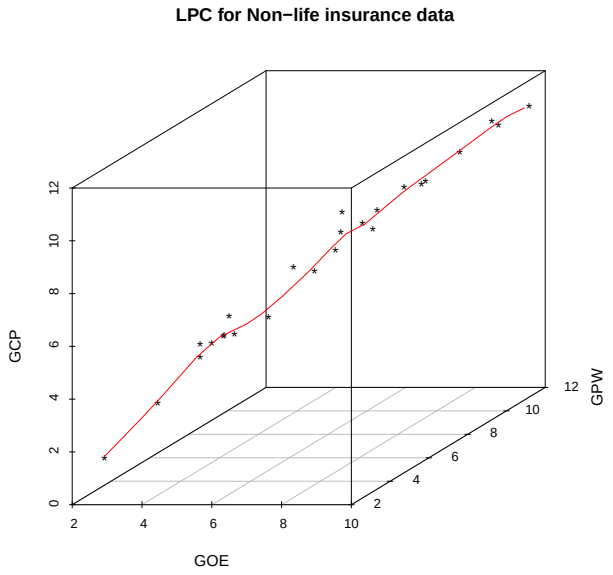

Figure 5.9: A 3D plot for the fitted LPC - Non-life insurance

The squared loadings for life and non-life data are displayed in Tables 5.9, 5.10 respectively. For life insurance, the three most influential variables were total

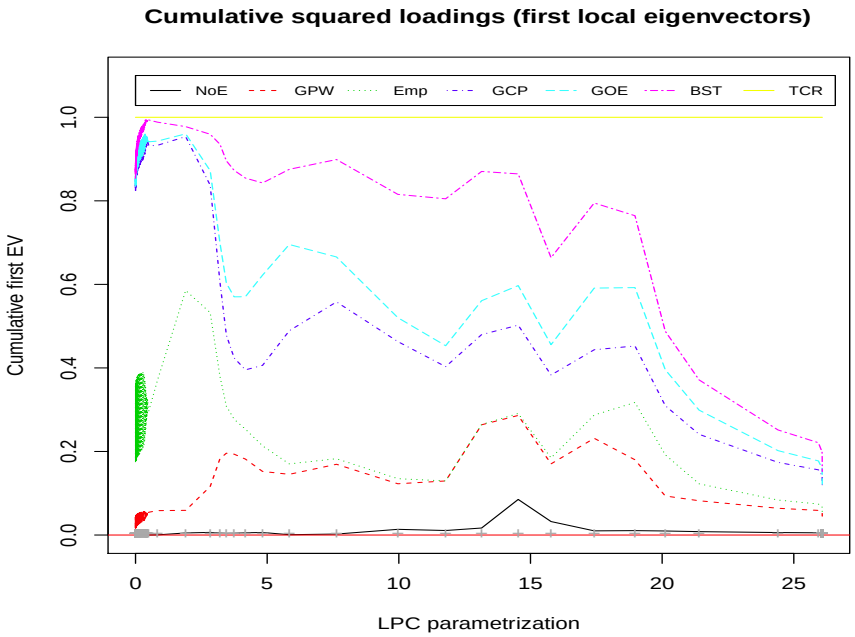

Figure 5.10: Cumulative squared loadings - life insurance

capital and reserves (TCR), gross claims paid (GCP) and number of employees (Emp) respectively. For non-life insurance, gross claims paid (GCP), gross operating expenses (GOE) and gross premiums written (GPW) were respectively the three most important variables.

Variable	NoE	GPW	Emp	GCP	GOE	BST	TCR
Average Sq. Loadings	0.003	0.054	0.136	0.349	0.020	0.056	0.358
Total Sq. Loadings	0.557	8.741	22.067	56.510	3.195	8.992	57.940

Table 5.9: Variables' loadings for the fitted LPC - life insurance

Variable	NoE	GPW	Emp	GCP	GOE	BST	TCR
Average Squared Loadings	0.015	0.139	0.054	0.327	0.283	0.133	0.048
Total Squared Loadings	2.847	26.166	10.142	61.445	53.123	25.068	8.960

Table 5.10: Variables' loadings for the fitted LPC - non-life insurance

We can compare the total squared loadings displayed in Table 5.10 (the LPC fit) with those displayed earlier in Table 5.8 (largest principal component fit). We

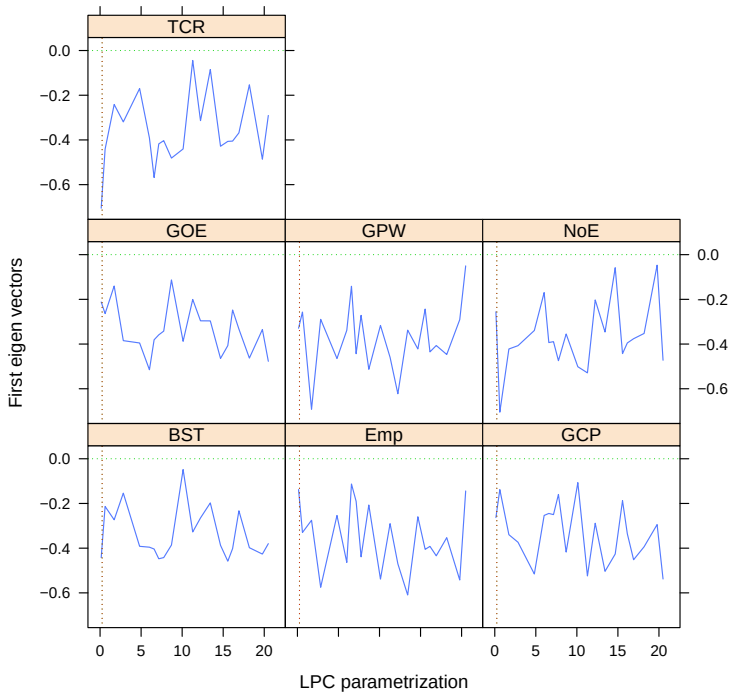

Figure 5.11: Loadings (first localised eigenvectors) - non-life insurance

can notice that the largest (GCP) and smallest (NoE) contributions are the same, while the order of the remaining variables has slightly changed.

Loadings in terms of the squared values of the first localised eigenvectors may have other useful interpretations, in particular, when linked to some other external variable(s) that may be closely related to the data. This is done through the curve parametrisation τ, a matter which shall be further illustrated later on in this chapter.

5.3 Phillips' Curves

'Phillips curves' is a famous term in economics. It refers to curves that study the relation between unemployment and the rate of inflation in an economy. It was

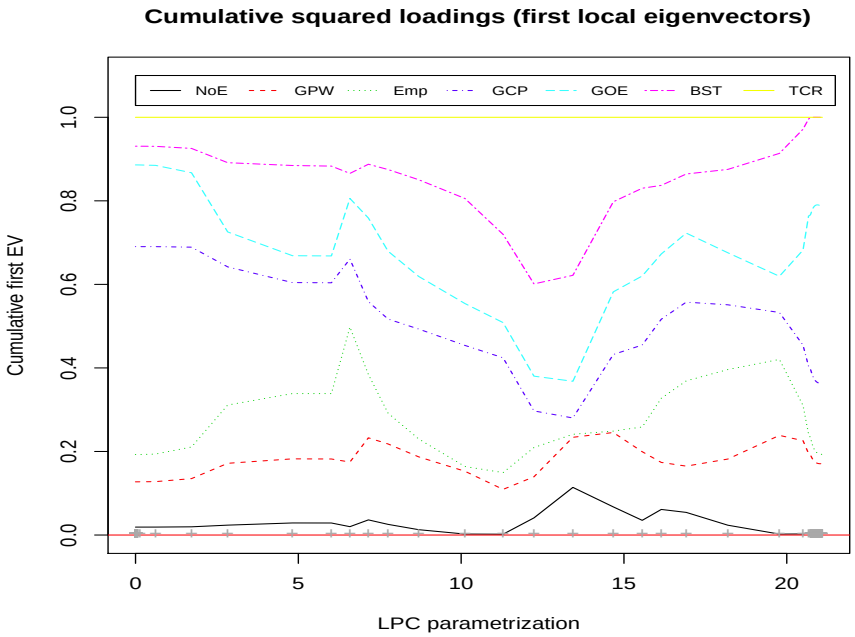

Figure 5.12: Cumulative squared loadings - non-life insurance

named after the famous economist *Alban William Phillips* who was responsible for the first appearance of Phillips curves when he wrote a paper in 1958 in which he observed an inverse relationship between money wage changes and unemployment in the British economy over the period 1861-1957 [62]. After Phillips' work in 1958, there were several and more recent versions of these curves [30, 32]. The basic conclusion suggested by analysing Phillips' curves is that the lower the unemployment in an economy, the higher the rate of inflation.

In this study, we shall introduce the local principal curve as a possible estimate (fit) for Phillips curves through exploring the relationship between unemployment rate and consumer price index (CPI) which is the most commonly used measure of inflation. Consumer price indices measure price changes for household goods and services.

The data sample used consists of[(1)]:

- Consumer price indices (all goods and services) for both UK and US. (monthly data)
- Unemployment rates for both UK and US. (monthly data)

Our data sample covers the period from Jan 1975 to April 2008.

Now, we shall try to fit a local principal curve through the two-dimensional unemployment-CPI data. First, we explore the data by a simple summary and a scatter plot (see Figure 5.13). This is important for identifying the basic characteristics of each variable, and hence forming an initial idea about the possible adequate choices for the parameters of the LPC algorithm, especially the bandwidth matrix $\boldsymbol{H}$ and the starting point $\boldsymbol{x}_{(o)}$.

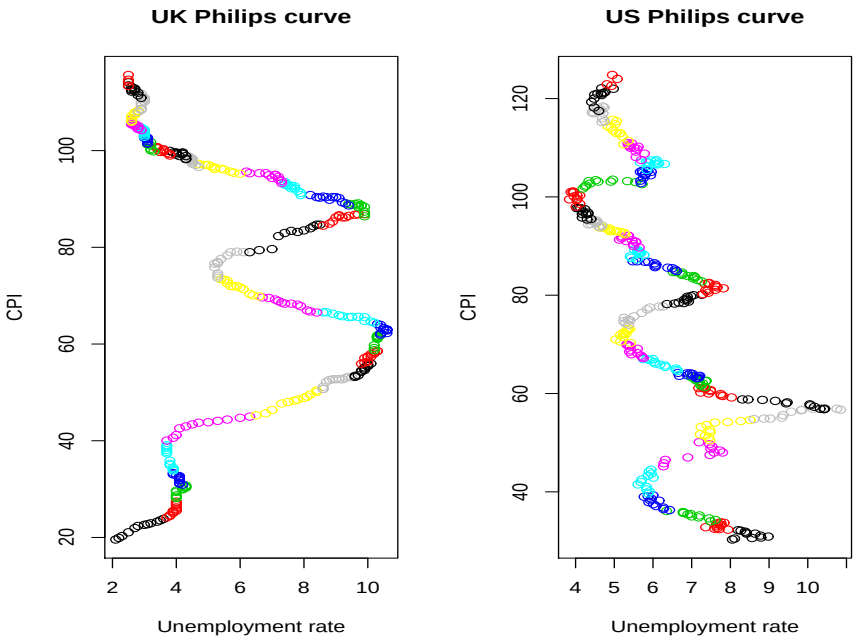

Figure 5.13: Unemployment vs. Inflation for UK and US(Jan 1975 - April 2008)

To fit a curve through the data, one usually starts with a random starting point

[(1)]Data source: University of Manchester, Source data from OECD, OECD Publications.

and an arbitrary choice of the bandwidth vector $\text{diag}(\boldsymbol{H}^{1/2}\boldsymbol{I}) = h_1, h_2$. We can then use the 'coverage' concept to check if the bandwidth selection is suitable for the data. We also see if there is any sensitivity to the starting point selection (if there is a trace of this, enabling the mean shift algorithm for modifying the choice of $\boldsymbol{x}_{(o)}$ is recommended).

After some effort, one should reach a reasonable combination of bandwidths that produces a 'good-looking' curve. The bandwidth vector used for both data (UK and US) was $\text{diag}(\boldsymbol{H}^{1/2}\boldsymbol{I}) = (0.5, 1.0)$. The starting points were automatically chosen through applying the MS algorithm. The LPC for our unemployment-inflation data is plotted in Figure 5.14.

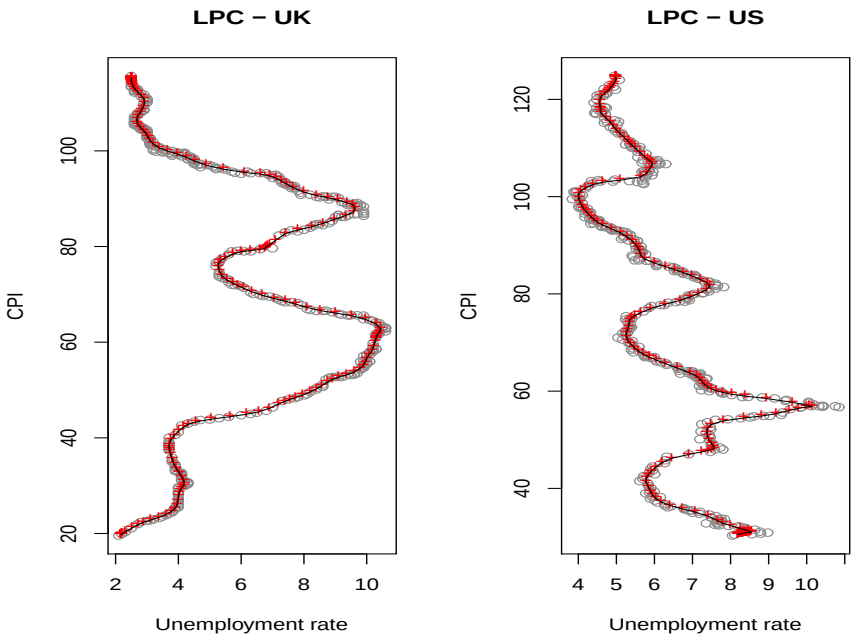

Figure 5.14: LPC fit for Phillips data

It may be useful to conclude something from our trials until reaching a good fit. In this context, the following about bandwidth selection is true: If the LPC is monotone in one of the variables, then the bandwidth orthogonal to this direc-

tion is not relevant as long as it is large enough. For our current data set, the curve flows into a vertical direction with the inflation rate, so the choice of the bandwidth for CPI is important, however, the bandwidth for unemployment rate is not actually a main concern. Another note that is worth saying is that if the curve does not seem to flow into one direction, and the variables are measured on significantly different scales, it is important to consider re-scaling or transforming some variables in order to increase data consistency and improve the curve fitting process.

After fitting the LPC, one should go further and think about prediction. The key issue now is to look for some link between the curve (actually what is meant here is the parametrisation along the curve) and another real variable. In most econometric applications, thinking about 'time' to play this link rule is a natural choice. The time variable has a major role in economics, specially when analysing past and expected future behaviour of some economic indicators. It can also explain and clarify some facts about systematic cyclic behaviour.

It is worth trying to investigate the link between time and parametrisation along the fitted LPC, since both of them increase in a monotone way. For our example, this link can be represented as in Figure 5.15 and can be referred to as 'calibration curve'. If the link is looking reliable and seems to be unique for each value, we can create some functional form or fit a spline to represent the relation between time and the LPC parametrisation. Prediction will be possible once a spline has been fit.

The prediction process typically goes as follows:

(i) Use the calibration curve to predict curve parametrisation given time.

(ii) Predict multidimensional data points, corresponding to the parametrisation

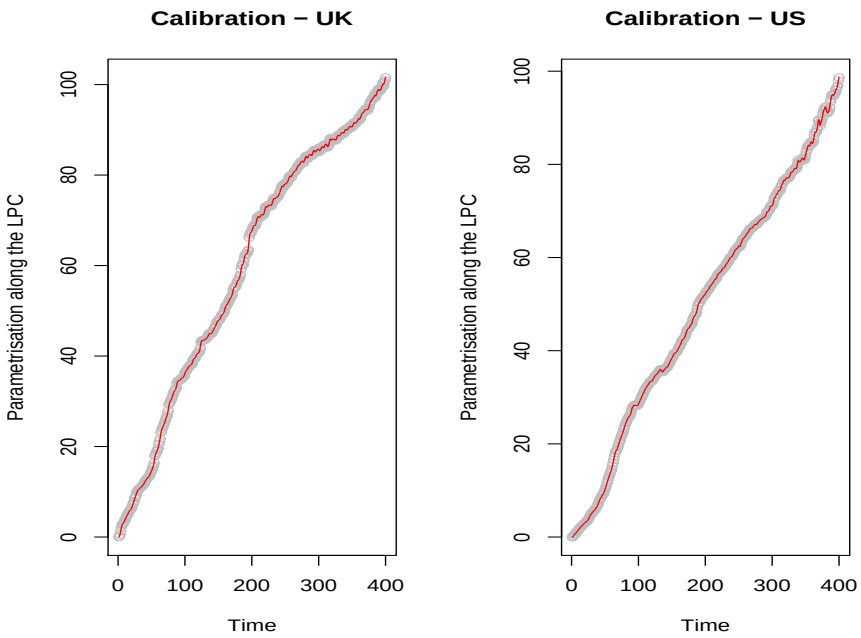

Figure 5.15: Calibration curves for Phillips data

value(s), simultaneously through the fitted LPC.

To check for this, using the US data set, assume that we want to do the prediction process given that time = 200. The following representation shows the prediction process from time=200 to the corresponding real two-dimensional data point:

$$\text{time} = 200 \Rightarrow \text{LPC-parameter} = 52.165 \Rightarrow \text{unemployment-rate} = 6.92, \text{CPI} = 79.34$$

The corresponding real values for unemployment rate and CPI were 6.87 and 79.30 respectively. This shows that using calibration and fitting splines do produce very good estimates.

One last thing that may need further investigation regarding this data is the cyclic[(1)] behaviour of data in both countries, UK and US. However, one can

[(1)] In Economy, a cycle shows the fluctuation between times of expansion and contraction. In Macroeconomics, a cycle is usually referred to as 'economic' or 'business' cycle.

already notice that:

- Cycles happen in both countries.
- Cycles happen more rapidly in the US compared to the UK (US cycle-time length is less).
- A cycle starts first in the US, then, after some time, a corresponding cycle starts in the UK.

The matter of detecting and measuring cycles within the context of LPCs and comparing two or more curves can be a subject for expected future research.

5.4 Gold and Currency

In the theory of Economy, it is thought that there is a relation between gold price and currency exchange rate. This comes from an argument which states that one of the reasons for holding gold is hedging against currency movements [52]. This suggests that there is some relationship between gold prices and currency exchange rates, which shall be subject to investigation through LPCs.

Data:

- Gold price per fine troy ounce in GBP.
- Average spot exchange rate[(1)] US Dollar $ into UK Sterling £.

The data sample covers the period from 2nd Jan. 1979 to 16th July 2008 - daily values (7468 observations - 253 working days per year)[(2)]. Logs are taken for all

[(1)] Also known as the 'current exchange rate'. This is the exchange rate used for immediate (spot) settlement for a transaction.

[(2)] Data source: Bank of England website - Statistical Interactive Database.

data to reduce measurement-scale effect. A two-dimensional scatterplot for this data is shown in Figure 5.16.

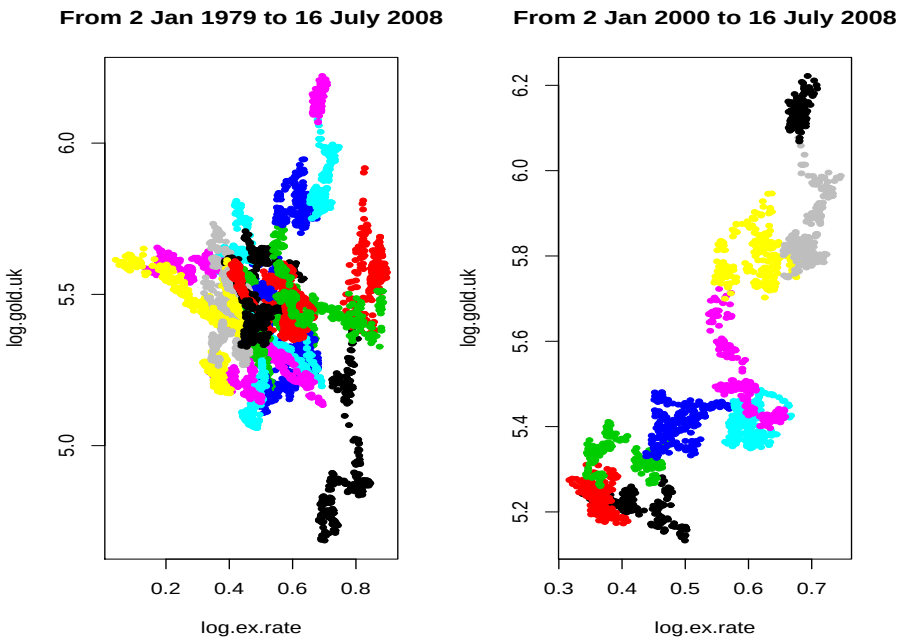

Figure 5.16: Log (£-$ exchange rate) vs. Log (gold price)

LPC for gold-dollar data:

In this part, for LPC application, only part of the total data sample (from 2 Jan 2000 to 16 July 2008) is used. This is for simplicity and to avoid the big data clustering that appears in the left panel of Figure 5.16. The reduced data sample contains 2158 observations.

A local principal curve for the gold-dollar reduced data sample is plotted in Figure 5.17. Using the MS algorithm, the starting point for the fitted curve was $\boldsymbol{x}_{(o)} = (0.36228, 5.21776)$. The small crosses along the curve in the plot represent the successive local centres of mass which construct the LPC.

Similar to the Phillips Curves' application, time is used as an additional external

variable that is assumed to have a link with the LPC parametrisation. Figure 5.18 represents the calibration curve for the gold-dollar reduced data sample.

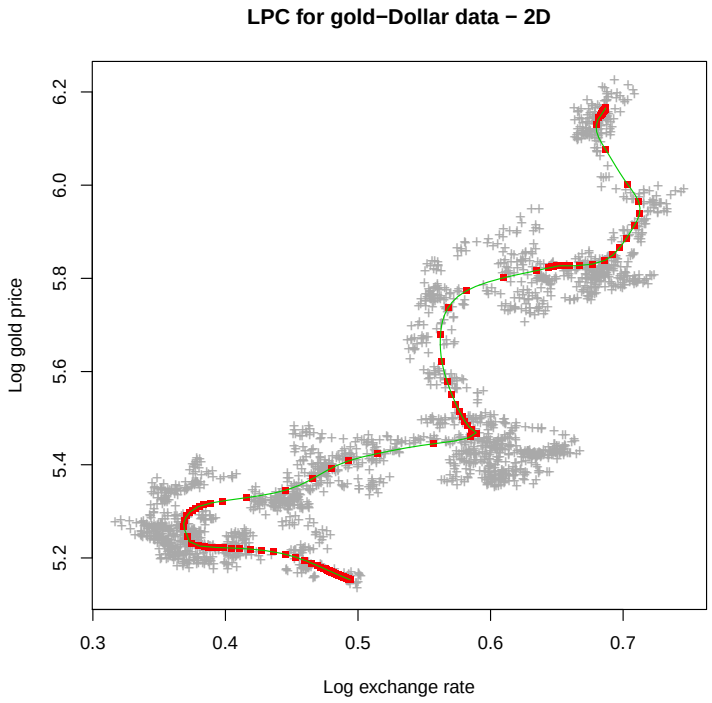

Figure 5.17: LPC : Gold price and the GBP(£)-USD($) exchange rate

Important questions immediately arise when examining the calibration curve: Does it represent a good calibration? Do we want this to be monotone? Can the link in this form be reliable in the prediction process? Does 'time' play a major role in forming a link with the curve? To get convincing answers for all these questions, we should make sure first that we have reached a reasonable trade-off between the smoothness of the local principal curve and that of the calibration curve. It is expected that as the smoothness of the LPC increases, the smoothness of the calibration curve decreases and vice versa. Maybe in the end, it is probably useful to accept some trade-off, and this may be affected by the main purpose of the analysis: Is it to reach a good representation or summary for the data rather than making predictions or not?

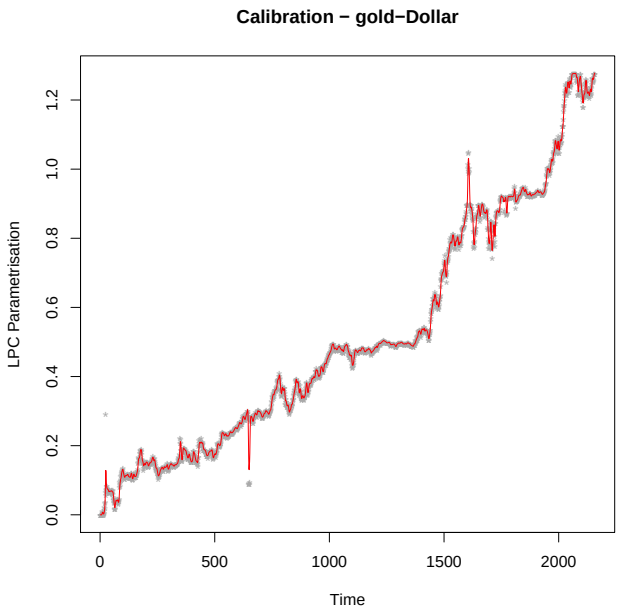

Figure 5.18: Calibration : Gold price and the GBP(£)-USD($) exchange rate

An alternative way for looking to the time variable is to add it to the main *lpc*(.) function as a third variable (dimension) and see if this improves the curve fitting for the data sample. In the sense of symmetry, this is not always the case, because other variables may depend on or be affected by time. (Of course, time is never expected to be dependent upon any other variable).

Figure 5.19 shows a 3D representation of the data with two alternative approaches to fit a principal curve, the first is the original HS approach (which has been applied using the R functions *principal.curve(.)* [39] and *pcurve(.)* [41]), and the second is the LPC approach.

We can observe some improvement in the fitted LPC compared to the fit shown in Figure 5.17. In terms of the total variance explained, the LPC fitted to the two-dimensional data set captured around 82.94% on average of the local variance in all neighbourhoods along the curve, while the fitted LPC for the three-dimensional data (after adding time) captured almost 99% of the variance.

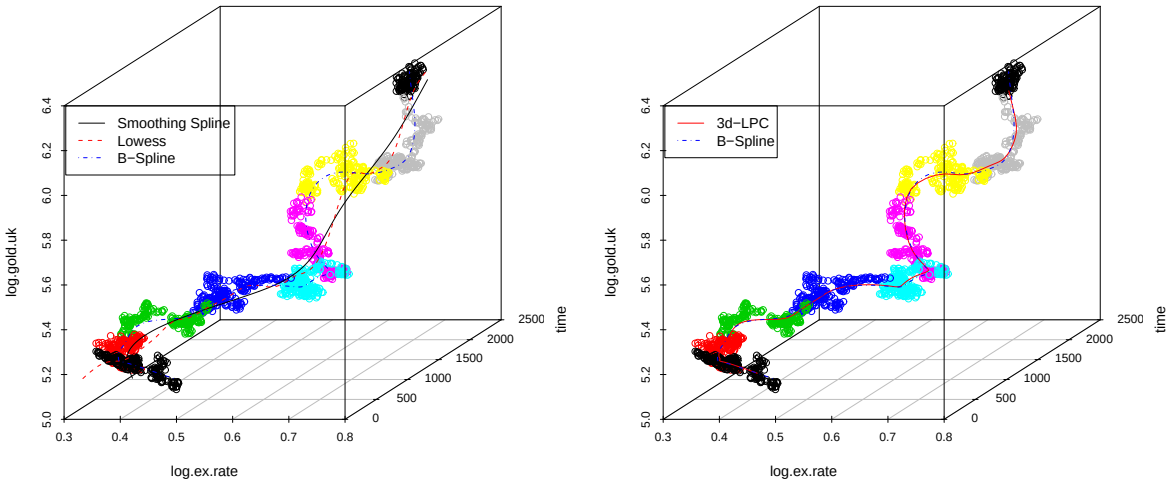

Figure 5.19: 3d HS(left) and LPC(right) curves for gold-dollar data

After all, a question remains; is it worth adding time to the original two-dimensional data? The value and benefits of this type of three-dimensional curve fitting is still to be explored in future research.

5.5 Consumer Price Index Construction

5.5.1 Introduction

Index number construction is an important and traditional subject in both the statistical and the economical sciences. A standard problem in Economics is the question of how to construct a single (summary) index from a series of individual (sub-)indices. For instance, the main measure of inflation for national macro-economic purposes is the Consumer Price Index (CPI), which covers essentially the monetary expenditures on all goods and services by all households of a certain economy (for instance, the UK). This index, say X_0, is usually computed from

sub-indices $X = (X_1, \ldots, X_p)^T$ by weighted averaging of the form

$$X_0 = w_1 X_1 + \ldots + w_p X_p = w^T X \tag{5.1}$$

where $w = (w_1, \ldots, w_p)^T$ is a set of weights relating to the composition of expenditure, which is allowed to vary over time, i.e. $w = w(t)$. Economists have taken substantial efforts to derive formulas which give appropriate or 'representative' weights for a certain economy. The actual process of averaging in (5.1) is rather crude from a statistical perspective for the following reasons:

- It is highly dependent on outlying (potentially erroneous) data.
- It is not able to deal with missing data.
- It does not allow an analysis of the relative contribution of the sub-indices over time.
- It does not take into account the differing variability (information) contained in the indices at different time points (other than through the weights, perhaps).

Potential alternatives addressing these issues were already suggested by Tintner [76] and Moser [54] in the context of production and price indices, and labour market indicators, respectively. They proposed to construct a linear summary index by finding that linear combination $\boldsymbol{\gamma}^T X$ of $X_1, \ldots, X_p$ with maximal variance $\mathrm{Var}(\boldsymbol{\gamma}^T X)$ among all unit vectors $\boldsymbol{\gamma}$. The solution to this problem is found via principal component analysis (PCA), and is given by the first eigenvector $\boldsymbol{\gamma}$ of the covariance matrix $\boldsymbol{\Sigma} = \mathrm{Cov}(X)$ of X. Assuming the existence of a 'price line' $X = aX_0 + \epsilon$, with $a \in \mathbb{R}^p$, Theil [74] developed a variant of PCA to estimate a and $\boldsymbol{\gamma}$ simultaneously. Neither of these authors used any additional weighting, though (external) weights w can be easily accommodated by considering

$X_w = (w_1 X_1, \ldots, w_p X_p)^T$ instead of X itself.

If we have a set of variables, each can be represented as a mix of a systematic component and an error, applying PCA to these variables results in constructing a number of independent factors, usually smaller in number than the data dimension, which capture most of the total variance in the data set. This is done by finding some linear function of the variables in the data set, which is least subject to errors. Principal components are of interest mainly in cases where the variables under consideration, the values of which formulate the data cloud, are considered to be symmetric, rather than one or more variable being generated from the remaining ones.

PCA-based approaches have not yet found widespread application in the context of economic index data. One reason for that is that PCA will find that line through the multidimensional cloud of indices which gives *globally* the best fit in terms of squared orthogonal distances; in other words 'one line has to fit it all'. The approximation done this way may be good in some parts of the data cloud but poor in others. As a consequence, the loadings (entries of $\boldsymbol{\gamma} = (\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_p)^T$) will reflect the contribution of the sub-indices $1, \ldots, p$ towards the overall index not equally well over the full data range — actually, the amount of information that individual indices contribute towards the overall index may vary greatly; an example for this is provided later in this chapter. Hence, what would be needed is a tool to maximise the variance locally, providing at each point the best local approximation to the data cloud. This implies that we need to fit a sequence of localised principal components, rather than one global principal component. The statistical concept corresponding to this viewpoint is a (local) principal curve.

Principal curves have recently attracted interest particularly in the engineering literature [53] due to their ability to extract low-dimensional 'features' from high-

dimensional data structures via the curve parametrisation τ. In particular, for $X \in \mathbb{R}^p$, one defines the *projection index* as the parameter of the closest point on the curve ($\boldsymbol{\nu}$) to X ((2.15)).

In our context, the extracted feature $\tau_{\boldsymbol{\nu}}(X)$ would be corresponding to the summary index of X, as we will illustrate in the following section. However, we are not only interested in this overall index, but also in the local contributions of the individual sub-indices, for which we need to determine loadings in terms of localised eigenvectors. The original HS algorithm for principal curves does not compute these, neither explicitly nor implicitly, so it is of limited use for our development. Alternatively using a method which is explicitly based on localised PCA is preferred.

As previously shown in Chapter 3, the local principal curve is determined by the series of the local centres of mass, $\boldsymbol{\mu}^{\boldsymbol{x}}$ values, and the actual localisation involved in the algorithm is performed through multivariate kernel functions. After termination of the algorithm, the parametrisation τ is calculated retrospectively through the Euclidean distances between neighbouring $\boldsymbol{\mu}^{\boldsymbol{x}}$, and interpolated between the $\boldsymbol{\mu}^{\boldsymbol{x}}$ through linear segments or cubic splines [24], yielding a fully parametrised one-dimensional curve $\boldsymbol{\nu}(\tau)$ through p-dimensional space, which passes precisely through all the local means $\boldsymbol{\mu}^{\boldsymbol{x}}$'s. Due to the localised way of averaging, the LPC algorithm is less robust to the outlying data points.

'Anchoring' the LPC:

For the LPC algorithm to be adapted for the role of summarising index data, an important adjustment is useful. Normally, there is some reference date for which all sub-indices take a baseline value, say 100, and also the overall index

takes this value. Hence, also the parametrised principal curve has to reflect this property and this can be realised through an *anchor:* This is a point of predetermined coordinates, say $\boldsymbol{x}_{(0)} = (100, \ldots, 100)^T$, and predetermined parameter value ('index') $\tau_{(o)} = 100$, through which the curve *is forced to pass.*

Recall that, normally, the LPC is fitted to the data through the following steps:

1. Choose a suitable starting point $\boldsymbol{x}_{(o)} \in \mathbb{R}^p$. Set $\boldsymbol{x} = \boldsymbol{x}_{(o)}$.
2. Calculate $\boldsymbol{\mu}^{\boldsymbol{x}}$.
3. Perform PCA locally at $\boldsymbol{x}$, yielding a localised eigenvector $\boldsymbol{\gamma}^{\boldsymbol{x}}$.
4. Find a new value for $\boldsymbol{x}$ by following $\boldsymbol{\gamma}^{\boldsymbol{x}}$ a predetermined step size, starting at $\boldsymbol{\mu}^{\boldsymbol{x}}$.
5. Repeat steps 2 to 4 until $\boldsymbol{\mu}^{\boldsymbol{x}}$ remains (approximately) constant.

Anchoring the curve is implemented by inverting steps 2 and 3 above, and recalculating τ by integrating over the arc length of the curve starting with the anchor point. Of course, this method is only feasible when the baseline time point is part of the time interval considered.

Forcing the curve to pass through the anchor at the start implies that we don't use the MS algorithm to select the starting point. For economic indices, in general, the anchor (base point) will be an adequate starting point as it will not reasonably be an outlier. This means that anchoring the LPC will still reserve the robustness regarding the choice of the starting point.

Now, we shall illustrate the functionality of the LPC algorithm as a 'feature extractor' for the summary index, in the subsequent section.

5.5.2 Analysis of CPI data

For the purpose of fitting a local principal curve as a summary price index, two sets of consumer price indices have been used, the first, as an introductory example, is a two dimensional set, and the second is a twelve dimensional set. All data are monthly UK data published through 'National Statistics Online' covering the period from January 1988 until December 2008. Both sets of indices are complemented subsets of the same total summary index, which is the total CPI for 'All Items'. The indices used for analysis are: (2005 is the base year for all indices , i.e. 2005=100)

D7BT: CPI INDEX 00 : ALL ITEMS

D7BU: CPI INDEX 01 : FOOD AND NON-ALCOHOLIC BEVERAGES

D7BV: CPI INDEX 02 : ALCOHOLIC BEVERAGES,TOBACCO & NARCOTICS

D7BW: CPI INDEX 03 : CLOTHING AND FOOTWEAR

D7BX: CPI INDEX 04 : HOUSING, WATER AND FUELS

D7BY: CPI INDEX 05 : FURN, HH EQUIP & ROUTINE REPAIR OF HOUSE

D7BZ: CPI INDEX 06 : HEALTH

D7C2: CPI INDEX 07 : TRANSPORT

D7C3: CPI INDEX 08 : COMMUNICATION

D7C4: CPI INDEX 09 : RECREATION & CULTURE

D7C5: CPI INDEX 10 : EDUCATION

D7C6: CPI INDEX 11 : HOTELS, CAFES AND RESTAURANTS

D7C7: CPI INDEX 12 : MISCELLANEOUS GOODS AND SERVICES

D7F4: CPI INDEX: ALL GOODS

D7F5: CPI INDEX: ALL SERVICES

Index construction from two sub-indices

The aim is to reconstruct the overall index (`CPI INDEX 00: ALL ITEMS`) using two sub-indices: the `CPI INDEX: ALL GOODS` and the `CPI INDEX: ALL SERVICES`. We use the modified LPC algorithm applying an anchor at $\boldsymbol{x}_{(0)} = 100 \times (w_1, w_2)^T$ and $\tau_{(o)} = 100$ corresponding to the reference point January 2005 as outlined earlier. For simplicity, a constant weight $w = 1/500 * (547, 453)^T$ for all years is used. The weighted version of any sub-index $X_{j(j=1,\cdots,p)}$ is given by

$$X_j^{(w)} = p \times \frac{w_j}{\sum_{i=1}^{p} w_i} \times X_j \tag{5.2}$$

Now, applying this adjusted LPC algorithm to fit a summary curve through the two weighted indices, one obtains the fit produced in Figure 5.20. It seems to give a reasonable summary for the two-dimensional data set in the form of a one-dimensional curve.

A first property of interest when using this statistical approach in CPI context could be; compared to the original overall index, how well is the resulting fit capturing the overall index behaviour? Figure 5.21 compares how the projection indices $\tau_{\boldsymbol{\nu}}(X)$ and the original `CPI INDEX 00` change over time. Figure 5.21 suggests that the statistically fitted overall index captures most movements in the true index, which is a desirable situation. Also, it can be seen that the fitted index looks smoother than the original index, due to the underlying smoothing properties implied by using the LPC algorithm.

The other useful informative tool accompanying the use of LPCs is related to the total variance explained by the curve and how each variable (sub-index) contributes to the fitted overall index. This is statistically measured through

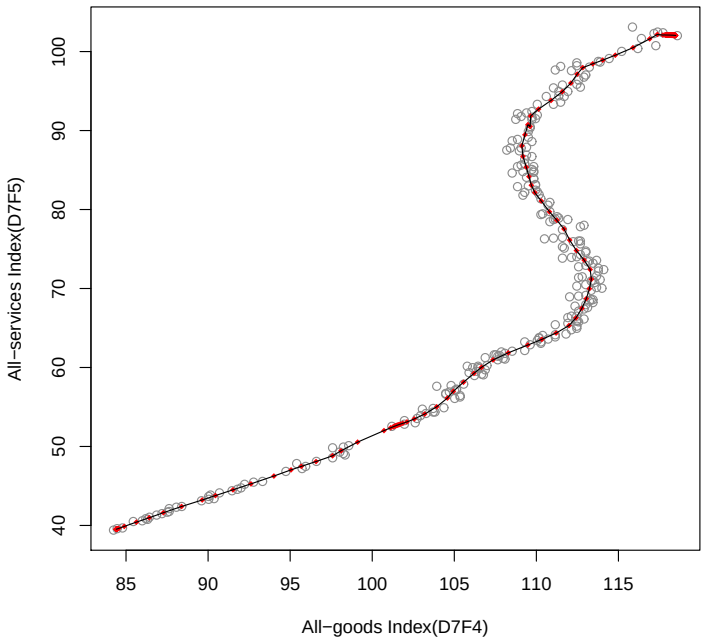

Figure 5.20: LPC fit for 2D CPI data.

'loadings', i.e. the entries of the (local) eigenvectors. At every point on the curve, the sum of squared loadings of the first eigenvector should be equal to one. This 'unity' property of eigenvectors provides a good tool to indicate how the sub-indices influence the fitted overall index at each point (time). Figure 5.22 shows the cumulative squared loadings of first eigenvectors for our example. Useful interpretations could be derived from such a figure, for instance, around the fitted curve's parameter values from 80 to 100, the second sub-index has a dominating effect on the fitted overall index.

As mentioned earlier in this chapter, one measure for the goodness of fit of the LPC in comparison with the first principal component line is the sum of squared Euclidean distances between data points and their projections onto the fitted LPC. For the LPC fitted to the two-dimensional CPI data, the latter sum of

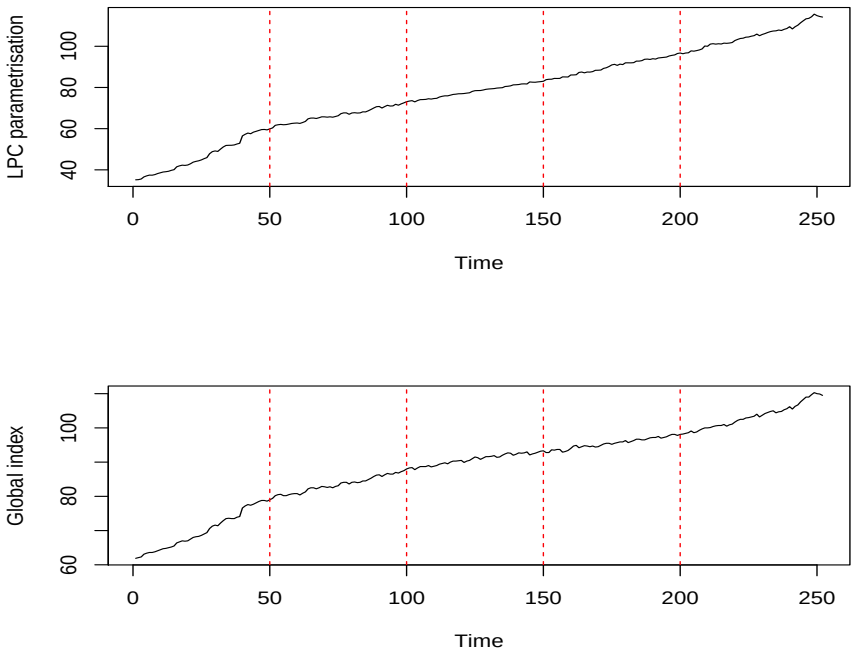

Figure 5.21: LPC-based (top) and average-based (bottom) CPI behaviour over time.

squares was 55.82 which was much better than the corresponding value for the first principal component line (156.54).

Index construction from twelve sub-indices

Adopting the same techniques used in the previous example, the LPC algorithm was applied to fit the overall consumer price index from the twelve sub-indices (`INDEX 01, INDEX 02, ..., INDEX 12`). Main indicators from the resulting fit are shown in Figure 5.23. Similar to the two-dimensional case, the procedure allows us to explore the index behaviour and the dominating underlying factors affecting it over time.

Looking at the bottom part of Figure 5.23, we can assess the contributions of the

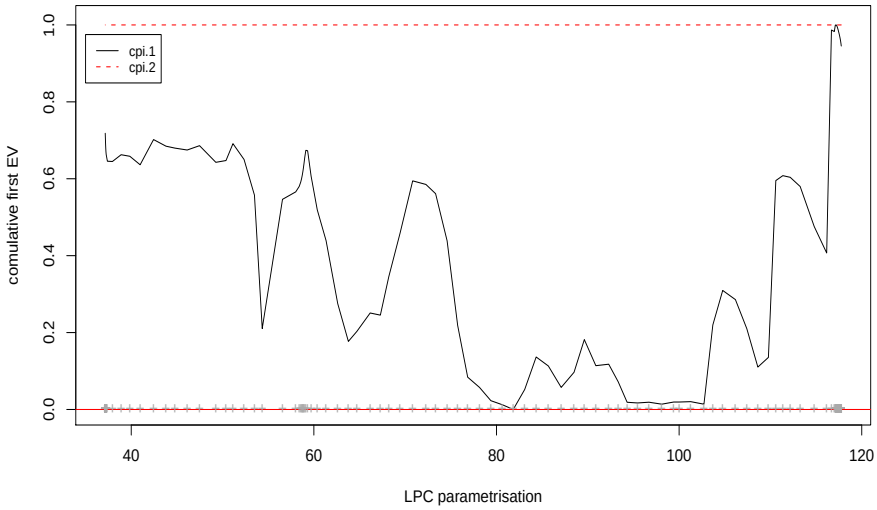

Figure 5.22: Cumulative squared loadings of first eigenvectors - 2D fit.

12 sub-indices over time. For example, it can be seen that the third index has the largest effect on the fitted overall index around the LPC parameter value 150 (which corresponds to some time point near 150), and the same can be said about index four around parameter values of 210 and 260 and that the first index alone approximately contributes 25% to the aggregate fitted index around parameter value of 260 (time near 250), and so on. All such interpretations can have useful meanings in the econometrics context.

The sum of squared Euclidean distances between data points and their projections onto the fitted LPC was 2383.79 compared to 12689.12 for the first principal component line. This confirms again that the non-linear summary CPI obtained through the LPC is outperforming linear summaries such as the first principal component line.

One remaining important feature of the proposed technique is the ability to predict missing data points at any given time (discrete or continuous) within the

data range. This is achieved, technically, through 'calibration' of time and the LPC parametrisation (by plotting them against each other and using a nonparametric smoother to find the functional relationship). Having done this, if we assume that we want to predict the data point that corresponds to, say, time = 220.5, we plug this value into the calibrated object which gives a parameter value of 215.34, then we get the corresponding estimated twelve-dimensional weighted point on the fitted curve, and applying a simple adverse-weight formula to each index, we get the real time estimated sub-indices' values (102.09, 102.59, 96.32, 107.92, 99.57, 102.48, 102.79, 99.82, 98.89, 105.93, 102.86, 103.27). This could be useful in handling missing data as well as predicting any assumed in-between data points. Recalling (5.2), the estimated non-weighted version of any sub-index X_j is given by

$$\hat{X}_j = \frac{\bar{w}}{w_j} \times \hat{X}_j^{(w)}$$

where $\bar{w} = (\sum_{i=1}^{p} w_i)/p$ is the average weight for all the j indices and $\hat{X}_j^{(w)}$ is the projected value of the weighted sub-index j onto the fitted LPC.

Finally, it is worth mentioning that the work presented in this part is merely a statistically-based approach to fit and analyse main economic indices. The computed index using the LPC algorithm has the ability to capture the basic trend of the original corresponding index over time. Being based upon principal component analysis, it allows to detect the influence of all variables (sub-indices) on the fitted index at all points (time), and would furthermore allow to assess the degree of 'local linearity' of the index, in terms of total local variance explained, at each point in time by looking at the localised first eigenvalues. The main novel feature of the proposed technique is that it is nonlinear and even nonparametric, while the traditional PCA-based methods are linear, which may be of limited

accuracy in particular if the time range considered is quite large.

It should be noted that the proposed technique, just as the PCA itself and the modified version by Theil (1960), is an 'ex-post' algorithm, i.e. one needs to have the full data available in order to reconstruct the indices retrospectively. However, unlike other principal curve algorithms, the LPC methodology would in principle allow for an updating algorithm, which would enable to extend the previously fitted curve and the associated statistics once new data have come in. This is a matter of future research.

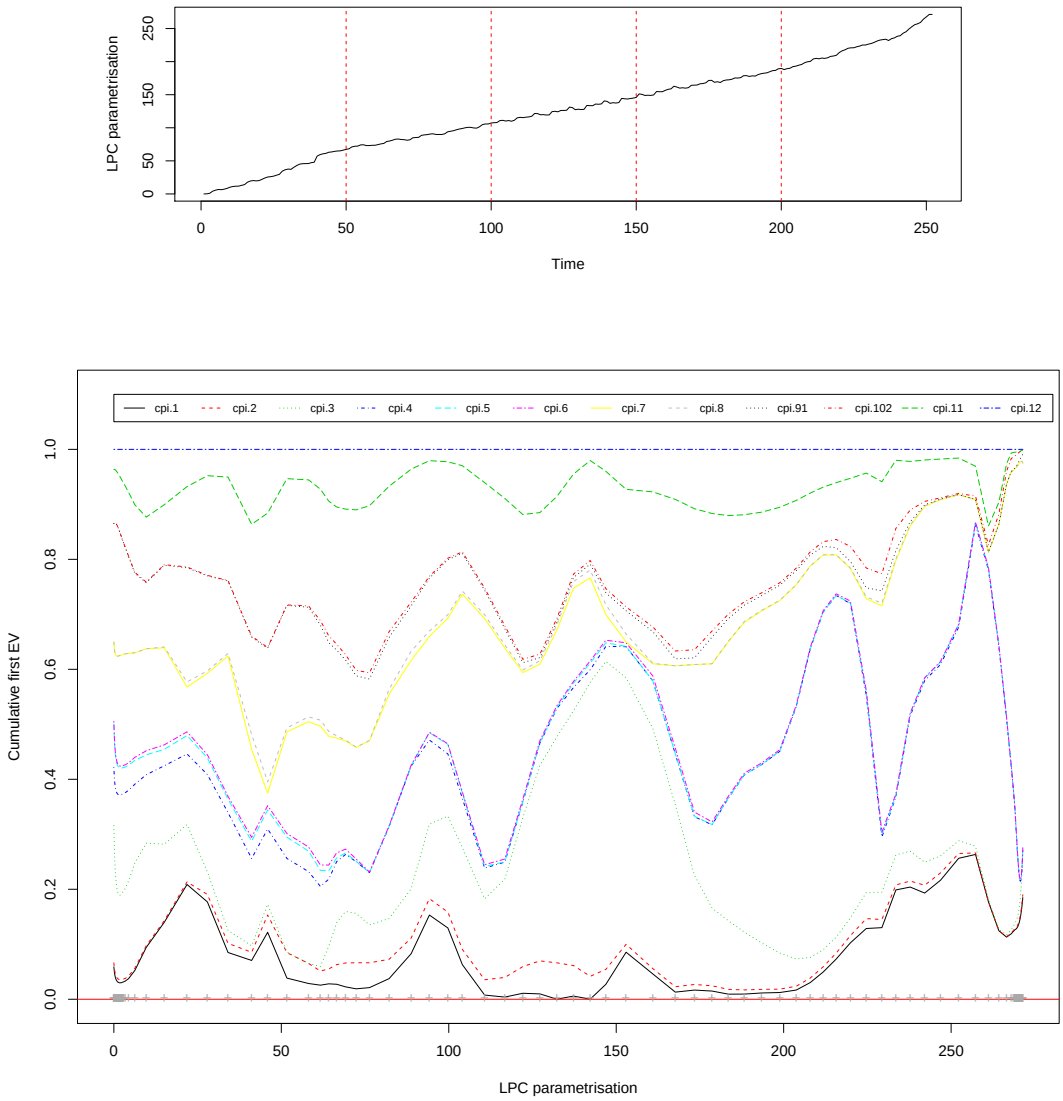

Figure 5.23: A 12-D example. Top: reconstructed summary index (LPC parametrisation over time); bottom: cumulative squared loadings (first eigenvector) over time.

Chapter 6

Conclusions

6.1 Summary Conclusions

In this work we have introduced some asymptotics for localised principal components using multivariate kernels as weights. It was shown that for small neighbourhoods and large sample sizes, at any data point $\boldsymbol{x}$, the largest eigenvector of the local covariance matrix $\boldsymbol{\Sigma}^{\boldsymbol{x}}$ can be approximated in terms of the density function and the bandwidth matrix $\boldsymbol{H}$.

For local principal curves (LPCs), this result implied that the LPC always follows the gradient of the density function, which means in practice that it will closely follow the density ridge. The previous approximation was extended to explore the behaviour of the local principal curve in terms of the difference between neighbouring local centres of mass which compose the fitted curve. Using the first order approximation of the latter, the stopping criteria for the LPC were further investigated. It was concluded that the position at which the curve stops only depends on the topology of data in the neighbourhood in terms of the density function and its derivative, the multivariate kernel function used and the bandwidth matrix. When a Gaussian kernel is applied, where the curve gets stuck

only depends on the bandwidth and the underlying density of the data. This was verified experimentally using a random sample from a bivariate standard normal distribution and simulating 20 local principal curves for this data. It was observed that the smaller the bandwidth, the larger the area of the data covered (visited) by the curves. The previous result was used to enhance the fitted LPC so that it reaches more data points at boundaries. This was accomplished through reducing the bandwidth adaptively relative to the step size of the local PCA as soon as the curve tends to converge. The boundary-extended LPC version was again tested using a time series data example, in which the curve was clearly going further at boundaries compared to the non-extended version of the LPC algorithm.

In addition to the LPC boundary extension based upon the asymptotics of localised principal components, another useful enhancement for the LPC algorithm was introduced. The mean shift algorithm as a mode seeking tool was employed to improve the choice of the starting point of the LPC algorithm. Instead of selecting this point at random, which may result in choosing an outlier or a mislocated data point which may affect the goodness of fit of the curve, the mean shift algorithm was integrated into that of the LPC so that the starting point is selected from the set of the local modes (located in high density areas) within the data cloud. Using this, the number of possible LPC fits for a given data is delimited by the number of its local modes.

In terms of application, local principal curves provide a useful and flexible tool to represent multidimensional and complex structures. It is important to apply the algorithm with the optimal options for the specific data set of interest. It is expected that LPCs and other PCA-based methods can be reliable for modelling econometric data in general, and specifically data of time series character.

Local principal curves, through the curve parametrisation, can well serve as a

prediction tool, especially when we can create a link between the parametrisation of the curve and another external underlying variable that is expected to be related to our data set. For econometric and financial data, time can play this role efficiently. This type of link can be represented through a 'calibration' curve, and a good link should give a precise and unique calibration as possible. In this context, we should keep in mind the smoothness trade-off between the LPC and the calibration curve.

When the prediction works fine with regard to predicting existing data points, it is then possible to think about testing further prediction abilities, such as:

- Given a certain data point, which may or may not be a part of the original data cloud, and after projecting this point onto the fitted curve, we could try to reconstruct the time at which this observation could have occurred.
- Estimating future (or past) time observations, at least in the short run. (i.e. extrapolating)

Both principal components and curves can produce a relatively meaningful analysis for multidimensional data. Local principal curves can be looked at as an efficient dimensionality reduction as well as feature and summary extraction tool. The fitted LPC provides a good graphical one-dimensional representation for complex multidimensional data. A well fitted curve, through the curve parametrisation, can also extract the main features of the multidimensional data providing a useful interpretable summary of the data.

The previous role of local principal curves was explored in the context of two econometric applications, insurance market key performance indicators and construction of aggregate consumer price indices. In such cases, the LPC capture the overall behaviour (trend) of the data. Also, using the standardised 'loadings'

obtained via localised principal components, the fitted LPC can be quite informative for analysing the total variance explained by the fitted curve and how each variable (dimension) contributes to the overall summary obtained through the parametrisation along the curve.

A flexibility issue related to the LPC algorithm as a summary is its ability to accommodate different weights for each dimension.

6.2 Suggestions for Future Research

One natural idea is to continue exploring the applicability of the LPC algorithm in other applications, which may suggest accommodating new features into the LPC algorithm, in particular, for high-dimensional actuarial and econometric data.

Besides this, there are several issues related to the LPC algorithm and its applications that may bring to mind some potential topics to be addressed. Among those are the following:

(i) The role of parametrisation in local principal curve modelling, and its possible relation to important latent variables, not necessarily time, in some applications.

(ii) Exploring statistical optimality for the choice of local principal curves parameters.

(iii) Using parametrised LPCs in prediction (forecasting), specially for time series data.

(iv) Measuring the correlation between principal curves and the possibility of

predicting a principal curve using another.

(v) Exploring the role that principal curves can play in missing value analysis and estimation.

(vi) Integrating local principal curves with other statistical methods providing new semi-parametric or non-parametric statistical modelling tools. Among those may be Bayesian methods and other statistical inference approaches.

(vii) The idea of 'functional principal curves'. Analogue to functional data analysis, there may arise situations where we fit more than one principal curve, each for a distinct group (cluster) of data. For example, in multidimensional time series data, a separate principal curve may be fit to one 'case' (individual, country, ... etc.) along time.

Appendix

(I) Abbreviations and Symbols Used

HS: The Hastie and Stuetzle version of principal curves.

LPC: Local Principal Curve.

MS algorithm: Mean shift algorithm.

PC: Principal Curve.

PCA: Principal Component Analysis.

$\boldsymbol{X}$: A d-variate random vector with density function $f(.)$, mean $\boldsymbol{\mu} \in \mathbb{R}^d$ and covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$.

X : $X = (X_1, X_2, \cdots, X_n)^T$, $X_i \in \mathbb{R}^d$ is a random sample from $\boldsymbol{X}$.

$\boldsymbol{x}$: A data point.

$\boldsymbol{H}$: A positive definite bandwidth matrix with diagonal elements: $h_1^2, \cdots, h_d^2$, $\boldsymbol{H} \in \mathbb{R}^{d \times d}$.

$K(\,)$: A multidimensional symmetric kernel function.

$\boldsymbol{\mu}^{\boldsymbol{x}}$: The local mean at $\boldsymbol{x}$.

$\boldsymbol{\gamma}^{\boldsymbol{x}}$: The first local eigenvector at $\boldsymbol{x}$.

$\boldsymbol{m}_{h,K}(\boldsymbol{x})$: The mean shift vector, with a bandwidth h and a kernel K.

$\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$: The sequence of successive locations of the mean shift procedure.

$\boldsymbol{\nu}(\tau)$: A curve $\boldsymbol{\nu}$ with parametrisation τ.

$d(\boldsymbol{x}, \boldsymbol{\nu})$: The Euclidean distance from a point $\boldsymbol{x}$ to its projection on the curve.

(II) Math

Math for Section 2.1

The matrix $\boldsymbol{A}_{\boldsymbol{\gamma}} = (\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T)$

$$\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \cdots, \gamma_q)^T$$

$$\boldsymbol{\gamma}\boldsymbol{\gamma}^T = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ . \\ . \\ . \\ \gamma_q \end{pmatrix} \begin{pmatrix} \gamma_1, & \gamma_2, & \cdots, & \gamma_q \end{pmatrix} = \begin{pmatrix} \gamma_1\gamma_1 & \cdots & \gamma_1\gamma_q \\ \cdots & \cdots & \cdots \\ \gamma_q\gamma_1 & \cdots & \gamma_q\gamma_q \end{pmatrix}$$

$$\boldsymbol{\gamma}\boldsymbol{\gamma}^T = \begin{pmatrix} \gamma_1^2 & \gamma_1\gamma_2 & \cdots & \gamma_1\gamma_q \\ \gamma_2\gamma_1 & \gamma_2^2 & \cdots & \gamma_2\gamma_q \\ \cdots & \cdots & \cdots & \cdots \\ \gamma_q\gamma_1 & \gamma_q\gamma_2 & \cdots & \gamma_q^2 \end{pmatrix}$$

It is evident that $\boldsymbol{\gamma}\boldsymbol{\gamma}^T$ is symmetric.

Now, examining the matrix $\boldsymbol{A}_{\boldsymbol{\gamma}} = [\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T]$, we get the following

$$(\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T) = \begin{pmatrix} 1 - \gamma_1^2 & \gamma_1\gamma_2 & \cdots & \gamma_1\gamma_q \\ \gamma_2\gamma_1 & 1 - \gamma_2^2 & \cdots & \gamma_2\gamma_q \\ \cdots & \cdots & \cdots & \cdots \\ \gamma_q\gamma_1 & \gamma_q\gamma_2 & \cdots & 1 - \gamma_q^2 \end{pmatrix}$$

And

$$\begin{aligned} \boldsymbol{A}_{\boldsymbol{\gamma}}^T\boldsymbol{A}_{\boldsymbol{\gamma}} &= [\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T][\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T] \\ &= \boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T - \boldsymbol{\gamma}\boldsymbol{\gamma}^T + \boldsymbol{\gamma}\boldsymbol{\gamma}^T\boldsymbol{\gamma}\boldsymbol{\gamma}^T \\ &= \boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T - \boldsymbol{\gamma}\boldsymbol{\gamma}^T + \boldsymbol{\gamma}\boldsymbol{\gamma}^T \end{aligned}$$

$$= \boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T$$
$$= \boldsymbol{A}_{\boldsymbol{\gamma}}$$

$\Rightarrow \boldsymbol{A}_{\boldsymbol{\gamma}}$ is idempotent[36].

For any matrix of type $A = c\boldsymbol{\psi}\boldsymbol{\psi}^T$, with $c \in \mathbb{R}, \boldsymbol{\psi} \in \mathbb{R}^d$, the only eigenvector of the matrix is (in standardised form) $\boldsymbol{\gamma} = \boldsymbol{\psi}/||\boldsymbol{\psi}||$, with eigenvalue $\lambda = c||\boldsymbol{\psi}||^2 = \text{Tr}(A)$. To verify this, multiplying A by ψ, we have

$$\begin{aligned} A\psi &= c.\psi(\psi^T\psi) \\ &= c.\psi\|\psi\|^2 \\ A\psi &= c.\|\psi\|^2\psi \end{aligned}$$

Then, ψ is an eigenvector of A with eigenvalue $c.\|\psi\|^2$. Also, if we consider

$$A\frac{\psi}{\|\psi\|} = c.\|\psi\|^2\frac{\psi}{\|\psi\|}$$

then $\boldsymbol{\gamma} = \frac{\psi}{\|\psi\|}$ is a standardised eigenvector of A with eigenvalue $c.\|\psi\|^2$.

The sum of the eigenvalues of a matrix is its trace, then

$$\begin{aligned} \lambda_1 + \lambda_2 + \cdots + \lambda_q &= tr(A) \\ &= c.\sum_{i=1}^{q}\psi_i^2 \\ &= c\|\psi\|^2 \end{aligned}$$

This leads to: $\lambda_2 = \cdots = \lambda_q = 0$, and $\boldsymbol{\gamma} = \frac{\psi}{\|\psi\|}$ is the only non-zero eigenvector of $A = c\psi\psi^T$.

Then, in the special case where $c = 1$ and $\psi = \boldsymbol{\gamma}$, the matrix $\boldsymbol{\gamma}\boldsymbol{\gamma}^T$ has only one non-zero eigenvector with an eigenvalue equals $\text{tr}(\boldsymbol{\gamma}\boldsymbol{\gamma}^T) = \|\boldsymbol{\gamma}\|^2 = 1$.

Also, let φ be an eigenvalue of an idempotent matrix A. This means that for some vector ν, $A\nu = \varphi\nu$:

$$\varphi\nu = A\nu = A^2\nu = A(A\nu) = A(\varphi\nu) = \varphi(A\nu) = \varphi^2 v$$

which implies that $\varphi^2 = \varphi$ so $\varphi(\varphi - 1) = 0$ which implies that the eigenvalues of A are either 1 or 0 and that the trace of A is the number of its non-zero eigenvalues.

We conclude that the matrix $\boldsymbol{\gamma}\boldsymbol{\gamma}^T$ has only one non-zero eigenvector with an eigenvalue equals one. Using the property of the eigenvalues that; if A is a symmetric matrix, then[61](p.30)

$$eig(I + cA) = 1 + c\lambda_i,$$

which implies that

$$eig(I - \boldsymbol{\gamma}\boldsymbol{\gamma}^T) = 1 - (1)(1, 0, \cdots, 0) = (0, 1, \cdots, 1)$$

and since all eigenvalues of $\boldsymbol{A}_{\boldsymbol{\gamma}}$ are greater than or equal to zero, then, $\boldsymbol{A}_{\boldsymbol{\gamma}}$ is positive semi-definite[61](p.50). Also, the trace of $\boldsymbol{A}_{\boldsymbol{\gamma}}$ is given by

$$\begin{aligned} tr(\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T) &= \sum_{i=1}^{n}(1 - \boldsymbol{\gamma}_i^2) \\ &= n - \sum_{i=1}^{n}\boldsymbol{\gamma}_i^2 \\ &= n - 1 \end{aligned}$$

which can be interpreted that the matrix $\boldsymbol{A}_{\boldsymbol{\gamma}}$ has n eigenvalues, all of them are ones except the first, and that the sum of those eigenvalues is $n - 1$.

$$\begin{aligned} \|\boldsymbol{A}_{\boldsymbol{\gamma}}u\|^2 &= (\boldsymbol{A}_{\boldsymbol{\gamma}}u)^T(\boldsymbol{A}_{\boldsymbol{\gamma}}u) \\ &= u^T\boldsymbol{A}_{\boldsymbol{\gamma}}^T\boldsymbol{A}_{\boldsymbol{\gamma}}u = u^T\boldsymbol{A}_{\boldsymbol{\gamma}}u \end{aligned}$$

$$\frac{\partial}{\partial\boldsymbol{\gamma}}\boldsymbol{A}_{\boldsymbol{\gamma}} = \frac{\partial}{\partial\boldsymbol{\gamma}}(\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T) = -2\boldsymbol{\gamma}$$

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{\gamma}} u^T \boldsymbol{A}_\gamma u &= \frac{\partial}{\partial \boldsymbol{\gamma}} u^T (\boldsymbol{I} - \boldsymbol{\gamma}\boldsymbol{\gamma}^T) u \\ &= \frac{\partial}{\partial \boldsymbol{\gamma}} (u^T u - u^T \boldsymbol{\gamma}\boldsymbol{\gamma}^T u) \\ &= 0 - \frac{\partial}{\partial \boldsymbol{\gamma}} u^T \boldsymbol{\gamma}\boldsymbol{\gamma}^T u \\ &= -2(uu^T)\boldsymbol{\gamma}\end{aligned}$$

Math for Section 3.1.2

Details for (3.11):

$\boldsymbol{x}_i \to s, \quad f(s) = \frac{1}{n} \; \forall \; s, \;\; u = \boldsymbol{H}^{-1/2}(s - \boldsymbol{x}),$
$(s - m) = (s - \boldsymbol{x} + \boldsymbol{x} - m) = \boldsymbol{H}^{1/2} u + \boldsymbol{x} - m$

Applying Taylor expansion as in [79](p.94),

$$\begin{aligned} f(\boldsymbol{x} + \boldsymbol{H}^{1/2}\boldsymbol{u}) = f(\boldsymbol{x}) + (\boldsymbol{H}^{1/2}\boldsymbol{u})^T f'(\boldsymbol{x}) + \frac{1}{2}(\boldsymbol{H}^{1/2}\boldsymbol{u})^T \mathcal{H}(\boldsymbol{x}) \boldsymbol{H}^{1/2}\boldsymbol{u} \\ + o((\boldsymbol{H}^{1/2}\boldsymbol{u})^T \boldsymbol{H}^{1/2}\boldsymbol{u}) \end{aligned} \tag{1}$$

where: $f'(\boldsymbol{x})$ is the first derivative of $f(\boldsymbol{x})$, $\mathcal{H}(\boldsymbol{x})$ is the Hessian matrix of of $f(\boldsymbol{x})$.

Considering the last term in (1), $o((\boldsymbol{H}^{1/2}\boldsymbol{u})^T \boldsymbol{H}^{1/2}\boldsymbol{u})$

$$\begin{aligned} o((\boldsymbol{H}^{1/2}\boldsymbol{u})^T \boldsymbol{H}^{1/2}\boldsymbol{u}) &= o(\mathbf{1}^T \boldsymbol{H}\, \mathbf{1}) \\ &= o(tr(\mathbf{1}^T \boldsymbol{H}\, \mathbf{1})) = o(tr(\boldsymbol{H}\, \mathbf{1}\, \mathbf{1}^T)) = o(tr(\boldsymbol{H})) \end{aligned}$$

The third term in (1), $\frac{1}{2}(\boldsymbol{H}^{1/2}\boldsymbol{u})^T \mathcal{H}(\boldsymbol{x}) \boldsymbol{H}^{1/2}\boldsymbol{u}$ gives

$$\begin{aligned} \frac{1}{2}(\boldsymbol{H}^{1/2}\boldsymbol{u})^T \mathcal{H}(\boldsymbol{x}) \boldsymbol{H}^{1/2}\boldsymbol{u} &= O(\mathbf{1}^T \boldsymbol{H}^{1/2}\, \mathbf{1}\, H^{1/2}\, \mathbf{1}) \\ &= O(tr(\mathbf{1}^T\, \mathbf{1}\, \boldsymbol{H}\, \mathbf{1})) = O(tr(\boldsymbol{H})) \end{aligned}$$

And the second term in (1), $(\boldsymbol{H}^{1/2}\boldsymbol{u})^T f'(\boldsymbol{x})$ yields

$$\begin{aligned}(\boldsymbol{H}^{1/2}\boldsymbol{u})^T f'(\boldsymbol{x}) &= O(\mathbf{1}^T \boldsymbol{H}^{1/2}\,\mathbf{1}) \\ &= O(tr(\mathbf{1}^T\,\boldsymbol{H}^{1/2}\,\mathbf{1})) = O(tr(\mathbf{1}^T\,\mathbf{1}\,\boldsymbol{H}^{1/2}) = O(tr(\boldsymbol{H}^{1/2}))\end{aligned}$$

Then (1) can be re-written as

$$\begin{aligned}f(\boldsymbol{x}+\boldsymbol{H}^{1/2}\boldsymbol{u}) &= f(\boldsymbol{x}) + O(tr(\boldsymbol{H}^{1/2})) + O(tr(\boldsymbol{H})) + o(tr(\boldsymbol{H})) \\ &= f(\boldsymbol{x}) + O(tr(\boldsymbol{H}^{1/2})) = f(\boldsymbol{x}) + o(\mathbf{1})\end{aligned}$$

$$E\left(\sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right)$$

Applying (3.3), the previous expression can be re-written as:

$$E\left(|\boldsymbol{H}|^{-1/2}\sum_{i=1}^{n} K(\boldsymbol{H}^{-1/2}(\boldsymbol{x}_i-\boldsymbol{x}))(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right)$$

and for large values of n and small values of $\boldsymbol{H}$

$$\begin{aligned}&E\left(|\boldsymbol{H}|^{-1/2}\sum_{i=1}^{n} K(\boldsymbol{H}^{-1/2}(\boldsymbol{x}_i-\boldsymbol{x}))(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right) \\ &= n|\boldsymbol{H}|^{-1/2}\int K\left(\boldsymbol{H}^{-1/2}(\boldsymbol{s}-\boldsymbol{x})\right)(\boldsymbol{s}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{s}-\boldsymbol{m})f(\boldsymbol{s})\,d\boldsymbol{s} \\ &= n\int K(\boldsymbol{u})(\boldsymbol{H}^{1/2}\boldsymbol{u}+\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{H}^{1/2}\boldsymbol{u}+\boldsymbol{x}-\boldsymbol{m})f(\boldsymbol{x}+\boldsymbol{H}^{1/2}\boldsymbol{u})\,d\boldsymbol{u} \\ &= n\int K(\boldsymbol{u})(\boldsymbol{H}^{1/2}\boldsymbol{u}+\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{H}^{1/2}\boldsymbol{u}+\boldsymbol{x}-\boldsymbol{m})\left\{f(\boldsymbol{x})+O(tr(\boldsymbol{H}^{1/2}))\right\} \\ &= n\int K(\boldsymbol{u})\left\{\boldsymbol{u}^T\boldsymbol{H}^{1/2}\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{H}^{1/2}\boldsymbol{u}+\boldsymbol{u}^T\boldsymbol{H}^{1/2}\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})+(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{H}^{1/2}\boldsymbol{u}\right. \\ &\quad \left.+(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})\right\}\left\{f(\boldsymbol{x})+O(tr(\boldsymbol{H}^{1/2}))\right\} \\ &= n\int K(\boldsymbol{u})\left\{\boldsymbol{u}^T\boldsymbol{H}^{1/2}\boldsymbol{A}_{\boldsymbol{\gamma}}\boldsymbol{H}^{1/2}\boldsymbol{u}+2\boldsymbol{u}^T\boldsymbol{H}^{1/2}\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})\right. \\ &\quad \left.+(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})\right\}\left\{f(\boldsymbol{x})+O(tr(\boldsymbol{H}^{1/2}))\right\} \\ &= n\int K(\boldsymbol{u})\left\{(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})+O(\mathbf{1}^T\boldsymbol{H}^{1/2}\boldsymbol{u})\right\}\left\{f(\boldsymbol{x})+O(\mathbf{1}^T\boldsymbol{H}^{1/2}\boldsymbol{u})\right\}\,d\boldsymbol{u}\end{aligned}$$

$$
\begin{aligned}
&= [nf(\boldsymbol{x})(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})] \\
&\quad \int K(\boldsymbol{u})\left\{(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m}) + O(\mathbf{1}^T\boldsymbol{H}^{1/2}\boldsymbol{u})\right\}\left\{+O(\mathbf{1}^T\boldsymbol{H}^{1/2}\boldsymbol{u})\right\}\, d\boldsymbol{u} \\
&= nf(\boldsymbol{x}) + o(n),
\end{aligned}
$$

$$
\begin{aligned}
&\mathrm{Var}\left(\sum_{i=1}^n K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})(\boldsymbol{x}_i-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\right) \\
&= \sum_{i=1}^n \mathrm{Var}\left(K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})\|\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\|^2\right) \\
&= \sum_{i=1}^n \left\{E\left(K_{\boldsymbol{H}}^2(\boldsymbol{x}_i-\boldsymbol{x})\|\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\|^4\right) - E^2\left(K_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})\|\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\|^2\right)\right\} \\
&= \sum_{i=1}^n \left\{E\left(K_{\boldsymbol{H}}^2(\boldsymbol{x}_i-\boldsymbol{x})\|\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}_i-\boldsymbol{m})\|^4\right)[f(\boldsymbol{x})(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m}) + O(\mathbf{1}^T\boldsymbol{H}^{1/2}\mathbf{1})]^2\right\} \\
&= n\Bigg[\int \frac{1}{|\boldsymbol{H}|}K^2(\boldsymbol{H}^{-1/2}(s-\boldsymbol{x}))\left[(s-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(s-\boldsymbol{m})\right]^2 f(s)ds \\
&\quad - f^2(\boldsymbol{x})[(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})]^2 - O(\mathbf{1}^T\boldsymbol{H}^{1/2}\mathbf{1}) - O(\mathbf{1}^T\boldsymbol{H}\mathbf{1})\Bigg] \\
&= \frac{n}{|\boldsymbol{H}|}\int K^2(u)[(\boldsymbol{H}^{1/2}u+\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{H}^{1/2}u+\boldsymbol{x}-\boldsymbol{m})]^2 f(\boldsymbol{x}+\boldsymbol{H}^{1/2}u)|\boldsymbol{H}|^{1/2}du \\
&\quad - nf^2(\boldsymbol{x})[(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})]^2 - nO(\mathbf{1}^T\boldsymbol{H}^{1/2}\mathbf{1}) \\
&= \frac{n}{|\boldsymbol{H}|^{1/2}}\int K^2(u)[(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m}) + O(\mathbf{1}^T\boldsymbol{H}^{1/2}u)]^2[f(\boldsymbol{x}) + O(\mathbf{1}^T\boldsymbol{H}^{1/2}u)]du \\
&\quad - nf^2(\boldsymbol{x})[(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})]^2 - O(n\mathbf{1}^T\boldsymbol{H}^{1/2}\mathbf{1}) \\
&= \frac{nf(\boldsymbol{x})}{|\boldsymbol{H}|^{1/2}}\cdot 1\cdot\left([(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})]^2 + O(\mathbf{1}^T\boldsymbol{H}^{1/2}\mathbf{1})\right) \\
&\quad - nf^2(\boldsymbol{x})[(\boldsymbol{x}-\boldsymbol{m})^T\boldsymbol{A}_{\boldsymbol{\gamma}}(\boldsymbol{x}-\boldsymbol{m})]^2 - O(n\mathbf{1}^T\boldsymbol{H}^{1/2}\mathbf{1}) \\
&= nf^2(\boldsymbol{x})\frac{O(\mathbf{1})}{n|\boldsymbol{H}|^{1/2}} = O(n^2)
\end{aligned}
$$

Math for Section 4.2

$$
\int \boldsymbol{u}\boldsymbol{u}^T K(\boldsymbol{u})\, d\boldsymbol{u} = \mu_2(K)\boldsymbol{I}
$$

$$
\begin{aligned}
f(\boldsymbol{x}) &= \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}\boldsymbol{x}^T\Sigma^{-1}\boldsymbol{x}\right\} \\
&= \frac{1}{(2\pi)^{d/2}\sigma^d} \exp\left\{-\frac{1}{2\sigma^2}\boldsymbol{x}^T\boldsymbol{x}\right\}
\end{aligned}
$$

$$
\begin{aligned}
\nabla f(\boldsymbol{x}) &= \frac{1}{(2\pi)^{d/2}\sigma^d} \exp\left\{-\frac{1}{2\sigma^2}\boldsymbol{x}^T\boldsymbol{x}\right\}\left\{-\frac{1}{2\sigma^2}\nabla(\boldsymbol{x}^T\boldsymbol{x})\right\} \\
&= \frac{1}{(2\pi)^{d/2}\sigma^d} \exp\left\{-\frac{1}{2\sigma^2}\boldsymbol{x}^T\boldsymbol{x}\right\}\left\{\frac{\boldsymbol{x}}{\sigma^2}\right\} \\
\|\nabla f(\boldsymbol{x})\| &= \frac{1}{(2\pi)^{d/2}\sigma^{d+2}} \exp\left\{-\frac{1}{2\sigma^2}\boldsymbol{x}^T\boldsymbol{x}\right\}\|\boldsymbol{x}\|
\end{aligned}
$$

Math for LPC boundary extension

$$
\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)} = \left[\frac{\mu_2(K)}{f(\boldsymbol{x}_{(j)})} \pm \frac{t}{||\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}\right]\boldsymbol{H}\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})
$$

when $\boldsymbol{H} = h^2\boldsymbol{I}$

$$
\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)} \overset{a}{=} h\left[\frac{h\,\mu_2(K)}{f(\boldsymbol{x}_{(j)})} \pm \frac{t}{h}\frac{1}{||\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}\right]\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})
$$

when $t = h$

$$
\boldsymbol{\mu}_{(j+1)} - \boldsymbol{\mu}_{(j)} \overset{a}{=} \left[\frac{\mu_2(K)}{f(\boldsymbol{x}_{(j)})} \pm \frac{1}{h\,||\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})||}\right]h^2\boldsymbol{\nabla} f(\boldsymbol{x}_{(j)})
$$

(III) Sample R Code

a. Sample R code for principal components representation

```
ddd<-read.table("data19", header=TRUE)
data<-ddd[, c(3,4)]
e<-eigen(cov(data))
e1 <- e$ve[,1]
e2 <- e$ve[,2]
m <- mean(data)
p1 <- m + 75*e1
p2 <- m - 75*e1
p3 <- m + 75*e2
p4 <- m - 75*e2

plot(data, pch=20)
segments(p2[1],p2[2], p1[1], p1[2], col=2, lty=1)
segments(p3[1],p3[2], p4[1], p4[2], col=3, lty=2)
l<-lm(data[,2]~data[,1])
abline(63.484, -0.201, col=4, lty=3)
legend(0,25, c("1st PC", "2nd PC", "LM"), lty=c(1,2,3), col=c(2,3,4))

----------------------------------------
# % of variance captured by each PC:

traffic.pc <- prcomp(data) # OR traffic.pc <- princomp(data)
traffic.pc$sdev^2/sum(traffic.pc$sdev^2)
----------------------------------------
```

b. Sample R code for methods to fit traffic data

```
plot(data, pch=20)
# Linear fit:
fit1 <- lm(data[,2]~ data[,1])
abline(lm(data[,2]~ data[,1]), col=2)
# Quadratic fit:
fit2 <- lm(data[,2]~ data[,1] + I(data[,1]^2))
```

```
lines(data[,1], fit2$fitted, col=3)
# Interpolation
fit3 <- approx(data, n=444)
lines(fit3$x, fit3$y, col=4)
legend(5, 27, c("Linear fit", "Quadratic fit", "Interpolation")
             , fill=c(2,3,4))
```

c. Sample R code for mean-shift function

```
g1 <- function(xi, x, h){
   1/2*exp(-1/2*((x-xi)/h)^2)
   }
gd <- function(Xi,x,h){
   d<-length(x)
   k<-1
   for (j in 1:d){k<- k* g1(Xi[,j],x[j],h[j])}
   k
   }

mean.shift<-function(data, x, h){
   gd <- gd(data, x, h)
   d <- dim(data)[2]
   ms <- NULL
   for (j in 1:d){
                ms[j]<-sum(data[,j]*gd)/sum(gd)
                }
   ms
   }

h<-c(20,5)
x<-c(50, 20)
x1<-mean.shift(data, x, h)

-----------------------------------------
v3<-runif(444, 0,1)
```

```
x.2 <- c(50, 20, 0.5)
h.2 <- c(10, 5, 0.05)

m1<-mean.shift(data2, x.2, h.2)

-----------------------------------------
norm <- function (x){
        d<-length(x)
        v<-0
        for(i in 1:d){v<-v+(x[i])^2}
        v
     }

norm(x.2)

-----------------------------------------

ms.rep <- function (data, x, h) {
          iter <-10
          th <- rep(0,iter)
          for (j in 1: 10){
            m     <- mean.shift(data, x, h)
            th[j] <- norm(m-x)/norm(x)
            x     <- m
          }
       #m
       #th
       return(list(m,th))
       }

ms.rep(data2, x.2, h.2)

======================

# FINAL MEAN-SHIFT CODE:

ms.rep <- function (data, x, h) {
```

```
        iter <-100
        s    <- 0
        th <- rep(0,iter)
        M  <-matrix(0, iter, length(x))
        for (j in 1: iter){
        m     <- mean.shift(data, x, h)
        M[j,] <- m
        th[j] <- norm(m-x)/norm(x)
        if (th[j]<0.000001){s<-j;
           print("required threshold reached"); break}
        x     <- m
        }
   return(list("Mean shift points"=M[1:s,],
               "Threshold values"=th[1:s], "iterations"=s))
   }

ms.rep(data2, x.2, h.2)
ms.rep(data2, c(50,20,1), h.2)
```

d. Sample R code for computing the squared distances (projections of data points) (2.1)

```
pc7 <- prcomp(d) # principal components
summary(pc7); names(pc7)
pc7$rot

library(LPCM)  # LPC
lpc77 = lpc(d)
lpc.proj = lpc.spline(lpc77, project=TRUE)
names(lpc.proj)

# ---------------------------------------
# Function for squared distances between data and their projections

dist.xg = function(data, gamma){
 n = dim(data)[1]
```

```
 m = mean(data)
 l = length(gamma)
 I = diag(l)
 g2 =  gamma %*% t(gamma)
 #print(g2)
 a = (I - g2) %*% m
 #print(a)
 xg = data-data
   for(i in 1:n){
   xg[i,] = t( a + g2 * data[i,] )
   }
 xd = data - xg
 sum(xd^2)
 }
# ---------------------------------------

dist.pc1 = dist.xg(d, pc7$rot[,1])
dist.lpc = sum((d - lpc.proj$closest.coords)^2)
```

(IV) Analysis of insurance data - country codes

Code	Country	Code	Country
AT	Austria	BG	Bulgaria
CY	Cyprus	CZ	Czech Republic
DE	Germany	DK	Denmark
EE	Estonia	EL	Greece (Hellenic Republic)
ES	Spain	FI	Finland
FR	France	HU	Hungary
IS	Iceland	IT	Italy
LT	Lithuania	LU	Luxemburg
LV	Latvia	NL	The Netherlands
NO	Norway	PL	Poland
PT	Portugal	RO	Romania
SE	Sweden	SK	Slovakia
UK	United Kingdom		

Table 1: Country codes for the insurance data application

References

[1] European insurance in figures: Data 1999-2008. Technical report, CEA Insurers of Europe, July 2010. CEA Statistics N° 40.

[2] Anderson, T. W. Asymptotic theory for principal component analysis. *The Annals of Mathematical Statistics*, 34(1):pp. 122–148, March 1963.

[3] Armeanu, Dan and Lache, Leonard. Application of the model of principal components analysis on romanian insurance market. *Theoretical and Applied Economics*, 6(523):11–20, 2008.

[4] Banfield, Jeffrey D. and Raftery, Adrian E. Ice floe identification in satellite images using mathematical morphology and clustering about principal curves. *Journal of the American Statistical Association*, 87(417):7–16, March 1992.

[5] Bishop, Christopher M., Svensén, Markus, and Williams, Christopher K. I. Gtm: The generative topographic mapping. *Neural Computation*, 10:215–234, 1998.

[6] Bishop, Yvonne M. and Fienberg, Stephen E. *Discrete Multivariate Analysis Theory and Practice*. Springer, New York, 2007.

[7] Bowman, Adrian W. and Azzalini, Adelchi. *Applied Smoothing Techniques for Data Analysis*. Clarendon Press, Oxford, 1997.

[8] Chacón, José E., Duong, Tarn, and Wand, M. P. Asymptotics for general multivariate kernel density derivative estimators. *Statistica Sinica*, 21(2):807–840, April 2011.

[9] Chang, Kui-yu and Ghosh, Joydeep. Three-dimensional model-based object recognition and pose estimation using. In *Probabilistic Principal Surfaces, SPIE: Applications of Artificial Neural Networks in Image Processing V*, pages 192–203, 2000.

[10] Chang, Kui-Yu and Ghosh, Joydeep. A unified model for probabilistic principal surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23 (1):22–41, January 2001.

[11] Charlton, M., Brunsdon, C., Demsar, Urska, Harris, Paul, and Fotheringham, Stewart. Principal component analysis: from global to local. In *13th AGILE International Conference on Geographic Information Science*. Guimarães, Portugal, 2010.

[12] Chen, Xiaopeng, Zhou, Youxue, Huang, Xiaosan, and Li, Chengrong. Adaptive bandwidth mean shift object tracking. In *IEEE Conference on Robotics, Automation and Mechatronics*, pages 1011 –1017, September 2008.

[13] Cheng, Y. Mean shift, mode seeking and clustering. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.

[14] Comaniciu, Dorin. An algorithm for data-driven bandwidth selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):281–288, 2003.

[15] Comaniciu, Dorin and Meer, Peter. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, May 2002.

[16] Comaniciu, Dorin, Ramesh, Visvanathan, and Meer, Peter. The variable bandwidth mean shift and data-driven scale selection. In *ICCV'01*, pages 438–445, 2001.

[17] Delicado, Pedro. Another look at principal curves and surfaces. *Journal of Multivariate Analysis*, 77:84–116, April 2001.

[18] Der, Ralf, Steinmetz, Ulrich, Balzuweit, Gerd, and Schüürmann, Gerrit. Nonlinear principal component analysis, 1998. URL http://www.informatik.uni-leipzig.de/~der/Veroeff/npcafin.ps.gz.

[19] Dong, D. and McAvoy, T.J. Nonlinear principal component analysis-based on principal curves and neural networks. In *American Control Conference, 1994*, volume 2, pages 1284–1288, 1994.

[20] Duchamp, Tom and Stuetzle, Werner. *Geometric properties of principal curves in the plane*, volume 109, pages 135–252. Springer, 1996.

[21] Einbeck, J. and Evers, L. *LPCM: Local principal curve methods*, 2010. URL http://CRAN.R-project.org/package=LPCM. R package version 0.41-6.

[22] Einbeck, J. and Evers, L. Localized regression on principal manifolds. In *25th International Workshop on Statistical Modelling (IWSM 2010)*, 2010. URL http://eprints.gla.ac.uk/45527/.

[23] Einbeck, J. and Zayed, M. Some asymptotics for localized principal components and curves. *Unpublished working paper, Durham University*, 2011.

[24] Einbeck, J., Evers, L., and Hinchliff, K. Data compression and regression based on local principal curves. In A. Fink, B. Lausen, W. Seidel, and A. Ultsch, editors, *Advances in Data Analysis, Data Handling and Business Intelligence*, pages 701–712, Heidelberg, 2010. Springer.

[25] Einbeck, J., Evers, L., and Powell, B. Data compression and regression through local principal curves and surfaces. *International Journal of Neural Systems*, 20 (3):177–192, 2010.

[26] Einbeck, Jochen. Bandwidth selection for mean-shift based unsupervised learning techniques: a unified approach via self-coverage. *Journal of Pattern Recognition Research*, 2:175–192, 2011.

[27] Einbeck, Jochen, Tutz, Gerhard, and Evers, Ludger. Local principal curves. *Statistics and Computing*, 15(4):301–313, October 2005.

[28] El Machkouri, Mohamed and Stoica, Radu. Asymptotic normality of kernel estimates in a regression model for random fields. *Journal of Nonparametric Statistics*, 22(8):955–971, 2010.

[29] Fan, J. and Gijbels, I. *Local Polynomial Modelling and Its Applications*. Chapman and Hall, 1995.

[30] Fisher, Irving. I discovered the phillips curve: A statistical relation between unemployment and price changes. *Journal of Political Economy*, 81(2):496–502, 1973. ISSN 00223808.

[31] Flett, T. M. *Mathematical Analysis*. McGraw-Hill Publishing Company Ltd., London, UK, 1966.

[32] Friedman, Milton. The role of monetary policy. *The American Economic Review*, 58(1):1–17, 1968. ISSN 00028282.

[33] Fukunaga, K. and Hostetler, L. D. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21:32–40, 1975.

[34] Fukunaga, K. and Olsen, D.R. An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, C-20(2):176 – 183, February 1971. ISSN 0018-9340.

[35] Grell, Michaela. Eu financial services: insurance. Technical report, European Comission - Eurostat, 2008. ISSN 1977-0316, Catalogue number: KS-SF-08-075-EN-N.

[36] Harville, David A. *Matrix Algebra From a Statistician's Perspective*. Springer, New York, 1997.

[37] Hastie, Trevor. *Principal Curves and Surfaces*. PhD thesis, Stanford University, Stanford Linear Accelerator Center, Stanford, California, 1984.

[38] Hastie, Trevor and Stuetzle, Werner. Principal curves. *Journal of the American Statistical Association*, 84(406):502–516, June 1989. ISSN 01621459.

[39] Hastie, Trevor and Weingessel, Andreas. *Package 'princurve'*, 2011. URL http://cran.r-project.org/web/packages/princurve/. R package version 1.1-11.

[40] Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome. *The Elements of Statistical Learning, Second Edition: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, 2nd edition, February 2009. ISBN 0387848576.

[41] Hastie, Trevor, De Ath, Glenn, and Walsh, Chris. *Principal curve analysis*, 2011. URL http://cran.r-project.org/web/packages/pcurve/. R package version 0.6-3.

[42] Jolliffe, I. T. *Principal Component Analysis*. Springer-Verlag New York, Inc., 2nd edition, 2002.

[43] Kambhatla, Nandakishore and Leen, Todd K. Dimension reduction by local principal component analysis. *Neural Computation*, 9:1493–1516, October 1997. ISSN 0899-7667.

[44] Kégl, B. and Krzyzak, A. Piecewise linear skeletonization using principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):59–74, January 2002.

[45] Kégl, B., Krzyzak, A., Linder, T., and Zeger, K. Learning and design of principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3): 281–297, March 2000.

[46] Kégl, Balazs. *Principal Curves: Learning, Design, and Applications*. PhD thesis, The Department of Computer Science, Concordia University, Montreal, Quebec, Canada, December 1999.

[47] Krzanowski, W. J. *Principles of multivariate analysis : a user's perspective*. Oxford University Press, 2000. ISBN 9780198507086.

[48] LeBlanc, Michael and Tibshirani, Robert. Adaptive principal surfaces. *Journal of the American Statistical Association*, 89(425):53–64, March 1994.

[49] Li, Xiangru, Hu, Zhanyi, and Wu, Fuchao. A note on the convergence of the mean shift. *Pattern Recognition*, 40(6):1756–1762, 2007. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/j.patcog.2006.10.016.

[50] Mardia, K. V., Bibby, J. M., and Kent, J. T. *Multivariate analysis*. Academic Press, London, 1979. ISBN 0124712525.

[51] Micheli-Tzanakou, Evangelia. *Supervised and unsupervised pattern recognition: feature extraction and computational intelligence*. CRC Press, Inc., Boca Raton, FL, USA, 2000. ISBN 0-8493-2278-2.

[52] Mills, Terence. Exploring the relationship between gold and the dollar. *Significance*, 1(3):113–115, 2004.

[53] Ming-Ming, Y., Jian, L., Chuan-CAI, L., and Jing-Yu, Y. Similarity preserving principal curve: an optimal one-dimensional feature extractor for data representation. *IEEE Transactions on Neural Networks*, 21(9):1445–1456, September 2010.

[54] Moser, J. W. A principal component analysis of labor market indicators. *Eastern Economic Journal*, X(3):243–257, 1984.

[55] Ould-Saïd, Elias and Lemdani, Mohamed. Asymptotic properties of a nonparametric regression function estimator with randomly truncated data. *Annals of the Institute of Statistical Mathematics*, 58(2):357–378, June 2006.

[56] Ozertem, Umut and Erdogmus, Deniz. Locally defined principal curves and surfaces. *Journal of Machine Learning Research*, 12:1249–1286, April 2011.

[57] Parzen, Emanuel. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[58] Pearson, K. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572, 1901. URL http://www.cimpa.ucr.ac.cr/socccad/pearson1901.pdf.

[59] Pēna, Marian, Barbakh, Wesam, and Fyfe, Colin. *Topology-Preserving Mappings for Data Visualisation*, pages 131–150. In A.N. Gorban et al., editors, Principal Manifolds for Data Visualization and Dimension Reduction. Springer, Berlin, 2008.

[60] Peres-Neto, Pedro, R., Jackson, Donald A., and Somers, Keith M. How many principal components? stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis*, 49(4):974–997, June 2005.

[61] Petersen, K. B. and Pedersen, M. S., et. al. The matrix cookbook. Technical report, 2006. URL http://www2.imm.dtu.dk/pubdb/views/edoc_download.php/3274/pdf/imm3274.pdf.

[62] Phillips, A. W. The relation between unemployment and the rate of change of money wage rates in the united kingdom, 1861-1957. *Economica*, 25(100):283–299, November 1958.

[63] Reinhard, K. and Niranjan, M. Subspace models for speech transitions using principal curves. In *Proceedings of Institute of Acoustics*, pages 53–60, 1998.

[64] Reinhard, K. and Niranjan, M. Parametric subspace modeling of speech transitions. *Speech Communication*, 27(1):19–42, 1999.

[65] Ruppert, D. and Wand, M. P. Multivariate locally weighted least squares regression. *Annals of Statistics*, 22:1346–1370, 1994.

[66] Schaal, S., Vijayakumar, S., and Atkeson, C.G. Local dimensionality reduction. In M.I. Jordan, M.J. Kearns, and S.A. Solla, editors, *Advances in Neural Information Processing Systems 10*. MIT Press, Cambridge, MA, 1998.

[67] Scott, David W. *Multivariate Density Estimation: Theory, Practice, and Visualization*. John Wiley & Sons, Inc., 1992. ISBN 9780471547709.

[68] Sheather, S. J. and Jones, M. C. A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, 53(3):683–690, 1991. ISSN 00359246.

[69] Silverman, R. W. *Density Estimation For Statistics And data Analysis*. Chapman and Hall, London, 1986.

[70] Smola, Alex J., Mika,Sebastian, and Bernhard Schölkopf. Quantization functionals and regularized principal manifolds, 1998. URL http://ida.first.fhg.de/publications/SmoMikSch98.ps.

[71] Stanford, Derek and Raftery, Adrian E. Principal curve clustering with noise. Technical report, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997.

[72] Stevens, James P. *Applied multivariate statistics for the social sciences*. Taylor & Francis Group, LLC, Hove, East Sussex, US, 5th edition, 2009.

[73] Tarpey, Thaddeus and Flury, Bernard. Self-consistency: A fundamental concept in statistics. *Statistical Science*, 11(3):229–243, 1996.

[74] Theil, H. Best linear index numbers of prices and quantities. *Econometrica*, 28 (2):464–480, 1960.

[75] Tibshirani, Robert. Principal curves revisited. *Statistics and Computing*, 2:183–190, 1992.

[76] Tintner, G. Some applications of multivariate analysis to economic data. *Journal of the American Statistical Association*, 41(236):472–500, 1946.

[77] Verbeek, J. J., Vlassis, N., and B. Kröse. A k-segments algorithm for finding principal curves. *Pattern Recognition Letters*, 23:1009–1017, 2000.

[78] Wand, M. P. and Jones, M. C. Comparison of smoothing parameterizations in bivariate kernel density estimation. *Journal of the American Statistical Association*, 88:520–528, 1993.

[79] Wand, M. P. and Jones, M. C. *Kernel Smoothing*. Chapman and Hall, London, 1995.

[80] Wand, Matt (R port and updates by Brian Ripley). *Functions for kernel smoothing for Wand & Jones (1995)*, June 2011. URL http://cran.r-project.org/web/packages/KernSmooth. R package version 2.23-6.

[81] Wang, L., Assadi, A. H., and Spalding, E. P. Tracing branched curvilinear structures with a novel adaptive local pca algorithm. In *Proceedings of the 2008 International Conference on Image Processing, Computer Vision, and Pattern Recognition*, volume 17, pages 557–563. CSREA Press, Athens, GA, 2008.

[82] Wilson, D.J.H., Irwin, G.W., and Lightbody, G. RBF principal manifolds for process monitoring. *IEEE Transactions on Neural Networks*, 10(6):1424–1434, November 1999.

[83] Yao, Fang. Asymptotic distributions of nonparametric regression estimates for longitudinal or functional data. *Journal of Multivariate Analysis*, 98:40–56, 2007.

[84] Yeomans, Keith A. and Golder, Paul A. The guttman-kaiser criterion as a predictor of the number of common factors. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 31(3):221–229, 1982. ISSN 00390526.

[85] Yu, Zhangsheng and Lin, Xihong. Nonparametric regression using local kernel estimating equations for correlated failure time data. *Biometrika*, 95(1):123–137, 2008.

[86] Zayed, M. and Einbeck, J. Constructing economic summary indexes via principal curves. In *COMPSTAT 2010 Proceedings (e-book)*, pages 1709–1716, 2010.

[87] Zhao, Qi, Yang, Zhi, Tao, Hai, and Wentai, Liu. Evolving mean shift with adaptive bandwidth: A fast and noise robust approach. In *ACCV (1)'09*, pages 258–268, 2009.