TUCS

Seppo Pulkkinen

Efficient Optimization Algorithms for Nonlinear Data Analysis

Turku Centre *for* Computer Science

TUCS Dissertations
No 189, December 2014

Efficient Optimization Algorithms for Nonlinear Data Analysis

Seppo Pulkkinen

To be presented, with the permission of the Faculty of Mathematics and Natural Sciences of the University of Turku, for public criticism in Auditorium Cal 1 on December 5, 2014, at 12 noon.

University of Turku
Department of Mathematics and Statistics
FI-20014 Turku
Finland

2014

Supervisors

Marko M. Mäkelä
Department of Mathematics and Statistics
University of Turku
Finland

Napsu Karmitsa
Department of Mathematics and Statistics
University of Turku
Finland

Reviewers

Adil M. Bagirov
School of Engineering and Information Technology
Federation University Australia
Australia

Jaakko Peltonen
Department of Information and Computer Science
Aalto University
and
School of Information Sciences
University of Tampere
Finland

Opponent

Leo Liberti
LIX
École Polytechnique
France

ISBN 978-952-12-3152-0
ISSN 1239-1883

Abstract

Identification of low-dimensional structures and main sources of variation from multivariate data are fundamental tasks in data analysis. Many methods aimed at these tasks involve solution of an optimization problem. Thus, the objective of this thesis is to develop computationally efficient and theoretically justified methods for solving such problems.

Most of the thesis is based on a statistical model, where ridges of the density estimated from the data are considered as relevant features. Finding ridges, that are generalized maxima, necessitates development of advanced optimization methods. An efficient and convergent trust region Newton method for projecting a point onto a ridge of the underlying density is developed for this purpose. The method is utilized in a differential equation-based approach for tracing ridges and computing projection coordinates along them. The density estimation is done nonparametrically by using Gaussian kernels. This allows application of ridge-based methods with only mild assumptions on the underlying structure of the data.

The statistical model and the ridge finding methods are adapted to two different applications. The first one is extraction of curvilinear structures from noisy data mixed with background clutter. The second one is a novel nonlinear generalization of principal component analysis (PCA) and its extension to time series data. The methods have a wide range of potential applications, where most of the earlier approaches are inadequate. Examples include identification of faults from seismic data and identification of filaments from cosmological data. Applicability of the nonlinear PCA to climate analysis and reconstruction of periodic patterns from noisy time series data are also demonstrated.

Other contributions of the thesis include development of an efficient semidefinite optimization method for embedding graphs into the Euclidean space. The method produces structure-preserving embeddings that maximize interpoint distances. It is primarily developed for dimensionality reduction, but has also potential applications in graph theory and various areas of physics, chemistry and engineering. Asymptotic behaviour of ridges and maxima of Gaussian kernel densities is also investigated when the kernel bandwidth approaches infinity. The results are applied to the nonlinear PCA and to finding significant maxima of such densities, which is a typical problem in visual object tracking.

Acknowledgements

Overall, my journey to the PhD degree was an enjoyable, though often a stressful experience. This would not have been possible without numerous people providing their support. First of all, I thank my supervisors Marko Mäkelä and Napsu Karmitsa for their guidance during this four-year endeavour. They provided me invaluable assistance especially during the early stages of my research. They taught me how to translate my unorganized ideas into a coherent text suitable for publication in a scientific journal. The early versions of my manuscripts were admittedly lacking the needed structure and mathematical formalism.

My supervisors were always willing to share their deep knowledge on optimization and numerical methods with me, which turned out to be crucial for this work. I also appreciate their conservative attitude towards my more innovative ideas. Those clear-minded and sensible people helped me to keep my feet on the ground when I got too excited about fancy research ideas. After all, the basic goals were: just get the necessary papers published and complete the PhD in due time.

I owe special thanks to Matti Vuorinen. He introduced me the fascinating field of optimization when I was completing my Master's degree in Helsinki. I took several of his courses that sparked my interest towards computational mathematics and numerical algorithms.

Naturally, I thank the pre-examiners Adil Bagirov and Jaakko Peltonen for carefully reviewing this manuscript and providing very insightful comments. It was an honour to have Leo Liberti to act as the opponent.

Several fellow students contributed to this thesis. In particular, the stimulating discussions with Paavo Nevalainen and Antti Nurkkala were a valuable source of new ideas and perspectives. This also applies to numerous people in the Turku Centre for Computer Science (TUCS) graduate school. My colleagues at the maths department with their colourful personalities provided a very vibrant working environment. Of course, I also thank the administrative staff for handling the practical matters.

I gratefully acknowledge the financial support received from TUCS and the maths department for providing me excellent working conditions. This work was also supported by Tapio Westerlund (Åbo Akademi) via Academy

of Finland (project no. 127992).

The Finnish Meteorological Institute (FMI) was my ”second home” during these four years. I worked there for several summers. Just to name a few people, Ari-Matti Harri, Jarmo Koistinen and Timo Kuitunen shared their expertise on meteorology with me and followed the progress of my PhD thesis with great interest. These summer periods filled with practical work on weather radars gave me the desperately needed breaks from mathematics. The people at FMI pushed me towards practical applications and encouraged me to explore career possibilities outside the academia.

Finally, I thank my parents for all their love and support.

List of original publications

I S. Pulkkinen, M. M. Mäkelä, and N. Karmitsa. A generative model and a generalized trust region Newton method for noise reduction. *Computational Optimization and Applications*, 57(1):129–165, 2014

II S. Pulkkinen, M. M. Mäkelä, and N. Karmitsa. A continuation approach to mode-finding of multivariate Gaussian mixtures and kernel density estimates. *Journal of Global Optimization*, 56(2):459–487, 2013

III S. Pulkkinen. Ridge-based method for finding curvilinear structures from noisy data. *Computational Statistics and Data Analysis*, 82:89–109, February 2015

IV S. Pulkkinen. Nonlinear kernel density principal component analysis with application to climate data. *Statistics and Computing*, 2014. accepted (based on TUCS Technical Report 1091), doi:10.1007/s11222-014-9539-0

V S. Pulkkinen. Finding graph embeddings by incremental low-rank semidefinite programming. submitted to Optimization Methods and Software, conditionally accepted (based on TUCS Technical Report 1069)

Contents

1 Introduction and outline of the work 1

1.1 Introduction 1

1.2 Outline of the work 4

2 Ridges and related statistical models 7

2.1 Ridge definition and basic properties 7

2.2 Applications of ridges and related concepts 10

2.2.1 Image processing 10

2.2.2 Theoretical chemistry 11

2.2.3 Global optimization 12

2.3 Estimation of underlying structures from density ridges . . . 12

2.4 Kernel density estimation 16

2.5 Practical error estimates and examples 19

2.5.1 Model bias 19

2.5.2 Combined model and density estimation bias 20

3 Algorithms for finding density ridges and global maxima 23

3.1 Mean shift-based methods 24

3.2 The subspace constraint 25

3.3 The trust region Newton method 26

3.3.1 Overview of the method 26

3.3.2 Solution of the trust region subproblem 28

3.3.3 Convergence to a ridge point 30

3.4 Comparison between trust region and mean shift methods . . 33

3.4.1 Theoretical considerations 33

3.4.2 Numerical experiments 34

3.5 Finding global modes of Gaussian mixtures and kernel densities 36

3.5.1 The Gaussian convolution and homotopy continuation 38

3.5.2 Choice of initial values 39

3.5.3 Implementation of the continuation method 42

3.5.4 Test results 43

4 Identification of curvilinear structures from noisy data 49
4.1 Relation to earlier research 49
4.2 The filament model 51
4.3 Properties of ridge curves 53
4.4 Differential equation formulation 55
4.5 The algorithmic framework 58
4.5.1 Main algorithms 58
4.5.2 Algorithm for tracing a ridge curve segment 61
4.6 Numerical experiments 64
4.6.1 Datasets and test setup 64
4.6.2 Illustrative examples 64
4.6.3 Performance evaluation 68

5 Nonlinear principal component analysis 69
5.1 Relation to earlier research 70
5.1.1 Linear PCA 70
5.1.2 Kernel- and neural network-based approaches 71
5.1.3 Local PCA methods 72
5.1.4 The proposed method 73
5.2 Nonlinear kernel density PCA 74
5.2.1 Properties of ridge sets 74
5.2.2 Estimation of principal component scores from ridges 76
5.2.3 Algorithm for computing principal component scores . 80
5.2.4 Extension to time series data 83
5.3 Experiments with climate and time series data 84

6 Graph-based dimensionality reduction 91
6.1 Problem definition and the proposed approach 92
6.2 Low-rank semidefinite programming 94
6.2.1 Notation and optimality conditions 94
6.2.2 Matrix formulation of the graph embedding problem . 96
6.3 Strong duality of the graph embedding SDP 96
6.4 Incremental low-rank algorithm 97
6.4.1 Problem formulation 97
6.4.2 The algorithm 98
6.5 Numerical experiments 99
6.5.1 Random graphs 101
6.5.2 Synthetic and machine learning datasets 104

7 Conclusions and discussion 109

Chapter 1

Introduction and outline of the work

1.1 Introduction

Analysis of nonlinear data and finding modes (maxima) of multimodal probability densities are tasks appearing in numerous research fielfs such as statistics and computer science. Many methods aimed at these tasks involve solution of an optimization problem. Therefore, the objective of this thesis is to develop efficient and theoretically justified methods for such problems, and thus bridge the gap between optimization and statistics.

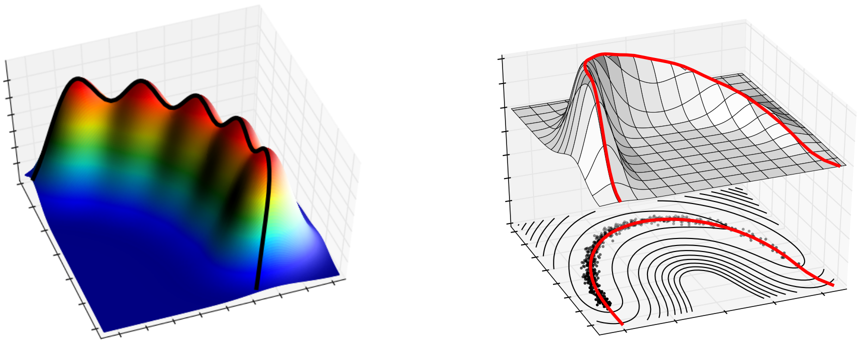
Three different data analysis and machine vision tasks are covered in this thesis with emphasis on nonlinear optimization methods.

1 **Dimensionality reduction:** Describe high-dimensional data in a low-dimensional coordinate system such that relevant information is preserved. The data can be any collection of real-valued vectors (e.g. digital images [119], speech signals [129], climate data [105] or biomedical data [79]). Dimensionality reduction can be used for visualization purposes. It also facilitates other tasks such as classification, clustering and identifying the main sources of variation. Such tasks can be carried out more efficiently and reliably on low-dimensional data.

2 **Shape extraction:** Extract curves and surfaces from low-dimensional scattered point sets or spatial data (e.g. earthquake patterns [115], filamentary and wall-like shapes in galaxy clusters [20], GPS tracks [22] or feature points extracted from images [6, 117]).

3 **Mode finding:** Efficiently find global or significant modes of probability densities. This global optimization problem arises, for instance, in real-time visual object tracking, where the most significant mode represents the most likely state of the tracked object [59, 112].

Our primary approach to tasks 1 and 2 is to use *ridges* of *density* functions for estimation of underlying structure from point sets. The assumption is that the points represent observations from some unknown distribution whose probability density is estimated from the observations. This idea originates from Ozertem and Erdogmus [96], and it has been later refined by Baş et al. [9–11] and Ghassabeh et al. [51]. A rigorous statistical treatment for this idea has been given by Genovese et al. [49]. Nowadays, ridge-based methods have become popular in many applications such as medical imaging (e.g. [10] and [11]) and analysis of seismic data (e.g. [55]). A comprehensive list of related applications will be given in Chapter 2.



(a) general function

(b) density of a point set

Figure 1.1: Examples of function ridges.

A ridge of a function surface corresponds to the intuition of a landscape ridge as a narrow elevated region between peaks. Paths lying on top of ridges are of particular interest in our applications. This is illustrated in Figure 1.1a showing a function ridge with a curve lying on top of it. A statistical interpretation for such a curve can be obtained via a density function, as illustrated in Figure 1.1b. That is, when a point set is distributed around a curve in a plane, the ridge curve of its density function passes through regions of high concentration, and thus describes the underlying structure. As we will see in the following, this idea can be generalized to r-dimensional ridges in d-dimensional space.

The following questions are studied in this thesis because addressing them is crucial for practical applicability of ridge-based methods.

(i) To what extent do ridges of the underlying density describe the structure of a point set?

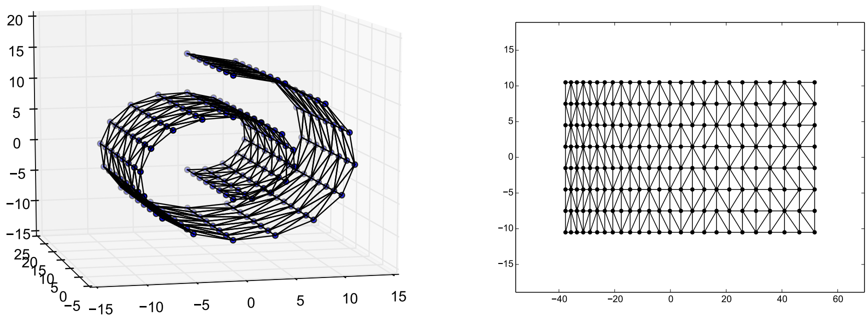
(ii) How to find a ridge of a density function reliably and efficiently?

(iii) Do ridges of the underlying density induce a coordinate system and how to obtain a representation for a point set in this coordinate system?

The statistical theory developed in [49] answers to question (i) to a large extent. As ridges are generalized maxima, addressing question (ii) necessitates development of advanced optimization methods. It has been partially answered in [51] and [96], where a simple method is developed for projecting a point onto a ridge of a density. Question (iii), on the other hand, has not been studied in the context of data analysis. Providing an answer to this question is crucial when using ridges for dimensionality reduction.

Unfortunately, the ridge-based approach is not ideal for high-dimensional data. This is due to inherent difficulties in high-dimensional density estimation (i.e. the "curse of dimensionality") [110]. For this reason, one part of this thesis is devoted to computational improvements to the *maximum variance unfolding* (MVU) method by Weinberger and Saul [131]. Differently to density-based methods, this geometric graph-based dimensionality reduction method is well-suited for high-dimensional data. However, it does not have an easy interpretation in terms of an underlying model for noisy data (see [5] for recent research on this topic).

The MVU method constructs a *neighbourhood graph* from a given point set and produces a low-dimensional representation based on the structure of the graph. Such a representation (i.e. an *embedding* of the graph) is obtained by maximizing interpoint distances so that distances between neighbouring points are preserved. This idea is illustrated in Figure 1.2 showing the input point set and its embedding. Obtaining such an embedding involves solution of a difficult optimization problem. Until now, the applicability of the MVU method has been severely limited due to lack of efficient methods for solving this problem. As we will see in the following, this problem has a rich theory, which also motivates choosing the MVU method as one research topic in this work.



(a) input point set in $\mathbb{R}^3$

(b) embedded graph in $\mathbb{R}^2$

Figure 1.2: Neighbourhood graph of a point set in the input space and the embedding obtained by unfolding the graph.

Task 3 mentioned on page 1, that is mode finding, is not directly related to the other ones. Nevertheless, it is relevant for this thesis. This is because ridge finding methods are also applicable to finding modes as a special case. For this task, our focus is on *Gaussian mixtures* and *kernel densities*. This is due to their special structure and the universal ability to model other probability distributions [110,128]. In addition, finding not only a significant one but all modes of such a density is of interest in many applications. This problem arises, for instance, in nonparametric clustering [34]. Finding modes of more general posterior distributions is also an important problem in Bayesian data analysis [48]. A multimodal density is shown in Figure 1.3.

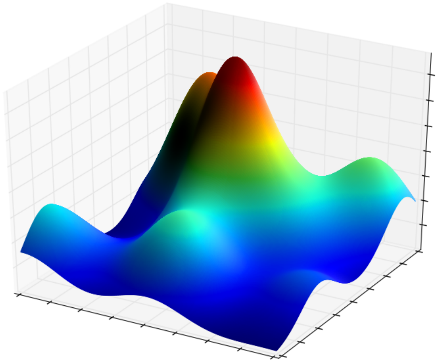


Figure 1.3: Example of a multimodal density.

The emphasis of this thesis is on algorithmic development, and the approaches taken are quite pragmatic. The theoretical results established in this work serve the purpose of justifying the algorithms or giving a guarantee that the algorithms give the desired results. We omit technical proofs in this introductory part, but they can be found in **Papers** [I]–[V] and the technical reports for those who are interested. Another focus area is performance of the developed algorithms. Numerical performance comparisons with existing algorithms are given whenever possible.

1.2 Outline of the work

This thesis is organized as follows. The notion of a ridge is rigorously defined in the form of an r-dimensional *ridge set* in the d-dimensional space $\mathbb{R}^d$ in Chapter 2. The theory developed in [49] is reviewed to provide a statistical justification for estimation of nonlinear structures from density ridges, which addresses question (i). *Nonparametric* density estimation from the data by using Gaussian *kernels* [30, 110, 128] is considered for computational implementation of ridge-based methods. As it turns out, this powerful density estimation approach offers a great amount of flexibility, as no restrictive assumptions are imposed on the data. On the other hand, this approach has its own computational challenges that the following chapters attempt to address.

Chapter 3 consists of two parts. The first part is based on **Paper I** that addresses question (ii). The contribution of this paper is a novel generalization of the classical *trust region* Newton method by Moré and Sorensen [91] to finding not only maxima but also ridges. It is shown that the method can be used for (approximately) projecting points onto r-dimensional ridges of their underlying density that is estimated by Gaussian kernels. Another important contribution of [I] is a rigorous proof for convergence of the method to a ridge point. Finally, it is empirically shown that the proposed method has significantly faster convergence rate than the earlier *mean shift* method [31, 34, 47] and its subspace-constrained variant [96] for finding modes and ridges, respectively. Fast convergence is desirable because of high computational cost of evaluating Gaussian kernel densities.

The second part of Chapter 3 deals with another application of a trust region Newton method based on **Paper II**. There the authors consider finding global or significant modes of Gaussian mixtures and kernel densities. The proposed approach is based on a *homotopy continuation* technique [92, 133], where the highly multimodal density is smoothly deformed into a unimodal one. Tracing the mode of the density along such a transformation yields a computationally efficient method that finds global modes with a high probability. A potential application area of this method is real-time visual object tracking [59, 112].

A more practically oriented approach is taken in Chapter 4 based on **Paper III**. There the ridge projection method presented in Chapter 3 is combined with the statistical theory presented in Chapter 2 and a kernel density estimator implemented in [41]. This results in a highly efficient method for extracting multiple curvilinear structures from noisy point sets. The method is based on the theory of ridge curves [36,89] that are formulated as a solution to a differential equation by utilizing the theory of gradient extremal curves [19, 64]. A predictor-corrector method utilizing the ridge projection method of Chapter 3 is used for the numerical implementation. As the predictor-corrector method yields a parametrization of a ridge curve, it addresses questions (ii) and (iii) in the case $r = 1$, but differently to the projection method presented in Chapter 3 that only produces a set of unordered points along such a curve. Applicability of the method to detection of faults from seismic data and to identification of filamentary structures from galaxy clusters are demonstrated.

The properties of ridge sets are explored further in Chapter 5 based on **Paper IV** addressing question (iii). The contribution of this paper is development of a nonlinear generalization of *principal component analysis*

(PCA) [69, 97]. The linear PCA is a well-established, but rather limited tool for reducing the dimensionality and identifying the main sources of variation from multivariate data. To address its limitations, the proposed method utilizes the structure of ridge sets to construct a nonlinear coordinate system. This is done by utilizing the results established by Miller [89]. It is shown that the principal component coordinates of a point set can be obtained one by one by successively projecting the points onto lower-dimensional ridge sets of a Gaussian kernel density. Such projection curves are defined as a solution to a differential equation. The equations are solved by a predictor-corrector method that utilizes the ridge projection method of Chapter 3. The applicability of the nonlinear PCA and its advantages over the linear PCA are demonstrated with two examples. These are obtaining a low-dimensional representation of a highly nonlinear climate model dataset and separation of a periodic component from an atmospheric time series.

Finally, Chapter 6 based on **Paper V** is devoted to the MVU method [131]. The graph embedding problem arising in MVU can either be formulated as a *semidefinite program* (SDP) or a *quadratically constrained quadratic program* (QCQP). The solution of the QCQP gives the SDP solution when the embedding dimension is sufficiently large. These two approaches are compared, and an efficient solution method based on the QCQP formulation is developed in [V]. The method solves a sequence of small-dimensional quadratic problems and increases the embedding dimension until the solution of the SDP is obtained. This approach is based on the theory of semidefinite programs and their quadratic low-rank formulations developed in [24, 56, 70] and the duality theory of semidefinite programs [123].

The research topics covered in this thesis and their relations are illustrated in Figure 1.4.

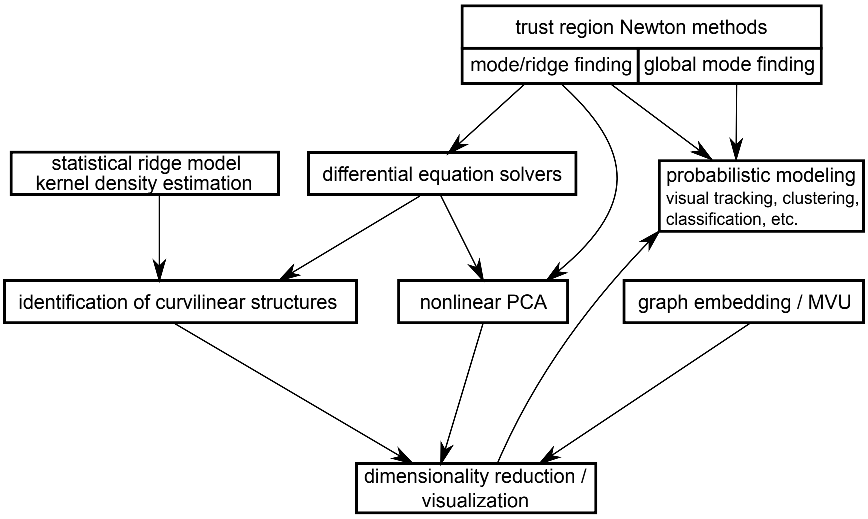

trust region Newton methods
mode/ridge finding
global mode finding
statistical ridge model
kernel density estimation
differential equation solvers
probabilistic modeling
visual tracking, clustering, classification, etc.
identification of curvilinear structures
nonlinear PCA
graph embedding / MVU
dimensionality reduction / visualization

Figure 1.4: Research topics covered in this thesis and their relations.

Chapter 2

Ridges and related statistical models

In this chapter we formally define the notion of a ridge and review the literature on ridge-based methods used in different application areas. We then proceed by reviewing the necessary statistical theory for estimation of underlying structure in point sets from density ridges. The last part of this chapter is devoted to density estimation by nonparametric kernel methods.

2.1 Ridge definition and basic properties

Generalizing the intuition of a ridge curve, we now define an r-dimensional ridge in the d-dimensional Euclidean space $\mathbb{R}^d$, where $r < d$. The definition is based on the theory presented by Eberly [43]. A key property defining a ridge is readily observed from Figure 1.1a, which illustrates the special case with $r = 1$ and $d = 2$. That is, the centerline of a ridge connects the maxima of the function, being a local maximum along the direction of greatest downward curvature at each point it passes through.

More formally, the curvature of a twice differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ at a given point $\boldsymbol{x}$ is determined by its second derivatives. By the chain rule of differentiation, we obtain the second *directional derivative*

$$\nabla^2_{\boldsymbol{v}} f(\boldsymbol{x}) = \frac{d^2}{dy^2} f(\boldsymbol{x} + y\boldsymbol{v})\bigg|_{y=0} = \boldsymbol{v}^T \nabla^2 f(\boldsymbol{x}) \boldsymbol{v}.$$

of f at $\boldsymbol{x}$ along a direction $\boldsymbol{v}$ such that $\|\boldsymbol{v}\| = 1$.

By defining the *Lagrangian*

$$L_{\boldsymbol{x}}(\boldsymbol{v}; \lambda) = \boldsymbol{v}^T \nabla^2 f(\boldsymbol{x}) \boldsymbol{v} - \lambda(\boldsymbol{v}^T \boldsymbol{v} - 1)$$

and equating its gradient with respect to $\boldsymbol{v}$ to zero, we make the following observation. The eigenvectors $\{\boldsymbol{v}_i(\boldsymbol{x})\}_{i=1}^d$ and the corresponding eigenvalues

$\lambda_1(\boldsymbol{x}) \geq \lambda_2(\boldsymbol{x}) \geq \cdots \geq \lambda_d(\boldsymbol{x})$ of the Hessian $\nabla^2 f(\boldsymbol{x})$ are stationary points and values of $h(\boldsymbol{v}) = \nabla^2_{\boldsymbol{v}} f(\boldsymbol{x})$, respectively, under the constraint $\|\boldsymbol{v}\| = 1$.

By the above observation, the Hessian eigenvectors $\boldsymbol{v}_i(\cdot)$ associated with the $d - r$ algebraically smallest eigenvalues correspond to the orthogonal directions of smallest second derivatives of f. When they are negative, the downward curvature of f is greatest along these directions. Thus, we define an r-dimensional ridge point as a local maximum of f restricted to a hyperplane via the function

$$g(\boldsymbol{y}) = f(\boldsymbol{u}(\boldsymbol{y})), \quad \text{where} \quad \boldsymbol{u}(\boldsymbol{y}) = \boldsymbol{x} + \sum_{i=r+1}^{d} y_{i-r} \boldsymbol{v}_i(\boldsymbol{x}) \tag{2.1}$$

for some $\boldsymbol{x} \in \mathbb{R}^d$.

By using the chain rule of differentiation, the conditions for $\boldsymbol{y}$ to be a local maximum of g are written as

$$\nabla g(\boldsymbol{y}) = \boldsymbol{V}(\boldsymbol{x})^T \nabla f(\boldsymbol{u}(\boldsymbol{y})) = \boldsymbol{0}, \tag{2.2}$$

$$\nabla^2 g(\boldsymbol{y}) = \boldsymbol{V}(\boldsymbol{x})^T \nabla^2 f(\boldsymbol{u}(\boldsymbol{y})) \boldsymbol{V}(\boldsymbol{x}) \quad \text{is negative definite,} \tag{2.3}$$

where

$$\boldsymbol{V}(\boldsymbol{x}) = \begin{bmatrix} \boldsymbol{v}_{r+1}(\boldsymbol{x}) & \boldsymbol{v}_{r+2}(\boldsymbol{x}) & \ldots & \boldsymbol{v}_d(\boldsymbol{x}) \end{bmatrix}.$$

Letting $\boldsymbol{y} = \boldsymbol{0}$ in equation (2.1) and applying the identities

$$\boldsymbol{V}(\boldsymbol{x})^T \boldsymbol{V}(\boldsymbol{x}) = \boldsymbol{I} \quad \text{and} \quad \boldsymbol{V}(\boldsymbol{x})^T \nabla^2 f(\boldsymbol{x}) = \boldsymbol{\Lambda}(\boldsymbol{x}) \boldsymbol{V}(\boldsymbol{x})^T,$$

where

$$\boldsymbol{\Lambda}(\boldsymbol{x}) = \operatorname{diag}\left[\lambda_{r+1}(\boldsymbol{x}), \lambda_{r+2}(\boldsymbol{x}), \ldots, \lambda_d(\boldsymbol{x})\right]$$

to (2.3), conditions (2.2) and (2.3) yield the first two conditions of the following definition for a ridge point $\boldsymbol{x}$ and a set of such points. In order to make the definition well-posed, we also require that the first r eigenvalues differ from the $r + 1$-th eigenvalue. In addition, we require that the first r eigenvalues are mutually distinct. These assumptions, that are required for continuity of the corresponding eigenvectors, will be justified later.

Definition 2.1.1 ([43]) *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a twice differentiable function and let $0 \leq r < d$. A point $\boldsymbol{x} \in \mathbb{R}^d$ belongs to the r-dimensional* ridge set *$\mathcal{R}^r_f$ if*

$$\nabla f(\boldsymbol{x})^T \boldsymbol{v}_i(\boldsymbol{x}) = 0, \quad \text{for all } i > r, \tag{2.4a}$$

$$\lambda_{r+1}(\boldsymbol{x}) < 0, \tag{2.4b}$$

$$\lambda_1(\boldsymbol{x}) > \lambda_2(\boldsymbol{x}) > \cdots > \lambda_{r+1}(\boldsymbol{x}), \quad \text{if } r > 0, \tag{2.4c}$$

where $\lambda_1(\boldsymbol{x}) \geq \lambda_2(\boldsymbol{x}) \geq \cdots \geq \lambda_d(\boldsymbol{x})$ and $\{\boldsymbol{v}_i(\boldsymbol{x})\}_{i=1}^d$ denote the eigenvalues and the corresponding eigenvectors of $\nabla^2 f(\boldsymbol{x})$, respectively.

A key property is that lower-dimensional ridge sets are contained within higher-dimensional ones. In particular, the zero-dimensional ridge points of a function are its maxima. This property that can be readily observed from Figure 1.1, follows directly from Definition 2.1.1. The methods to be described in Chapters 4 and 5 extensively use this property.

Proposition 2.1.1 *If $f : \mathbb{R}^d \to \mathbb{R}$ is a twice differentiable function, then $\mathcal{R}_f^r \subseteq \mathcal{R}_f^{r+1}$ for all $r = 0, 1, \dots, d-1$.*

Another important property is that the ridge sets of a function and its logarithm coincide. This property will be extensively utilized later, as it allows interchangeable use of f and $\log f$ when dealing with ridge sets. Another reason is that many of the theoretical results derived in this thesis hold for the logarithm of a density function but not for the density itself.

Proposition 2.1.2 ([96]) *If $f : \mathbb{R}^d \to \mathbb{R}$ is twice differentiable, then $\mathcal{R}_f^r = \mathcal{R}_{\log f}^r$ for all $r = 0, 1, 2, \dots, d-1$.*

The algorithms described in the following chapters either project points onto ridges or trace ridges. This is done by tracing curves determined by Hessian eigenvectors. Therefore we need conditions to ensure their continuity and differentiability. First, we state the following result that is a direct consequence of the well-known result about continuity of eigenvalues of a matrix with respect to its elements (e.g. [95], Theorem 3.1.2).

Theorem 2.1.1 *If $f : \mathbb{R}^d \to \mathbb{R}$ is twice continuously differentiable, then there exist continuous functions $\{\lambda_i\}_{i=1}^d : \mathbb{R}^d \to \mathbb{R}$ representing the eigenvalues of $\nabla^2 f$ such that $\lambda_1(\boldsymbol{x}) \geq \lambda_2(\boldsymbol{x}) \geq \cdots \geq \lambda_d(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^d$.*

If we make an additional assumption that condition (2.4c) is satisfied in some open set, then the corresponding eigenvectors are infinitely many times differentiable in this set. The following result is a direct consequence of the well-known results about continuity and differentiability of eigenvectors (e.g. [86] and [95], Theorem 3.1.3).

Theorem 2.1.2 *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a twice continuously differentiable function, let $r > 0$ and let $U \subset \mathbb{R}^d$ be an open set such that*

$$\lambda_1(\boldsymbol{x}) > \lambda_2(\boldsymbol{x}) > \cdots > \lambda_{r+1}(\boldsymbol{x}) \quad \textit{for all } \boldsymbol{x} \in U,$$

where the functions λ_i are defined as in Theorem 2.1.1. Then there exists a set of C^∞-functions $\{\boldsymbol{v}_i\}_{i=1}^r : U \to \mathbb{R}^d$ representing the eigenvectors of $\nabla^2 f$ corresponding to the eigenvalues $\{\lambda_i\}_{i=1}^r$.

2.2 Applications of ridges and related concepts

Research on function ridges and related concepts has been done in many different disciplines such as image processing, theoretical chemistry and global optimization. This makes some earlier results directly applicable in this work. An overview of related research is given below.

2.2.1 Image processing

There exists a rich theory of ridge-like structures in digital image processing. The so-called *height ridges*, as defined by Eberly [43] similarly to Definition 2.1.1, have been widely used for extracting curvilinear and tree-like features from images. Such features are of interest in analysis of aerial, satellite and solar images and in medical imaging. Road and river networks [60], solar flares [68] and blood vessels [114] are examples of these.

The theory of ridges in image processing is closely related to the research of this thesis. Since a digital image is a discrete set of pixels, such an image is approximated by a smooth function in a majority of mathematical papers dealing with image processing.

The usual approach is to convolve a discrete image by using a Gaussian kernel. For an intensity function $I : [1, 2, \ldots, m] \times [1, 2, \ldots, n] \to \mathbb{R}$ representing a $m \times n$ grayscale image, the convolved image can be obtained as

$$\hat{I}(x, y) = \sum_{i=1}^{m} \sum_{j=1}^{n} I(x_i, y_j) K_\sigma(r_{ij}), \quad r_{ij} = \sqrt{(x - x_i)^2 + (y - y_i)^2}.$$

Here the two-dimensional Gaussian kernel with standard deviation σ is defined as

$$K_\sigma(r) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{r^2}{2\sigma^2}\right).$$

The above case, where the data points are aligned in a regular grid, is in fact a special instance of a more general scattered point set that will be considered in the following chapters.

As Definition 2.1.1 only gives pointwise conditions, it does not guarantee any kind of connectivity of ridge sets. The theory developed by Damon [36] and Miller [89] in the context of image processing addresses this issue. Their results give conditions ensuring that an r-dimensional ridge set of a C^∞-function forms a connected smooth *manifold*. These results are directly applicable to the kernel density estimates to be introduced in Section 2.4, as such a density with a Gaussian kernel is a C^∞-function. As it turns out, these results are crucial for the methods described in Chapters 4 and 5.

2.2.2 Theoretical chemistry

The so-called *gradient extremal* curves originally introduced by Hoffman et al. [64] are a standard tool in theoretical chemistry for modeling reaction paths. An important special case of these are valley curves along potential surfaces, as they connect minima representing equilibrium states. For a given function f, a valley curve is a ridge curve of $-f$.

A gradient extremal point is defined as a stationary point of gradient norm along a contour curve (i.e. a curve where $f(\boldsymbol{x}) = c$). In order to show the relation between such a point and a ridge curve point, we consider points at which the gradient norm is minimal. For a three times differentiable function $f : \mathbb{R}^d \to \mathbb{R}$, such a point is a solution to

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \frac{1}{2}\|\nabla f(\boldsymbol{x})\|^2 \quad \text{s.t. } f(\boldsymbol{x}) = c. \tag{2.5}$$

The Lagrangian of the above problem is

$$L(\boldsymbol{x};\lambda) = \frac{1}{2}\|\nabla f(\boldsymbol{x})\|^2 - \lambda[f(\boldsymbol{x}) - c]$$

with Lagrange multiplier λ, yielding the first-order Karush-Kuhn-Tucker (KKT) condition

$$\nabla_{\boldsymbol{x}} L(\boldsymbol{x}^*;\lambda^*) = \nabla^2 f(\boldsymbol{x}^*)\nabla f(\boldsymbol{x}^*) - \lambda^*\nabla f(\boldsymbol{x}^*) = \mathbf{0}. \tag{2.6}$$

This condition is equivalent to saying that the gradient is an eigenvector of the Hessian. Hence, by orthogonality of the eigenvectors of the symmetric matrix $\nabla^2 f(\boldsymbol{x}^*)$, this condition is equivalent to condition (2.4a) when the ridge dimension r is one and the optimal Lagrange multiplier is $\lambda^* = \lambda_1(\boldsymbol{x}^*)$.

To further explore the relation between solutions of problem (2.5) and ridge points, we note that the second derivative of the Lagrangian L with respect to $\boldsymbol{x}$ is

$$\nabla^2_{\boldsymbol{x}} L(\boldsymbol{x};\lambda) = \nabla^3 f(\boldsymbol{x})\nabla f(\boldsymbol{x}) + [\nabla^2 f(\boldsymbol{x})]^2 - \lambda\nabla^2 f(\boldsymbol{x}). \tag{2.7}$$

Plugging the optimal Lagrange multiplier $\lambda = \lambda_1(\boldsymbol{x})$ into the above equation and taking a vector product with any Hessian eigenvector $\boldsymbol{v}_i(\boldsymbol{x})$ for $i > 1$ from both sides yields

$$\begin{aligned}\boldsymbol{v}_i(\boldsymbol{x})^T\nabla^2_{\boldsymbol{x}} L(\boldsymbol{x};\lambda)\boldsymbol{v}_i(\boldsymbol{x}) =& \boldsymbol{v}_i(\boldsymbol{x})^T[\nabla^3 f(\boldsymbol{x})\nabla f(\boldsymbol{x})]\boldsymbol{v}_i(\boldsymbol{x}) + \\ & \lambda_i(\boldsymbol{x})[\lambda_i(\boldsymbol{x}) - \lambda_1(\boldsymbol{x})],\end{aligned} \tag{2.8}$$

where

$$[\nabla^3 f(\boldsymbol{x})\nabla f(\boldsymbol{x})]_{i,k} = \sum_{j=1}^{d}[\nabla^3 f(\boldsymbol{x})]_{i,j,k}[\nabla f(\boldsymbol{x})]_j, \quad i,k = 1,2,\dots,d. \tag{2.9}$$

By conditions (2.4b) and (2.4c), the right hand side of equation (2.8) is strictly positive when $\boldsymbol{x} \in \mathcal{R}_f^1$ and the third derivative term $\nabla^3 f(\boldsymbol{x}) \nabla f(\boldsymbol{x})$ is sufficiently small. In this case, since $\boldsymbol{v}_i(\boldsymbol{x})^T \boldsymbol{v}_1(\boldsymbol{x}) = \boldsymbol{v}_i(\boldsymbol{x})^T \nabla f(\boldsymbol{x}) = 0$ for all $i > 1$ and $\nabla f(\boldsymbol{x})$ is the gradient of the constraint in (2.5), we deduce that any ridge curve point $\boldsymbol{x} \in \mathcal{R}_f^1$ also satisfies the second-order KKT conditions of problem (2.5) (see e.g. [7] for the definition of such conditions).

The above observations imply that a ridge curve point is a special case of a gradient extremal point. Thus, some of the results presented in Chapter 4 dealing with ridge curves rely on existing results for gradient extremals.

2.2.3 Global optimization

Global optimization is another important application of ridge and valley curves since such curves pass through local maxima and minima, respectively. The so-called *terrain method* developed by Lucia et al. [84] builds a network of extremum points of a function by following such curves. In that paper and a series of related papers (e.g. [85]), the authors apply the terrain method to molecular modeling. A related method has been developed by Sminchisescu and Triggs [113] for global optimization problems arising in computer vision.

2.3 Estimation of underlying structures from density ridges

As noted in the previous section, ridge-based methods have been used in various application areas. However, estimation of underlying low-dimensional structure in scattered point set from density ridges appears to be a new research area. This idea was first proposed by Ozertem and Erdogmus [96]. However, they do not provide any statistical model for such structure. Consequently, the quality of the estimates obtained from density ridges is not rigorously analyzed. To the knowledge of the author, such analysis has not been carried out until very recently by Genovese et al. [49]. Therefore we recall in this section the necessary results to justify the use of density ridges as estimators.

One common approach to measure the quality of an estimator for some underlying structure in a point set is to assume a *generative model* (e.g. [18], [49], [61] and [120]). Such a model typically describes a random process where the points are sampled from one or more a priori known *generating functions* with additive noise. When low-dimensional structure is modeled, the generating functions are mappings from some subset of a low-dimensional space to a higher-dimensional space (e.g. a curve segment in $\mathbb{R}^d$ with $d > 1$). Figure 1.1b shows an example of a point set sampled from such a model.

Assuming a generative model, it can be shown in certain special cases that the density ridges coincide with the generating function. One such case is when the points are sampled from an m-dimensional hyperrectangle with normally distributed isotropic noise (i.e. with covariance matrix $\sigma^2\boldsymbol{I}$ for some $\sigma > 0$).

More specifically, let us assume that the points are sampled from an m-dimensional hyperrectangle

$$\mathcal{D} = \{\boldsymbol{\theta} \in \mathbb{R}^m \mid a_i \leq \theta_i \leq b_i, i = 1, 2, \dots, m\} \tag{2.10}$$

with $a_i < b_i$, $i = 1, 2, \dots, m$ and then embedded into the d-dimensional space $\mathbb{R}^d$ via the generating function

$$\boldsymbol{f}(\boldsymbol{\theta}) = \boldsymbol{x}_0 + \sum_{i=1}^{m} \theta_i \boldsymbol{u}_i \tag{2.11}$$

with some $\boldsymbol{x}_0 \in \mathbb{R}^d$ and mutually orthogonal vectors $\{\boldsymbol{u}_i\}_{i=1}^m \subset \mathbb{R}^d \setminus \{\boldsymbol{0}\}$.

The coordinates on $\mathcal{D}$ are modeled by an uniformly distributed random variable $\boldsymbol{\Theta}$. Given a sample from $\boldsymbol{\Theta}$, denoted as $\boldsymbol{\theta}$, the above noise assumption yields the conditional density

$$p_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta}) = \frac{1}{(\sqrt{2\pi}\sigma)^d} \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{\theta})\|^2}{2\sigma^2}\right)$$

for the observed variable denoted by $\boldsymbol{X}$.

Integration of the joint density $p_{\boldsymbol{X},\boldsymbol{\Theta}}(\boldsymbol{x};\boldsymbol{\theta}) = p_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\Theta} = \boldsymbol{\theta})p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ then yields the *marginal density*

$$p_{\boldsymbol{X}}(\boldsymbol{x}) = \frac{1}{(\sqrt{2\pi}\sigma)^d V(\mathcal{D})} \int\limits_{\mathcal{D}} \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{\theta})\|^2}{2\sigma^2}\right) d\boldsymbol{\theta}, \tag{2.12}$$

where

$$V(\mathcal{D}) = \prod_{i=1}^{m} (b_i - a_i)$$

denotes the volume of the hyperrectangle $\mathcal{D}$.

The following result is proven in **Paper I** for the above model.

Theorem 2.3.1 *Let* $0 < m < d$, *let* $\mathcal{D}$ *be defined by* (2.10) *and let* $\boldsymbol{f}$ *be defined by* (2.11). *If* $p_{\boldsymbol{X}}$ *is defined by* (2.12), *then* $\{\boldsymbol{f}(\boldsymbol{\theta}) \mid \boldsymbol{\theta} \in \mathcal{D}\} \subseteq \mathcal{R}^m_{p_{\boldsymbol{X}}}$.

In addition, we have $\mathcal{R}^m_{p_{\boldsymbol{X}}} = \{\boldsymbol{x}_0\} + \operatorname{span}(\boldsymbol{u}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_m)$. Unfortunately the result of the above theorem does not generally hold when the points are sampled from a general nonlinear hypersurface. That is, the estimates for the underlying structure obtained from density ridges are *biased*.

Error bounds for ridge-based estimates are derived in [49] in terms of the noise deviation σ. Instead of a hyperrectangle, the points are assumed to be sampled from a more general smooth manifold M. The sampling from M is done under some probability distribution W defined on M and having density w. The noise, that is added to the samples, is assumed to be normally distributed with isotropic covariance $\sigma^2\boldsymbol{I}$ as above. The error bounds guarantee that as σ approaches zero, the density ridges converge to the manifold M. In addition, it is shown that when two densities and their derivatives are close to each other, this is also the case for their ridge sets.

In analogy with (2.12), for the more general manifold model we obtain the marginal density

$$p_\sigma(\boldsymbol{x}) = \int\limits_M \phi_\sigma(\boldsymbol{x}-\boldsymbol{z})w(\boldsymbol{z})d\boldsymbol{z} \tag{2.13}$$

for the observed variable $\boldsymbol{X}$, where

$$\phi_\sigma(\boldsymbol{u}) = \frac{1}{(\sqrt{2\pi}\sigma)^d}\exp\left(-\frac{\|\boldsymbol{u}\|^2}{2\sigma^2}\right).$$

Here we have omitted the subscript $\boldsymbol{X}$. The subscript σ is added to denote the dependence on σ, which is needed for stating the error bounds. For the remainder of this section, we assume that the dimension d of the input space and the dimension $0 \leq m < d$ of the manifold M are fixed.

The concept of an ε-*dilation* of a set is needed for stating the results of this section.

Definition 2.3.1 ([49]) *For $\varepsilon > 0$, the ε-dilation of a set $A \subset \mathbb{R}^d$, denoted by $A \oplus \varepsilon$, is*

$$A \oplus \varepsilon = \left\{ \boldsymbol{x} \in \mathbb{R}^d \;\middle|\; \inf_{\boldsymbol{y}\in A} \|\boldsymbol{x}-\boldsymbol{y}\| \leq \varepsilon \right\}.$$

The following assumption is made in [49] for the manifold M. Due to space constraints we omit the formal definition of a reach. Essentially this assumption states that the manifold is a closed smooth surface with no self-intersections (e.g. a sphere or a torus).

Assumption 2.3.1 *M is a compact manifold such that* $\operatorname{reach}(M) > 0$ *and has no boundary.*

Assumptions on the structure of the manifold M alone are not sufficient because the data points are assumed to be sampled from M under some probability distribution w. Therefore the following assumption is needed.

Assumption 2.3.2 *The density w is twice differentiable and $0 < w(\boldsymbol{x}) < \infty$ for all $\boldsymbol{x} \in M$.*

The results established in [49] include the following condition. This condition is a strengthening of the pointwise conditions (2.4b) and (2.4c) to hold uniformly in a dilation of a given set.

Condition 2.3.1 *For a given function $p : \mathbb{R}^d \to \mathbb{R}$ and a set $U \subset \mathbb{R}^d$, there exist $\beta > 0$ and $\delta > 0$ such that*

$$\lambda_{m+1}(\boldsymbol{x}) < -\beta \quad \textit{and} \quad \lambda_m(\boldsymbol{x}) - \lambda_{m+1}(\boldsymbol{x}) > \beta$$

for all $\boldsymbol{x} \in U \oplus \delta$, where $\lambda_1(\boldsymbol{x}) \geq \lambda_2(\boldsymbol{x}) \geq \cdots \geq \lambda_d(\boldsymbol{x})$ denote the eigenvalues of $\nabla^2 p(\boldsymbol{x})$.

In the statement of the following results, the distance between two sets A and B is given by the *Hausdorff distance* defined as

$$\operatorname{Haus}(A, B) = \max\{\sup_{x \in A} \inf_{y \in B} \|\boldsymbol{x} - \boldsymbol{y}\|, \sup_{y \in B} \inf_{x \in A} \|\boldsymbol{x} - \boldsymbol{y}\|\}. \tag{2.14}$$

Finally, we can formulate the main results of [49]. The first one states that under appropriate assumptions the density (2.13) has a ridge and the ridge converges to the manifold M as σ tends to zero.

Theorem 2.3.2 ([49]) *Suppose that Assumptions 2.3.1 and 2.3.2 are satisfied. Let $M_\sigma = M \oplus r_\sigma$ with $r_\sigma = \alpha\sigma$ for any $0 < \alpha < 1$ and let $\mathcal{R}^*_\sigma = \mathcal{R}^m_{p_\sigma} \bigcap M_\sigma$. Let $A \geq 2$ and define*

$$K_\sigma = \sqrt{2\sigma^2 \log\left(\frac{1}{\sigma^{A+d}}\right)}.$$

Then for all sufficiently small $\sigma > 0$ we have that

(i) Condition 2.3.1 holds for p_σ and $\mathcal{R}^*_\sigma$ with $\beta = c\sigma^{-(d-m+2)}$ for some $c > 0$.

(ii) $\operatorname{Haus}(M, \mathcal{R}^*_\sigma) = \mathcal{O}(K^2_\sigma)$ as $\sigma \to 0$.[1]

If p_σ is replaced by $\log p_\sigma$, the above conditions hold with $\beta = c\sigma^{-2}$ and $M_\sigma = M \oplus \kappa$, where $\kappa = \operatorname{reach}(M)$.

Theorem 2.3.2 is applicable when the marginal density p is known and can be computed exactly. However, in practice this is not the case and the density needs to be estimated. The following result applies when the density estimate and its derivatives are sufficiently close to those of the marginal density p.

[1] For two functions f and g and a constant a, $f(x) = \mathcal{O}(g(x))$ as $x \to a$ if and only if there exists $C > 0$ and $\delta > 0$ such that $|f(x)| \leq C|g(x)|$ for all x such that $|x - a| < \delta$.

Theorem 2.3.3 ([49]) *Suppose that functions p and $\hat{p}$ are three times differentiable and all the derivatives are bounded. Assume that Condition 2.3.1 holds for p and $\mathcal{R}^m_p$. Let $\boldsymbol{g}$, $\boldsymbol{H}$, $\boldsymbol{H}'$, $\hat{\boldsymbol{g}}$, $\hat{\boldsymbol{H}}$ and $\hat{\boldsymbol{H}}'$ denote the gradient, Hessian and the third derivatives of p and $\hat{p}$, respectively. Define*

$$\varepsilon = \|p - \hat{p}\|_\infty, \qquad \varepsilon' = \max_j \|g_j - \hat{g}_j\|_\infty,$$
$$\varepsilon'' = \max_{jk} \|H_{jk} - \hat{H}_{jk}\|_\infty, \qquad \varepsilon''' = \max_{ijk} \|H'_{ijk} - \hat{H}'_{ijk}\|_\infty,$$

where

$$\|f\|_\infty = \sup_{\boldsymbol{x} \in \mathcal{R}^m_p \oplus \delta} |f(\boldsymbol{x})|$$

for a given function f. Let $\psi = \max\{\varepsilon, \varepsilon', \varepsilon''\}$ and let $\Psi = \max\{\varepsilon, \varepsilon', \varepsilon'', \varepsilon'''\}$. Then there exists $C > 0$ such that for any sufficiently small Ψ

(i) *Condition 2.3.1 holds for $\hat{p}$ and $\mathcal{R}^m_p$.*

(ii) $\mathrm{Haus}(\mathcal{R}^m_p, \mathcal{R}^m_{\hat{p}}) \leq \frac{2C\psi}{\beta}$.

Assuming the generative model described above, Theorems 2.3.2 and 2.3.3 give a theoretical justification for the ridge projection and tracing algorithms described in the following chapters. For sufficiently small σ, the marginal density p_σ satisfies Condition 2.3.1 in $\mathcal{R}^*_\sigma$ by condition (i) of Theorem 2.3.2. When this is the case and a density estimate $\hat{p}$ and its derivatives are sufficiently close to those of p_σ, Condition 2.3.1 also holds for $\hat{p}$ and $\mathcal{R}^*_\sigma$ by Theorem 2.3.3. Consequently, the ridge sets of $\hat{p}$ are well-defined when σ is sufficiently small. In addition, condition (ii) of Theorem 2.3.3, and consequently condition (ii) of Theorem 2.3.2 imply that for sufficiently small σ, it is reasonable expect that ridges of $\hat{p}$ give good estimates of the underlying manifold M.

Due to restrictive assumptions, the above results are mostly of theoretical interest. The error bounds do not generally hold when the manifold M intersects itself or is not closed or when the model has multiple manifolds. Though not formally verified, it is conjectured in [49] that relaxing the closedness assumption yields an error bound of order $\mathcal{O}(K_\sigma)$ in condition (ii) of Theorem 2.3.2.

2.4 Kernel density estimation

The density estimation methods considered in this thesis are based on *Gaussian kernels* [110, 128]. The idea of kernel density estimation is to assign each sample point a kernel function. A Gaussian kernel density estimate is essentially a sum of such kernels, as stated in the following definition.

Definition 2.4.1 ([128]) *The* Gaussian kernel density estimate $\hat{p}_{\boldsymbol{H}}$ *obtained by drawing a set of samples* $\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1}^N \subset \mathbb{R}^d$ *from a probability density* $p : \mathbb{R}^d \to \mathbb{R}$ *is*

$$\hat{p}_{\boldsymbol{H}}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{y}_i), \tag{2.15}$$

where the kernel $K_{\boldsymbol{H}} : \mathbb{R}^d \to]0, \infty[$ *is the* Gaussian *function*

$$K_{\boldsymbol{H}}(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^d |\boldsymbol{H}|}} \exp\left(-\frac{1}{2}\boldsymbol{x}^T \boldsymbol{H}^{-1}\boldsymbol{x}\right) \tag{2.16}$$

with a symmetric and positive definite kernel bandwidth *matrix* $\boldsymbol{H} \in \mathbb{R}^{d\times d}$.

Gaussian kernels have certain advantages over other possible choices. First, a Gaussian kernel density is infinitely many times differentiable. As we shall see in the following, the ridges of such a function are well-defined. Second, rigorous error bounds have been derived for such kernel estimates. Such error bounds typically give the *asymptotic* rate of convergence for an appropriately chosen sequence of bandwidth matrices $\boldsymbol{H}$. That is, for a desired order k, the k-th derivatives of the estimator $\hat{p}_{\boldsymbol{H}}$ converge to the k-th derivatives of the true density p as the number of samples N approaches infinity.

The error bounds for kernel density estimators in the literature are typically derived with respect to a squared error measure. With the notation introduced in [30], such a quantity is written as

$$\mathrm{SE}(\boldsymbol{x}; \boldsymbol{H}) = \|\widehat{D^{\otimes k}p}(\boldsymbol{x}; \boldsymbol{H}) - D^{\otimes k}p(\boldsymbol{x})\|^2, \tag{2.17}$$

where $D^{\otimes k}p$ denotes a vector containing all partial derivatives of p of order k and $\widehat{D^{\otimes k}p}$ denotes a kernel estimator for the k-th derivatives of p.

Integrating the pointwise error (2.17) and taking the expectation $\mathbb{E}$ over the samples $\boldsymbol{Y}$ yields the *mean integrated squared error*

$$\mathrm{MISE}(\boldsymbol{H}) = \mathbb{E}\left[\int_{\mathbb{R}^d} \mathrm{SE}(\boldsymbol{x}; \boldsymbol{H}) d\boldsymbol{x}\right]. \tag{2.18}$$

Consequently, the problem of density estimation is transformed into a problem of determining the optimal bandwidth matrix $\boldsymbol{H}$ with respect to MISE.

The MISE error measure is computationally intractable except in certain special cases (e.g. when p is a normal density or a mixture of normal densities [87]). Thus, MISE is usually approximated by a computable formula that converges to this quantity as the sample size N approaches infinity. It can be shown that for any sequence of bandwidth matrices $\boldsymbol{H}$ such that $\boldsymbol{H} \to \boldsymbol{0}$ elementwise and $N^{-1}|\boldsymbol{H}|(\boldsymbol{H}^{-1})^{\otimes k} \to \boldsymbol{0}$ as $N \to \infty$, the expansion

$$\mathrm{MISE}(\boldsymbol{H}) = \mathrm{AMISE}(\boldsymbol{H}) + o(N^{-1}|\boldsymbol{H}|^{-\frac{1}{2}}[\mathrm{tr}(\boldsymbol{H}^{-1})]^k + [\mathrm{tr}(\boldsymbol{H})]^2) \qquad (2.19)$$

is valid as $N \to \infty$. The formula for the asymptotic MISE (AMISE) appearing in the above equation is derived in [30].[2][3]

Asymptotic error bounds for density derivative estimators of a given order k are derived in [30] under the following assumptions.

Assumption 2.4.1 *The density p and the kernel function $K_{\boldsymbol{H}}$ satisfy the following conditions.*

(i) The density p has all partial derivatives up to order $k+2$, all its partial derivatives of order k are square integrable, and all its partial derivatives of order $k+2$ are bounded, continuous and square integrable.

(ii) $K_{\boldsymbol{H}}$ is a positive, symmetric, square integrable density function such that

$$\int_{\mathbb{R}^d} \boldsymbol{x}\boldsymbol{x}^T K_{\boldsymbol{H}}(\boldsymbol{x}) d\boldsymbol{x} = c\boldsymbol{I}$$

for some constant c, and all its partial derivatives of order k are square integrable.

Theorem 2.4.1 ([30]) *Under Assumption 2.4.1, every element of the optimal (symmetric and positive definite) bandwidth matrix*

$$\boldsymbol{H}_{\mathrm{AMISE}} = \arg\min_{\boldsymbol{H}} \mathrm{AMISE}(\boldsymbol{H})$$

is of order $\mathcal{O}(N^{-2/(d+2k+4)})$. Furthermore, the minimal AMISE($\boldsymbol{H}$) *is of order $\mathcal{O}(N^{-4/(d+2k+4)})$.*

Plugging the above estimates to the formula (2.19) shows that any AMISE-optimal bandwidth matrix $\boldsymbol{H}$ is also asymptotically MISE-optimal. That is, for any such bandwidth, MISE approaches zero as the sample size N approaches infinity.

It can be shown that the density p satisfies condition (i) of Assumption 2.4.1 for any k when it is a marginal density of the form (2.13) and Assumptions 2.3.1 and 2.3.2 are satisfied. This is because p is a Gaussian convolution of a bounded and compactly supported function. In addition, it can be shown by using the standard formulae for Gaussian integrals (e.g. [2]) that the Gaussian kernel (2.16) satisfies condition (ii). Under these assumptions, Theorem 2.4.1 thus justifies the use of an AMISE-optimal Gaussian

[2] The symbol $\otimes k$ denotes the k-fold *Kronecker product* of a matrix with itself.

[3] For two functions f and g and a constant a, $f(x) = o(g(x))$ as $x \to a$ if and only if for all $\varepsilon > 0$ there exists $\delta > 0$ such that $|f(x)| \leq \varepsilon |g(x)|$ for all x such that $|x - a| < \delta$.

kernel density estimator assuming that the data is sampled from the model described in Section 2.3.

Various different bandwidth estimators have been implemented based on the MISE and AMISE criteria. Duong and Hazelton [42] propose determining the bandwidth $\boldsymbol{H}$ by smoothed cross-validation and give asymptotic error bounds. Chacón and Duong [29] propose a plug-in bandwidth selector. Chacón et al. [30] extend the earlier methods to density derivatives. Bandwidth selectors based on the above references and more recent ones have been implemented in the `ks` package for the `R` software by Duong [41].

2.5 Practical error estimates and examples

The results of Sections 2.3 and 2.4 give asymptotic error bounds for the ridge and kernel density estimates as the noise standard deviation σ approaches zero and the sample size N approaches infinity. Unfortunately, these results provide no insight on how good the estimates are in practice with nonzero σ and finite N. Therefore this section is devoted to analysis of the estimation errors with realistic test cases. The results presented here are based on **Paper III** dealing with one-dimensional ridges (i.e. ridge curves).

2.5.1 Model bias

A practical analysis of the bias of ridge estimates is given in [III]. That is, the bias when a ridge estimator is applied directly to the (a priori known) marginal density without kernel estimation. In [III], the author considers a special case that also reflects the general behaviour of the bias. In this example, the model consists of a single generating function $\boldsymbol{f}(\theta) = (\cos\theta, \sin\theta)$ parametrizing the unit circle on a plane.

In analogy with (2.12), the marginal density induced by $\boldsymbol{f}$ at a point $\boldsymbol{x} = (x_1, x_2)$ is

$$p_\sigma(x_1, x_2) = \frac{1}{4\pi^2\sigma^2} \int_0^{2\pi} G_\sigma(x_1, x_2; \theta) d\theta,$$

where

$$G_\sigma(x_1, x_2; \theta) = \exp\left(-\frac{(x_1 - \cos\theta)^2 + (x_2 - \sin\theta)^2}{2\sigma^2}\right).$$

The components of the gradient of p_σ at a given point $\boldsymbol{x} = (x_1, x_2)$ with $x_2 = 0$ are given by

$$\frac{\partial p_\sigma}{\partial x_1}(x_1, 0) = -\frac{1}{4\pi^2\sigma^4} \int_0^{2\pi} (x_1 - \cos\theta) G_\sigma(x_1, 0; \theta) d\theta, \quad \text{and} \quad \frac{\partial p_\sigma}{\partial x_2}(x_1, 0) = 0.$$

For the Hessian, we have

$$\frac{\partial^2 p_\sigma}{\partial x_1^2}(x_1,0) = \frac{1}{4\pi^2\sigma^4}\int_0^{2\pi}\left[\frac{(x_1-\cos\theta)^2}{\sigma^2}-1\right]G_\sigma(x_1,0;\theta)d\theta, \tag{2.20}$$

$$\frac{\partial^2 p_\sigma}{\partial x_2^2}(x_1,0) = \frac{1}{4\pi^2\sigma^4}\int_0^{2\pi}\left(\frac{\sin^2\theta}{\sigma^2}-1\right)G_\sigma(x_1,0;\theta)d\theta$$

and

$$\frac{\partial^2 p_\sigma}{\partial x_1\partial x_2}(x_1,0) = \frac{\partial^2 p_\sigma}{\partial x_2\partial x_1}(x_1,0) = 0. \tag{2.21}$$

It can be shown by numerical integration that the Hessian element (2.20) has exactly one root in the interval $[0,1]$ when $\sigma \in]0, \frac{\sqrt{2}}{2}[$. Furthermore,

$$\frac{\partial^2 p_\sigma}{\partial x_1^2}(x_1,0) < 0 \quad \text{and} \quad \frac{\partial^2 p_\sigma}{\partial x_1^2}(x_1,0) < \frac{\partial^2 p_\sigma}{\partial x_2^2}(x_1,0) \quad \text{for all } x_1 \in [x_1^{**},1],$$

where x_1^{**} denotes the root of (2.20). In view of equation (2.21), this implies that the normalized eigenvectors of the Hessian $\nabla^2 p_\sigma(x_1,0)$ are $\boldsymbol{v}_1(x_1,0)=(0,1)$ and $\boldsymbol{v}_2(x_1,0)=(1,0)$ for all $x_1 \in]x_1^{**},1]$. They correspond to the eigenvalues $\lambda_1(x_1,0) = \frac{\partial^2 p_\sigma}{\partial x_2^2}(x_1,0)$ and $\lambda_2(x_1,0) = \frac{\partial^2 p_\sigma}{\partial x_1^2}(x_1,0) < 0$, respectively.

By the above observations, the x-coordinate of any ridge point $\boldsymbol{x}^* = (x_1^*,0)$ of p_σ is a zero point of the derivative $\frac{\partial p_\sigma}{\partial x_1}(x_1,0)$ in the interval $]x_1^{**},1]$. Such a point can be computed by numerical integration and root-finding. The distance of such a ridge point $\boldsymbol{x}^*$ to the actual generating curve (i.e. the model bias) relative to σ as a function of σ is plotted in Figure 2.1. The bias occurs towards the curvature center and is inversely proportional to the ratio between the curvature radius (which is one) and σ.

Furthermore, the ridge curve of the marginal density p_σ gives an accurate estimate of the generating function. In the interval $[0,0.35]$, the relative distance between the point $\boldsymbol{x}^*$ and the generating curve grows linearly. With $\sigma = 0.35$ that corresponds to a large amount of noise, the distance is only 0.2. This is also illustrated in Figure 2.2 showing both the generating curves and density ridge curves. Though the results shown here represent theoretically attainable accuracy with a "perfect" density estimator, they nevertheless suggest that the model bias is expected to be small in most cases.

2.5.2 Combined model and density estimation bias

Finally, we demonstrate by examples that ridge curves can give good estimates of the generating functions even when computed from a kernel density

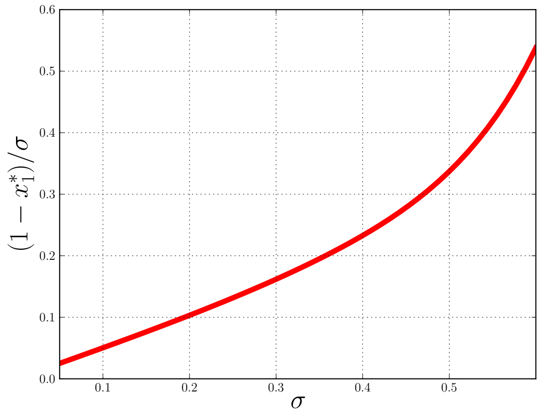


Figure 2.1: Distance between the generating curve $\boldsymbol{f}(\theta)$ and the x-coordinate x_1^* of a ridge point of p_σ relative to noise standard deviation σ as a function of σ.

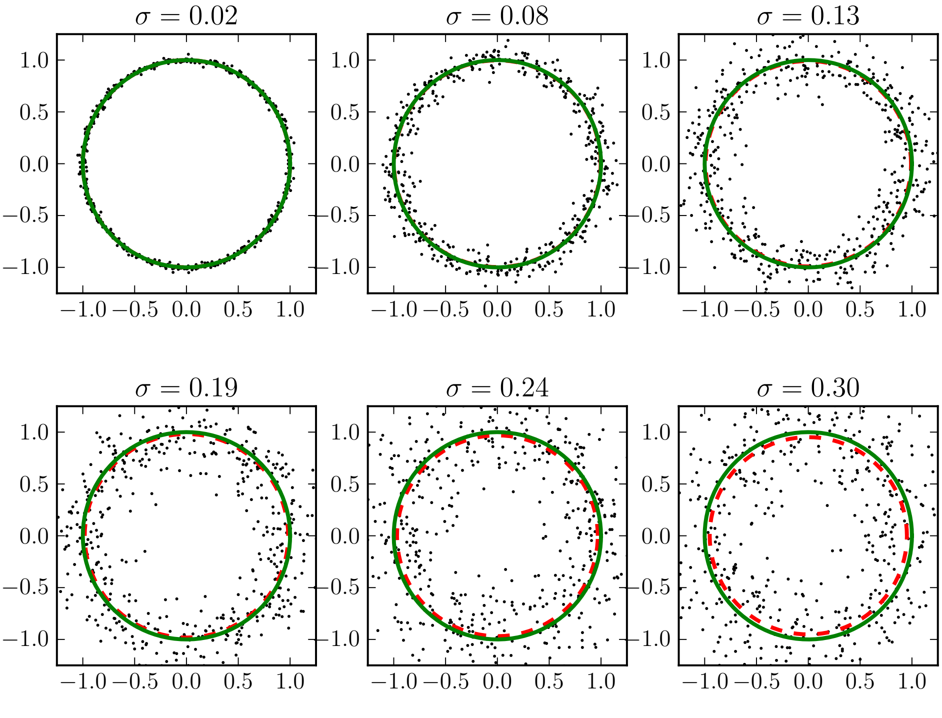
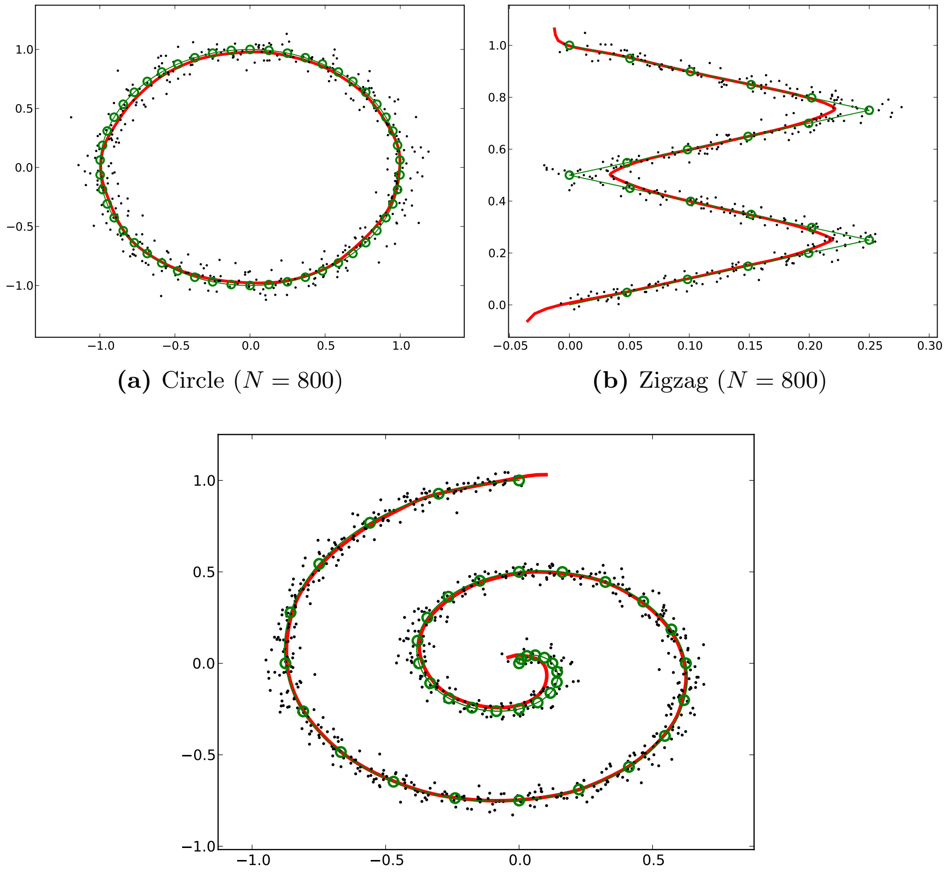


Figure 2.2: Circular data distributions with different values of σ, generating functions (green lines) and ridge curves of p_σ (red dashed lines).

estimate. Some datasets, their generating functions and ridge curves of kernel density estimates are shown in Figure 2.3. The bandwidth matrices $\boldsymbol{H}$ are chosen by using the `Hpi` estimator implemented in the `ks` package [41]. The bandwidths are chosen to be optimal for gradient estimation. Though this choice does not yield the best possible estimate for the Hessian required for ridge extraction, it nevertheless gives optimal estimates for modes because they are stationary points of the estimated density.

The ridge curves shown in Figure 2.3 contain both model bias and error due to kernel estimation. Nevertheless, they give good estimates of the generating functions. However, when the generating curve has sharp turns, of which Figure 2.3b shows an extreme example, the deviation between the ridge curve and the generating curve can be large. These observations agree with the results presented for the model bias in Subsection 2.5.1.



(a) Circle ($N = 800$)

(b) Zigzag ($N = 800$)

(c) Spiral ($N = 1400$)

Figure 2.3: Generating functions (green curves and circles) and kernel density ridge curves (red curves) of some datasets obtained by using the algorithm described in Chapter 4.

Chapter 3

Algorithms for finding density ridges and global maxima

This chapter deals with optimization methods for finding ridges and modes of density functions, and it is divided into two parts. The first part (Sections 3.1–3.4) consists of a literature review on the earlier mean shift-based methods and a summary of **Paper I**. The contribution of [I] is development of a trust region Newton method for projecting a given point onto an r-dimensional ridge set (a set of maxima when $r = 0$). Assuming a generative model and applying the method to Gaussian kernel densities, applicability of the method to extraction of underlying structures from noisy point sets is demonstrated. Numerical comparison with mean shift-based methods shows that the Newton method has superior performance.

The second part of this chapter (Section 3.5) is based on **Paper II**. The contribution of this paper is development of a Newton-based method for finding significant modes of Gaussian mixtures and kernel densities at a low computational cost. This problem arises, for instance, in real-time visual tracking. It is shown in [II] that by applying a Gaussian convolution, such a highly multimodal density can be smoothly deformed into a unimodal one. Applying this idea reversely, a homotopy continuation method is proposed. The method starts from the mode of the unimodal density and traces the mode while deforming the density into the original one. This process is formulated as a solution to a differential equation. It is demonstrated by numerical experiments that this approach is highly efficient and finds global modes with a high probability.

3.1 Mean shift-based methods

The mean shift method was first introduced by Fukunaga and Hostetler [47] and later refined by Cheng [31] and Comaniciu and Meer [34]. By now, this method has become a popular approach to finding modes of densities that can be expressed in the form (2.15). It has been utilized in a wide variety of applications. Examples include clustering [31], image smoothing and segmentation [34] and object tracking [35].

The mean shift method is a first-order method based on a fixed-point iteration. The iteration formula is obtained by equating the gradient of the kernel density (2.15) given by

$$\nabla \hat{p}_{\boldsymbol{H}}(\boldsymbol{x}) = \frac{1}{N}\sum_{i=1}^{N} K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{y}_i)\boldsymbol{H}^{-1}(\boldsymbol{x}-\boldsymbol{y}_i)$$

to zero and solving for $\boldsymbol{x}$. This yields a fixed-point iteration

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k, \quad \text{where } \boldsymbol{s}_k = \boldsymbol{f}_{\boldsymbol{H}}(\boldsymbol{x}_k) - \boldsymbol{x}_k \tag{3.1}$$

and

$$\boldsymbol{f}_{\boldsymbol{H}}(\boldsymbol{x}) = \frac{\displaystyle\sum_{i=1}^{N} K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{y}_i)\boldsymbol{y}_i}{\displaystyle\sum_{i=1}^{N} K_{\boldsymbol{H}}(\boldsymbol{x}-\boldsymbol{y}_i)}. \tag{3.2}$$

With an appropriately chosen kernel function $K_{\boldsymbol{H}}$ and under certain assumptions, this simple iterative process converges to a stationary point of the kernel density $\hat{p}_{\boldsymbol{H}}$.

Li et al. [80] prove convergence of the above mean shift iteration under mild assumptions on the kernel function by assuming that the number of stationary points of the density is finite. Carreira-Perpiñán [27] shows that the mean shift method is a variant of an expectation-maximization (EM) method and claims its convergence based on this argument but without a rigorous proof. A similar argument is given by Fashing and Tomasi [45], but again without proof. Refining the earlier results, Ghassabeh et al. [50, 51] give a rigorous convergence analysis. Unfortunately, none of the earlier research addresses the limitation that as a first-order method, the mean shift method can only be proven to converge to a first-order stationary point.

Carreira-Perpiñán [27] shows that the convergence rate of the mean shift method is generally Q-linear (see e.g. [94] for definitions of convergence rates) for isotropic Gaussian kernel densities (i.e. the case $\boldsymbol{H} = h^2\boldsymbol{I}$). When the iteration converges to a mode $\boldsymbol{x}^*$, the convergence rate r is shown to be

$$r = \frac{h^2}{\hat{p}_{h^2\boldsymbol{I}}(\boldsymbol{x}^*)}\lambda_1(\boldsymbol{x}^*) + 1,$$

where $\lambda_1(\boldsymbol{x}^*)$ denotes the greatest eigenvalue of the Hessian $\nabla^2\hat{p}_{h^2\boldsymbol{I}}(\boldsymbol{x}^*)$.

Superlinear convergence rate is obtained in the special cases when $h \to 0$ or $h \to \infty$. Unfortunately, these cases are irrelevant for practical applications since very narrow or wide kernels usually give poor density estimates. Considering our application, the above result reveals an unsettling fact. That is, the mean shift method is expected to have very slow convergence when finding modes lying on a ridge. At a ridge point, the greatest Hessian eigenvalue is usually near zero (cf. Figure 1.1b).

3.2 The subspace constraint

Ozertem and Erdogmus [96] propose the *subspace-constrained* mean shift (SCMS) method. Generalizing the original mean shift method for finding modes, this method (approximately) projects a given point onto an r-dimensional ridge set $\mathcal{R}^r_{\hat{p}_{\boldsymbol{H}}}$ of a Gaussian kernel density $\hat{p}_{\boldsymbol{H}}$.

The idea of the SCMS method is to constrain the mean shift step (3.1) according to

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{P}_r(\boldsymbol{x})\boldsymbol{s}_k, \tag{3.3}$$

where the matrix

$$\boldsymbol{P}_r(\boldsymbol{x}) = \boldsymbol{I} - \sum_{i=1}^{r} \boldsymbol{v}_i(\boldsymbol{x})\boldsymbol{v}_i(\boldsymbol{x})^T$$

projects the step onto the subspace spanned by the Hessian eigenvectors $\{\boldsymbol{v}_i(\boldsymbol{x})\}_{i=r+1}^{d}$ corresponding to the $d-r$ algebraically smallest eigenvalues. Here the eigenvectors are those of the log-Hessian

$$\nabla^2 \log \hat{p}_{\boldsymbol{H}}(\boldsymbol{x}) = \frac{\nabla^2\hat{p}_{\boldsymbol{H}}(\boldsymbol{x})}{\hat{p}_{\boldsymbol{H}}(\boldsymbol{x})} - \frac{\nabla\hat{p}_{\boldsymbol{H}}(\boldsymbol{x})\nabla\hat{p}_{\boldsymbol{H}}(\boldsymbol{x})^T}{\hat{p}_{\boldsymbol{H}}(\boldsymbol{x})^2}. \tag{3.4}$$

A convergence proof for the SCMS method, which appears to be the first one in the literature, is given in [51].

The subspace constraint stems from Definition 2.1.1, which states that an r-dimensional ridge point is a local maximum in the subspace spanned by the $d-r$ last Hessian eigenvectors. Taking the logarithm is justified by the special case where the objective function is a normal density [96]. In this case, the SCMS iteration converges to a point that is an orthogonal projection of the starting point $\boldsymbol{x}_0$ onto the ridge set $\mathcal{R}^r_p$. The set $\mathcal{R}^r_p = \mathcal{R}^r_{\log p}$ is a hyperplane spanned by the first r Hessian eigenvectors. These properties follow from the fact that the logarithm of a normal density p with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ is a quadratic function with

$$\nabla \log p(\boldsymbol{x}) = -\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \quad \text{and} \quad \nabla^2 \log p(\boldsymbol{x}) = -\boldsymbol{\Sigma}^{-1}. \tag{3.5}$$

More generally, any iteration of the form (3.3) can be viewed as an approximate method for tracing a solution curve of a differential equation. Namely, for a starting point $\boldsymbol{x}_0$ we can define a curve $\boldsymbol{\gamma}_r : [0,\infty[\to \mathbb{R}^d$ satisfying

$$\frac{d}{dt}\left\{\left[\sum_{i=1}^{r} \boldsymbol{v}_i(\boldsymbol{\gamma}_r(t))\boldsymbol{v}_i(\boldsymbol{\gamma}_r(t))^T\right]\nabla \log \hat{p}_{\boldsymbol{H}}(\boldsymbol{\gamma}_r(t))\right\} = \boldsymbol{0}, \qquad t \geq 0, \tag{3.6a}$$

$$\boldsymbol{\gamma}_r(0) = \boldsymbol{x}_0. \tag{3.6b}$$

By Theorems 2.1.1 and 2.1.2, the Hessian eigenvectors are differentiable in some neighbourhood of $\boldsymbol{x}_0$ whenever $\lambda_1(\boldsymbol{x}_0) > \lambda_2(\boldsymbol{x}_0) > \cdots > \lambda_{r+1}(\boldsymbol{x}_0)$.

Let us denote $\boldsymbol{\gamma}_r(t_k) = \boldsymbol{x}_k$ and $\boldsymbol{\gamma}_r'(t_k) = \boldsymbol{s}_k$ for some nonnegative and monotonous sequence $\{t_k\}$. When the density $\hat{p}_{\boldsymbol{H}}$ is replaced by its quadratic approximation at each iterate $\boldsymbol{x}_k$, implying that the eigenvectors $\boldsymbol{v}_i(\cdot)$ are constant, we observe that the steps $\boldsymbol{s}_k$ obtained from (3.3) satisfy condition (3.6a). This can be observed by taking the derivative with respect to t and using the chain rule and eigendecomposition of the Hessian $\nabla^2 \log \hat{p}_{\boldsymbol{H}}$. Another observation is that the solution curve $\boldsymbol{\gamma}_r$ gives an orthogonal projection when $\hat{p}_{\boldsymbol{H}}$ is a normal density.

It should be noted that the orthogonality of ridge projection does not generally hold for arbitrary densities whose Hessian eigenvectors are not constant. Furthermore, ignoring the eigenvector derivatives in (3.6a), which leads to the iteration formula (3.3), leads to deviation from the actual solution curve. Nevertheless, in this way it is possible to obtain approximate ridge projections that are computationally cheap and accurate enough for most purposes. A more detailed analysis of projection curves of the form (3.6) will be given in Chapter 5.

3.3 The trust region Newton method

The conceptually simple mean shift method is easy to implement and works well in many applications, but it suffers from slow convergence. Some improvements have been proposed to alleviate this shortcoming. For instance, Carreira-Perpiñán [26] proposes a hybrid method. This method alternates between the Newton step and a gradient ascent step and resorts to the latter when the Hessian is not negative definite. A more sophisticated trust region Newton method for finding not only modes but also r-dimensional ridge points of kernel densities is developed in **Paper I**.

3.3.1 Overview of the method

The method developed in [I] is an extension of the classical trust region Newton method by Moré and Sorensen [91]. As in [91], the method successively maximizes the quadratic model

$$Q_k(\boldsymbol{s}) = \log \hat{p}(\boldsymbol{x}_k) + \nabla \log \hat{p}(\boldsymbol{x}_k)^T \boldsymbol{s} + \frac{1}{2}\boldsymbol{s}^T \nabla^2 \log \hat{p}(\boldsymbol{x}_k)\boldsymbol{s} \tag{3.7}$$

of the objective function that is taken to be the logarithmic kernel density $\log \hat{p}$. Here we omit the bandwidth $\boldsymbol{H}$ for notational convenience.

The difference to the classical trust region method is that as in SCMS, the iteration is constrained to the subspace

$$S_r(\boldsymbol{x}_k) = \operatorname{span}(\boldsymbol{v}_{r+1}(\boldsymbol{x}_k), \boldsymbol{v}_{r+2}(\boldsymbol{x}_k), \ldots, \boldsymbol{v}_d(\boldsymbol{x}_k)) \tag{3.8}$$

spanned by the last $d - r$ eigenvectors of the Hessian. This constraint is incorporated into the trust region subproblem

$$\max_{\boldsymbol{s}} Q_k(\boldsymbol{s}) \quad \text{s.t.} \quad \begin{cases} \|\boldsymbol{s}\| \leq \Delta_k, \\ \boldsymbol{s} \in S_r(\boldsymbol{x}_k), \end{cases} \tag{3.9}$$

whose solution yields the step $\boldsymbol{s}_k$ at each iteration. The solution method is described in Subsection 3.3.2.

In order to control the quality of the quadratic model (3.7), the ratio

$$\rho_k = \frac{\log \hat{p}(\boldsymbol{x}_k + \boldsymbol{s}_k) - \log \hat{p}(\boldsymbol{x}_k)}{Q_k(\boldsymbol{s}_k) - Q_k(\boldsymbol{0})} \tag{3.10}$$

between the actual increase of the objective function and the increase predicted by the model is tested. Based on this ratio, the algorithm adjusts the trust region radius Δ_k and chooses whether to accept the step $\boldsymbol{s}_k$ obtained by solving the subproblem (3.9). The radius Δ_k is adjusted according to the rules

$$\Delta_{k+1} = \begin{cases} \frac{1}{2}\Delta_k, & \text{if } \rho_k < \frac{1}{4}, \\ \min\{2\Delta_k, \Delta_{\max}\}, & \text{if } \|\boldsymbol{s}_k\| = \Delta_k \text{ and } \rho_k > \frac{3}{4}, \\ \Delta_k, & \text{otherwise}, \end{cases} \tag{3.11}$$

where the constants are good-known values based on numerical experiments. The parameter $\Delta_{\max}$ specifies the maximum trust region radius, and it can be used to adjust the accuracy of the ridge projection.

If the increase given by the quadratic model is sufficient, the step $\boldsymbol{s}_k$ is accepted, and otherwise rejected according to

$$\boldsymbol{x}_{k+1} = \begin{cases} \boldsymbol{x}_k + \boldsymbol{s}_k, & \text{if } \rho_k > \frac{1}{10}, \\ \boldsymbol{x}_k, & \text{otherwise}. \end{cases} \tag{3.12}$$

Adapting the standard gradient norm stopping criterion to the ridge Definition 2.1.1, the above iteration uses the criteria

$$\|\nabla_{\text{pr}} \log \hat{p}(\boldsymbol{x}_k)\| < \varepsilon_{\text{pr}} \quad \text{and} \quad \lambda_{r+1}(\boldsymbol{x}_k) \leq 0. \tag{3.13}$$

Here $\varepsilon_{\text{pr}} > 0$ is some small user-chosen threshold value, and

$$\nabla_{\text{pr}} \log \hat{p}(\boldsymbol{x}_k) = \boldsymbol{P}_r(\boldsymbol{x}_k)\nabla \log \hat{p}(\boldsymbol{x}_k)$$

is the projection of the gradient onto the subspace $\boldsymbol{S}_r(\boldsymbol{x}_k)$.

The iterative method described above is listed as Algorithm 3.1.

Algorithm 3.1: GTRN (generalized trust region Newton)

```
input  : Gaussian kernel density $\hat{p} : \mathbb{R}^d \to \mathbb{R}$
         starting point $\boldsymbol{x}_0 \in \mathbb{R}^d$
         ridge set dimension $0 \leq r < d$
         maximum trust region radius $\Delta_{\max} > 0$
         stopping criterion threshold $\varepsilon_{\text{pr}} > 0$
         maximum number of iterations $k_{\max}$
output: ridge point $\boldsymbol{x}^* \in \mathcal{R}^r_{\log \hat{p}}$.
for $k = 0, 1, \dots, k_{\max} - 1$ do
    Evaluate $\log \hat{p}(\boldsymbol{x}_k)$, $\nabla \log \hat{p}(\boldsymbol{x}_k)$, $\nabla_{\text{pr}} \log \hat{p}(\boldsymbol{x}_k)$ and $\nabla^2 \log \hat{p}(\boldsymbol{x}_k)$.
    Compute the eigendecomposition of $\nabla^2 \log \hat{p}(\boldsymbol{x}_k)$.
    if conditions (3.13) are satisfied then terminate with $\boldsymbol{x}^* = \boldsymbol{x}_k$.
    Obtain $\boldsymbol{s}_k$ as a solution to (3.9).
    Compute $\rho_k$ according to (3.10).
    Choose $\Delta_{k+1}$ and $\boldsymbol{x}_{k+1}$ according to (3.11) and (3.12),
    respectively.
Return with $\boldsymbol{x}^* = \boldsymbol{x}_k$.
```

3.3.2 Solution of the trust region subproblem

It is shown in [I] that the solution to the subspace-constrained trust region subproblem (3.9) can be obtained by using a projection of the eigendecomposition of the Hessian $\nabla^2 \log \hat{p}_{\boldsymbol{H}}$ onto the subspace $S_r(\boldsymbol{x}_k)$.

To simplify the notation, in the following we consider the equivalent problem

$$\max_{\boldsymbol{s}} \quad \boldsymbol{g}^T \boldsymbol{s} + \frac{1}{2} \boldsymbol{s}^T \boldsymbol{A} \boldsymbol{s} \tag{3.14a}$$

$$\text{s.t.} \quad \|\boldsymbol{s}\| \leq \Delta \text{ and } \boldsymbol{s} \in \operatorname{span}(\boldsymbol{v}_{r+1}, \boldsymbol{v}_{r+2}, \dots, \boldsymbol{v}_d), \tag{3.14b}$$

where $\Delta > 0$, $0 \leq r < d$, $\boldsymbol{g} \in \mathbb{R}^d$ and $\{\boldsymbol{v}_i\}_{i=r+1}^d \subset \mathbb{R}^d$ denote the normalized eigenvectors of a matrix $\boldsymbol{A} \in \mathbb{R}^{d \times d}$ corresponding to the $d - r$ smallest eigenvalues $\lambda_{r+1} \geq \lambda_{r+2} \geq \cdots \geq \lambda_d$.

The solution method is based on the following lemma giving the sufficient KKT optimality conditions for problem (3.9).

Lemma 3.3.1 ([I]) *A vector $\boldsymbol{s}^* \in \mathbb{R}^d$ is a solution to problem (3.14) if $\boldsymbol{s}^*$ satisfies conditions (3.14b) and the conditions*

$$\boldsymbol{V}(\boldsymbol{\Lambda} - \kappa \boldsymbol{I})\boldsymbol{V}^T \boldsymbol{s}^* = -\boldsymbol{V}\boldsymbol{V}^T \boldsymbol{g}, \tag{3.15}$$

$$\kappa(\Delta - \|\boldsymbol{s}^*\|) = 0, \tag{3.16}$$

$$\boldsymbol{V}(\boldsymbol{\Lambda} - \kappa \boldsymbol{I})\boldsymbol{V}^T \quad \textit{is negative semidefinite} \tag{3.17}$$

hold for some $\kappa \geq 0$, where

$$\begin{aligned}
\boldsymbol{V} &= [\boldsymbol{v}_{r+1}, \boldsymbol{v}_{r+2}, \dots, \boldsymbol{v}_d] \in \mathbb{R}^{d\times(d-r)},\\
\boldsymbol{\Lambda} &= \operatorname{diag}[\lambda_{r+1}, \lambda_{r+2}, \dots, \lambda_d] \in \mathbb{R}^{(d-r)\times(d-r)}
\end{aligned}$$

and $\boldsymbol{I}$ denotes the $(d-r)\times(d-r)$ identity matrix.

By noting that condition (3.15) is equivalent to

$$\boldsymbol{s}^* = -\boldsymbol{V}(\boldsymbol{\Lambda} - \kappa^*\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{g} \tag{3.18}$$

when $\boldsymbol{s}^* \in \operatorname{span}(\boldsymbol{v}_{r+1}, \boldsymbol{v}_{r+2}, \dots, \boldsymbol{v}_d)$ and $\lambda_{r+1} - \kappa^* < 0$, we obtain a formula for the optimal solution to (3.14).

Formula (3.18) shows the main advantage of constraining the step computation to the subspace defined by equation (3.8) instead of the tangent space of solutions to (3.6). Though the ability to follow the solution space of (3.6) accurately is lost, the computationally convenient form (3.18) cannot be obtained when the terms containing eigenvector derivatives are included.

Using the step formula (3.18), solving problem (3.14) amounts to finding a scalar $\kappa^* \geq 0$ such that conditions (3.16) and (3.17) are satisfied. To this end, we observe that by parametrizing the set of possible solutions $\boldsymbol{s}^*$ with respect to κ we obtain

$$\|\boldsymbol{s}(\kappa)\| = \|\boldsymbol{V}(\boldsymbol{\Lambda} - \kappa\boldsymbol{I})^{-1}\boldsymbol{V}^T\boldsymbol{g}\| = \left[\sum_{i=r+1}^{d} \frac{(\boldsymbol{g}^T\boldsymbol{v}_i)^2}{(\lambda_i - \kappa)^2}\right]^{\frac{1}{2}}. \tag{3.19}$$

For $\kappa = 0$, the above step reduces to the standard Newton step

$$\boldsymbol{s}(0) = \boldsymbol{V}\boldsymbol{\Lambda}^{-1}\boldsymbol{V}^T\boldsymbol{g} = \boldsymbol{A}^{-1}\boldsymbol{g}.$$

The Hessian eigenvalue λ_{r+1} and the Newton step length $\|\boldsymbol{s}(0)\|$ determine the approach for solving the step $\boldsymbol{s}^*$. Special cases occur when $\boldsymbol{g}^T\boldsymbol{v}_{r+1} = 0$. An exhaustive list of all possible cases is given below.

(i) $\lambda_{r+1} < 0$ and $\|\boldsymbol{s}(0)\| \leq \Delta$

(ii) conditions (i) are not satisfied and $\boldsymbol{g}^T\boldsymbol{v}_{r+1} \neq 0$

(iii) conditions (i) are not satisfied, $\boldsymbol{g}^T\boldsymbol{v}_{r+1} = 0$ and either

   (a) $\lambda_{r+1} < 0$

   (b) $\lambda_{r+1} \geq 0$ and $\boldsymbol{g}^T\boldsymbol{v}_i \neq 0$ for some $i \geq r+1$ such that $\lambda_i = \lambda_{r+1}$

   (c) $\lambda_{r+1} \geq 0$ and $\boldsymbol{g}^T\boldsymbol{v}_i = 0$ for all $i \geq r+1$ such that $\lambda_i = \lambda_{r+1}$ and $\|\boldsymbol{s}(\max\{\lambda_{r+1}, 0\})\| > \Delta$

   (d) $\lambda_{r+1} \geq 0$, $\boldsymbol{g}^T\boldsymbol{v}_i = 0$ for all $i \geq r+1$ such that $\lambda_i = \lambda_{r+1}$ and $\|\boldsymbol{s}(\max\{\lambda_{r+1}, 0\})\| \leq \Delta$

Adapting the results of [91], each of the above cases is analyzed in [I]. In case (i), the Newton step $\boldsymbol{s}(0)$ is well-defined and lies inside the trust region. This step is a solution to (3.9) since conditions (3.16) and (3.17) are satisfied with $\kappa = 0$. In cases (ii) and (iiia)-(iiic), it is shown that the optimal κ^* satisfying conditions (3.16) and (3.17) can be obtained as a solution to the equation $\|\boldsymbol{s}(\kappa)\| = \Delta$. Once the eigendecomposition of $\boldsymbol{A}$ has been obtained, solving this equation amounts to univariate root-finding involving no matrix computations, which can be seen from (3.19). A method for this purpose is developed in [I]. The last case (iiid) is analogous to the "hard case" described in [91], in which the root-finding method is not applicable. In this case the solution can be obtained by using a formula derived in [I].

Differently to the algorithm of [91] that uses a Cholesky decomposition of the Hessian, the proposed algorithm uses a full eigendecomposition. The rationale behind this choice is that in typical applications of the algorithm such as shape extraction, the dimension d is small, and thus computing a full eigendecomposition does not incur a significant additional cost. This is also because the projection onto the subspace (3.8) in any case requires computation of either the first r or last $d - r$ Hessian eigenvectors. Another advantage of the proposed approach is that the root-finding iterations do not require any matrix factorizations since the matrix-vector products in (3.19) can be precomputed.

3.3.3 Convergence to a ridge point

A rigorous convergence proof for the trust region Newton method described in Subsections 3.3.1 and 3.3.2 is given in [I]. Generalizing the earlier results by Moré and Sorensen [91] for convergence to a second-order stationary point, the authors prove convergence of the modified method to an r-dimensional ridge point.

The analysis in [I] is done for Gaussian kernel densities, but the results in fact hold for any twice continuously differentiable function f under mild assumptions. The basic idea in [I] is to extend the proof construction of [91] by using the reduced quadratic model

$$\tilde{Q}_k(\tilde{\boldsymbol{s}}) = f(\boldsymbol{x}_k) + \tilde{\nabla} f(\boldsymbol{x}_k)^T \tilde{\boldsymbol{s}} + \frac{1}{2} \tilde{\boldsymbol{s}}^T \tilde{\nabla}^2 f(\boldsymbol{x}_k) \tilde{\boldsymbol{s}}$$

and show that the algorithm converges to a ridge point when the steps $\boldsymbol{s}_k$ satisfy the extended KKT conditions (3.15)–(3.17). Here the reduced gradient and Hessian are defined as

$$\tilde{\nabla} f(\boldsymbol{x}_k) = \boldsymbol{V}_k^T \nabla f(\boldsymbol{x}_k) \quad \text{and} \quad \tilde{\nabla}^2 f(\boldsymbol{x}_k) = \boldsymbol{V}_k^T \nabla^2 f(\boldsymbol{x}_k) \boldsymbol{V}_k,$$

$\tilde{\boldsymbol{s}}_k = \boldsymbol{V}_k^T \boldsymbol{s}_k$ denotes a projected step and

$$\begin{aligned}\boldsymbol{V}_k &= [\boldsymbol{v}_{r+1}(\boldsymbol{x}_k), \boldsymbol{v}_{r+2}(\boldsymbol{x}_k), \ldots, \boldsymbol{v}_d(\boldsymbol{x}_k)] \in \mathbb{R}^{d \times (d-r)}, \\ \boldsymbol{\Lambda}_k &= \operatorname{diag}[\lambda_{r+1}(\boldsymbol{x}_k), \lambda_{r+2}(\boldsymbol{x}_k), \ldots, \lambda_d(\boldsymbol{x}_k)] \in \mathbb{R}^{(d-r)\times(d-r)}.\end{aligned}$$

The proofs in [I] are carried out under the following assumptions on the objective function f and the starting point $\boldsymbol{x}_0$.

Assumption 3.3.1 *The following conditions are satisfied.*

(i) The superlevel set $\mathcal{L}_c = \{\boldsymbol{x} \in \mathbb{R}^d \mid f(\boldsymbol{x}) \geq c\}$ with $c = f(\boldsymbol{x}_0)$ is compact.

(ii) The Hessian $\nabla^2 f$ is locally Lipschitz continuous on some superlevel set $\mathcal{L}_c$ whose interior contains $\boldsymbol{x}_0$.

For any given starting point $\boldsymbol{x}_0$, the former condition is shown to hold for the logarithm of a Gaussian kernel density in [I]. Consequently, as a C^∞-function it also satisfies the latter condition.

The convergence proof given in [I] not only extends the proof of [91] to ridge sets, but also adds some missing parts to the original one. In particular, the authors show that after a finite number of steps the rule (3.12) always yields an iterate $\boldsymbol{x}_{k+1}$ such that $\boldsymbol{x}_{k+1} \neq \boldsymbol{x}_k$ (i.e. that $\rho_k > \frac{1}{10}$). This holds provided that the step $\boldsymbol{s}_k$ satisfies conditions (3.15)–(3.17) for some $\kappa_k \geq 0$ at each iteration.

The main convergence result states the following. Given a starting point $\boldsymbol{x}_0$ and a twice differentiable function f satisfying Assumption 3.3.1, iteration of Algorithm 3.1 converges to a ridge point in a weak sense.

Theorem 3.3.1 ([I]) *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a twice differentiable function satisfying Assumption 3.3.1 for a starting point $\boldsymbol{x}_0 \in \mathbb{R}^d$. Define the set*

$$\tilde{\mathcal{R}}_f^r = \{\boldsymbol{x} \in \mathbb{R}^d \mid \nabla f(\boldsymbol{x})^T \boldsymbol{v}_i(\boldsymbol{x}) = 0 \text{ for all } i > r \text{ and } \lambda_{r+1}(\boldsymbol{x}) \leq 0\}.$$

If $\{\boldsymbol{x}_k\}$ is a sequence generated by Algorithm 3.1 applied to f with $0 \leq r < d$, then either the algorithm terminates at some $\boldsymbol{x}_k \in \tilde{\mathcal{R}}_f^r$ or $\{\boldsymbol{x}_k\}$ has a subsequence converging to a point $\boldsymbol{x}^* \in \tilde{\mathcal{R}}_f^r$.

If we make stronger assumptions, the sequence $\{\boldsymbol{x}_k\}$ generated by Algorithm 3.1 converges to a ridge point in the sense of Definition 2.1.1. Assuming that the iterates lie in a set U such that

$$\lambda_{r+1}(\boldsymbol{x}) < 0 \quad \text{and} \quad \lambda_r(\boldsymbol{x}) > \lambda_{r+1}(\boldsymbol{x}) \quad \text{for all } \boldsymbol{x} \in U, \tag{3.20}$$

then the limit point $\boldsymbol{x}^*$ is a ridge point except condition (2.4c) for the first r eigenvectors. Conditions (3.20) also guarantee the desired property that the choice of the first r eigenvalues and eigenvectors is unique.

Assuming conditions (3.20) is reasonable when Algorithm 3.1 is used in the application it is primarily designed for. That is, projection of a sample point $\boldsymbol{y}_i$ onto a ridge of a kernel density estimate in the statistical framework described in Sections 2.3 and 2.4. For this assumption to be valid, we need certain additional assumptions.

(i) The points $\boldsymbol{y}_i$ are sampled from an m-dimensional manifold with additive noise (i.e. from the generative model described in Section 2.3).

(ii) The amount of noise (i.e. the standard deviation σ) is sufficiently small.

(iii) The kernel density $\hat{p}_{\boldsymbol{H}}$ gives an accurate estimate of the marginal density p and its derivatives up to third order.

(iv) The starting point $\boldsymbol{x}_0$ is sufficiently close to a ridge of $\hat{p}_{\boldsymbol{H}}$.

(v) The ridge dimension r in Algorithm 3.1 is chosen as m.

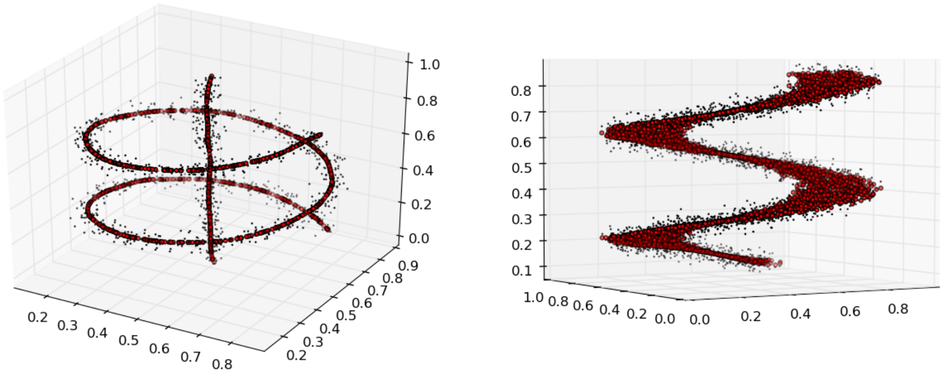
Assuming conditions (i) and (ii), Theorem 2.3.2 implies that the logarithmic marginal density $\log p$ satisfies conditions (3.20) uniformly in some ε-dilation of a subset of the ridge set $\mathcal{R}^m_{\log p}$ lying near the underlying manifold. For any asymptotically optimal kernel density estimator, condition (iii) is satisfied when the sample size N is sufficiently large.[1] When conditions (i)–(iii) are satisfied, also the kernel density estimate $\log \hat{p}_{\boldsymbol{H}}$ satisfies conditions (3.20) uniformly in some ε-dilation of such a subset of $\mathcal{R}^m_{\log \hat{p}_{\boldsymbol{H}}}$ by Theorem 2.3.3.

By "sufficiently close" in condition (iv), we mean by the above remarks that the starting point $\boldsymbol{x}_0$ lies in some ε-dilation of $\log \hat{p}_{\boldsymbol{H}}$ where conditions (3.20) are satisfied. Condition (iv) is implied by condition (ii). This is because the points sampled from the model, that are used as starting points for Algoritm 3.1, are expected to be near the underlying manifold. Condition (v) is the most difficult to satisfy, because it requires a priori information on the manifold dimension m. In principle, it can be estimated from the data, but this topic is beyond the scope of this thesis.

If we in addition to conditions (3.20) assume that the kernel density $\log \hat{p}_{\boldsymbol{H}}$ satisfies condition (2.4c) in the set U containing the iterates, we can say more about the iteration of Algorithm 3.1. In this case, the iteration path can be interpreted as an approximate projection in a curvilinear coordinate system. This is because the first r Hessian eigenvectors are continuous along the iteration path by the assumption that condition (2.4c) holds in U and Theorem 2.1.2. This, in its turn, implies continuity of the orthogonal subspace spanned by the last $d-r$ eigenvectors from which the steps $\boldsymbol{s}_k$ are obtained.

[1] The kernel density estimators discussed in Section 2.4 are not optimal for all derivatives, but only for derivatives of the chosen order and not in the sense of the sup-norm as in Theorem 2.3.3. Utilization of a more advanced method is left as a topic of future research.

Though assumptions (i)–(v) might seem restrictive, they are often satisfied in practice. Some data sets and their projections onto kernel density ridges shown in Figure 3.1 demonstrate cases where these assumptions are plausible. Figure 1.1b shows another example of such a case.



(a) Spiral3d ($r = 1$)

(b) Helix ($r = 2$)

Figure 3.1: Projections of synthetically generated point sets onto r-dimensional kernel density ridges.

3.4 Comparison between trust region and mean shift methods

In this section we discuss some theoretical advantages that the Newton-based method has over the mean shift-based methods. We also summarize the results of the numerical experiments done in [I] to compare the performance of these methods.

3.4.1 Theoretical considerations

The trust region Newton method has several advantages compared to the mean shift method and the SCMS variant. The most important ones are listed below.

- As shown in the following, the trust region Newton method consistently outperforms the mean shift-based methods due to significantly faster convergence rate.
- The mean shift methods are only applicable to kernel densities under certain assumptions on the kernel function. The Newton-based method is applicable to any function satisfying Assumption 3.3.1 for any given starting point $\boldsymbol{x}_0$.

- The mean shift methods lack the ability to choose the step size. This is conveniently incorporated in the trust region method via the maximum trust region radius $\Delta_{\max}$.
- As pointed out by Carreira-Perpiñán [27] and Carreira-Perpiñán and Williams [28], the mean shift method does not always converge to a nearby mode. The attraction basins can be nonconvex or even disconnected. The trust region mechanism effectively forces the iteration to converge to a nearby mode or a ridge point.
- Once the eigendecomposition has been obtained, computing the trust region step comes at a nominal cost. The SCMS method cannot utilize the $d-r$ smallest eigenvalues and eigenvectors in the step computation.

Projection of a set of N d-dimensional points by using the trust region Newton and SCMS methods with a full Hessian eigendecomposition has a computational cost of $\mathcal{O}(N^2d^2 + Nd^3)$. This is because the projections are done for all N points. The cost of evaluating the Hessian of the density estimate (2.15) with a Gaussian kernel is $\mathcal{O}(Nd^2)$. After each evaluation, the Hessian eigendecomposition has a cost of $\mathcal{O}(d^3)$. For the standard mean shift iteration (3.1), the eigendecomposition is not necessary, and the total cost reduces to $\mathcal{O}(N^2d^2)$. The above observations imply that kernel density- and ridge-based methods scale poorly to large or high-dimensional data. Possible improvements to alleviate this shortcoming are discussed in Chapter 7.

3.4.2 Numerical experiments

A set of numerical experiments is carried out in [I]. The authors compare performance of the GTRN algorithm (Algorithm 3.1) to the mean shift algorithm and its subspace-constrained variant SCMS. The algorithms are implemented in Fortran 95.

The numerical tests in [I] are done on test problems, where a set of points is generated from a set of curves or surfaces with additive normally distributed noise according to the model described in Section 2.3. A more defailed description of the test problems is given in [I]. The densities are estimated by using Gaussian kernel estimates of the form (2.15) with diagonal bandwidth matrices $\boldsymbol{H} = h^2\boldsymbol{I}$ and hand-picked values for h.

The test runs are done by starting each algorithm from each sample point in the dataset. This yields a projection of the points onto the ridge set of their kernel density, as illustrated in Figure 3.1. The GTRN and mean shift algorithms are run with $\varepsilon_{\mathrm{pr}} = 10^{-6}$ and 200 as the maximum number of iterations. For the GTRN algorithm, the parameter $\Delta_{\max}$ is set to $3h$. For the mean shift algorithms, only the first condition in (3.13) is tested because convergence to a second-order stationary point cannot be guaranteed.

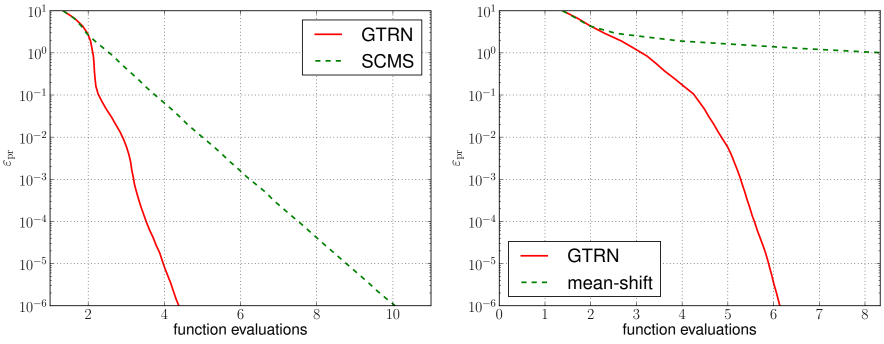
Table 3.1 taken from [I] shows the number of function evaluations used by the GTRN and the SCMS algorithms on some test problems. The ridge

	SCMS		GTRN	
	num. eval.	CPU time	num. eval.	CPU time
Circle	13 965	7.85	3 783	2.13
DistortedHalfCircle	12 161	6.81	3 684	2.08
DistortedSShape	10 561	5.93	3 384	1.91
HalfCircle	9 878	5.54	3 341	1.88
Helix	24 520	91.94	14 463	54.24
Spiral	15 984	15.19	5 701	5.44
Spiral3d	12 070	14.37	5 253	6.26
Zigzag	12 214	7.12	3 796	2.26

Table 3.1: Function evaluations and CPU times used by the `SCMS` and `GTRN` algorithms for ridge projection.

	Mean shift		GTRN	
	num. eval.	CPU time	num. eval.	CPU time
Circle	85 496	35.51	4 541	2.54
DistortedHalfCircle	137 335	57.05	4 826	2.68
DistortedSShape	114 078	47.42	4 770	2.67
HalfCircle	119 016	49.46	4 699	2.62
Spiral	220 654	159.11	8 170	7.72
Spiral3D	163 564	107.56	7 363	8.71
Zigzag	111 563	47.07	4 772	2.73

Table 3.2: Function evaluations and CPU times used by the mean shift and `GTRN` algorithms for mode finding.



(a) Ridge projection

(b) Mode finding

Figure 3.2: Average number of function evaluations needed to reach stopping criteria (3.13) on the Spiral3d test problem.

dimension r is in these tests set to the dimension of the generating function of the data ($r = 1$ for all datasets except Helix). These results show that the subspace-constrained method inherits the rapid convergence rate from the standard Newton method, and it consistently outperforms `SCMS` on all test problems. The performance difference is even larger when the methods are applied to mode finding (i.e. ridge projection with $r = 0$), which can be seen from Table 3.2.

The results shown in Tables 3.1 and 3.2 might not give a complete picture. This is because the stopping criterion ε_{pr} is too strict for most practical applications, and the relative performance of the algorithms may depend on the desired accuracy. In order to address this gap, the average function evaluations needed to reach a given threshold ε_{pr} on the Spiral3d test problem are plotted in Figure 3.2 taken from [I]. Here the evaluation counts are averaged over the algorithm executions started from each sample point $\boldsymbol{y}_i$ in the dataset.

The plots shows in Figure 3.2 can be interpreted as convergence rates. This is because each iteration of the `GTRN` algorithm typically uses k evaluations of the objective function and its gradient and Hessian, where k is a small number. For the mean shift-based methods, this number is exactly one. For ridge projection ($r > 0$), the `GTRN` algorithm converges faster than the `SCMS` algorithm on a wide range of threshold parameters. This strengthens the conclusion made from the results of Table 3.1. For mode finding ($r = 0$), the `GTRN` algorithm achieves a superlinear convergence rate, whereas the mean shift method converges at a very slow linear rate. This result agrees with the theoretical convergence rate discussed in Section 3.1. An interesting observation is that the convergence rate of `GTRN` seems to degrade to linear when the subspace constraint is imposed. A theoretical explanation of this behaviour remains as a topic of future research.

3.5 Finding global modes of Gaussian mixtures and kernel densities

The second part of this chapter based on **Paper II** deals with finding global modes of Gaussian mixtures and kernel densities. Performing exhaustive mode finding of such a density, as described in [26], is not feasible in applications where the computational budget is limited. When this is the case, one must resort to seeking for a single mode that preferably is the global one or in some sense significant. A good example of such an application is real-time object tracking in computer vision [59, 112].

The mean shift method is a standard tool for finding modes of kernel densities. However, as a local method, it tends converge to an irrelevant local mode when applied to a highly multimodal density and the global mode is sought. To address this shortcoming, Han et al. [59] and Shen et al. [112] propose a variant of a mean shift method that traces modes at multiple scales by adjusting the kernel bandwidth. They demonstrate that the method efficiently finds global modes of Gaussian kernel densities arising in visual object tracking. Other approaches for finding a global mode include the branch and bound method by Wirjadi et al. [132].

Tracing intensity maxima of images through different scales is also a key task in digital image processing. Such methods are closely related to the theme of this section. This is because an image may be viewed as a set of points convolved by a Gaussian kernel (cf. Subsection 2.2.1). For example, Collins [33] describes a method based on this idea and the *scale space* theory from image processing (e.g. [81]).

A method for finding global modes of Gaussian mixtures and kernel densities is developed in [II]. This paper extends the ideas of Han et al. [59] and Shen et al. [112] to more general density functions and gives a more rigorous mathematical treatment. The mathematical theory also borrows some elements from the homotopy continuation methods by Moré and Wu [92] and Wu [133] that were originally developed for applications in theoretical chemistry.

In [II], the authors consider densities of the form

$$p(\mathbf{x}) = \sum_{i=1}^{n} w_i g(\mathbf{x}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{3.21}$$

with weighting coefficients $w_i \in \mathbb{R}$ such that $\sum_{i=1}^{n} w_i = 1$ and

$$g(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})}{2}\right) \tag{3.22}$$

denoting a d-variate normal distribution with mean $\boldsymbol{\mu}$ and positive definite covariance matrix $\boldsymbol{\Sigma}$. The Gaussian kernel density estimate defined according to (2.15) and (2.16) is a special case of the density (3.21) with $w_i = \frac{1}{N}$ and $\boldsymbol{\Sigma}_i = \boldsymbol{H}$ for all $i = 1, 2, \dots, n$. Though various different interpretations could be given to the density (3.21), in the following we will simply call it a *Gaussian mixture.*

The Gaussian mixture (3.21) is a generalization of the standard Gaussian mixture, where the weights w_i are assumed to be positive. Mixture densities with negative weights appear in many applications such as target tracking and sensor data fusion. For instance, Koch [73] describes a model where negative weights describe ”negative information”. That is, expected but missing sensor measurements.

In [II], the analysis is restricted to the *isotropic* case, where

$$\boldsymbol{\Sigma}_i = \sigma_i^2 \mathbf{I}, \quad \sigma_i > 0, \quad i = 1, 2, \dots, n$$

and $\mathbf{I}$ denotes the $d \times d$ identity matrix. In this case, the Gaussian mixture (3.21) reduces to

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}} \sum_{i=1}^{n} \frac{w_i}{\sigma_i^d} \exp\left(-\frac{\|\mathbf{x} - \boldsymbol{\mu}_i\|^2}{2\sigma_i^2}\right). \tag{3.23}$$

3.5.1 The Gaussian convolution and homotopy continuation

The global mode finding method developed in [II] is based on a homotopy continuation approach adapted from [92] and [133]. The idea is to apply the Gaussian convolution

$$\langle p \rangle_\gamma(\boldsymbol{x}) = \frac{1}{(\sqrt{\pi}\gamma)^d} \int\limits_{\mathbb{R}^d} p(\boldsymbol{y}) \exp\left(-\frac{\|\boldsymbol{y} - \boldsymbol{x}\|^2}{\gamma^2}\right) d\boldsymbol{y},$$

where $\gamma > 0$ is a smoothing parameter. This transformation can be intuitively interpreted as a local averaging operation. Larger values of γ produce a "smoother" function, and the original function p is obtained at the limit $\gamma \to 0$.

For the isotropic Gaussian mixture (3.23), the Gaussian convolution has a closed-form expression given by

$$\langle p \rangle_\gamma(\boldsymbol{x}) = \sum_{i=1}^{n} \tilde{C}_{\gamma,i} w_i \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2}{\gamma^2 + 2\sigma_i^2}\right), \tag{3.24}$$

where $\gamma > 0$ and

$$\tilde{C}_{\gamma,i} = \frac{1}{(2\pi)^{\frac{d}{2}}} \left(\frac{2}{\gamma^2 + 2\sigma_i^2}\right)^{\frac{d}{2}}, \quad i = 1, 2, \ldots, n. \tag{3.25}$$

The idea of the continuation principle is to start from some initial transformation parameter $\gamma_0 > 0$ and let γ approach zero. A local maximizer of the objective function is traced along this transformation, which effectively carries the optimization over undesired local maxima. This is illustrated in Figure 3.3.

Formally, the continuation idea can be defined in terms of a *homotopy mapping*. The Gaussian convolution (3.24) induces a $\mathcal{C}^\infty$-homotopy $\rho : \mathbb{R}^d \times [0, \infty[\to \mathbb{R}$ defined as

$$\begin{aligned} \rho(\boldsymbol{x}, \gamma) &= \langle p \rangle_\gamma(\boldsymbol{x}), \\ \rho(\boldsymbol{x}, 0) &= p(\boldsymbol{x}). \end{aligned}$$

The conditions that a maximum is traced along the transformation are stated as

$$\left.\begin{array}{l} \nabla_{\boldsymbol{x}} \rho(\boldsymbol{x}(\gamma), \gamma) = \mathbf{0}, \\ \nabla^2_{\boldsymbol{x}} \rho(\boldsymbol{x}(\gamma), \gamma) \text{ is negative definite} \end{array}\right\} \quad \text{for all } \gamma \in [0, \gamma_0], \quad \gamma_0 > 0. \tag{3.26}$$

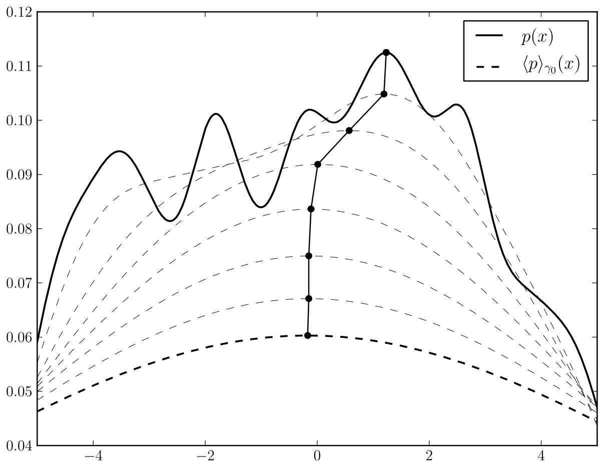


Figure 3.3: A curve connecting the maximizers of the smoothed Gaussian mixture $\langle p\rangle_\gamma$ with different values of transformation parameter γ.

These conditions implicitly define the curve

$$\begin{aligned} &\boldsymbol{x}'(\gamma) = -\nabla^2_{\boldsymbol{x}}\rho(\boldsymbol{x}(\gamma),\gamma)^{-1}\frac{\partial}{\partial\gamma}\nabla_{\boldsymbol{x}}\rho(\boldsymbol{x}(\gamma),\gamma), \quad \gamma\in]0,\gamma_0], \\ &\boldsymbol{x}(\gamma_0) = \boldsymbol{x}_0 \end{aligned} \tag{3.27}$$

that is obtained by differentiating the condition $\nabla_{\boldsymbol{x}}\rho(\boldsymbol{x}(\gamma),\gamma) = \mathbf{0}$ with respect to γ.

The multiscale mean shift methods developed in [59] and [112] can be viewed as simplified implementations of the above approach. In these methods, a convolved Gaussian kernel density is successively maximized by using a sequence of hand-chosen values of the parameter γ. On the other hand, the above approach combined with a numerical method for tracing the solution curve of (3.27) provides two advantages. That is, a rigorous way of choosing the sequence of transformation parameters γ and also the starting point for each maximization of $\langle p\rangle_\gamma$.

3.5.2 Choice of initial values

Figure 3.3 illustrates how a transformed mixture $\langle p\rangle_\gamma$ becomes unimodal when the parameter γ is sufficiently large. A rigorous proof for this property involves showing concavity of $\langle p\rangle_\gamma$ for such γ. The proof is carried out in [II] for the isotropic Gaussian mixture (3.23). Furthermore, the authors derive a computable condition for testing the concavity for a given γ. Differently to the heuristic choices used in multiscale mean shift methods, this approach provides a rigorous way of choosing the initial value γ_0.

A closely related result is that any density of the form (2.15) with a compactly supported kernel with bandwidth matrix $\boldsymbol{H} = h\boldsymbol{I}$ becomes unimodal in the whole $\mathbb{R}^d$ when h is sufficiently large [58]. However, because the support of the Gaussian (3.22) is infinite, we need to restrict the analysis to a ball $B(\boldsymbol{z}; r)$ containing the mean vectors $\boldsymbol{\mu}_i$.

Assumption 3.5.1 *The center point $\boldsymbol{z} \in \mathbb{R}^d$ and radius $r > 0$ of the closed ball $B(\boldsymbol{z}; r)$ defined as*

$$B(\boldsymbol{z}; r) = \{\boldsymbol{x} \in \mathbb{R}^d \mid \|\boldsymbol{x} - \boldsymbol{z}\| \leq r\}, \quad \boldsymbol{z} \in \mathbb{R}^d, \quad r > 0$$

are chosen such that $\boldsymbol{\mu}_i \in B(\boldsymbol{z}; r)$ for all $i = 1, \ldots, n$.

Concavity of $\langle p \rangle_\gamma$ in the ball $B(\boldsymbol{z}; r)$ is equivalent to negative definiteness of its Hessian in $B(\boldsymbol{z}; r)$. In order to utilize this fact, the following result is proven in [II] for the homotopy mapping ρ. In particular, it gives a condition for testing whether the greatest eigenvalue $\lambda_{\max}(\cdot)$ of the Hessian $\nabla^2 \rho(\cdot, \gamma)$ is negative in $B(\boldsymbol{z}; r)$, implying negative definiteness. The proof utilizes the Weyl inequality for matrix eigenvalues (e.g. [65]).

Theorem 3.5.1 *Assume 3.5.1. Then*

$$\lambda_{\substack{\max \\ \boldsymbol{x} \in B(\boldsymbol{z}; r)}} \left(\nabla^2_{\boldsymbol{x}} \rho(\boldsymbol{x}, \gamma) \right) \leq 2 \left[\Lambda_1(\gamma) + \Lambda_2(\gamma) \right]$$

for all $\gamma \geq 0$, where

$$\Lambda_1(\gamma) = -\sum_{w_i > 0} \frac{\tilde{C}_{\gamma,i}}{\gamma^2 + 2\sigma_i^2} w_i \exp\left(-\frac{(\|\boldsymbol{z} - \boldsymbol{\mu}_i\| + r)^2}{\gamma^2 + 2\sigma_i^2} \right) - \sum_{w_i < 0} \frac{\tilde{C}_{\gamma,i}}{\gamma^2 + 2\sigma_i^2} w_i,$$
$$\Lambda_2(\gamma) = \sum_{w_i > 0} \frac{2\tilde{C}_{\gamma,i}}{(\gamma^2 + 2\sigma_i^2)^2} w_i \hat{r}_i(\gamma)^2 \exp\left(-\frac{\hat{r}_i(\gamma)^2}{\gamma^2 + 2\sigma_i^2} \right),$$

the constants $\tilde{C}_{\gamma,i}$ are defined according to (3.25) and

$$\hat{r}_i(\gamma) = \min\left\{ \|\boldsymbol{z} - \boldsymbol{\mu}_i\| + r, \sqrt{\gamma^2 + 2\sigma_i^2} \right\}.$$

Furthermore, there exists $\gamma^* > 0$ such that

$$\Lambda_1(\gamma) + \Lambda_2(\gamma) < 0$$

for all $\gamma > \gamma^*$.

In order to guarantee unimodality of $\rho(\cdot, \gamma)$ in $B(\boldsymbol{z}; r)$, it is also necessary to show that it has at least one stationary point in this ball. For a homoscedastic Gaussian mixture with positive weights (e.g. a density of the form (2.15)), one can easily observe from equation (3.2) that its stationary points lie in the convex hull of the mean vectors $\boldsymbol{\mu}_i$ for any admissible bandwidth matrix $\boldsymbol{H}$ [28]. This property does not, however, hold for Gaussian mixtures with negative weights. Instead, a weaker result is obtained in [II]. The result states that the stationary points of $\rho(\cdot, \gamma)$ in $B(\boldsymbol{z}; r)$ converge to a weighted sum of the mean vectors $\boldsymbol{\mu}_i$ as the parameter γ tends to infinity.

Theorem 3.5.2 *Define the set*

$$S_\gamma = \{\boldsymbol{x} \in \mathbb{R}^d \mid \nabla_{\boldsymbol{x}} \rho(\boldsymbol{x}, \gamma) = \mathbf{0}\}$$

and let

$$\boldsymbol{x}^* = \sum_{i=1}^{n} w_i \boldsymbol{\mu}_i \tag{3.28}$$

and $r > 0$*. Then for all* $\epsilon > 0$ *there exists* $\gamma^* > 0$ *such that* $\|\boldsymbol{x} - \boldsymbol{x}^*\| < \epsilon$ *for all* $\boldsymbol{x} \in S_\gamma \cap B(\boldsymbol{x}^*; r)$ *and* $\gamma > \gamma^*$*.*

Theorems 3.5.1 and 3.5.2 give rise to Algorithm 3.2 for finding the starting point $\boldsymbol{x}_0 = \boldsymbol{x}(\gamma_0)$ and the initial transformation parameter γ_0 for tracing the solution curve of (3.27). The idea is to increase γ_0 until two conditions are satisfied. The first one is that the Hessian is strictly negative definite in the ball $B(\boldsymbol{x}^*; r)$ containing the mean vectors $\boldsymbol{\mu}_i$ and the limit point $\boldsymbol{x}^*$ given by equation (3.28). The second one is that a stationary point $\boldsymbol{x}_0$ is also found inside the ball $B(\boldsymbol{x}^*; r)$. That is,

$$\lambda_{\max\limits_{\boldsymbol{x} \in B(\boldsymbol{x}^*; r)}} (\nabla^2_{\boldsymbol{x}} \rho(\boldsymbol{x}, \gamma)) < 0 \quad \text{and} \quad \nabla_{\mathbf{x}} \rho(\mathbf{x}_0, \gamma_0) = \mathbf{0}. \tag{3.29}$$

Algorithm 3.2: Initial values

```
input : Gaussian mixture of the form (3.23)
output: starting point $\boldsymbol{x}_0 \in B(\boldsymbol{x}^*; r)$                  } satisfying conditions (3.29)
        transformation parameter $\gamma_0 > 0$              }
$\mathbf{x}^* \leftarrow \sum_{i=1}^{n} w_i \boldsymbol{\mu}_i$
$r \leftarrow \max\{\|\mathbf{x}^* - \boldsymbol{\mu}_i\| \mid i = 1, 2, \ldots, n\}$
Choose the initial $\gamma_0 > 0$.
while $\Lambda_1(\gamma_0) + \Lambda_2(\gamma_0) \geq 0$ do increase $\gamma_0$.
repeat
    Obtain $\mathbf{x}_0 \in \mathbb{R}^d$ such that $\nabla_{\mathbf{x}} \rho(\mathbf{x}_0, \gamma_0) = \mathbf{0}$, use $\mathbf{x}^*$ as starting
    point.
    if $\mathbf{x}_0 \notin B(\mathbf{x}^*; r)$ then Increase $\gamma_0$.
until $\mathbf{x}_0 \in B(\mathbf{x}^*; r)$
```

Remark 3.5.1 *The proof for the property that $\rho(\cdot,\gamma)$ has at least one stationary point in $B(\boldsymbol{x}^*; r)$ for all sufficiently large γ is not given in [II]. However, the proof can be carried out showing that*

$$\lim_{\gamma\to\infty} M(\gamma)\nabla_{\boldsymbol{x}}\rho(\boldsymbol{x},\gamma) = \boldsymbol{x}^* - \boldsymbol{x},$$

where

$$M(\gamma) = \max_{i=1,2,\dots,n} (\gamma^2 + 2\sigma_i^2)^{\frac{d}{2}+1}$$

and the convergence is uniform in $B(\boldsymbol{x}^*; r)$ (see the proof of Theorem 4.4 in [II]). The claim then follows from the fact that at any boundary point $\boldsymbol{x}$ of $B(\boldsymbol{x}^*; r)$, the vector $\boldsymbol{x}^* - \boldsymbol{x}$ is parallel to the inward-pointing normal vector.

3.5.3 Implementation of the continuation method

A predictor-corrector algorithm for tracing the solution curve of the initial value problem (3.27) is developed in [II]. At each iteration, the algorithm takes a predictor step along a tangent direction of the solution curve of problem (3.27). The expression for the predictor is given by

$$\tilde{\boldsymbol{x}}_k(\tau) = \boldsymbol{x}_k - \tau\boldsymbol{T}(\boldsymbol{x}_k,\gamma_k), \tag{3.30}$$

where

$$\boldsymbol{T}(\boldsymbol{x},\gamma) = -\nabla^2_{\boldsymbol{x}}\rho(\boldsymbol{x},\gamma)^{-1}\frac{\partial}{\partial\gamma}\nabla_{\boldsymbol{x}}\rho(\boldsymbol{x},\gamma) \tag{3.31}$$

is obtained from the right hand side of (3.27).

The choice of the step size τ is based on the rules

$$\|\tilde{\mathbf{x}}_k(\tau) - \mathbf{x}_k\| = \Delta_p \quad \text{and} \quad \gamma_k - \tau \geq \frac{1}{4}\gamma_k.$$

The first rule attempts to keep the length of the predictor step at some user-specified value $\Delta_p > 0$ to avoid too short step sizes. The second rule is needed to avoid too large step sizes. These rules are imposed by choosing

$$\tau = \min\left\{\frac{\Delta_p}{\|\mathbf{T}(\mathbf{x}_k,\gamma_k)\|}.\frac{3\gamma_k}{4}\right\}. \tag{3.32}$$

After each predictor step, the algorithm starts a corrector iteration from the predictor iterate $\tilde{\boldsymbol{x}}_k(\tau)$ to find a point $\hat{\boldsymbol{x}}_k$ satisfying the condition $\nabla_{\boldsymbol{x}}\rho(\hat{\boldsymbol{x}}_k,\gamma_k-\tau) = \boldsymbol{0}$. As a corrector method, the algorithm uses a trust region Newton method based on the ideas described in Section 3.3 (see [II] for a detailed description). This method is also used in Algorithm 3.2 for finding the starting point $\boldsymbol{x}_0$.

The method described in Subsection 3.3.2 is applicable to the trust region subproblem, though in [II] the authors use a truncated conjugate gradient method developed in [116]. The main advantage of the Newton method over the mean shift method is faster convergence. In addition, when using the method described in Subsection 3.3.2, the point $\hat{\boldsymbol{x}}_k$ is guaranteed to be second-order optimal. It is also worthwhile to note that the mean shift iteration may diverge when applied to a Gaussian mixture with negative weights, which is not an issue for trust region Newton methods.

The predictor-corrector algorithm for tracing a solution curve of the initial value problem (3.27) is listed as Algorithm 3.3.

Algorithm 3.3: Homotopy continuation

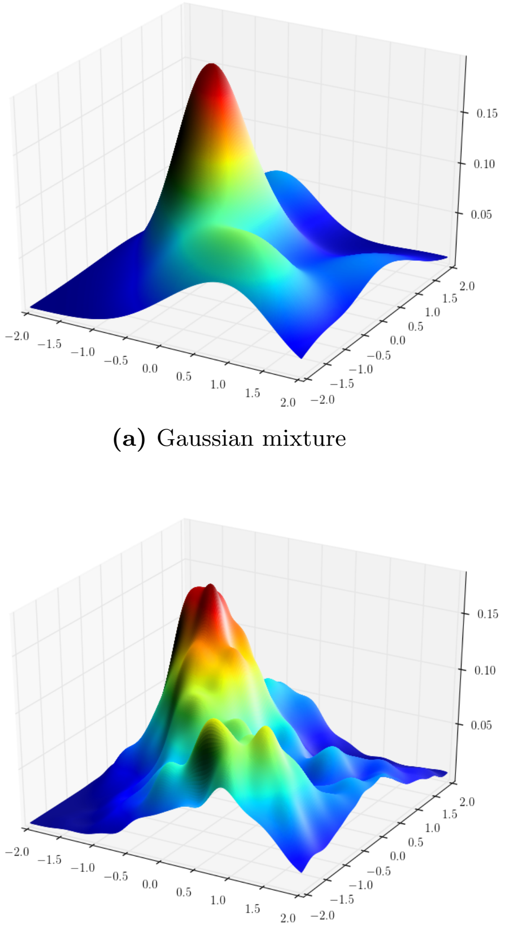
```
input : Gaussian mixture of the form (3.23)
        maximum number of iterations $k_{\max}$
output: estimate of the global mode $\boldsymbol{x}^*$
Choose $\mathbf{x}_0 \in \mathbb{R}^d$ and $\gamma_0 > 0$ satisfying (3.26) by using Algorithm 3.2.
while $\gamma_k > \gamma_{\min}$ and $k \leq k_{\max}$ do
    Choose $\tau$ according to (3.32).
    Compute $\tilde{\boldsymbol{x}}_k(\tau)$ from (3.30).
    Solve $\nabla_{\mathbf{x}} \rho(\hat{\mathbf{x}}_k, \gamma_k - \tau) = \mathbf{0}$ for $\hat{\mathbf{x}}_k$, use $\tilde{\mathbf{x}}_k(\tau)$ as starting point.
    $\mathbf{x}_{k+1} \leftarrow \hat{\mathbf{x}}_k$
    $\gamma_{k+1} \leftarrow \gamma_k - \tau$
    $k \leftarrow k + 1$
if $\gamma_{\min} = 0$ then
    Solve $\nabla_{\mathbf{x}} \rho(\hat{\mathbf{x}}_k, 0) = \mathbf{0}$ for $\hat{\mathbf{x}}_k$, use $\mathbf{x}_k$ as starting point.
Return with $\boldsymbol{x}^* = \boldsymbol{x}_k$.
```

3.5.4 Test results

Two potential applications for the global mode finding method (Algorithm 3.3) are identified in [II]. The first one is finding modes of general Gaussian mixtures with possibly negative weights such as those described in [73]. The second one, which is emphasized in [II], is finding global modes of kernel densities. Though a kernel density is a Gaussian mixture with strictly positive weights $w_i = \frac{1}{n}$ and identical covariance matrices $\boldsymbol{\Sigma}_i = \boldsymbol{H}$, finding the global mode of such a density is still a computationally demanding problem when the sample size n is large.

To stress the inherent difficulty of finding the global mode of a Gaussian kernel density estimate, such an estimate, and the true density are shown in Figure 3.4 taken from [II]. This figure shows that a density estimate may have a large number of spurious local maxima that are not present in the true density. This is possible even when the true density is unimodal.

The case shown in Figure 3.4 is common in practical applications. The



(a) Gaussian mixture

(b) kernel density estimate

Figure 3.4: A bivariate Gaussian mixture with 10 components and a Gaussian kernel density estimate from 5000 simulated samples. The bandwidth $\boldsymbol{H} = h\boldsymbol{I}$ is obtained by approximate minimization of the MISE (2.18) between the true density and the estimate.

spurious maxima are present even with the MISE-optimal bandwidth when a diagonal matrix $\boldsymbol{H} = h^2\boldsymbol{I}$ is used. The situation is usually not better when using a cheap ”plugin” bandwidth chooser. Such bandwidth choosers are commonly used to avoid the high computational cost of optimal bandwidth calculation. Even in the case shown in Figure 3.4, the continuation method, that traces the global mode through different bandwidths, can be used to obtain a good estimate of the global mode of the true density.

A set of numerical tests are carried out in [II] to demonstrate the applicability of the method to Gaussian kernel densities. The continuation method is not guaranteed to give a global mode in all cases. A failure can occur when the density has a very narrow and high peak. This may happen, for instance, when the density is a Gaussian mixture for which the σ-parameter of one component is small compared to the others. In such a case the method is prone to converge to some broader, but lower peak. Another example is a density having peaks of similar shape. Therefore, the success probability P_{succ} defined as the ratio between the number of successful and total test runs is an appropriate measure for the reliability of the method.

In the first set of tests conducted in [II], the algorithm is run on randomly generated Gaussian mixtures of the form (3.23). The parameters are chosen as $w_i = \frac{1}{n}$ and $\sigma_i = h$ for all $i = 1, 2, \ldots, n$. In effect, the h-parameter determines the width of individual modes and the number of modes of the Gaussian mixture. For each run, the means $\boldsymbol{\mu}_i$ are sampled from the uniform distribution such that $\boldsymbol{\mu}_i \in [-2, 2]^d$ for $i = 1, 2, \ldots, n$ and $d = 1, 2, 3$. With this choice and h in the range $[0.25, 0.6]$, the peaks of the mixture have roughly equal shapes and heights.

The success probabilities P_{succ} from 1000 test runs together with the average number of modes as a function of h are plotted in Figure 3.5. The results show that the reliability of the algorithm depends on the choice of the h-parameter. With small values of h, the modes of the Gaussian mixtures are narrowly peaked. The number of modes in this case is also high, as shown in Figure 3.5. This makes the global mode very difficult to identify, which corresponds to the low success rates. On the other hand, with larger values of h the peaks are broader, and the average number of modes decreases. When the number of modes is less than ten, the algorithm achieves over 60% success rate. Also, with on average five modes, the algorithm achieves 75% success rate at identifying global modes.

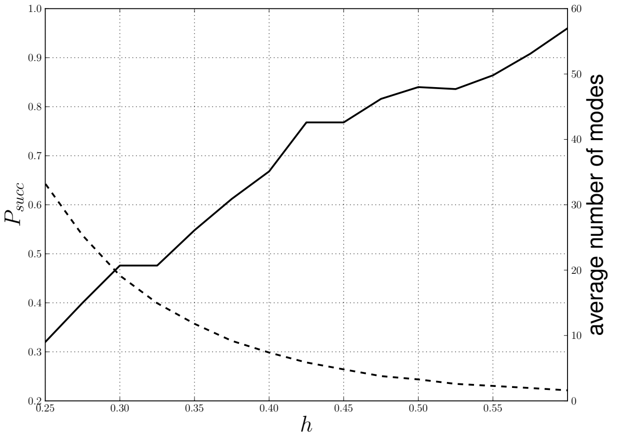


Figure 3.5: Success probability (solid curve) and number of modes (dashed curve) as a function of h in the first tests with $d = 3$ and $n = 1000$.

The function evaluation counts in the tests corresponding to Figure 3.5 are plotted in Figure 3.6 as a function of the bandwidth parameter h. Here a function evaluation means one evaluation of the objective function, gradient and Hessian at each iteration of the trust region Newton method. The mixed-derivative evaluation means the evaluation of the tangent vector (3.31). The conclusion from these results is that the number of expensive function evaluations is small compared to that of exhaustive mode finding for a wide range of bandwidths h. An exhaustive mode finding is typically done by starting a local iteration from each of the n means of the Gaussian functions, as in [26]. This takes roughly $c \cdot n$ objective function and derivative evaluations, where $c > 1$. For $n = 1000$, this number is much larger than the numbers shown in Figure 3.6.

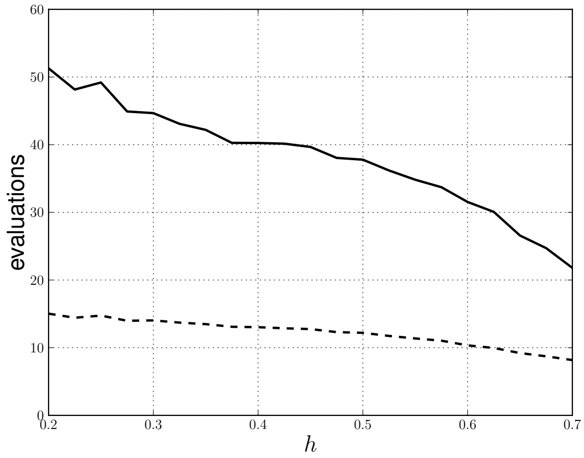


Figure 3.6: The average numbers of combined function/gradient/Hessian evaluations (solid line) and mixed-derivative evaluations (dashed line) as a function of h with $d = 3$ and $n = 1000$.

In the second set of tests conducted in [II], the algorithm is applied to Gaussian kernel densities estimating simulated data from Gaussian mixtures of the form (3.23). Ten random Gaussian mixtures p are generated with random weights $w_i \in [0.25, 0.75]$, means $\boldsymbol{\mu}_i \in [-2, 2] \times [-2, 2]$ and standard deviations $\sigma_i \in [0.4, 0.7]$ sampled from the uniform distribution. The weights w_i are normalized to one. For each of the Gaussian mixtures, 5000 points are sampled and a kernel density estimate is constructed from these samples. The bandwidh h is computed by approximately minimizing the MISE (2.18) between the true density and the kernel density estimate.

The results of the above tests are listed in Table 3.3. Only for two of the ten Gaussian mixtures, the algorithm was not able to find the global mode of the kernel density in any of the three attempts with different sample sets. The failures were identified to occur when the true density had a narrow peak.

p	Modes in p	$\hat{p}_h$	Modes in $\hat{p}_h$	Success	p	Modes in p	$\hat{p}_h$	Modes in $\hat{p}_h$	Success
1	4	1	40	yes	6	2	1	37	no
		2	35	yes			2	36	no
		3	40	yes			3	40	no
2	2	1	24	yes	7	1	1	23	no
		2	21	yes			2	23	yes
		3	19	yes			3	23	yes
3	3	1	36	yes	8	3	1	35	no
		2	25	yes			2	41	no
		3	31	yes			3	36	no
4	3	1	37	yes	9	1	1	26	yes
		2	27	no			2	24	yes
		3	35	yes			3	27	yes
5	2	1	23	yes	10	4	1	57	yes
		2	26	yes			2	51	yes
		3	24	yes			3	49	yes

Table 3.3: Convergence of the algorithm to global modes of kernel density estimates and the number of modes in the Gaussian mixtures p and kernel densities $\hat{p}_h$.

Chapter 4

Identification of curvilinear structures from noisy data

This chapter is based on **Paper III**. In this paper, a ridge curve of a Gaussian kernel density is formulated as a solution to a differential equation. A predictor-corrector method is developed for tracing the set of such solution curves. The idea of the method is to find the modes of the density and then trace ridge curves passing through the modes. The trust region Newton method described in Chapter 3 is utilized for these tasks. Differently to this method that yields only an unordered set of ridge points, the method described in this chapter traces a ridge curve by moving along it. This yields a parametrization of such a curve. The ridge curve tracing method is combined with a generalization of the statistical model described in Chapter 2 to multiple filamentary structures and a kernel bandwidth estimator. The resulting algorithm can be used for extraction of such structures from scattered point sets with background clutter. Numerical experiments show superior performance of the algorithm compared to ridge tracing algorithms based on the mean-shift method. The algorithm also implements a disciplined way to determine endpoints of ridge curves. This is essential when there are multiple curvilinear structures in the input data.

4.1 Relation to earlier research

There exists a vast amount of literature on identifying curvilinear structures from point sets. In a pioneering work, Hastie and Stuetzle [61] introduce the notion of a *self-consistent* principal curve that does not self-intersect and has a finite length within any bounded subset of $\mathbb{R}^d$. They define a principal curve point as the conditional expectation of the data distribution on a hyperplane orthogonal to the curve. For a computational implementation, they use a scatterplot smoother and develop an iterative algorithm that alternates between projection and conditional expectation steps. An alternative formulation in a rigorous statistical framework is given by Tibshirani [120]. Banfield and Raftery [6] and Stanford and Raftery [115] combine the Hastie and Stuetzle algorithm with a clustering algorithm and apply their algorithms to finding multiple curvilinear patterns from satellite images and earthquake catalogs.

Kégl et al. [78] consider principal curves with bounded length. They show that imposing the length constraint guarantees existence of such a curve when the data distribution has finite second moments. Furthermore, their definition allows relaxing the assumption that the principal curve does not intersect itself. Based on this definition, they propose the *polygonal line* algorithm. Using nearest-neighbour (Voronoi) partitions, the algorithm constructs a piecewise linear curve fitted to the data. Kégl and Krzyzak [77] further extend the algorithm to multiple principal curves with intersections.

Unfortunately, the above methods have serious limitations. Those based on the Hastie and Stuetzle definition can only fit a single nonintersecting curve to the data unless combined with a clustering algorithm or a special-purpose scatterplot smoother. On the other hand, the polygonal line algorithm requires a large number of user-supplied parameters and heuristic rules when multiple curves with intersections are fitted to the data. Furthermore, the methods that fit a curve based on a global goodness of fit criterion are sensitive to the choice of the initial guess.

Recently, the need to address the shortcomings of the earlier methods has given rise to local principal curve definitions. The idea is to construct such a curve in a "bottom-up" fashion based on local conditions rather than a global criterion. This approach offers a large degree of flexibility as neither the number of principal curves is restricted nor any parametric assumptions need to be made on the data distribution. Einbeck et al. [44] propose a heuristic method that iteratively traces a principal curve defined in terms of locally weighted mean and covariance estimates. This approach can be viewed as a simplified version of the *principal oriented point* method by Delicado [37] and Delicado and Huerta [38].

Using density ridges is among the most recent approaches for extracting curves from point patterns. The idea of defining a principal curve as a ridge curve of a density is in fact closely related to the local principal curve definitions proposed in [37], [38] and [44]. More insight on this aspect will be given in Chapter 5. There we give a ridge point of a Gaussian kernel density an interpretation in terms of locally weighted mean and covariance, where the weights are Gaussian functions.

Most (if not all) of the research on ridge-based curve estimation has been focused on projecting point sets onto density ridges. Among the most notable examples is the work by Ozertem and Erdogmus [96]. However, the only algorithms that proceed along a ridge curve to obtain a parametrization appear to be those developed by Baş [8] and Baş et al. [10, 11]. In these papers, a mean shift-based algorithm is applied to extraction of filamentary and tree-like structures from biomedical images.

The aim of **Paper III** presented in this chapter is to fill the apparent gaps in the algorithmic development of ridge-based methods. The proposed algorithm uses the highly efficient Newton method described in Chapter 3 to locate the modes of a kernel density estimate obtained from the data. A predictor-corrector method utilizing the Newton-based ridge projection method is developed for tracing ridge curves originating from the modes.

In addition, the algorithm utilizes the theory of ridge curves in order to guarantee proper termination at their endpoints. The earlier local principal curve or ridge-based methods do not implement any rigorous rules for this purpose. Finally, successful numerical experiments are conducted with an automatic kernel bandwidth estimator. Combining such an estimator with a ridge tracing method has not been extensively studied so far (see Grillenzoni [55] for some earlier results).

4.2 The filament model

In [III], density ridges are incorporated into a generative model describing multiple filamentary structures with background clutter. In this sense, the model is a generalization of the model of [49] presented in Section 2.3.

The sample points represented by an observed variable $\boldsymbol{X}$ are assumed to be generated in a random process. The type of a sample point is given by the random variable

$$T = \begin{cases} 1, & \text{if the sample belongs to a filament,} \\ 0, & \text{if the sample is background clutter} \end{cases}$$

having probabilities

$$P(T = 1) = \rho \quad \text{and} \quad P(T = 0) = 1 - \rho \tag{4.1}$$

with some $\rho \in]0, 1]$.

When a sample drawn from $\boldsymbol{X}$ is background clutter (i.e. when $T = 0$), we assume that it is uniformly distributed in some compact domain $\Omega \subset \mathbb{R}^d$. That is,

$$\boldsymbol{X} \mid (T = 0) \sim \mathcal{U}(\Omega). \tag{4.2}$$

On the other hand, when the sample belongs to some of the n filaments (i.e. when $T = 1$), we assume that it is obtained from some of the n generating functions with noise. The generating functions $\{\boldsymbol{f}_i\}_{i=1}^n : \mathcal{D}_i \to \Omega$ are continuous mappings from some compact and connected domains $\mathcal{D}_i \subset \mathbb{R}$. When $T = 1$, the outcome of the random variable $\boldsymbol{X}$ depends on three random variables: I, Θ and $\boldsymbol{\varepsilon}$. The random variable I with domain $\{1, 2, \ldots, n\}$

specifies which filament the sample belongs into, and the random variable Θ gives coordinate along the specified filament. In addition, we assume that the sample is generated with additive noise represented by a random variable $\boldsymbol{\varepsilon}$.

An example of a point set drawn from the model is shown in Figure 4.1.

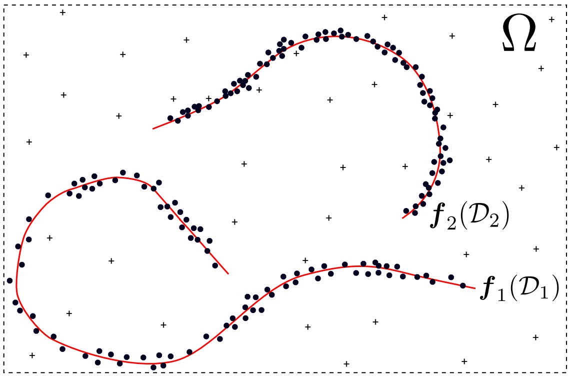

Ω
$\boldsymbol{f}_2(\mathcal{D}_2)$
$\boldsymbol{f}_1(\mathcal{D}_1)$

Figure 4.1: Filaments parametrized by two generating functions $\boldsymbol{f}_1 : \mathcal{D}_1 \to \Omega$ and $\boldsymbol{f}_2 : \mathcal{D}_2 \to \Omega$ with $\mathcal{D}_1 \subset \mathbb{R}$, $\mathcal{D}_2 \subset \mathbb{R}$ and $\Omega \subset \mathbb{R}^2$, noisy samples and background clutter.

We assume that the random variables I and $\boldsymbol{\varepsilon}$ are distributed according to

$$P(I = i) = w_i \quad \text{and} \quad \boldsymbol{\varepsilon} \sim \mathcal{N}_d(0, \sigma^2) \tag{4.3}$$

with $\boldsymbol{w} > \mathbf{0}$ such that $\sum_{i=1}^{n} w_i = 1$ and with $\mathcal{N}_d(0, \sigma^2)$ denoting a d-variate normal distribution with zero mean and covariance $\sigma^2 \boldsymbol{I}$. Given $i \in \{1, 2, \dots, n\}$, the conditional variable $\Theta \mid (I = i)$ is assumed to follow some distribution defined in the domain $\mathcal{D}_i$.

The above assumptions yield the conditional random variable

$$\boldsymbol{X} \mid (T = 1, I = i, \Theta = \theta) = \boldsymbol{f}_i(\theta) + \boldsymbol{\varepsilon} \tag{4.4}$$

having the density

$$p_{\boldsymbol{X}}(\boldsymbol{x} \mid T = 1, I = i, \Theta = \theta) = \frac{1}{(\sqrt{2\pi}\sigma)^d} \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{f}_i(\theta)\|^2}{2\sigma^2}\right). \tag{4.5}$$

By applying the relation between the joint and conditional densities we obtain

$$p_{\boldsymbol{X},T,I,\Theta}(\boldsymbol{x}, 1, i, \theta) = p_{\boldsymbol{X}}(\boldsymbol{x} \mid T = 1, I = i, \Theta = \theta) p_\Theta(\theta \mid I = i) P(I = i) P(T = 1)$$

and

$$p_{\boldsymbol{X},T,I,\Theta}(\boldsymbol{x}, 0, i, \theta) = p_{\boldsymbol{X}}(\boldsymbol{x} \mid T = 0) P(T = 0).$$

Summing the joint density $p_{\boldsymbol{X},T,I,\Theta}(\boldsymbol{x}, t, i, \theta)$ over the domains of the discrete random variables T and I and integrating over the domain of the continuous variable Θ together with equations (4.1)–(4.5) then yields the marginal density. Analogously to the densities defined by equations (2.12) and (2.13), this density is given by

$$p_{\boldsymbol{X}}(\boldsymbol{x}) = \frac{\rho}{(\sqrt{2\pi}\sigma)^d} \sum_{i=1}^{n} w_i \int\limits_{\mathcal{D}_i} \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{f}_i(\theta)\|^2}{2\sigma^2}\right) p_{\Theta}(\theta \mid I = i) d\theta + \frac{1-\rho}{V(\Omega)}, \quad (4.6)$$

where $V(\Omega)$ denotes the volume of the domain Ω. In particular, from this we observe that the ridges of $p_{\boldsymbol{X}}$ are invariant with respect to uniformly distributed background clutter.

Assuming that the samples are drawn from the above model, the aim is to estimate the image sets

$$\boldsymbol{F}_i = \{\boldsymbol{f}_i(\theta) \mid \theta \in \mathcal{D}_i\}, \quad i = 1, 2, \ldots, n$$

representing the filaments. The approach proposed in [III] uses ridge curves of the marginal density (4.6) for this purpose. While this density cannot be used directly without prior knowledge on the generating functions or model parameters, it can be estimated by using Gaussian kernels.

Based on the above ideas, the method developed in [III] proceeds in two stages. First, the method employs a bandwidth estimator for determining the optimal kernel bandwidth matrix $\boldsymbol{H}$ from the samples. Then the method obtains estimates for the sets $\boldsymbol{F}_i$ by tracing the connected components of the *ε-separated* ridge curve set

$$\mathcal{R}_{\hat{p}_{\boldsymbol{H}},\varepsilon} = \mathcal{R}_{\hat{p}_{\boldsymbol{H}}} \cap \{\boldsymbol{x} \in \mathbb{R}^d \mid \hat{p}_{\boldsymbol{H}}(\boldsymbol{x}) > \varepsilon\}$$

of the kernel density estimate $\hat{p}_{\boldsymbol{H}}$. The purpose of the user-specified threshold parameter ε is to exclude low-density areas that are not likely to contain significant features in the data.

4.3 Properties of ridge curves

In the presence of multiple filaments in the data, a key problem is how to deal with intersections. In such a case, the density estimated from the data is also expected to have multiple ridge-like structures. Consequently, a ridge tracing algorithm needs to implement a set of rules to ensure proper termination at the endpoints of ridge curves.

In order to address the above issue, we first note that ridge curves belong to a more general set of *critical curves* that we define as follows.

Definition 4.3.1 *Let $p \in C^{\infty}(\mathbb{R}^d, \mathbb{R})$ and let $\{\boldsymbol{v}_j\}_{j=1}^{d} : \mathbb{R}^d \to \mathbb{R}^d$ denote the eigenvectors of $\nabla^2 p$ corresponding to the eigenvalues $\lambda_1(\cdot) \geq \lambda_2(\cdot) \geq \cdots \geq \lambda_d(\cdot)$. The* set of critical curves *of p of index $i \in \{1, 2, \ldots, d\}$ is*

$$\mathcal{C}_p^i = \{\boldsymbol{x} \in \mathbb{R}^d \mid \nabla p(\boldsymbol{x})^T \boldsymbol{v}_j(\boldsymbol{x}) = 0 \textit{ and } \lambda_j(\boldsymbol{x}) \neq \lambda_i(\boldsymbol{x}) \textit{ for all } j \neq i\}.$$

A ridge curve belongs to a set of critical curves of index one with the additional condition (2.4b). To illustrate this fact, the critical and ridge curve sets of a Gaussian kernel density estimate obtained from a point set are plotted in Figure 4.2 taken from [III].

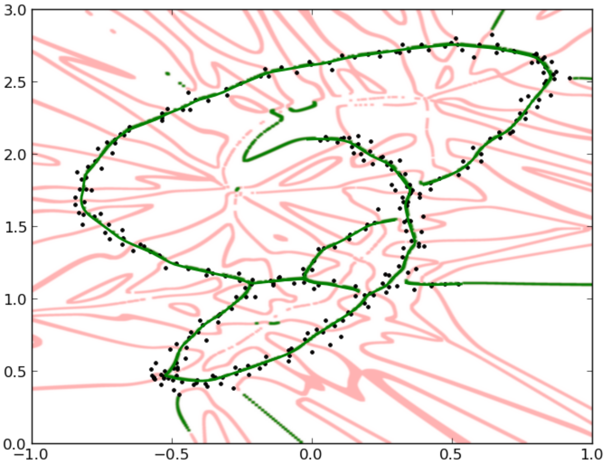


Figure 4.2: Critical and ridge curves of a bivariate function. The set of critical curves are plotted in red and its subset, the set of ridge curves is plotted in green.

In the following, we recall the main results on the relation between critical and ridge curves from Damon [36] and Miller [89]. Motivated by applications in digital image processing, they give a rigorous analysis for such curves of C^∞-functions in a differential geometric framework.

Based on Definition 4.3.1, the following characterizations for different types of critical curve points are given in [36] and [89].

Definition 4.3.2 *Let $p \in C^\infty(\mathbb{R}^d, \mathbb{R})$ and let $\lambda_1(\cdot) \geq \lambda_2(\cdot) \geq \cdots \geq \lambda_d(\cdot)$ denote the eigenvalues of the Hessian $\nabla^2 p$. If $\boldsymbol{x} \in \mathcal{C}_p^i$ for some index i, then $\boldsymbol{x}$ is a*

(i) ridge point *of p if $i = 1$ and $\lambda_2(\boldsymbol{x}) < 0$.*

(ii) valley point *of p if $i = d$ and $\lambda_{d-1}(\boldsymbol{x}) > 0$.*

(iii) r-connector point *of p if $i = 1$ and $\lambda_2(\boldsymbol{x}) > 0$.*

(iv) v-connector point *of p if $i = d$ and $\lambda_{d-1}(\boldsymbol{x}) < 0$.*

(v) m-connector point *of p if $i > 1$ and $i < d$.*

Generalizing the earlier results by Damon [36] for bivariate functions, Miller [89] shows that the one-dimensional ridge set of a C^∞-function of any dimension *generically* defines a set of smooth curves. Here a generic property means that if a function p does not satisfy this property, then any arbitrarily small perturbation of p measured in the L_2-norm does.

One of the main results of [89] is that the following properties hold generically for C^{∞}-functions, of which the Gaussian kernel density estimate is a special case. For such a density, a generic property is satisfied for almost all data point configurations except isolated special cases. For a rigorous definition of genericity, we refer to [36] and [89].

Theorem 4.3.1 *For $p \in C^{\infty}(\mathbb{R}^d, \mathbb{R})$, the following properties are generically satisfied.*

(i) The set $\mathcal{C}_p = \bigcup_{i=1}^{d} \mathcal{C}_p^i$ consists of a discrete (i.e. finite or countably infinite) set of C^{∞}-curves. The curves in $\mathcal{R}_p^1$, which is a subset of $\mathcal{C}_p^1$, may have endpoints.

(ii) The curves $\mathcal{C}_p^i$ intersect orthogonally at stationary points of p where the Hessian $\nabla^2 p$ has distinct eigenvalues. There are no other intersection points between curves $\mathcal{C}_p^i$ having different indices.

(iii) The curves in $\mathcal{R}_p^1$ do not intersect at any point and they have no self-intersections.

(iv) A connected component curve of $\mathcal{R}_p^1$ can have an endpoint $\boldsymbol{x}$ only when $\lambda_1(\boldsymbol{x}) = \lambda_2(\boldsymbol{x})$ or $\lambda_2(\boldsymbol{x}) = 0$.

(v) When a ridge curve ends at a point $\boldsymbol{x}$ such that $\lambda_2(\boldsymbol{x}) = 0$, it is smoothly continued by an r-connector curve.

(vi) When a ridge curve ends at a point $\boldsymbol{x}$ such that $\lambda_1(\boldsymbol{x}) = \lambda_2(\boldsymbol{x})$, it is smoothly continued by an m-connector curve (when $d > 2$) or a v-connector curve (when $d = 2$).

4.4 Differential equation formulation

In this section, we formulate a ridge curve of a function $p \in C^3(\mathbb{R}^d, \mathbb{R})$ as a solution to a differential equation. That is, assuming that p has a nonempty ridge curve set $\mathcal{R}_p^1$, we give the equation for the tangent of a ridge curve passing through a given point $\boldsymbol{x}_0 \in \mathcal{R}_p^1$. In what follows, we omit the superscript 1 for notational convenience.

In [III], condition (2.4a) for $\boldsymbol{x}_0$ is reformulated by using the fact that it is equivalent to the condition

$$\nabla^2 p(\boldsymbol{x}_0) \nabla p(\boldsymbol{x}_0) = \lambda_1(\boldsymbol{x}_0) \nabla p(\boldsymbol{x}_0) \tag{4.7}$$

defining a gradient extremal curve (cf. equation (2.6)).

It is then shown that this condition implicitly defines a curve $\boldsymbol{x} : [0, \infty[\to \mathbb{R}^d$ that is a solution to the initial value problem

$$\frac{d}{d\theta} \left[\boldsymbol{P}(\boldsymbol{x}(\theta)) \nabla^2 p(\boldsymbol{x}(\theta)) \frac{\nabla p(\boldsymbol{x}(\theta))}{\|\nabla p(\boldsymbol{x}(\theta))\|} \right] = \boldsymbol{0}, \quad \boldsymbol{x}(0) = \boldsymbol{x}_0. \tag{4.8}$$

Here the matrix

$$\boldsymbol{P}(\boldsymbol{x}) = \boldsymbol{I} - \frac{\nabla p(\boldsymbol{x}) \nabla p(\boldsymbol{x})^T}{\|\nabla p(\boldsymbol{x})\|^2} \tag{4.9}$$

projects a given vector onto the subspace orthogonal to the gradient $\nabla p(\boldsymbol{x})$ that is also the first Hessian eigenvector by condition (4.7).

It is shown, for instance, in [19] that the tangent vector $\boldsymbol{x}'(\theta)$ for a solution curve of problem (4.8) can be obtained as a solution to

$$\boldsymbol{P}(\boldsymbol{x}(\theta))\boldsymbol{A}(\boldsymbol{x}(\theta))\boldsymbol{x}'(\theta) = \boldsymbol{0}, \tag{4.10}$$

where

$$\boldsymbol{A}(\boldsymbol{x}) = \nabla^3 p(\boldsymbol{x}) \nabla p(\boldsymbol{x}) + [\nabla^2 p(\boldsymbol{x})]^2 - \frac{\nabla p(\boldsymbol{x})^T \nabla^2 p(\boldsymbol{x}) \nabla p(\boldsymbol{x})}{\|\nabla p(\boldsymbol{x})\|^2} \nabla^2 p(\boldsymbol{x}) \tag{4.11}$$

and the product $\nabla^3 p(\boldsymbol{x}) \nabla p(\boldsymbol{x})$ is defined according to (2.9). The matrix $\boldsymbol{A}(\boldsymbol{x})$ is in fact equivalent to the Hessian of the Lagrangian (2.7).

Whenever the matrix $\boldsymbol{P}(\boldsymbol{x}(\theta))\boldsymbol{A}(\boldsymbol{x}(\theta))$ has one-dimensional null space, the tangent vector $\boldsymbol{x}'(\theta)$ can be uniquely determined from equation (4.10) up to a scalar factor. The following result is an adaptation of a similar result for gradient extremals given in [19].

Theorem 4.4.1 ([III],[100]) *Let $p \in C^3(\mathbb{R}^d, \mathbb{R})$, $\boldsymbol{x} \in \mathbb{R}^d$ and let*

$$\boldsymbol{P}(\boldsymbol{x}) = \boldsymbol{U}(\boldsymbol{x})\boldsymbol{U}(\boldsymbol{x})^T, \quad \textit{where } \boldsymbol{U}(\boldsymbol{x}) \in \mathbb{R}^{d \times (d-1)} \tag{4.12}$$

be the eigendecomposition of the matrix $\boldsymbol{P}(\boldsymbol{x})$ defined by equation (4.9). Assume that $\nabla p(\boldsymbol{x}) \neq \boldsymbol{0}$ and that the matrix $\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{U}(\boldsymbol{x})^T \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{U}(\boldsymbol{x})$ is nonsingular, where $\boldsymbol{A}(\cdot)$ is defined by equation (4.11). Then the vector

$$\boldsymbol{u}^* = \frac{\nabla p(\boldsymbol{x})}{\|\nabla p(\boldsymbol{x})\|} - \boldsymbol{U}(\boldsymbol{x})\boldsymbol{C}(\boldsymbol{x})^{-1}\boldsymbol{b}(\boldsymbol{x}) \tag{4.13}$$

with

$$\boldsymbol{b}(\boldsymbol{x}) = \boldsymbol{U}(\boldsymbol{x})^T [\nabla^3 p(\boldsymbol{x}) \nabla p(\boldsymbol{x})] \frac{\nabla p(\boldsymbol{x})}{\|\nabla p(\boldsymbol{x})\|} \tag{4.14}$$

and its scalar multiples are the only solutions to the equation

$$\boldsymbol{P}(\boldsymbol{x})\boldsymbol{A}(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{0}. \tag{4.15}$$

It is important to note that due to the third derivative term in equation (4.11), the ridge curve tangent is not in general parallel to the first Hessian eigenvector as condition (2.4a) would suggest. In addition, the tangent vector given by equation (4.13) is not defined at a stationary point, that is when $\nabla p(\boldsymbol{x}) = \boldsymbol{0}$. Nevertheless, the following result gives a limiting direction for the tangent vector when an isolated stationary point of p belonging to $\mathcal{R}_p$

is approached along a ridge curve. By an isolated stationary point we mean a point with a neighbourhood containing no other stationary points of p. The limiting direction is parallel to the eigenvector $\boldsymbol{v}_1(\boldsymbol{x}_0)$ at the stationary point $\boldsymbol{x}_0$. This result follows from equations (4.11)–(4.14).

Theorem 4.4.2 ([III],[100]) *Let $p \in C^3(\mathbb{R}^d, \mathbb{R})$ and assume that there exists a continuous curve $\boldsymbol{x} : \mathcal{D} \to \mathbb{R}^d$ defined on some domain $\mathcal{D} \subset \mathbb{R}$ such that condition (2.4a) is satisfied for all $\boldsymbol{x}(\theta)$ with $\theta \in \mathcal{D}$. Further, assume that $\boldsymbol{x}(0) = \boldsymbol{x}_0$ for some isolated stationary point $\boldsymbol{x}_0 \in \mathcal{R}_p$. If we define*

$$\boldsymbol{u}(\theta) = \frac{\nabla p(\boldsymbol{x}(\theta))}{\|\nabla p(\boldsymbol{x}(\theta))\|} - \boldsymbol{U}(\boldsymbol{x}(\theta))\boldsymbol{C}(\boldsymbol{x}(\theta))^{-1}\boldsymbol{b}(\boldsymbol{x}(\theta)),$$

where the matrix $\boldsymbol{U}(\cdot)$ is defined according to (4.12) and the vector $\boldsymbol{b}(\cdot)$ is defined according to (4.14), then

$$\lim_{\theta \to 0} \left| \frac{\boldsymbol{u}(\theta)^T}{\|\boldsymbol{u}(\theta)\|} \boldsymbol{v}_1(\boldsymbol{x}_0) \right| = 1.$$

The matrix $\boldsymbol{C}(\cdot)$ needed for computation of the tangent vector from (4.13) may become singular in two distinct ways. The first case is covered by the following theorem that is a simplified version of the one proven in [100]. This result is a generalization of the one given in [19] for gradient extremals.

Theorem 4.4.3 ([III],[100]) *Let $p \in C^3(\mathbb{R}^d, \mathbb{R})$, $\boldsymbol{x} \in \mathbb{R}^d$ and let the matrices $\boldsymbol{U}(\boldsymbol{x})$, $\boldsymbol{A}(\boldsymbol{x})$ and $\boldsymbol{C}(\boldsymbol{x})$ be defined as in Theorem 4.4.1 and assume that the matrix $\boldsymbol{C}(\boldsymbol{x})$ is singular with eigenvalues $\lambda_i = 0$ for $i \in I$, where $I \subset \{1, 2, \dots, d-1\}$. Let*

$$\boldsymbol{C}(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{D}\boldsymbol{W}^T$$

with $\boldsymbol{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_{d-1}] \in \mathbb{R}^{(d-1)\times(d-1)}$ and the diagonal matrix $\boldsymbol{D} \in \mathbb{R}^{(d-1)\times(d-1)}$ be the eigendecomposition of $\boldsymbol{C}(\boldsymbol{x})$ and define the vector $\boldsymbol{b}(\boldsymbol{x})$ according to equation (4.14). If $\boldsymbol{w}_i^T \boldsymbol{b}(\boldsymbol{x}) \neq 0$ for some $i \in I$, then solutions to equation (4.15) with respect to $\boldsymbol{u}$ are of the form

$$\boldsymbol{u}(\boldsymbol{\beta}) = \boldsymbol{U}(\boldsymbol{x}) \sum_{i \in I} \beta_i \boldsymbol{w}_i \tag{4.16}$$

with $\boldsymbol{\beta} \in \mathbb{R}^{|I|}$.

The singular points of the matrix $\boldsymbol{C}(\cdot)$ covered by the above theorem are called *turning points*. At such points, the solutions of equation (4.15) become orthogonal to the gradient ∇p. This is because by equation (4.16) the solution vectors are spanned by the columns of the matrix $\boldsymbol{U}(\cdot)$ defined by equation (4.12). Consequently, a ridge curve has a sharp turn near a turning point, implying that it is not likely to give any meaningful estimate for the underlying structure in the input data.

On the other hand, the so-called *bifurcation points* occur when $\boldsymbol{w}_i^T \boldsymbol{b}(\boldsymbol{x}) = 0$ for all $i \in I$. As this condition involves all indices in I and all derivatives of the density p, bifurcation points are not expected to occur except in special cases where p is highly symmetric. However, turning points are common, and hence the ridge tracing algorithm described in the next section implements a stopping criterion to terminate at such points.

4.5 The algorithmic framework

In this section we describe an algorithm for constructing the ε-separated ridge curve set $\mathcal{R}_{\hat{p},\varepsilon}$ of a Gaussian kernel density estimate $\hat{p}$. The subscript $\boldsymbol{H}$ is omitted for notational convenience. The algorithm first finds the modes of $\hat{p}$ belonging to the set $\mathcal{R}_{\hat{p}}$. Then, by using these modes as starting points the algorithm constructs the set $\mathcal{R}_{\hat{p}}$ by tracing its component curves that pass through the modes by Proposition 2.1.1.

4.5.1 Main algorithms

In this subsection we describe the main algorithms `RCURVES` and `RCCOMP` for extracting the ridge curve set $\mathcal{R}_{\hat{p},\varepsilon}$. That is, generating sequences of points along the ridge curves. The algorithms presented here are simplified versions of those developed in [III]. In particular, we omit some thresholds that are needed in practical implementation because of limited numerical precision.

For computational reasons, the algorithms are applied to a scaled Gaussian kernel density estimate whose bandwidth $\boldsymbol{H}$ is an identity matrix. This is done by utilizing the fact that by using the Cholesky factorization $\boldsymbol{H} = \boldsymbol{L}\boldsymbol{L}^T$ the density $\hat{p}$ defined by equations (2.15) and (2.16) can be written as

$$\tilde{p}(\boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} K_{\boldsymbol{I}}(\boldsymbol{x} - \boldsymbol{L}^{-1}\boldsymbol{y}_i).$$

The scaled density estimate $\tilde{p}$ is related to the original one via the identity

$$\tilde{p}(\boldsymbol{L}^{-1}\boldsymbol{x}) = \sqrt{|\boldsymbol{H}|}\hat{p}(\boldsymbol{x}).$$

The transformations $\boldsymbol{L}^{-1}\boldsymbol{y}_i$ are precomputed as the first step of the algorithm. As a final step, the extracted ridge points $\tilde{\boldsymbol{x}}$ are transformed to the original coordinate system by applying the inverse transformation $\boldsymbol{x} = \boldsymbol{L}\tilde{\boldsymbol{x}}$.

The main algorithm `RCURVES` is listed as Algorithm 4.1. The algorithm proceeds in two stages. First it finds a set of non-duplicate modes of the scaled kernel density $\tilde{p}$ belonging to the ridge set $\mathcal{R}_{\tilde{p},\varepsilon}$. This is done by using

Algorithm 4.1: RCURVES (extract ridge curve set)

```
   input : point set $\boldsymbol{Y} = \{\boldsymbol{y}_i\}_{i=1}^N \subset \mathbb{R}^d$
           Gaussian kernel density estimate $\tilde{p} : \mathbb{R}^d \to \mathbb{R}$
           density threshold $\varepsilon > 0$
   output: collection of approximate ridge curves $\boldsymbol{X} \subset \mathcal{P}(\mathcal{R}_{\tilde{p},\varepsilon})$
1  $\boldsymbol{Z}^* \leftarrow \emptyset$
2  for $\boldsymbol{y} \in \boldsymbol{Y}$ do
3      Apply GTRN to $\tilde{p}$ with $r = 0$ to obtain $\boldsymbol{y}^*$ from the starting point $\boldsymbol{y}$.
4      if $\boldsymbol{y}^* \notin \boldsymbol{Z}^*$ and $\boldsymbol{y}^* \in \mathcal{R}_{\tilde{p}}^0 \cap \mathcal{R}_{\tilde{p},\varepsilon}$ then
5          $\boldsymbol{Z}^* \leftarrow \boldsymbol{Z}^* \cup \{\boldsymbol{y}^*\}$
6  $\boldsymbol{X} \leftarrow \emptyset$
7  $\boldsymbol{M} \leftarrow \emptyset$
8  for $\boldsymbol{z}^* \in \boldsymbol{Z}^*$ do
9      if $\boldsymbol{z}^* \notin \boldsymbol{M}$ then
10         $\boldsymbol{X}^+, \boldsymbol{M} \leftarrow \mathrm{RCCOMP}(\tilde{p}, \boldsymbol{M}, \boldsymbol{z}^*, 1, \varepsilon)$
11         if $\boldsymbol{x}^+_{|\boldsymbol{x}^+|-1} \neq \boldsymbol{z}^*$ then
12             $\boldsymbol{X}^-, \boldsymbol{M} \leftarrow \mathrm{RCCOMP}(\tilde{p}, \boldsymbol{M}, \boldsymbol{z}^*, -1, \varepsilon)$
13         Concatenate the sequences $\boldsymbol{X}^-$ and $\boldsymbol{X}^+$ to a sequence $\tilde{\boldsymbol{X}}$.
14         $\boldsymbol{X} \leftarrow \boldsymbol{X} \cup \tilde{\boldsymbol{X}}$.
```

the GTRN algorithm described in Chapter 3. The starting points are chosen as the sample points $\boldsymbol{y}_i$. This is because the modes of a Gaussian kernel density lie in the convex hull of the points $\boldsymbol{y}_i$ (cf. equation (3.2)). The modes obtained in this way are collected to the set $\boldsymbol{Z}^* \subset \mathcal{R}_{\tilde{p}}^0 \cap \mathcal{R}_{\tilde{p},\varepsilon}$.

In the second stage of Algorithm 4.1, the RCCOMP algorithm is called from each mode $\boldsymbol{z}^* \in \boldsymbol{Z}^*$ to obtain point sequences $\boldsymbol{X}^+$ and $\boldsymbol{X}^-$ along a ridge curve in two opposite directions. Recalling Theorem 4.4.2, these directions are along the Hessian eigenvector corresponding to the greatest eigenvalue. The test at line 9 of Algorithm 4.1 is done to prevent tracing the same ridge curve components multiple times. This is done by using the set $\boldsymbol{M}$ where the algorithm stores the modes visited during the executions of RCCOMP. The test at line 11 is needed to avoid extraction of the same ridge curve component two times when the first call of RCCOMP yields a closed loop.

Finally, the output of the RCURVES algorithm, denoted by $\boldsymbol{X} \subset \mathcal{P}(\mathcal{R}_{\tilde{p},\varepsilon})$, is a collection of point sequences along the ridge curve components, one for each connected component in the set $\mathcal{R}_{\tilde{p},\varepsilon}$.

The RCCOMP algorithm used in RCURVES is listed as Algorithm 4.2. Given a mode $\boldsymbol{x}_0^*$ of $\tilde{p}$ lying on a ridge curve (i.e. a point $\boldsymbol{x}_0^* \in \mathcal{R}_{\tilde{p}}^0 \cap \mathcal{R}_{\tilde{p},\varepsilon}$), the algorithm traces a part of a ridge curve component passing through $\boldsymbol{x}_0^*$. In the following, we denote such a component by $\mathcal{R}_{\tilde{p},\varepsilon,\boldsymbol{x}_0^*}$. The component is traced along positive or negative direction of the first Hessian eigenvector depending on the specified sign parameter $s^* \in \{-1, 1\}$.

The RCCOMP algorithm successively invokes the RCSEGMENT algorithm for extracting ridge curve segments. The RCSEGMENT algorithm is listed as Algorithm 4.3 in the next subsection. By a segment we mean a part of a ridge curve component that starts from a mode and either terminates at another mode or an endpoint of a ridge curve. The latter case occurs when the conditions defining a ridge curve become violated (cf. condition (iv) of Theorem 4.3.1) or the density $\tilde{p}$ falls below the threshold ε.

Algorithm 4.2: RCCOMP (extract a part of a ridge curve component)

```
input : Gaussian kernel density estimate $\tilde{p} : \mathbb{R}^d \to \mathbb{R}$
        visited modes $\boldsymbol{M} \subset \mathcal{R}_{\tilde{p}}^0 \cap \mathcal{R}_{\tilde{p},\varepsilon}$
        starting point $\boldsymbol{x}_0^* \in \mathcal{R}_{\tilde{p}}^0 \cap \mathcal{R}_{\tilde{p},\varepsilon}$
        sign parameter $s^* \in \{-1, 1\}$
        low probability density threshold $\varepsilon > 0$
output: subset of a ridge curve component $\boldsymbol{X} \subset \mathcal{R}_{\tilde{p},\varepsilon,\boldsymbol{x}_0^*}$
        visited modes $\boldsymbol{M} \subset \mathbb{R}^d$
$\boldsymbol{X} \leftarrow \emptyset$
$\boldsymbol{x}^* \leftarrow \boldsymbol{x}_0^*$
while not terminated do
    $\boldsymbol{X}^{**}, \boldsymbol{x}^{**}, c, s^{**} \leftarrow$ RCSEGMENT$(\tilde{p}, \boldsymbol{x}^*, s^*, \varepsilon)$
    Concatenate the sequence $\boldsymbol{X}^{**}$ with $\boldsymbol{X}$.
    $\boldsymbol{M} \leftarrow \boldsymbol{M} \cup \{\boldsymbol{x}^*\}$
    if $c = 0$ then
        if $\boldsymbol{x}^{**} \notin \boldsymbol{M}$ then
            $\boldsymbol{x}^* \leftarrow \boldsymbol{x}^{**}$
            $s^* \leftarrow s^{**}$
        else
            Terminate.
    else
        Terminate.
```

Each call of RCSEGMENT returns a sequence $\boldsymbol{X}^{**}$ along the traced ridge curve segment. These sequences are concatenated with the sequence $\boldsymbol{X}$ that is the output of RCCOMP. In addition, the previously visited mode $\boldsymbol{x}^*$ is added to the set $\boldsymbol{M}$. When RCSEGMENT terminates at a mode $\boldsymbol{x}^{**}$ (i.e. when it returns $c = 0$) and $\boldsymbol{x}^{**}$ is not in $\boldsymbol{M}$, the next ridge curve segment is traced by calling RCSEGMENT with $\boldsymbol{x}^{**}$ as starting point. The sign s^{**} returned by the previous call is used to ensure that the direction of the tracing does not change. On the other hand, when RCSEGMENT terminates at an endpoint of the ridge curve component (i.e. when $c = 1$), RCCOMP is also terminated.

Note that at line 8 the above algorithm tests a more restrictive condition than the theoretical results would require. Namely, Theorem 4.3.1 states that ridge curves generically cannot intersect each other or have self-intersections, unless such a curve forms a closed loop. Theoretically, the algorithm can thus arrive at a previously visited mode only in this kind of case (that is when $\boldsymbol{x}^{**} = \boldsymbol{x}_0^*$). However, in some cases the stopping criteria in the RCSEGMENT algorithm can fail to detect the endpoint of a ridge curve. As a result, the algorithm "jumps" from a ridge curve component to another (see Figure 4.2 on page 54 for an example where this is possible). Therefore the more restrictive test is used as a precautionary measure to prevent extracting the same ridge curves multiple times.

4.5.2 Algorithm for tracing a ridge curve segment

The RCSEGMENT algorithm invoked from RCCOMP to extract ridge curve segments is described in this subsection. The algorithm listed here as Algorithm 4.3 is a simplified version of the one developed in [III]. Based on the theory given in Section 4.4, the algorithm implements a predictor-corrector method for tracing a ridge curve.

Predictor-corrector algorithm

The RCSEGMENT algorithm generates a sequence of points $\boldsymbol{X} = (\boldsymbol{x}_0, \boldsymbol{x}_1, \dots) \subset \mathcal{R}_{\tilde{p},\varepsilon,\boldsymbol{x}_0}$ along a ridge curve segment passing through $\boldsymbol{x}_0$. At each iteration, the algorithm takes a predictor step

$$\tilde{\boldsymbol{x}}_k = \boldsymbol{x}_k + \tau_k s_k \boldsymbol{u}_k$$

along the normalized solution curve tangent $\boldsymbol{u}_k$ with step size $\tau_k > 0$ and sign parameter $s_k \in \{-1, 1\}$. The tangent vector $\boldsymbol{u}_k$ at $\boldsymbol{x} = \boldsymbol{x}_k$ is computed from equation (4.13) and normalized. A detailed description of the rules for choosing the step size τ_k is given in [III]. For the predictor estimate $\tilde{\boldsymbol{x}}_k$, the algorithm tests the stopping criteria (4.18) given in the following subsection.

The sign s_k is used to ensure that the iteration moves forward along the ridge curve. At the first iteration $k = 0$, it is chosen as the user-supplied parameter $s_0 \in \{-1, 1\}$. For the subsequent iterations $k = 1, 2, \dots$, it is chosen according to

$$s_k = \begin{cases} 1, & \text{if } s_{k-1}\boldsymbol{u}_{k-1}^T\boldsymbol{u}_k > 0, \\ -1, & \text{otherwise.} \end{cases} \tag{4.17}$$

After the predictor step, a corrector step is applied to project the predictor estimate $\tilde{\boldsymbol{x}}_k$ back to the ridge curve. For this purpose, the algorithm uses the GTRN algorithm described in Chapter 3 with ridge dimension $r = 1$.

Algorithm 4.3: RCSEGMENT (extract ridge curve segment)

```
input : Gaussian kernel density estimate $\tilde{p} : \mathbb{R}^d \to \mathbb{R}$
        starting point $\boldsymbol{x}_0 \in \mathcal{R}_{\tilde{p}}^0 \cap \mathcal{R}_{\tilde{p},\varepsilon}$
        initial sign parameter $s_0 \in \{-1, 1\}$
        initial step size $\tau_0 > 0$
        low probability density threshold $\varepsilon > 0$
output: points $\boldsymbol{X} = (\boldsymbol{x}_0, \boldsymbol{x}_1, \dots) \subset \mathcal{R}_{\tilde{p},\varepsilon,\boldsymbol{x}_0}$ on a ridge curve segment
        stopping criterion type $c \in \{0, 1\}$
        (0=mode, 1=low density or iteration has left a ridge curve)
        Returned when terminated at a mode:
        the mode $\boldsymbol{x}^* \in \mathcal{R}_{\tilde{p},\varepsilon,\boldsymbol{x}_0}$
        the current sign parameter $s^* \in \{-1, 1\}$
$\boldsymbol{X} \leftarrow (\boldsymbol{x}_0)$
$\boldsymbol{u}_0 \leftarrow \boldsymbol{v}_1(\boldsymbol{x}_0)$
for $k = 0, 1, \dots$ do
    if $\tilde{p}(\boldsymbol{x}_k) < \varepsilon$ or $\frac{\lambda_1(\boldsymbol{x}_k)}{\lambda_2(\boldsymbol{x}_k)} > 1 - \varepsilon_e$ then Terminate with $c = 1$.
    if $k > 0$ then
        Obtain $\boldsymbol{u}_k$ from equation (4.13).
        if $\boldsymbol{u}_k^T \boldsymbol{v}_1(\boldsymbol{x}_k) < 1 - \varepsilon_a$ then Terminate with $c = 1$.
        $s_k \leftarrow \mathrm{sgn}(s_{k-1} \boldsymbol{u}_{k-1}^T \boldsymbol{u}_k)$
        if conditions (4.19) are satisfied then
            Apply GTRN to $\tilde{p}$ with $r = 0$ to obtain $\boldsymbol{x}^*$ from starting
            point $(\boldsymbol{x}_k + \boldsymbol{x}_{k-1})/2$.
            $\boldsymbol{X} \leftarrow (\boldsymbol{x}_0, \dots, \boldsymbol{x}_{k-1}, \boldsymbol{x}^*)$
            $s^* \leftarrow \mathrm{sgn}(s_{k-1} \boldsymbol{u}_{k-1}^T \boldsymbol{v}_1(\boldsymbol{x}^*))$
            Terminate with $c = 0$.
    if conditions (4.18) are satisfied then
        Increase $\tau_k$.
    else
        Decrease $\tau_k$ until conditions (4.18) are satisfied. If $\tau_k$ is small,
        terminate with $c = 1$.
    $\tilde{\boldsymbol{x}}_k \leftarrow \boldsymbol{x}_k + \tau_k s_k \boldsymbol{u}_k$
    Apply GTRN to $\tilde{p}$ with $r = 1$ to obtain $\boldsymbol{x}_{k+1}$ from starting point $\tilde{\boldsymbol{x}}_k$.
    $\tau_{k+1} \leftarrow \tau_k$
    $\boldsymbol{X} \leftarrow (\boldsymbol{x}_0, \boldsymbol{x}_1, \dots, \boldsymbol{x}_k, \boldsymbol{x}_{k+1})$
```

Step size adaptation and stopping criteria

After each predictor step, the algorithm tests the conditions

$$\frac{|\nabla\tilde{p}(\tilde{\boldsymbol{x}}_k)^T\boldsymbol{v}_1(\tilde{\boldsymbol{x}}_k)|}{\|\nabla\tilde{p}(\tilde{\boldsymbol{x}}_k)\|} > 1-\varepsilon_r \quad \text{and} \quad \lambda_2(\tilde{\boldsymbol{x}}_k) < 0 \tag{4.18}$$

with some small $\varepsilon_r \in]0,1[$, where the first condition corresponds to (2.4a) and the second condition corresponds to (2.4b). If either one of conditions (4.18) is not satisfied, the algorithm decreases τ_k. This is repeated until conditions (4.18) are satisfied or τ_k is below some small value (e.g. 10^{-6}). The latter case indicates that the iterate $\boldsymbol{x}_k$ lies near an endpoint of a ridge curve, and the algorithm terminates with $c = 1$. On the other hand, when conditions (4.18) are satisfied for $\tilde{\boldsymbol{x}}_k$ for the first attempt, then the step size τ_k is increased.

In addition to the predictor conditions (4.18), the RCSEGMENT algorithm uses the stopping criteria

$$\tilde{p}(\boldsymbol{x}_k) < \varepsilon, \quad \frac{\lambda_1(\boldsymbol{x}_k)}{\lambda_2(\boldsymbol{x}_k)} > 1-\varepsilon_e \quad \text{and} \quad \boldsymbol{u}_k^T\boldsymbol{v}_1(\boldsymbol{x}_k) < 1-\varepsilon_a$$

to detect if the density falls below the threshold ε or the iteration crosses an endpoint of the ridge curve. The second criterion with some small $\varepsilon_e \in]0,1[$ tests whether the first and second eigenvalue of the Hessian become identical (cf. condition (2.4c) and condition (iv) of Theorem 4.3.1). The third criterion, where $\varepsilon_a \in]0,1[$, measures the cosine of the angle between the ridge curve tangent $\boldsymbol{u}_k$ and the eigenvector $\boldsymbol{v}_1(\boldsymbol{x}_k)$. When this quantity is below the threshold $1-\varepsilon_a$, these directions deviate significantly from each other. By Theorem 4.4.3, this indicates that the iteration is approaching a turning point.

The last stopping criterion tests whether the iteration has crossed a mode of the density estimate $\tilde{p}$. This is detected by testing if the gradient changes direction along the ridge curve. Before crossing a mode, the curve tangent is approximately parallel to the gradient and after crossing the mode approximately parallel to the negative gradient (cf. Theorem 4.4.2). For $k > 0$, this yields the criteria

$$s_{k-1}\frac{\nabla\tilde{p}(\boldsymbol{x}_{k-1})^T\boldsymbol{u}_{k-1}}{\|\nabla\tilde{p}(\boldsymbol{x}_{k-1})\|} > 1-\varepsilon_c \quad \text{and} \quad s_k\frac{\nabla\tilde{p}(\boldsymbol{x}_k)^T\boldsymbol{u}_k}{\|\nabla\tilde{p}(\boldsymbol{x}_k)\|} < -(1-\varepsilon_c) \tag{4.19}$$

with some small $\varepsilon_c \in]0,1[$. When these criteria are met, the algorithm terminates and returns the mode $\boldsymbol{x}^*$ found by the GTRN algorithm started from the midpoint of the current iterate $\boldsymbol{x}_k$ and the previous iterate $\boldsymbol{x}_{k-1}$. In analogy with equation (4.17), the algorithm also determines the sign parameter s^* at the mode $\boldsymbol{x}^*$ by comparing the directions of the previous tangent vector $s_{k-1}\boldsymbol{u}_{k-1}$ and the eigenvector $\boldsymbol{v}_1(\boldsymbol{x}^*)$.

4.6 Numerical experiments

In this section we demonstrate the applicability of the RCURVES algorithm (Algorithms 4.1–4.3) to extraction of curvilinear structures from noisy data. Illustrative examples on a representative selection of synthetic as well as two observational datasets from seismology and cosmology will be given. Numerical test results will also be provided to assess the computational performance of the algorithm.

4.6.1 Datasets and test setup

Three different types of datasets are used in [III]: synthetic datasets where the points are samples from a set of generating curves and an earthquake and a galaxy dataset.

Earthquake epicenters are typically clustered around seismic *faults*. Due to this fact, identification of faults from earthquake catalogs is a potential application for the proposed method. This is illustrated in [III] with a seismological dataset. The dataset covers the New Madrid seismic region extending from Illinois to Arkansas. It contains the locations of observed earthquakes in this region from 1974 to 2013 with magnitude one and above, consisting of 6157 samples.

In cosmology, galaxies typically form clusters and filamentary structures. A well-known example of this is the *Shapley Supercluster* [40]. The dataset consists of the angular sky coordinates and recession velocities of 4215 galaxies in the supercluster. As a preprocessing step, the original data is transformed in [III] into three-dimensional Cartesian coordinates by utilizing the fact that recession velocities of galaxies are proportional to their radial distances [40].

In all tests, the kernel bandwidth matrix $\boldsymbol{H}$ is estimated by using the Hpi estimator implemented in the ks package for the R software [41]. The density estimator is chosen to be optimal for the first derivatives (see Subsection 2.5.2 for the rationale of this choice). As discussed in Chapter 3, the RCURVES algorithm is applied to the logarithm of the kernel density. A detailed description of the test setup is given in [III].

4.6.2 Illustrative examples

A key feature of the RCURVES algorithm is its ability to separate different components of the ridge curve set and properly terminate at endpoints of ridge curves. An example of the latter is shown in Figure 4.3. This figure shows the "Jakob" dataset from Verbeek et al. [127] and the kernel density ridge curves obtained by the RCURVES algorithm. Each component of the ridge curve set is plotted with a different color.

It is also worthwhile to note that in the tests conducted in [III] with synthetic datasets, the algorithm is able to accurately extract each generating curve and give a correct number of generating curves in all test cases where the curves do not have intersections. However, when they do, the ridge curves are split into two parts at each intersection point due to the non-intersecting nature of ridge curves (cf. Theorem 4.3.1 and Figure 4.2).

Examples of point sets with multiple generating curves are shown in Figure 4.4. The algorithm is also applicable for extracting curves with closed loops, as shown in Figure 2.3a. The ridge curves extracted from the New Madrid and Shapley datasets are shown in Figures 4.5 and 4.6.

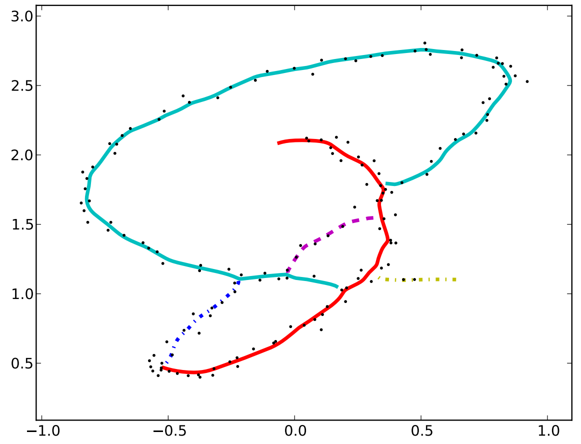
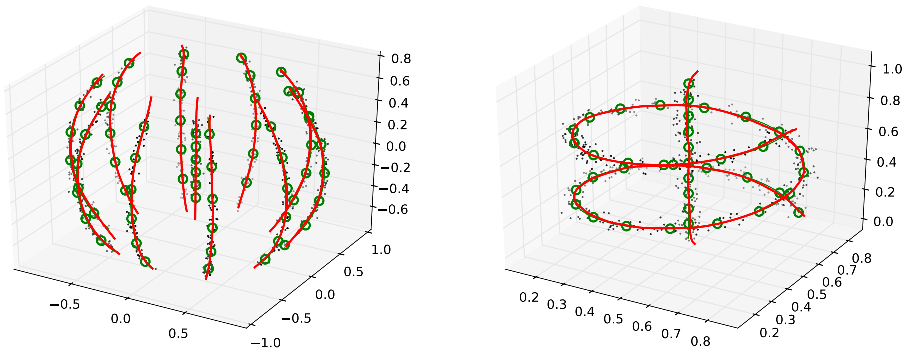


Figure 4.3: Kernel density ridge curves of the Jakob dataset.



(a) Arcs

(b) Spiral3d

Figure 4.4: Estimates from kernel density ridges (red) and known generating curves (green lines and circles) of three-dimensional synthetic datasets.

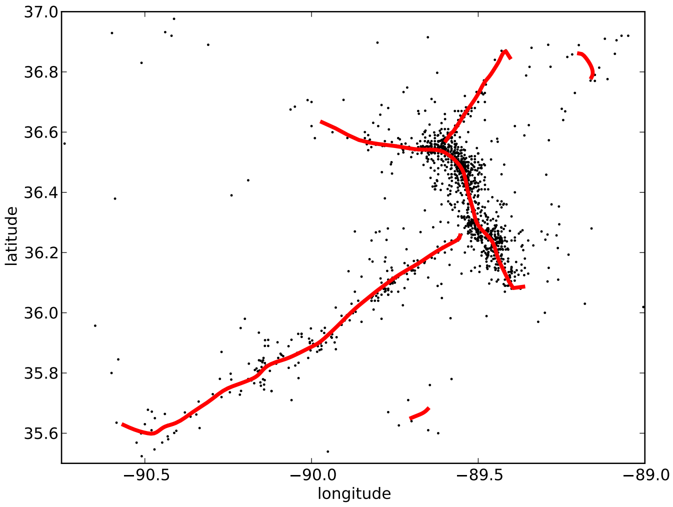
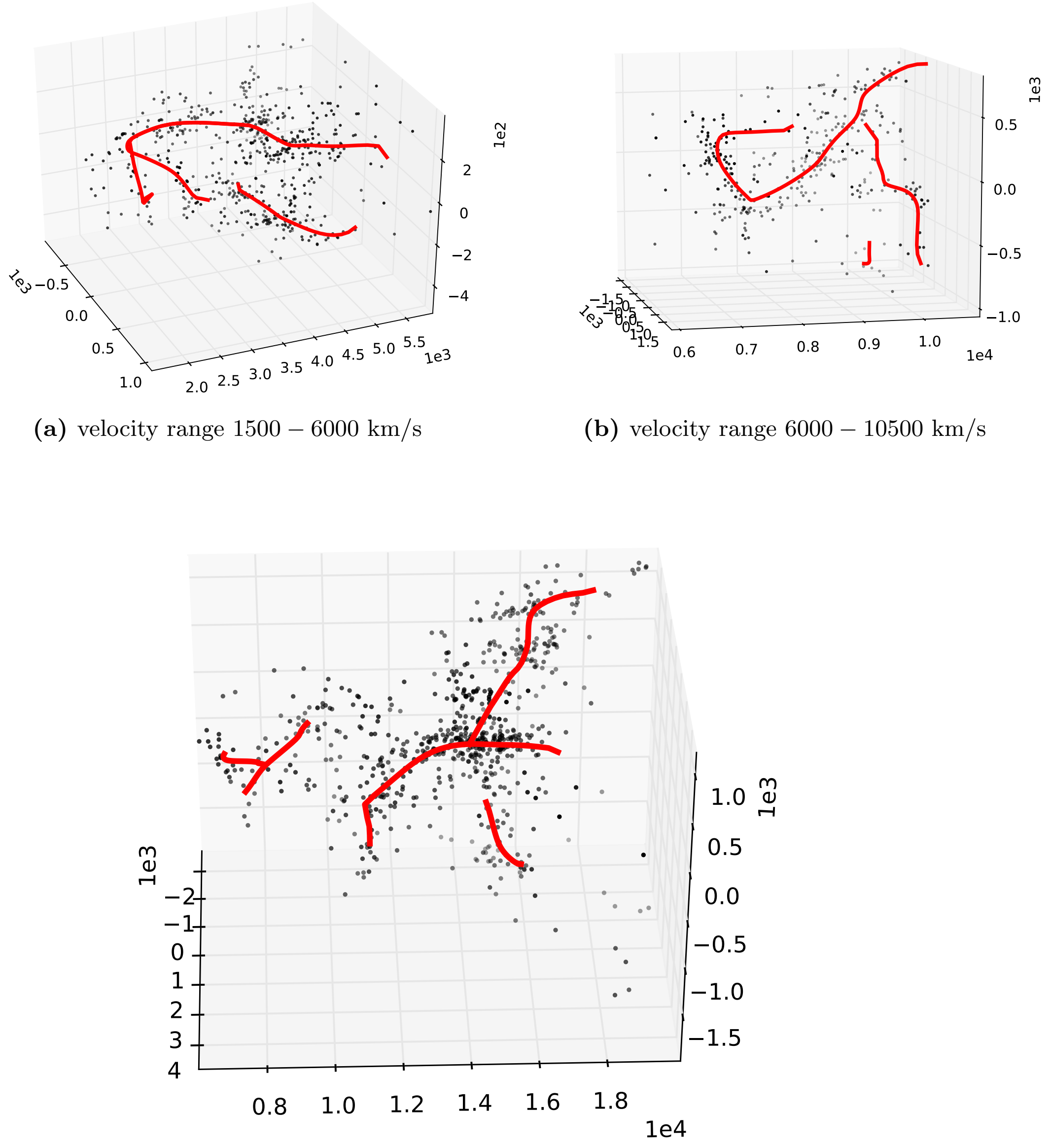


Figure 4.5: Faults extracted from the New Madrid dataset.

	RCURVES-MS			RCURVES		
Dataset	$\#f$	$\#\nabla^3$	time	$\#f$	$\#\nabla^3$	time
Arcs	208 705	601	11.968	18 156	601	2.132
Circle	107 078	263	2.294	9 261	263	0.297
DistortedHalfCircle	133 724	137	2.843	9 138	137	0.284
DistortedSShape	140 888	241	3.003	9 333	241	0.298
HalfCircle	176 654	199	3.751	9 417	199	0.297
Jakob	30 726	685	0.366	9 691	643	0.195
Ladder	328 113	2 002	28.360	34 258	2 016	6.506
New Madrid	365 772	372	63.388	61 349	348	14.282
Shapley (Figure 4.6a)	29 099	135	0.574	6 746	132	0.272
Shapley (Figure 4.6b)	24 756	135	0.308	4 069	132	0.122
Shapley (Figure 4.6c)	122 095	202	9.973	26 271	176	3.944
Spiral	278 515	472	10.137	15 683	472	0.826
Spiral3d	216 673	518	7.559	11 854	518	0.917
Zigzag	108 493	188	2.311	8 721	188	0.276

Table 4.1: Function evaluations, third derivative evaluations and wall clock times used by the RCURVES-MS and RCURVES algorithms for kernel density estimates obtained from the test datasets.



(a) velocity range 1500 – 6000 km/s

(b) velocity range 6000 – 10500 km/s

(c) velocity range 6000 – 20000 km/s

Figure 4.6: Filaments extracted from the Shapley dataset in three-dimensional Cartesian coordinates.

4.6.3 Performance evaluation

Numerical experiments are done in [III] to assess the performance of the RCURVES algorithm. A Fortran 95 implementation of the algorithm is compared to a variant named as RCURVES-MS. In this variant, the Newton-based mode finding and ridge projection methods are replaced with the mean shift method and the SCMS method, respectively. The RCURVES-MS algorithm is performance-wise comparable to the algorithms proposed by Baş [8] and Baş et al. [10, 11]. However, those algorithms do not implement any rigorously derived formulae for tangent vector calculations or stopping criteria based on the theory of ridge curves.

The conclusion made in [III] is that using the rapidly converging Newton method instead of the mean shift-based methods gives a decisive performance advantage. This is shown in Table 4.1 taken from [III]. Here the objective function and derivative evaluations up to second order are denoted by $\#f$. The more expensive third derivative evaluations needed for computing the tangent vector are denoted by $\#\nabla^3$. The measured computation times are wall-clock times for running the RCURVES algorithm (time used for kernel bandwidth estimation is not included).

The long computation times used by the RCURVES-MS algorithm are explained by the large number of function evaluations. This, in its turn, mostly results from very slow convergence of the mean shift method during the mode finding step. The modes of the densities lie on ridges, and thus they have highly elongated peaks. This is another manifestation of the poor convergence rates discussed in Section 3.1 and empirically observed from Figure 3.2b. The observations also strengthen the claim that the GTRN method is particularly well-suited for finding modes of highly curved densities.

Chapter 5

Nonlinear principal component analysis

Principal component analysis (PCA) is a ubiquitous tool for identifying the main sources of variation from multivariate data. The basic idea of the method is to use an orthogonal transformation to identify linear subspaces in which the data has maximal variance. However, as a linear method it cannot adequately describe complex nonlinear shapes.

Based on **Paper IV**, this chapter deals with *kernel density PCA* (KDPCA) that is a nonlinear generalization of PCA. The key idea is to define the principal components of a point set in terms of an m-dimensional ridge set of its Gaussian kernel density. Using the coordinate system induced by a nested collection of ridge sets of dimension $r = 0, 1, \ldots, m$, a key result is that the first m principal component coordinates of the data points can be obtained one by one by successively projecting the points onto lower-dimensional ridge sets until $r = 0$. A projection path in a curvilinear coordinate system is defined as a solution to a differential equation of the form (3.6). In addition, KDPCA is extended to time series analysis by adopting the notion of *phase space* of a time series from the linear *singular spectrum analysis* (SSA).

Estimation of principal component coordinates from kernel density ridges necessitates the use of advanced algorithms. To this end, the Newton-based ridge projection algorithm described in Chapter 3 is combined with a predictor-corrector algorithm for tracing curves that project points onto lower-dimensional ridge sets. In addition, the ridge curve tracing algorithm described in Chapter 4 is applied in the nonlinear SSA to parametrization of phase space representations of time series.

Finally, KDPCA and its SSA-based extension are applied to climate data. It is demonstrated that KDPCA is able to describe highly nonlinear structure of a climate model output, giving an accurate low-dimensional representation. The SSA-based extension is applied to reconstruction of a periodic pattern from an atmospheric time series, whose phase space representation forms a closed loop. In both applications, KDPCA and the SSA-based extension are shown to give a significant improvement over their linear counterparts.

5.1 Relation to earlier research

In this section we give a literature review on the linear PCA and its nonlinear extensions. Particular emphasis will be given to this method and the so-called local PCA methods, as they form the basis of the method developed in [IV].

5.1.1 Linear PCA

Since its introduction by Pearson [97], principal component analysis (PCA) has become the standard tool for dimensionality reduction and identifying the main sources of variation from multivariate data. This method has appeared in numerous application areas with different names such as *empirical orthogonal functions* (EOF) in climate analysis [130], *proper orthogonal decomposition* (POD) in fluid mechanics [13] and the *Karhunen-Loève transform* (KLT) in the theory of stochastic processes [83]. In the following, we give a brief overview of PCA based on [69].

The linear PCA attempts to capture the variability of a given data

$$\boldsymbol{Y} = [\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad \cdots \quad \boldsymbol{y}_n]^T \in \mathbb{R}^{n \times d}$$

by transforming the data into some m-dimensional coordinate system, where $0 < m < d$, via an orthogonal transformation. The remaining $d - m$ components, that are interpreted as noise, are discarded. In this coordinate system, the axes point along directions of maximal variance.

Let us denote the mean-centered samples by

$$\tilde{\boldsymbol{y}}_i = \boldsymbol{y}_i - \hat{\boldsymbol{\mu}}, \quad \text{where } \hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{y}_i. \tag{5.1}$$

A projection of the samples $\tilde{\boldsymbol{y}}_i$ into an m-dimensional space can be obtained via the mapping

$$\boldsymbol{\theta}_i(\boldsymbol{A}) = \boldsymbol{A}^T \tilde{\boldsymbol{y}}_i,$$

where $\boldsymbol{A}$ is a $d \times m$ matrix with $0 < m < d$ and with orthonormal columns.

Conversely, for the given coordinates $\boldsymbol{\theta}_i$ in the m-dimensional space, the corresponding *reconstruction* (i.e. projection onto the hyperplane spanned by the m first principal components) of $\boldsymbol{y}_i$ in the input space is obtained as

$$\hat{\boldsymbol{y}}_i(\boldsymbol{A}) = \hat{\boldsymbol{\mu}} + \boldsymbol{A}\boldsymbol{\theta}_i.$$

With the above definitions, it can be shown that finding the matrix $\boldsymbol{A}$ that minimizes the reconstruction error is equivalent to maximizing the variance in the transformed coordinate system. That is,

$$\min_{\boldsymbol{A}\in O(d,m)} \sum_{i=1}^{n} \|\hat{\boldsymbol{y}}_i(\boldsymbol{A}) - \hat{\boldsymbol{\mu}} - \tilde{\boldsymbol{y}}_i\|^2 = \max_{\boldsymbol{A}\in O(d,m)} \sum_{i=1}^{n} \|\boldsymbol{\theta}_i(\boldsymbol{A})\|^2, \tag{5.2}$$

where $O(d, m)$ denotes the set of $d\times m$ matrices having orthonormal columns. Furthermore, any i-th principal component corresponds to the direction of the i-th largest variance, and these directions form an orthogonal set.

The solution to the above optimization problems is the matrix $\boldsymbol{V}_m = [\boldsymbol{v}_1 \quad \boldsymbol{v}_2 \quad \cdots \quad \boldsymbol{v}_m]$, where the column vectors $\boldsymbol{v}_i$ are the (normalized) eigenvectors of the $d \times d$ sample *covariance* matrix

$$\hat{\boldsymbol{\Sigma}}_{\boldsymbol{Y}} = \frac{1}{n-1} \sum_{i=1}^{n} (\boldsymbol{y}_i - \hat{\boldsymbol{\mu}})(\boldsymbol{y}_i - \hat{\boldsymbol{\mu}})^T \tag{5.3}$$

corresponding to the m largest eigenvalues. Thus, projection of the mean-centered sample set $\tilde{\boldsymbol{Y}}$ onto the m-dimensional subspace corresponding to the directions of largest variance is given by

$$\boldsymbol{\Theta} = \tilde{\boldsymbol{Y}} \boldsymbol{V}_m.$$

In statistical literature, the coordinates $\boldsymbol{\Theta} \in \mathbb{R}^{n\times m}$ obtained in this way are called principal component *scores*.

5.1.2 Kernel- and neural network-based approaches

The *kernel PCA* (KPCA) developed by Schölkopf et al. [109] is among the most well-known nonlinear extensions of PCA. The idea is to map the data into a high-dimensional *feature space* by using a kernel function. The rationale behind this approach is that with an appropriate choice of kernel function, the data is well-described by linear principal components in the high-dimensional feature space. Rather than directly applying PCA in the feature space, the method utilizes properties of kernel functions, which leads to an $n \times n$ eigenvalue problem where n is the number of data points.

Unfortunately, KPCA requires a careful choice of a kernel function, which depends on the input data. As the method is based on an artificially chosen kernel function rather than a statistical model, the output might not reflect the actual structure of the data. Another limitation is that the method produces only principal component coordinates and not a reconstruction in the input space.

Nonlinear PCA methods based on *neural networks* have gained popularity especially in chemical engineering and climate analysis. A neural network is essentially a layer of functions for performing a series of mappings. The outcome of such mappings is a regression curve and a mapping between the input points and the curve. Typically the shape of the curve is determined by weights assigned to the individual functions. The optimal weights for a given data can be found by minimizing a cost function (e.g. a sum of squared residuals). For a comprehensive review of neural network-based PCA methods, see Scholz et al. [108].

The method by Kramer [74] designed for applications in chemical engineering is among the earliest neural network-based nonlinear PCA (NLPCA) methods. Kirby and Miranda [72] develop a neural network method that is capable of fitting closed curves. Monahan [90] demonstrates applicability of NLPCA to climate analysis. Other applications to climate data and some improvements are given by Hsieh [66] and Hsieh and Hamilton [67]. Based on NLPCA, they also develop a nonlinear variant of singular spectrum analysis (SSA) that is a PCA-based method for time series data.

Neural networks have the advantage of being able to represent very complex functions. Furthermore, they provide both a low-dimensional coordinate representation as well as a reconstruction. However, as pointed out by Christiansen [32], such methods are sensitive to overfitting, which necessitates the use of additional penalty parameters. As a result, the fitted curve or surface might not reflect the actual structure of the data. In addition, Newbigging et al. [93] and Ross [105] point out that the principal component coordinates do not have an arc length parametrization, which may introduce a significant bias. Another issue is the increasing complexity and the additional degrees of freedom in a neural network when fitting higher-dimensional surfaces. Some higher-dimensional extensions are described in [90] and [105]. A more recent approach is the *hierarchical* NLPCA [108] that, similarly to PCA, is constrained to produce ordered principal components according to their explained variance.

5.1.3 Local PCA methods

The so-called *local PCA* methods constitute another important research area. The crucial difference to the above approaches is that these methods construct the principal components in a ”bottom-up” fashion. Rather than minimizing a global goodness of fit criterion as in NLPCA, these methods operate directly in the input space by using local structure of the data. Thus, the difficult global minimization of an auxiliary cost function with a potentially large number of suboptimal solutions is avoided.

Kambhatla and Leen [71] propose a method that divides the input data into disjoint partitions. The method then carries out principal component analysis by computing the mean (5.1) and covariance (5.3) locally in each partition. Successive clustering and principal component projection steps are carried out until the improvement of the reconstruction error (5.2) in each partition is below a given threshold. The clustering is done with respect to locally computed principal component hyperplanes rather than mean vectors. Thus, the method can be viewed as a generalization of the well-known k-means clustering [62].

Einbeck et al. [44] give an alternative definition for a local principal component. In their approach, the localization of the mean vector (5.1) and covariance matrix (5.3) is done via weighting by a Gaussian kernel (2.16) with a diagonal bandwidth matrix $\boldsymbol{H} = h^2\boldsymbol{I}$. This approach eliminates the need to use a separate clustering algorithm to partition the data, though it requires choosing the kernel bandwidth. Based on this idea, the authors propose a heuristic algorithm for tracing the first principal component (i.e. a principal curve). However, they do not extend this method to higher-dimensional principal surfaces.

Unfortunately, the local PCA methods are not directly applicable when one desires to find a global coordinate system. Aligning the local principal component coordinates into a global coordinate system is a nontrivial task. One well-known approach to solving this problem is the *local tangent space alignment* (LTSA) method developed by Zhang and Zha [139]. Assuming that the data is sampled from a smooth manifold that does not intersect itself, the method first computes the nearest neighbours of each data point. Then the method obtains an approximate tangent space of the manifold at each point via local PCA restricted to nearest neighbours. As a final step, the local principal component coordinates are aligned into a global coordinate system by solving an eigenvalue problem. The main shortcoming of LTSA is that the low-dimensional coordinates tend to become severely distorted when the input data is noisy or the underlying manifold has high curvature [138]. Another limitation is that the method only produces low-dimensional coordinates but not a reconstruction in the input space.

5.1.4 The proposed method

In the following section, we describe the kernel density PCA (KDPCA) developed in [IV]. The basic idea of the method is to construct a Gaussian kernel density from the data and estimate the principal components from its ridge sets. The method is essentially a local one, as the ridge definition is based on pointwise conditions, but it also constructs a global coordinate system by exploiting the structure of ridge sets.

The main features of the KDPCA method are listed below.

- KDPCA produces a set of m first principal components ordered according to their significance. Differently to the linear PCA, they are obtained from an m-dimensional ridge set that can describe nonlinear structure.

- When the kernel bandwidth is parametrized as $\boldsymbol{H} = h^2\boldsymbol{I}$, the parameter h has an intuitive interpretation as a scale parameter. As h approaches infinity, we obtain the linear PCA as a special case.
- KDPCA produces low-dimensional coordinates as well as a reconstruction in the input space. Most nonlinear dimensionality reduction methods are not capable of doing the latter. The reconstruction ability is a desired feature, for instance, in climate analysis [105].
- KDPCA does not involve solution of any auxiliary problem. The principal components are solely determined by the structure of ridge sets. Connectivity of ridge sets can be guaranteed under mild assumptions. This, in its turn, guarantees existence of a well-defined global coordinate system.
- The nonlinear principal component scores are obtained by tracing projection curves defined by a differential equation. Such curves are parametrized by arc length, which avoids the bias problem of NLPCA.
- The ridge-based approach is also applicable when the data is concentrated around a closed curve. Namely, the `RCURVES` algorithm described in Chapter 4 can be applied to obtain a consistent parametrization and automatically detect if the curve is closed, which NLPCA is not capable of. An example of this will be given in Section 5.3.

5.2 Nonlinear kernel density PCA

In this section, we provide the necessary mathematical theory for estimation of nonlinear principal components from a ridge set of a Gaussian kernel density. We then describe an algorithm for estimating the corresponding principal component scores. Finally, we give a brief overview of the SSA method for time series analysis.

5.2.1 Properties of ridge sets

We begin with the following result that shows the connection between the ridge set of a normal density and the linear principal components. This result follows trivially from Proposition 2.1.2 and the fact that the logarithm of a normal density with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ is a quadratic function having gradient and Hessian given by equation (3.5).

Proposition 5.2.1 *Let p be a d-variate normal density with mean $\boldsymbol{\mu}$ and symmetric positive definite covariance matrix $\boldsymbol{\Sigma}$. Denote the eigenvalues of $\boldsymbol{\Sigma}$ by $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$ and the corresponding eigenvectors by $\{\boldsymbol{v}_i\}_{i=1}^d$. Then for any $0 \leq r < d$ such that $\lambda_1 > \lambda_2 > \cdots > \lambda_{r+1}$ we have*

$$\mathcal{R}_p^r = \mathcal{R}_{\log p}^r = \begin{cases} \{\boldsymbol{\mu}\}, & r = 0, \\ \{\boldsymbol{\mu}\} + \operatorname{span}(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_r), & r = 1, 2, \ldots, d-1. \end{cases}$$

More generally, the m-dimensional ridge set of a Gaussian kernel density is related to an m-dimensional principal component hyperplane via its gradient and Hessian. Here we use a function $\hat{p}_h$ to denote a Gaussian kernel density with bandwidth matrix $\boldsymbol{H} = h^2\boldsymbol{I}$. The following result can be obtained from a straightforward calculation by using equation (3.4). It shows that an m-dimensional ridge point lies on an m-dimensional principal component hyperplane. This hyperplane is determined by a weighted mean and the eigenvectors of a weighted sample covariance matrix, where the weights are Gaussian functions.

Theorem 5.2.1 ([IV]) *Let $\hat{p}_h : \mathbb{R}^d \to \mathbb{R}$ be a Gaussian kernel density, let $0 < r < d$ and denote the eigenvectors of $\nabla^2 \log \hat{p}_h(\cdot)$ corresponding to the r greatest eigenvalues by $\{\boldsymbol{v}_i(\cdot)\}_{i=1}^r$. Define*

$$\begin{aligned}\tilde{\boldsymbol{\mu}}(\boldsymbol{x}) &= \sum_{i=1}^n c_i(\boldsymbol{x})\boldsymbol{y}_i,\\ \tilde{\boldsymbol{\Sigma}}(\boldsymbol{x}) &= \sum_{i=1}^n c_i(\boldsymbol{x})[\boldsymbol{y}_i - \tilde{\boldsymbol{\mu}}(\boldsymbol{x})][\boldsymbol{y}_i - \tilde{\boldsymbol{\mu}}(\boldsymbol{x})]^T,\end{aligned}$$

where

$$c_i(\boldsymbol{x}) = \frac{\exp\left(-\dfrac{\|\boldsymbol{x}-\boldsymbol{y}_i\|^2}{2h^2}\right)}{\displaystyle\sum_{j=1}^n \exp\left(-\dfrac{\|\boldsymbol{x}-\boldsymbol{y}_j\|^2}{2h^2}\right)}, \quad i = 1, 2, \dots, n.$$

Assume that the eigenvalues of $\nabla^2 \log \hat{p}_h(\boldsymbol{x})$ satisfy the condition $\lambda_1(\boldsymbol{x}) > \lambda_2(\boldsymbol{x}) > \cdots > \lambda_{r+1}(\boldsymbol{x})$. Then

$$\nabla \log \hat{p}_h(\boldsymbol{x})^T \boldsymbol{v}_i(\boldsymbol{x}) = 0 \quad \textit{for all } i > r$$

if and only if

$$\boldsymbol{x} - \tilde{\boldsymbol{\mu}}(\boldsymbol{x}) \in \operatorname{span}(\tilde{\boldsymbol{v}}_1(\boldsymbol{x}), \tilde{\boldsymbol{v}}_2(\boldsymbol{x}), \dots, \tilde{\boldsymbol{v}}_r(\boldsymbol{x})),$$

where $\{\tilde{\boldsymbol{v}}_i(\boldsymbol{x})\}_{i=1}^r$ denote the eigenvectors of $\tilde{\boldsymbol{\Sigma}}(\boldsymbol{x})$ corresponding to the r greatest eigenvalues.

The above result provides an intuitive interpretation for the bandwidth h. Namely, the weights $c_i(\boldsymbol{x})$ can be viewed as the influence of the i-th data point at a given point $\boldsymbol{x}$. More weight is given for farther points for large bandwidth h in computation of the mean and covariance estimates $\tilde{\boldsymbol{\mu}}(\boldsymbol{x})$ and $\tilde{\boldsymbol{\Sigma}}(\boldsymbol{x})$. On the other hand, for small h, the points in a small neighbourhood of $\boldsymbol{x}$ are given a large weight. Thus, h effectively determines the scale of the structure sought from the data. Another point of interest from the above result is that a ridge point of a Gaussian kernel density $\hat{p}_h$ in fact coincides with a principal point defined according to Einbeck et al. [44].

Another way of viewing the connection between linear principal components and ridge sets of a Gaussian kernel density is to analyze their asymptotic behaviour as the bandwidth h approaches infinity. Namely, it can be shown that in this case any r-dimensional ridge set converges to the r-dimensional principal component hyperplane in a given compact set. The convergence occurs with respect to a norm that is essentially the Hausdorff distance (2.14). Consequently, by letting h approach infinity, the linear PCA is achieved as a special case when desired. The following assumption does not pose an additional restriction, as it is also required by the linear PCA when the r first unique principal components are sought.

Assumption 5.2.1 *The $r+1$ greatest eigenvalues of the sample covariance matrix $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{Y}}$ defined by equation (5.3) satisfy the conditions $\lambda_1 > \lambda_2 > \cdots > \lambda_{r+1} > 0$.*

Theorem 5.2.2 ([IV]) *Let $\hat{p}_h : \mathbb{R}^d \to \mathbb{R}$ be a Gaussian kernel density, let $0 \leq r < d$ and let Assumption 5.2.1 be satisfied. Define the set*

$$S_\infty^r = \left\{ \hat{\boldsymbol{\mu}} + \sum_{i=1}^{r} \alpha_i \boldsymbol{v}_i \mid \boldsymbol{\alpha} \in \mathbb{R}^r \right\},$$

where $\hat{\boldsymbol{\mu}}$ denotes the sample mean (5.1) and $\{\boldsymbol{v}_i\}_{i=1}^r$ denote the eigenvectors of the sample covariance matrix $\hat{\boldsymbol{\Sigma}}_{\boldsymbol{Y}}$ corresponding to the eigenvalues $\{\lambda_i\}_{i=1}^r$. Then for any compact set $U \subset \mathbb{R}^d$ such that $U \cap S_\infty^r \neq \emptyset$ and $\varepsilon > 0$ there exists $h_0 > 0$ such that

$$\left.\begin{array}{l} \operatorname{dist}(\mathcal{R}_{\hat{p}_h}^r \cap U, S_\infty^r) < \varepsilon, \\ \operatorname{dist}(S_\infty^r \cap U, \mathcal{R}_{\hat{p}_h}^r) < \varepsilon \end{array}\right\} \quad \textit{for all } h \geq h_0,$$

where

$$\operatorname{dist}(S_1, S_2) = \sup_{\boldsymbol{x} \in S_1} \inf_{\boldsymbol{y} \in S_2} \|\boldsymbol{x} - \boldsymbol{y}\|.$$

5.2.2 Estimation of principal component scores from ridges

In this subsection, we provide the theoretical basis for estimating the first m nonlinear principal component scores of a given point set. The idea is to obtain the scores one by one by successively projecting the points onto lower-dimensional ridge sets of a Gaussian kernel density. The projections are done along eigenvector curves that are defined by a differential equation. The arc lengths of the curves are interpreted as the principal component scores.

For now, we assume that a given point has already been projected onto an m-dimensional ridge set of its underlying density p with some $m \leq d$.

The aim is to obtain coordinates for the point in the m-dimensional coordinate system induced by such a ridge set. For $r = 1, 2, \dots, m$, we define a projection curve $\boldsymbol{\gamma}_r : \mathbb{R} \to \mathbb{R}^d$ onto the $r - 1$ -dimensional ridge set as a solution to the initial value problem

$$\frac{d}{dt}\left[\boldsymbol{P}_r(\boldsymbol{\gamma}_r(t))\nabla \log p(\boldsymbol{\gamma}_r(t))\right] = \mathbf{0}, \quad t \geq 0, \qquad (5.4)$$

$$\boldsymbol{\gamma}_r(0) = \boldsymbol{x}_0, \quad \boldsymbol{x}_0 \in \mathcal{R}^r_{\log p} \setminus \mathcal{R}^{r-1}_{\log p},$$

where $\boldsymbol{P}_r(\cdot) = \boldsymbol{I} - \boldsymbol{v}_r(\cdot)\boldsymbol{v}_r(\cdot)^T$ and $\{\boldsymbol{v}_i(\cdot)\}_{i=1}^d$ denote the eigenvectors corresponding to the eigenvalues $\lambda_1(\cdot) \geq \lambda_2(\cdot) \geq \cdots \geq \lambda_d(\cdot)$ of $\nabla^2 \log p$.

The above definition is motivated by a special case that shows its connection to the linear PCA projection. Namely, for any d-dimensional normal density p, a ridge point $\boldsymbol{x}_0 \in \mathcal{R}^r_{\log p}$, where $1 \leq r \leq m$, can be projected onto the lower-dimensional ridge set $\mathcal{R}^{r-1}_{\log p} \subset \mathcal{R}^r_{\log p}$ by following the solution curve of (5.4). The curve is a straight line parallel to the eigenvector $\boldsymbol{v}_r$. This property follows trivially from the fact that $\log p$ is a quadratic function with gradient and Hessian given by (3.5).

Proposition 5.2.2 ([IV]) *Let p be a d-variate normal density with symmetric and positive definite covariance matrix $\boldsymbol{\Sigma}$ and let $1 \leq r \leq d$. If the eigenvalues of $\boldsymbol{\Sigma}$ satisfy the condition $\lambda_1 > \lambda_2 > \cdots > \lambda_{r+1}$, then for any solution curve $\boldsymbol{\gamma}_r$ of the initial value problem (5.4) we have*

$$\boldsymbol{\gamma}_r'(t)/\|\boldsymbol{\gamma}_r'(t)\| = \pm\boldsymbol{v}_r$$

for all $t \geq 0$. Furthermore, if the sign of $\boldsymbol{\gamma}_r'$ is chosen such that

$$\boldsymbol{\gamma}_r'(t)^T\nabla \log p(\boldsymbol{\gamma}_r(t)) > 0 \quad \textit{for all } t \geq 0,$$

then $\log p$ has a unique maximum point $\boldsymbol{x}^* \in \mathcal{R}^{r-1}_{\log p}$ along the curve $\boldsymbol{\gamma}_r$.

When the density p is not normal, obtaining an expression for the tangent vector $\boldsymbol{\gamma}_r'(t)$ is nontrivial. However, by utilizing the formula for the derivatives of eigenvectors (e.g. [86]), equation (5.4) can be rewritten as

$$\boldsymbol{A}_r(\boldsymbol{\gamma}_r(t))\boldsymbol{\gamma}_r'(t) = \mathbf{0}, \qquad (5.5)$$

where

$$\boldsymbol{A}_r(\boldsymbol{x}) = \boldsymbol{P}_r(\boldsymbol{x})\nabla^2 \log p(\boldsymbol{x}) - \boldsymbol{F}_r(\boldsymbol{x}), \qquad (5.6)$$

$$\boldsymbol{F}_r(\boldsymbol{x}) = \boldsymbol{v}_r(\boldsymbol{x})^T\nabla \log p(\boldsymbol{x})\nabla \boldsymbol{v}_r(\boldsymbol{x})^T + \boldsymbol{v}_r(\boldsymbol{x})\nabla \log p(\boldsymbol{x})^T\nabla \boldsymbol{v}_r(\boldsymbol{x}) \qquad (5.7)$$

and

$$\nabla \boldsymbol{v}_r(\boldsymbol{x}) = \left[\lambda_r(\boldsymbol{x})\boldsymbol{I} - \nabla^2 \log p(\boldsymbol{x})\right]^+ \nabla^3 \log p(\boldsymbol{x})\boldsymbol{v}_r(\boldsymbol{x}) \qquad (5.8)$$

and the operator "$^+$" denotes the Moore-Penrose pseudoinverse (e.g. [52]). It can be shown by straightforward calculation that when $r = 1$, equations (5.5)–(5.8) are in fact equivalent to equations (4.9)–(4.11) defining the tangent vector of a ridge curve.

For a general density p, projection onto the ridge set $\mathcal{R}^{r-1}_{\log p}$ can still be done by maximizing $\log p$ along the curve $\boldsymbol{\gamma}_r$, but this requires additional justification. To this end, we first give a result showing that when $\boldsymbol{\gamma}_r$ approaches a ridge point $\boldsymbol{x}^* \in \mathcal{R}^{r-1}_{\log p}$, the tangent vector $\boldsymbol{\gamma}'_r$ becomes parallel to the eigenvector $\boldsymbol{v}_r$. This property is a higher-dimensional generalization of the one stated in Theorem 4.4.2. It can be verified by inspecting the null space of the matrix $\boldsymbol{A}_m(\boldsymbol{x})$ defined by equations (5.6)–(5.8). Here we need a technical assumption that will be justified later in this subsection.

Assumption 5.2.2 *The eigenvalues of $\nabla^2 \log p$ satisfy the conditions*

(i) $\lambda_1(\boldsymbol{\gamma}_r(t)) > \lambda_2(\boldsymbol{\gamma}_r(t)) > \cdots > \lambda_{r+1}(\boldsymbol{\gamma}_r(t))$,

(ii) $\lambda_1(\boldsymbol{\gamma}_r(t)) < 0$

for all $t \geq 0$.

Proposition 5.2.3 ([IV]) *Let $1 \leq r \leq d$ and let $\boldsymbol{\gamma}'_r$ denote the normalized tangent vector of a solution curve of (5.4). If Assumption 5.2.2 is satisfied and*

$$\lim_{t \to t^*} \boldsymbol{v}_r(\boldsymbol{\gamma}_r(t))^T \nabla \log p(\boldsymbol{\gamma}_r(t)) = 0$$

for some $t^* > 0$, then

$$\lim_{t \to t^*} |\boldsymbol{\gamma}'_r(t)^T \boldsymbol{v}_r(\boldsymbol{\gamma}_r(t))| = 1.$$

Proposition 5.2.3 implies the following properties that justify seeking for a lower-dimensional ridge point by maximizing $\log p$ along the curve $\boldsymbol{\gamma}_r$.

Proposition 5.2.4 ([IV]) *If $\boldsymbol{\gamma}_r$ is a solution to (5.4) for some $1 \leq r \leq d$ and Assumption 5.2.2 is satisfied, then either $\boldsymbol{\gamma}_r(t) \in \mathcal{R}^r_{\log p} \setminus \mathcal{R}^{r-1}_{\log p}$ for all $t \geq 0$ or $\lim_{t \to t^*} \boldsymbol{\gamma}_r(t) \in \mathcal{R}^{r-1}_{\log p}$ for some $t^* > 0$. In the latter case, $\log p$ attains its local maximum along $\boldsymbol{\gamma}_r$ at the limit point $\boldsymbol{\gamma}_r(t^*)$.*

It is not formally proven in [IV] that the curve $\boldsymbol{\gamma}_r(t)$ always converges to a point in $\mathcal{R}^{r-1}_{\log p}$. Proposition 5.2.2 nevertheless suggests that this is the case when p is sufficiently close to a normal density (i.e. when h is large).

Recall that our aim is to use projection curves $\boldsymbol{\gamma}_r$ defined by equation (5.4) to obtain the first m nonlinear principal component scores of the given sample points $\boldsymbol{y}_i$. This is to be done by using the kernel density $\log \hat{p}_h$ as the objective function. Differently to the normal density in Proposition 5.2.2, this density is not guaranteed to be unimodal or have connected ridge sets.

For instance, it is clear that such a density becomes multimodal when h is too small.

Unimodality of the density and connectedness of its ridge sets are essential here. This is because as in the linear PCA, our aim is to describe the data in a global coordinate system having the mode as the origin (cf. Proposition 5.2.1). Hence, we assume the following.

Assumption 5.2.3 *Define the set*

$$U_h = \bigcup_{i=1}^{n} \mathcal{L}_i^h,$$

where

$$\mathcal{L}_i^h = \{\boldsymbol{x} \in \mathbb{R}^d \mid \log \hat{p}_h(\boldsymbol{x}) \geq \log \hat{p}_h(\boldsymbol{y}_i)\}.$$

Let $\lambda_1(\cdot) \geq \lambda_2(\cdot) \geq \cdots \geq \lambda_d(\cdot)$ denote the eigenvalues of $\nabla^2 \log \hat{p}_h$. Assume that for all $\boldsymbol{x} \in U_h$ we have

$$0 > \lambda_1(\boldsymbol{x}) > \lambda_2(\boldsymbol{x}) > \cdots > \lambda_{m+1}(\boldsymbol{x})$$

and that U_h is compact and connected.

Miller [89] gives a rigorous analysis on the structure of ridge sets of C^∞-functions. Under the above assumption, the results of [89] guarantee that the r-dimensional ridge set of the density $\log \hat{p}_h$ forms a connected manifold in the set U_h for any $1 \leq r \leq m$. Furthermore, $\log \hat{p}_h$ is unimodal in the compact set U_h. In addition, Assumption 5.2.3 implies differentiability of the Hessian eigenvectors via Theorem 2.1.2, which is essential for the definition of the initial value problem (5.4). This assumption also entails Assumption 5.2.2 when the curves $\boldsymbol{\gamma}_r$ lie in U_h.

Assumption 5.2.3 can be satisfied by choosing a sufficiently large h, though in this way the density estimate might not reflect the true density.

Theorem 5.2.3 ([IV]) *Under Assumption 5.2.1 for $r = m$, for any Gaussian kernel density $\hat{p}_h$ there exists $h_0 > 0$ such that Assumption 5.2.3 is satisfied for all $h \geq h_0$.*

The arc length of a curve $\boldsymbol{\gamma}_r$ gives the (curvilinear) distance of its starting point to the ridge set $\mathcal{R}^{r-1}_{\log \hat{p}_h}$. Assume that we have projected a given sample point $\boldsymbol{y}_i$ onto the ridge set $\mathcal{R}^{m}_{\log \hat{p}_h}$. Starting from such a point, tracing the curves $\boldsymbol{\gamma}_r$ successively for $r = m, m-1, \ldots, 1$ then yields the first m principal component scores of $\boldsymbol{y}_i$. When Assumption 5.2.3 is satisfied, imposing the conditions (cf. Proposition 5.2.2)

$$\boldsymbol{\gamma}_r'(t)^T \nabla \log \hat{p}_h(\boldsymbol{\gamma}_r(t)) > 0 \quad \text{and} \quad \|\boldsymbol{\gamma}_r'(t)\| = 1 \tag{5.9}$$

for all $r = m, m-1, \ldots, 1$ and $t \geq 0$ guarantees that the curves $\boldsymbol{\gamma}_r$ lie in the set U_h.

Denote the projection of a given sample point $\boldsymbol{y}_i$ onto the set $\mathcal{R}^m_{\log \hat{p}_h}$ as $\tilde{\boldsymbol{y}}_i$ and the starting points of the curves $\boldsymbol{\gamma}_r$ as $\boldsymbol{x}^r_0$. The first m principal component scores of $\boldsymbol{y}_i$ are then obtained recursively as

$$\theta_r = s^*_r \int\limits_0^{t^*_r} \|\boldsymbol{\gamma}'_r(t)\| dt, \tag{5.10}$$

where

$$\boldsymbol{x}^r_0 = \begin{cases} \tilde{\boldsymbol{y}}_i, & r = m, \\ \boldsymbol{\gamma}_{r+1}(t^*_{r+1}), & 1 \leq r < m \end{cases}$$

for $r = m, m-1, \dots, 1$. Here we assume that for each r there exists $t^*_r \geq 0$ such that $\boldsymbol{\gamma}_r(t^*_r) \in \mathcal{R}^{r-1}_{\log \hat{p}_h}$. The multiplier $s^*_r = \lim\limits_{t \to t^*_r -} s_r(t)$, where

$$s_r(t) = \begin{cases} 1, & \text{if } \boldsymbol{\gamma}'_r(t)^T \boldsymbol{v}_r(\boldsymbol{\gamma}_r(t)) > 0, \\ -1, & \text{otherwise,} \end{cases} \tag{5.11}$$

is introduced to ensure that the principal component score θ_r has the correct sign.

5.2.3 Algorithm for computing principal component scores

Based on the theory given in Subsection 5.2.2, we now describe the algorithm for estimating the nonlinear principal component scores

$$\boldsymbol{\Theta} = [\boldsymbol{\theta}_1 \quad \boldsymbol{\theta}_2 \quad \cdots \quad \boldsymbol{\theta}_n]^T \in \mathbb{R}^{n \times m}$$

of a given sample set

$$\boldsymbol{Y} = [\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad \cdots \quad \boldsymbol{y}_n]^T \in \mathbb{R}^{n \times d}$$

for a given $0 < m \leq d$. This amounts to first projecting the samples $\boldsymbol{y}_i$ onto the ridge set $\mathcal{R}^m_{\log \hat{p}_h}$ and then successively projecting them onto the lower-dimensional ridge sets $\mathcal{R}^r_{\log \hat{p}_h}$ by tracing the curves $\boldsymbol{\gamma}_r$ until $r = 0$.

The initial ridge projection is done by using the `GTRN` algorithm (Algorithm 3.1). The rationale behind this choice is that this algorithm produces an approximate projection onto a ridge set in a computationally efficient way. As pointed out in Subsection 3.3.3, the iteration of `GTRN` yields a well-defined projection curve under Assumption 5.2.3. This is due to the continuity of the first m Hessian eigenvectors in the set U_h by Theorem 2.1.2.

After the initial ridge projection, the remaining m projections are done by tracing the curves $\boldsymbol{\gamma}_r$ by using a predictor-corrector method. The principal component scores $\theta_{i,r}$ are obtained from a numerical approximation of the integral (5.10) along the projection curves.

Algorithm 5.1: NLPCS (nonlinear principal component scores)

```
    input : sample points $\boldsymbol{Y} = [\boldsymbol{y}_1 \quad \boldsymbol{y}_2 \quad \cdots \quad \boldsymbol{y}_n]^T \in \mathbb{R}^{n \times d}$
            Gaussian kernel density $\hat{p}_h : \mathbb{R}^d \to \mathbb{R}$
            ridge dimension $0 < m \leq d$
            step size $\tau > 0$
    output: principal component scores
            $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1 \quad \boldsymbol{\theta}_2 \quad \cdots \quad \boldsymbol{\theta}_n]^T \in \mathbb{R}^{n \times m}$
 1  $\boldsymbol{\Theta} \leftarrow \boldsymbol{0}$
 2  for $i = 1, 2, \ldots, n$ do
 3      Apply GTRN to $\log \hat{p}_h$ with ridge dimension $m$ to obtain $\boldsymbol{x}_i^*$ from
        starting point $\boldsymbol{y}_i$.
 4  for $r = m, m-1, \ldots, 1$ do
 5      for $i = 1, 2, \ldots, n$ do
 6          $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}_i^*$
 7          for $k = 0, 1, \ldots$ do
 8              Obtain the tangent $\boldsymbol{u}_k$ from (5.5) such that $\|\boldsymbol{u}_k\| = 1$.
 9              $s_k \leftarrow \mathrm{sgn}(\boldsymbol{u}_k^T \nabla \log \hat{p}_h(\boldsymbol{x}_k))$
10              if $k > 0$ then
11                  if $s_{k-1}\boldsymbol{u}_{k-1}^T \boldsymbol{u}_k s_k < 0$ then
12                      $\bar{\boldsymbol{x}} \leftarrow (\boldsymbol{x}_{k-1} + \boldsymbol{x}_k)/2$
13                      Apply GTRN to $\log \hat{p}_h$ with ridge dimension $r-1$ to
                        obtain $\boldsymbol{x}_i^*$ from starting point $\bar{\boldsymbol{x}}$.
14                      $\theta_{i,r} \leftarrow \theta_{i,r} + \|\boldsymbol{x}_i^* - \boldsymbol{x}_{k-1}\|$
15                      if $(\boldsymbol{x}_i^* - \boldsymbol{x}_{k-1})^T \boldsymbol{v}_r(\boldsymbol{x}_i^*) < 0$ then
16                          $\theta_{i,r} \leftarrow -\theta_{i,r}$
17                      Return to line 5.
18                  else
19                      $\theta_{i,r} \leftarrow \theta_{i,r} + \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|$
20              $\tilde{\boldsymbol{x}}_k \leftarrow \boldsymbol{x}_k + \tau s_k \boldsymbol{u}_k$
21              Apply GTRN to $\log \hat{p}_h$ with ridge dimension $r$ to obtain
                $\boldsymbol{x}_{k+1}$ from starting point $\tilde{\boldsymbol{x}}_k$.
```

The algorithm for estimating the principal component scores is listed as Algorithm 5.1. The first step is the initial projection onto the ridge set $\mathcal{R}^m_{\log \hat{p}_h}$. After that the algorithm carries out $m \times n$ iterations. Each iteration for $r = m, m-1, \ldots, 1$ projects each of the n sample points onto the ridge set $\mathcal{R}^{r-1}_{\log \hat{p}_h}$. The intermediate projections are stored in the variables $\{\boldsymbol{x}_i^*\}_{i=1}^n$.

In the following, we describe the steps for carrying out one ridge projection (i.e. one iteration of the loop over the index i) for a given r. The starting point $\boldsymbol{x}_0$ for $\boldsymbol{\gamma}_r$ is chosen as $\boldsymbol{x}_i^*$ representing the projection of the sample point $\boldsymbol{y}_i$ onto the set $\mathcal{R}^r_{\log \hat{p}_h}$. For a monotonously increasing sequence $\{t_k\}$ such that $\boldsymbol{\gamma}_r(t_{k^*}) \in \mathcal{R}^{r-1}_{\log \hat{p}_h}$ for some k^*, we introduce the notation $\boldsymbol{x}_k = \boldsymbol{\gamma}_r(t_k)$ for the iterates along the curve $\boldsymbol{\gamma}_r$. With this notation, an approximation to the integral in (5.10) is given by

$$\int_0^{t_r^*} \|\boldsymbol{\gamma}_r'(t)\| dt \approx \sum_{k=1}^{k^*} \|\boldsymbol{\gamma}_r(t_k) - \boldsymbol{\gamma}_r(t_{k-1})\| = \sum_{k=1}^{k^*} \|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|.$$

The algorithm uses a predictor-corrector method to generate the iterates $\boldsymbol{x}_k$. At the predictor step (line 20), the algorithm proceeds along a tangent vector $\boldsymbol{u}_k = \boldsymbol{\gamma}_r'(t_k)$ solved from equation (5.5). That is,

$$\tilde{\boldsymbol{x}}_k = \boldsymbol{x}_k + \tau s_k \boldsymbol{u}_k,$$

where $\tau > 0$ is some user-supplied step size, $\|\boldsymbol{u}_k\| = 1$ and the multiplier

$$s_k = \begin{cases} 1, & \text{if } \boldsymbol{u}_k^T \nabla \log \hat{p}_h(\boldsymbol{x}_k) > 0 \\ -1, & \text{otherwise} \end{cases}$$

is introduced to impose conditions (5.9). To project the predictor estimate $\tilde{\boldsymbol{x}}_k$ back to the ridge set $\mathcal{R}^r_{\log \hat{p}_h}$, the algorithm takes a corrector step by invoking the `GTRN` algorithm at the last line.

A stopping criterion is imposed to terminate the tracing of the curve $\boldsymbol{\gamma}_r$ when a maximum of $\log \hat{p}_h$ along $\boldsymbol{\gamma}_r$ is encountered (line 11). For $k > 0$, the condition

$$s_{k-1} \boldsymbol{u}_{k-1}^T \boldsymbol{u}_k s_k < 0$$

tests whether the gradient changes sign along the curve. When this condition is met, the algorithm projects the midpoint of the current and previous iterate onto a nearby ridge point $\boldsymbol{x}_i^* \in \mathcal{R}^{r-1}_{\log \hat{p}_h}$ by invoking the `GTRN` algorithm. At lines 15–16, the algorithm computes the sign s_r^* for the integral (5.10) by approximately testing condition (5.11). The inner iteration (i.e. iteration of the loop over the index k) is then terminated, and the point $\boldsymbol{x}_i^*$ is retained as starting point for projection onto lower-dimensional ridge set.

Tests for unimodality or connectedness of the ridge sets of $\log \hat{p}_h$ are not included in Algorithm 5.1 for conciseness. Unimodality can be tested by finding its modes as in the initial step of Algorithm 4.1. This can also be guaranteed by the approach described in Subsection 3.5.2. On the other hand, disconnectedness of ridge sets can be detected by testing if a projection curve crosses a point $\boldsymbol{x}$ where $\lambda_{r+1}(\boldsymbol{x}) = 0$ or $\lambda_i(\boldsymbol{x}) = \lambda_j(\boldsymbol{x})$ for some $i, j = 1, 2, \ldots, r+1$ such that $i \neq j$, where $\lambda_i(\cdot)$ denote the eigenvalues of $\nabla^2 \log \hat{p}_h$ [89]. When multimodality or a disconnected ridge set is detected, the algorithm can be restarted with a larger h or smaller initial ridge dimension m.

Remark 5.2.1 *When only the first principal component is sought, a more efficient approach is to use the method of Subsection 3.5.2 to guarantee unimodality (this does not necessarily guarantee connectedness of ridge sets). Then one can use Algorithms 4.2 and 4.3 to extract a sequence of line segments along the principal curve. The principal component scores can be obtained by finding the nearest line segment for each data point and computing distance along the line segments to the origin (the mode of the density).*

5.2.4 Extension to time series data

The extension of KDPCA to time series data, that we call KDSSA, is based on the singular spectrum analysis (SSA) developed by Golyandina et al. [53] and Vautard et al. [125]. In SSA, a time series is embedded into a multidimensional *phase space.* This is done by constructing a *trajectory matrix* from time-lagged copies of the time series. That is, such a matrix of a time series $\boldsymbol{x} = (x_1, x_2, \dots, x_n)$ is given by

$$\boldsymbol{Y}_{\boldsymbol{x},L} = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_L \\ x_2 & x_3 & x_4 & \cdots & x_{L+1} \\ x_3 & x_4 & x_5 & \cdots & x_{L+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n-L+1} & x_{n-L+2} & x_{n-L+3} & \cdots & x_n \end{bmatrix}, \tag{5.12}$$

where L is a user-supplied time window length.

Applying the linear PCA to the above matrix, one can obtain the principal components and the reconstructed time series by using the formulae given by Vautard et al. [125]. Generalizing this PCA-based approach, we minimize the reconstruction error

$$E(\boldsymbol{x}) = \sum_{i=1}^{n-L+1} \sum_{j=1}^{L} (\tilde{y}_{i,j} - x_{i+j-1})^2 \tag{5.13}$$

using the first m nonlinear principal components, where $m \leq L$. Here the vectors $\tilde{\boldsymbol{y}}_i$ denote the projections of the row vectors $\boldsymbol{y}_i$ of $\boldsymbol{Y}_{\boldsymbol{x},L}$ onto the m-dimensional ridge set of their Gaussian kernel density.

A straightforward calculation shows that by equating the gradient $\nabla E(\boldsymbol{x})$ to zero, we obtain the formulae

$$x_i^* = \begin{cases} \dfrac{1}{L} \displaystyle\sum_{j=1}^{L} \tilde{y}_{i-j+1,j}, & L \leq i \leq n-L+1 \\ \dfrac{1}{i} \displaystyle\sum_{j=1}^{i} \tilde{y}_{i-j+1,j}, & 1 \leq i \leq L-1 \\ \dfrac{1}{n-i+1} \displaystyle\sum_{j=i-n+L}^{L} \tilde{y}_{i-j+1,j}, & n-L+2 \leq i \leq n \end{cases} \tag{5.14}$$

for the elements of the reconstructed time series such that $E(\boldsymbol{x}^*)$ minimizes the reconstruction error (5.13).

Here the nonlinear SSA is applied to quasiperiodic time series (i.e. noisy time series having some underlying periodic pattern). The motivation is as follows. Assuming that a time series follows the model

$$X(t) = f(t) + \varepsilon(t)$$

for some periodic function f and ε representing the noise, it is reasonable to model the trajectory samples (i.e. the rows of the matrix $\boldsymbol{Y}_{\boldsymbol{x},L}$) as a point set that is concentrated around a closed curve.

5.3 Experiments with climate and time series data

In this section we give a summary of the experiments carried out in [IV]. That is, we demonstrate the applicability of KDPCA and its SSA-based extension KDSSA to climate data where highly nonlinear phenomena are common. Three potential applications are presented with illustrations.

- Estimation and analysis of principal component scores from a climate model output representing gridded sea surface temperatures. The scores are estimated by using the `NLPCS` algorithm (Algorithm 5.1).
- Reconstruction of a periodic component from an atmospheric time series. This is done by projecting its phase space representation (5.12) onto the ridge curve of its kernel density and then applying formula (5.14). The projection is done by using the `GTRN` algorithm (Algorithm 3.1).
- Parametrization of the reconstructed phase space trajectory by using the `RCURVES` algorithm (Algorithm 4.1). The parametrization is used for detection of unusually short or long periods in the above time series.

In the first example, the `RCURVES` and `NLPCS` algorithms are applied to a simulated sea surface temperature dataset obtained from the GFDL-CM2.1 climate model [39]. This dataset has been previously used by Ross et al. [106] for evaluating dimensionality reduction methods on climate data. The dataset used in [IV] consists of 6000 samples where seasonal variation has been removed by subtracting monthly mean values. For computational reasons, the high-dimensional data ($d = 10073$) is first projected onto the ten first linear principal components explaining 87.3% of the variance in the data.

Figure 5.1 shows the first principal component (i.e. principal curve) estimated from the kernel density ridge of the GFDL-CM2.1 dataset with kernel bandwidth $h = 40$. Here the dataset and the curve are projected onto the

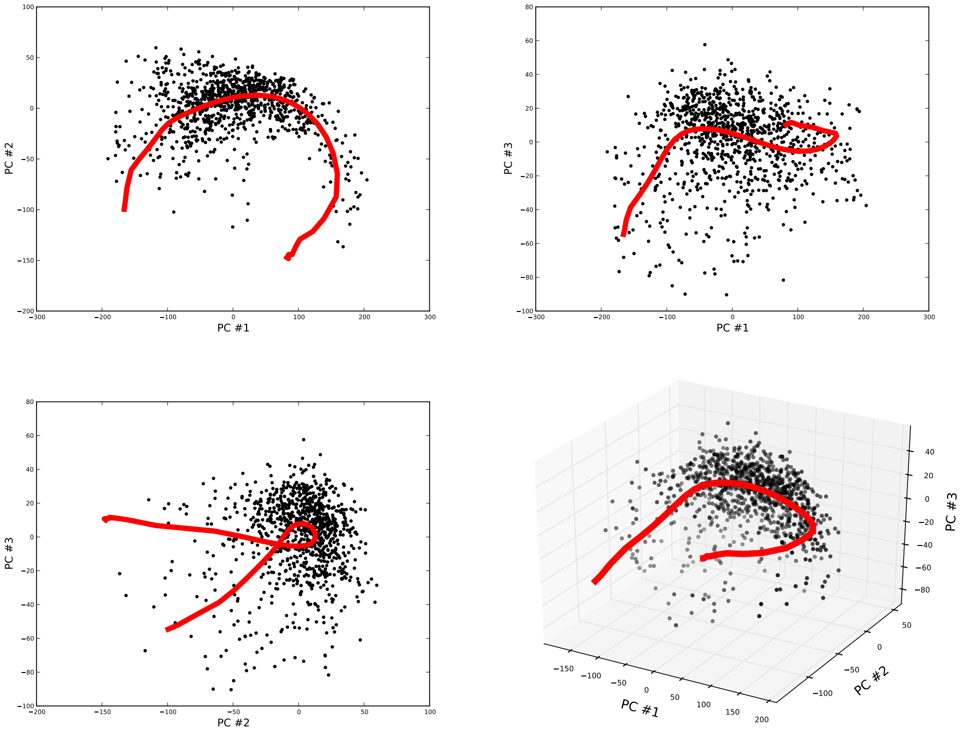


Figure 5.1: The first nonlinear principal component estimated from the GFDL-CM2.1 dataset (only a subset of the curve is drawn).

subspace spanned by the first three linear principal components. The principal curve segment has been extracted by using the `RCURVES` algorithm. Clearly, the ridge curve is able to capture the highly nonlinear shape of the data (differently to the examples shown in Chapters 2 and 4, the true generating curve is not known).

The curves shown in Figure 5.1 are plotted for illustration purposes. When doing statistical analysis, the principal component scores are of main interest. For the above dataset, the first principal component correlates with the so-called NINO3 index related to the El Niño Southern Oscillation (ENSO) phenomenon. The second one correlates with the Pacific warm water volume [105].

To illustrate the application of the `NLPCS` algorithm, the two first principal component scores obtained by this algorithm are plotted in Figure 5.2. The corresponding reconstruction with ridge dimension $r = 2$ obtained by `NLPCS` is plotted in Figure 5.3. Examples of using such reconstructions to visualize dominant ocean circulation patterns are given in [105].

Figures 5.2 and 5.3 clearly show the ability of KDPCA to capture nonlinear shapes that cannot be described by the linear PCA. That is, Figure

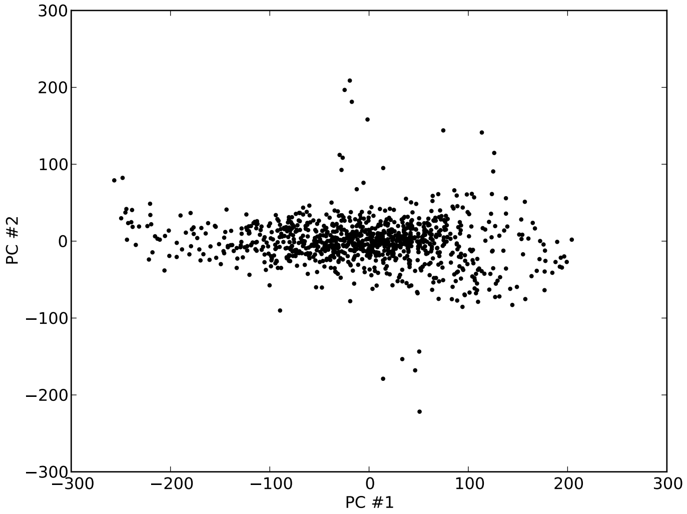


Figure 5.2: Two first nonlinear principal component scores estimated from the GFDL-CM2.1 dataset.

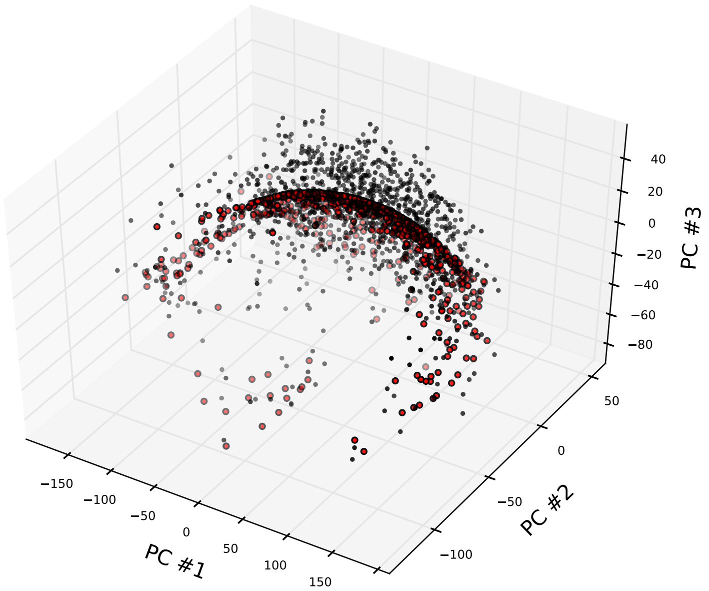


Figure 5.3: Projection of the GFDL-CM2.1 dataset onto its nonlinear principal component surface.

5.2 represents an unfolding of the nonlinear principal surface shown in Figure 5.3. Such an unfolding cannot be obtained from the linear principal component projections shown in Figure 5.1.

In the second example, the KDSSA described in Subsection 5.2.4 is applied to a wind measurement time series. The time series consists of monthly mean zonal winds constructed from balloon measurements. It represents the well-known periodic atmospheric phenomenon called quasi-biennial oscillation (QBO) in the tropical stratosphere.

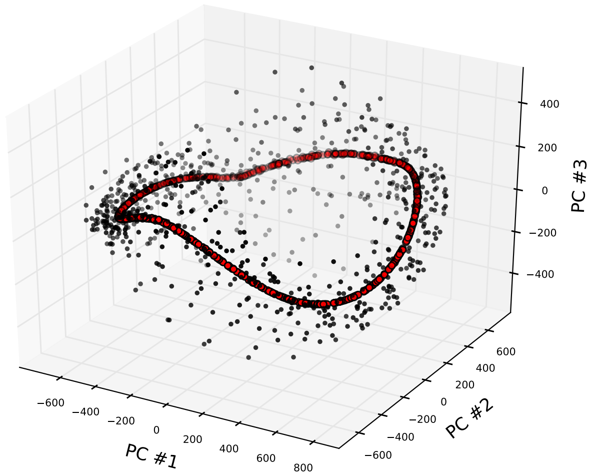


Figure 5.4: Phase space representation of the QBO time series and the reconstructed trajectory curve obtained by kernel density ridge projection.

Due to the strong periodicity of the QBO time series, its phase space representation is expected to be concentrated around a closed loop. This is indeed the case, as seen from Figure 5.4 showing the phase space representation and the reconstructed trajectory. This figure represents projections onto the first three linear principal components. The phase space representation is obtained by constructing the matrix (5.12) with $L = 18$. The reconstruction is obtained by projecting the trajectory points (i.e. the rows of the matrix (5.12)) onto the ridge curve of the Gaussian kernel density with $h = 200$ by using the `GTRN` algorithm with $r = 1$. For computational reasons, the `GTRN` algorithm is applied to a projection onto the subspace spanned by the four first linear principal components.

The estimate for the underlying periodic pattern in the QBO time series is plotted in Figure 5.5. This estimate has been obtained by applying formula (5.14) to the reconstructed trajectory points $\tilde{\boldsymbol{y}}_i$ shown in Figure 5.4. The original time series and the reconstructions obtained from the first two linear principal components of the trajectory matrix (5.12) are also plotted for comparison. The conclusion is that the reconstruction obtained by the nonlinear SSA gives the best estimate for the periodic component. The reconstruction obtained from the first linear principal component is insufficient to describe the structure of the time series. On the other hand, including the second principal component only replicates random fluctuations and not the periodic pattern. This deficiency follows from the fundamental limitation that as a linear method PCA cannot describe data that is concentrated around a closed curve.

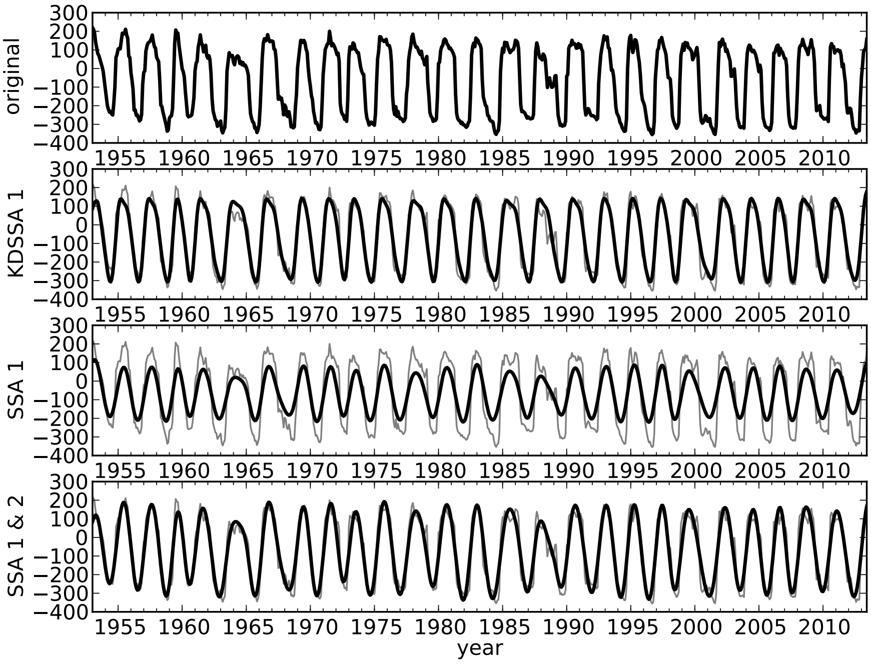


Figure 5.5: The QBO time series and the reconstructed time series obtained by using the first KDSSA component, the first linear SSA component and the two first linear SSA components combined. The original time series is plotted in gray in the lower figures.

In the last example, the `RCURVES` algorithm is applied to obtaining an approximate parametrization of the reconstructed phase space trajectory shown in Figure 5.4. The following analysis is done by following the approach by Hamilton and Hsieh [67] who apply an NLPCA-based nonlinear SSA method to the QBO time series.

By a parametrization we mean an ordered sequence of points along the trajectory curve shown in Figure 5.4. The parametrization is not unique because the curve forms a closed loop, but one can be obtained by fixing one point along the curve as the origin. Figure 5.6 shows the coordinates of the QBO time series along the parametrized trajectory curve (the t-values) normalized to the interval $[-\pi, \pi]$. At each time step, the coordinate t for the sample point is obtained by finding the closest point along the parametrized trajectory curve and then computing the distance of this point to the origin along the curve.

The parametrization can be used to analyze the lengths of individual cycles in the time series. To demonstrate this, here we also compute at each time step the expected total distance that the t-coordinate has moved along the phase space trajectory since the beginning of the observation period. By this quantity we mean the total distance along the trajectory assuming that the speed is constant during the whole observation period. The "t-anomaly" values (in the terminology of Hsieh and Hamilton [67]), that represent deviations of the actual total distances from the expected value are also plotted in Figure 5.6 with normalization to the interval $[-1, 1]$. Comparison of the t-anomaly time series with Figure 5.5 shows that rapid increases in t-anomaly values represents unusually short cycles and rapid decreases represent unusually long ones.

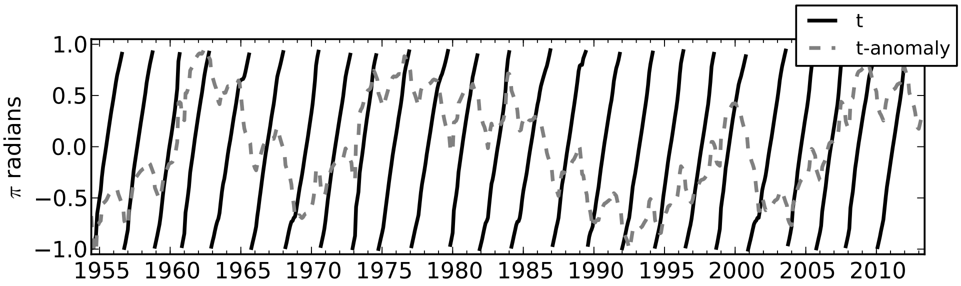


Figure 5.6: The first nonlinear principal component coordinate of the QBO time series (t) and the deviation from the expected value (t-anomaly).

Chapter 6

Graph-based dimensionality reduction

Though the KDPCA described in Chapter 5 is likely to perform well on datasets of moderate size and dimensionality, it might not be the best approach for all dimensionality reduction tasks. When the number of samples is large (say $n > 100000$) or the data is high-dimensional (say $d > 100$) without any trivial low-dimensional structure, the computational cost of evaluating the Gaussian functions becomes prohibitively high. In many applications, one is dealing with such data. A typical example is a large collection of digital images, where the data dimension is the number of pixels in the image.

In particular, when the input data is high-dimensional, a more efficient approach is to construct a low-dimensional representation based on a neighbourhood graph of the data. Efficient methods for constructing such graphs from high-dimensional data have been developed (e.g. [76]). In fact, a majority of nonlinear dimensionality reduction methods are based on the graph-based approach. Just to name a few, there are Isomap [119], Laplacian eigenmaps [12], locally linear embedding [107] and maximum variance unfolding (MVU) [131] belonging to this category. Other neighbour-based methods include stochastic neighbour embedding [63] and its variants [122, 126].

Due to the inherent computational difficulties of the kernel density PCA, the final part of this thesis based on **Paper V** is devoted to graph-based dimensionality reduction. The focus is on exploring the computational feasibility of the MVU method by Weinberger and Saul [131]. This method can be conveniently formulated as a semidefinite optimization problem. Unfortunately this comes with a high computational cost, and the development of the MVU method has been hindered by computational difficulties. Only recently, some attempts based on quadratic reformulations of the semidefinite MVU problem have been made to overcome this difficulty [75, 124]. Refining the ideas presented in these references, a reformulation framework for solving the MVU problem is presented in this chapter. State-of-the-art optimization methods are utilized in this framework, which is shown to yield drastic performance improvements compared to the standard semidefinite solvers.

6.1 Problem definition and the proposed approach

The optimization problem arising in the MVU method is a special instance of a more general graph embedding problem. Therefore we present the problem in a broader context. To this end, let $G = (V, E)$ denote an undirected graph with nodes $V = \{1, 2, \ldots, n\}$ and edges $E \subset V \times V$.

The problem of embedding the graph G into the d-dimensional space $\mathbb{R}^d$ is to assign each node i a point $\boldsymbol{y}_i \in \mathbb{R}^d$ such that distances between adjacent points are equal to the edge weights. This requirement is often too restrictive. Thus, a typical graph embedding problem is formulated as

$$\begin{aligned} &\text{find} \quad \boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n \in \mathbb{R}^d \\ &\text{s.t.} \quad c_1 D_{ij} \leq \|\boldsymbol{y}_i - \boldsymbol{y}_j\| \leq c_2 D_{ij}, \quad \{i, j\} \in E \end{aligned} \tag{6.1}$$

for the given constants $0 \leq c_1 \leq c_2 \leq 1$, edges $\{i, j\} \in E$ and edge weights $D_{ij} \geq 0$ (see [82] for a general discussion on graph embeddings).

From the viewpoint of dimensionality reduction, it is desirable to find a low-dimensional embedding that preserves the structure of the graph. As discussed in [57], [118] and [131], this can be done by maximizing pairwise distances between the points $\boldsymbol{y}_i$ under the above distance constraints. This approach is based on the intuition that, differently to the arbitrary embedding (6.1), stretching the points apart from each other is likely to "flatten" the graph and thus produce a representation lying on some low-dimensional subspace.

Here we consider graph embeddings that maximize the sum of squared pairwise point distances under distance constraints with only upper bounds. Together with a centering constraint, such an embedding problem is formulated as the quadratically constrained quadratic program (QCQP)

$$\begin{aligned} \max_{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_n \in \mathbb{R}^d} \quad & \sum_{i=1}^n \sum_{j=1}^n \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2 \\ \text{s.t.} \quad & \|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2 \leq D_{ij}^2, \quad \{i, j\} \in E, \\ & \sum_{i=1}^n \boldsymbol{y}_i = \mathbf{0} \end{aligned} \tag{6.2}$$

with some embedding dimension d. The advantage of relaxing the distance constraints is that the feasible set is convex and has a nonempty interior for all $d \geq 1$ when $D_{ij} > 0$ for all $\{i, j\} \in E$. This would not be the case with strict equalities. For instance, it is not possible to embed a general graph into a line while strictly preserving the edge weights.

For solving graph embedding problems of the form (6.2), the standard approach is to consider a semidefinite relaxation. By introducing the matrix $\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{Y}^T$ with $\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{y}_1, & \boldsymbol{y}_2, & \ldots, & \boldsymbol{y}_n \end{bmatrix}^T \in \mathbb{R}^{n \times d}$, problem (6.2) can be reformulated as the semidefinite program (SDP)

$$\begin{aligned} \max_{\boldsymbol{K} \in \mathcal{S}^n} \quad & \operatorname{tr}(\boldsymbol{K}) \\ \text{s.t.} \quad & \boldsymbol{K} \succeq 0, \\ & K_{ii} - 2K_{ij} + K_{jj} \leq D_{ij}^2, \quad \{i, j\} \in E, \\ & \sum_{i=1}^{n} \sum_{j=1}^{n} K_{ij} = 0, \end{aligned} \tag{6.3}$$

where $\mathcal{S}^n$ denotes the cone of symmetric $n \times n$ matrices.

Reformulation of the difficult convex maximization problem (6.2) as the SDP (6.3) has the advantage that the SDP is a concave maximization problem over a convex set, and thus any local maximum is a global one. Moreover, the SDP formulation eliminates the need of knowing the embedding dimension d in problem (6.2) a priori. Once the SDP has been solved, an embedding can be obtained from the eigenvectors of the matrix $\boldsymbol{K}$ corresponding to an appropriately chosen small number of the largest eigenvalues (e.g. [131]). The standard interior-point SDP solvers are applicable to the SDP (6.3) (e.g. [21] and [137]).

Unfortunately, the interior-point SDP solvers scale poorly to large problems since the SDP (6.3) has $\mathcal{O}(n^2)$ variables. Moreover, these solvers require factorization and storage of a dense $m \times m$ matrix, where m is the number of constraints in the problem [21]. Since the number of edges can be significantly larger than the number of nodes in the graph, this incurs a major computational bottleneck.

Based on [V], we describe a computationally efficient approach for solving the SDP (6.3). The approach is an adaptation of the theory of semidefinite programs and their low-rank quadratic formulations developed in [24,25,56] and [70] to problems (6.2) and (6.3). Based on these results, an incremental low-rank method for solving the SDP (6.3) is developed in [V]. The idea is to apply a NLP solver to sequence of small quadratic problems (6.2) with increasing dimension d until a solution to the SDP is obtained. Furthermore, it is shown that such a solution is globally optimal for (6.2) with dimension d that gives an upper bound for the rank of the optimal solution of (6.3). Due to zero duality gap, which is established in [V] for the SDP (6.3) and its dual under mild assumptions, solution of the primal problem yields a solution to the dual problem and vice versa. The dual problem is equivalent to determining the fastest mixing Markov process on a graph (e.g. [118]), and similar problems also appear in graph theory (e.g. [57]).

6.2 Low-rank semidefinite programming

In this section we recall the main results concerning the relation between optimal solutions of semidefinite programs and their low-rank reformulations.

6.2.1 Notation and optimality conditions

In what follows, we recall the main results by Burer and Monteiro [25] and Grippo et al. [56]. These results give necessary and sufficient conditions that a (local) solution to the QCQP

$$\begin{aligned} \min_{\boldsymbol{y}\in\mathbb{R}^{nd}} \quad & \boldsymbol{y}^T(\boldsymbol{I}_d \otimes \boldsymbol{C})\boldsymbol{y} \\ \text{s.t.} \quad & \boldsymbol{y}^T(\boldsymbol{I}_d \otimes \boldsymbol{A}_i)\boldsymbol{y} = b_i, \quad i = 1, 2, \dots, m \end{aligned} \tag{6.4}$$

arising from the change of variables $\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{Y}^T$ with

$$\boldsymbol{Y} = \begin{bmatrix} y_1 & y_{n+1} & \cdots & y_{nd-n+1} \\ y_2 & y_{n+2} & \cdots & y_{nd-n+2} \\ \vdots & \vdots & \ddots & \vdots \\ y_n & y_{2n} & \cdots & y_{nd} \end{bmatrix} \in \mathbb{R}^{n\times d} \tag{6.5}$$

such that $d \le n$ yields a solution to the standard form SDP

$$\begin{aligned} \min_{\boldsymbol{K}\in\mathcal{S}^n} \quad & \boldsymbol{C} \bullet \boldsymbol{K} \\ \text{s.t.} \quad & \boldsymbol{K} \succeq 0, \\ & \boldsymbol{A}_i \bullet \boldsymbol{K} = b_i, \quad i = 1, 2, \dots, m. \end{aligned} \tag{6.6}$$

Here the operator $\otimes$ between two matrices $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{p\times q}$ defined as

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1n}\boldsymbol{B} \\ \vdots & \ddots & \vdots \\ a_{m1}\boldsymbol{B} & \cdots & a_{nm}\boldsymbol{B} \end{bmatrix} \in \mathbb{R}^{mp\times nq}$$

denotes the *Kronecker product*. The operator $\bullet$ between two $n \times n$ matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as

$$\boldsymbol{A} \bullet \boldsymbol{B} = \operatorname{tr}(\boldsymbol{A}^T\boldsymbol{B}) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}b_{ij},$$

and $\boldsymbol{I}_d$ denotes a $d \times d$ identity matrix.

For a vector $\boldsymbol{y} \in \mathbb{R}^{nd}$ obtained by stacking the columns of the matrix $\boldsymbol{Y}$ defined by equation (6.5), we shall use the notation $\mathrm{vec}(\boldsymbol{Y})$. For a matrix $\boldsymbol{Y}$ obtained from a vector $\boldsymbol{y} \in \mathbb{R}^{nd}$ according to equation (6.5) we shall use the notation $\mathrm{mat}(\boldsymbol{y})$.

A key assumption made in [56] is that the SDP (6.6) and its dual have nonempty solution sets and the gap between the primal and dual solutions of (6.6) is zero. The dual of problem (6.6) is

$$\begin{aligned} \max_{\boldsymbol{\lambda} \in \mathbb{R}^m} \quad & \boldsymbol{b}^T \boldsymbol{\lambda} \\ \text{s.t.} \quad & \boldsymbol{C} - \sum_{i=1}^m \lambda_i \boldsymbol{A}_i \succeq 0. \end{aligned} \tag{6.7}$$

Assumption 6.2.1 *Problem (6.6) and its dual (6.7) have nonempty solution sets. In addition, if $\boldsymbol{K}^* \in \mathbb{R}^{n \times n}$ is a solution of (6.6) and $\boldsymbol{\lambda}^* \in \mathbb{R}^m$ is a solution of (6.7), then $\boldsymbol{C} \bullet \boldsymbol{K}^* = \boldsymbol{b}^T \boldsymbol{\lambda}^*$.*

Under Assumption 6.2.1, the following theorem provides a sufficient condition for a local solution of (6.4) to be the global one and also the solution of the SDP (6.6). For the KKT optimality conditions, we refer to [7].

Theorem 6.2.1 ([56]) *Under Assumption 6.2.1, if $\boldsymbol{y}^* \in \mathbb{R}^{nd}$ is a first-order KKT point of (6.4) with Lagrange multipliers $\boldsymbol{\lambda}^* \in \mathbb{R}^m$ and the condition*

$$\boldsymbol{C} + \sum_{i=1}^m \lambda_i^* \boldsymbol{A}_i \succeq 0 \tag{6.8}$$

holds, then the matrix $\boldsymbol{K}^* = \boldsymbol{Y}^* \boldsymbol{Y}^{*^T}$ with $\boldsymbol{Y}^* = \mathrm{mat}(\boldsymbol{y}^*)$ is a solution to (6.6) and $\boldsymbol{y}^*$ is a global solution to (6.4).

Conversely, the following theorem provides a necessary condition.

Theorem 6.2.2 ([56]) *Under Assumption 6.2.1, if $\boldsymbol{y}^* \in \mathbb{R}^{nd}$ is a global solution to (6.4) with Lagrange multipliers $\boldsymbol{\lambda}^* \in \mathbb{R}^m$ and the matrix $\boldsymbol{K}^* = \boldsymbol{Y}^* \boldsymbol{Y}^{*^T}$ with $\boldsymbol{Y}^* = \mathrm{mat}(\boldsymbol{y}^*)$ is a solution to (6.6), then condition (6.8) holds.*

Finally, a lower bound for d ensuring that any local solution $\boldsymbol{y}^*$ of (6.4) yields a solution $\boldsymbol{K}^* = \boldsymbol{Y}^* \boldsymbol{Y}^{*^T}$ of (6.6) is given in [25]. This holds when problem (6.6) has a nonempty solution set and

$$d \geq \bar{d} = \max \left\{ d \in \mathbb{N} \mid \frac{d(d+1)}{2} \leq m + 1 \right\}. \tag{6.9}$$

As we will see in Section 6.5, this bound is rather conservative, and optimal solutions of problem (6.3) can in practice be obtained by solving problems (6.2) with a much smaller dimension d. In fact, it turns out that in practice the required d is exactly the rank of the optimal SDP solution $\boldsymbol{K}^*$. However, giving a theoretical justification for this remains as a topic of future work.

6.2.2 Matrix formulation of the graph embedding problem

Problems (6.2) and (6.3) are not in the standard forms (6.4) and (6.6), but they can be stated in this form. For $G = (V, E)$ denoting a graph with n nodes and n_E edges, we denote an ordered sequence of the edges by

$$E = ((i_1, j_1), (i_2, j_2), \ldots, (i_{n_E}, j_{n_E})). \tag{6.10}$$

In addition, we define the matrices

$$\boldsymbol{C} = -\boldsymbol{I}_n \quad \text{and} \quad \boldsymbol{A}_k = \boldsymbol{a}_k \boldsymbol{a}_k^T, \quad k = 1, 2, \ldots, n_E, \tag{6.11}$$

where the vectors $\boldsymbol{a}_k \in \mathbb{R}^n$ are defined as

$$a_{k,l} = \begin{cases} 1, & \text{if } l = i_k, \\ -1, & \text{if } l = j_k, \\ 0, & \text{otherwise.} \end{cases} \tag{6.12}$$

We define the last constraint matrix as $\boldsymbol{A}_{n_E+1} = \mathbf{1}_n \mathbf{1}_n^T$, where $\mathbf{1}_n$ denotes a vector of ones having length n.

Finally, we set $m = n_E + 1$ and define the elements of the vector $\boldsymbol{b}$ appearing in problems (6.4) and (6.6) as

$$b_k = \begin{cases} D^2_{i_k, j_k}, & k = 1, 2, \ldots, n_E, \\ 0, & k = m. \end{cases} \tag{6.13}$$

6.3 Strong duality of the graph embedding SDP

Assumption 6.2.1 is essential in order to apply the results of Section 6.2 to the SDP (6.3) and its quadratic low-rank reformulation (6.2). This assumption is shown to hold in [V]. In the following, we give a summary of the duality results established in [V] and the required assumptions.

Sun et al. [118] and Xiao et al. [136] show that the dual problem of the SDP (6.3) can be written as

$$\begin{aligned} \min_{\boldsymbol{\lambda} \geq \mathbf{0}} \quad & \boldsymbol{b}^T \boldsymbol{\lambda} \\ \text{s.t.} \quad & \kappa\Big(\sum_{i=1}^{n_E} \lambda_i \boldsymbol{A}_i\Big) \geq 1, \end{aligned} \tag{6.14}$$

where n_E denotes the number of edges in the edge set E of the graph G, the function $\kappa(\cdot)$ denotes the second smallest eigenvalue of a matrix and the matrices $\boldsymbol{A}_i$ are defined according to (6.11) and (6.12). We assume that the edges of the graph G are ordered according to (6.10) and define the vector $\boldsymbol{b}$ according to (6.13).

The duality results of [V] are established under two mild assumptions on the input graph G. The first one is a nondegeneracy condition that excludes zero edge weights. The second one is connectedness of the graph (i.e. any two nodes can be connected by a path in the edge set).

Assumption 6.3.1 *The edge weights D_{ij} of the graph $G = (V, E)$ satisfy the condition $D_{ij} > 0$ for all $\{i, j\} \in E$.*

Assumption 6.3.2 *The graph $G = (V, E)$ is connected.*

In order to apply the standard duality results for semidefinite programs (e.g. [123]), it is necessary show that the feasible sets of problems (6.3) and its dual have nonempty (relative) interior. This is stated in the following result. Consequently, Assumption 6.2.1 holds for the standard forms of the primal and dual problems (6.3) and (6.14).

Theorem 6.3.1 ([V]) *Under Assumptions 6.3.1 and 6.3.2, the primal problem (6.3) and its dual (6.14) have nonempty solution sets. Furthermore, if $\boldsymbol{K}^* \in \mathbb{R}^{n\times n}$ is a solution to (6.3) and $\boldsymbol{\lambda}^* \in \mathbb{R}^{n_E}$ is a solution to (6.14), then $\operatorname{tr}(\boldsymbol{K}^*) = \boldsymbol{b}^T\boldsymbol{\lambda}^*$.*

6.4 Incremental low-rank algorithm

Based on the above results, we now describe the algorithm developed in [V]. The algorithm obtains a solution to the graph embedding SDP (6.3) by means of the smaller QCQP (6.2). The idea is to successively solve problem (6.2) with increasing dimension d until a solution of problem (6.3) is attained. The algorithm is an adaptation of the incremental low-rank algorithms by Journée et al. [70] and Piacentini [98] to the graph embedding problem.

6.4.1 Problem formulation

It is shown in [V] that problem (6.2) can be equivalently stated in the standard NLP form as

$$\begin{aligned} \min_{\boldsymbol{y}\in\mathbb{R}^{nd}} \quad & f(\boldsymbol{y}) \\ \text{s.t.} \quad & g_i^d(\boldsymbol{y}) \le b_i, \qquad i = 1, 2, \dots, n_E, \\ & h_i^d(\boldsymbol{y}) = 0, \qquad i = 1, 2, \dots, d, \end{aligned} \tag{NLP$_d$}$$

where

$$\begin{aligned} f(\boldsymbol{y}) &= -\|\boldsymbol{y}\|^2, \\ g_i^d(\boldsymbol{y}) &= \boldsymbol{y}^T(\boldsymbol{I}_d \otimes \boldsymbol{A}_i)\boldsymbol{y}, \qquad i = 1, 2, \dots, n_E, \\ h_i^d(\boldsymbol{y}) &= \boldsymbol{y}^T\boldsymbol{c}_i^d, \qquad i = 1, 2, \dots, d \end{aligned}$$

and the nd-dimensional vectors $\boldsymbol{c}_i^d$ are defined as

$$c_{i,j}^d = \begin{cases} 1, & \text{if } j = (i-1)n+1, (i-1)n+2, \ldots, (i-1)n+n, \\ 0, & \text{otherwise} \end{cases}$$

for $i = 1, 2, \ldots, d$. This problem is equivalent to problem (6.2) up to scaling of the objective function. This is a concave minimization problem under convex constraints, and thus the feasible set is convex. Furthermore, it is shown in [V] that its feasible set is bounded under Assumption 6.3.2.

6.4.2 The algorithm

In this subsection we describe an algorithm that solves a sequence of quadratic problems (NLP_d) starting with some small dimension $d = d_0 \geq 1$ and increases d as long as the solution is not optimal to the SDP (6.3).

The following theorem gives a computationally convenient form of condition (6.8) adapted to the graph embedding SDP (6.3) and its quadratic low-rank formulation (NLP_d). For the sequel, we introduce the function $\kappa : \mathcal{S}^n \to \mathbb{R}$ to denote the second smallest eigenvalue (among n, not necessarily distinct eigenvalues) of a symmetric $n \times n$ matrix.

Theorem 6.4.1 ([V]) *Let $\boldsymbol{y}^* \in \mathbb{R}^{nd}$ be a solution to (NLP_d) with Lagrange multipliers $\boldsymbol{\lambda}^* \in \mathbb{R}^{n_E}$ corresponding to the constraints $g_i^d(\boldsymbol{y}^*) \leq b_i$. The matrix $\boldsymbol{K}^* = \boldsymbol{Y}^* \boldsymbol{Y}^{*^T}$ with $\boldsymbol{Y}^* = \text{mat}(\boldsymbol{y}^*)$ is a solution to (6.3) and $\boldsymbol{y}^*$ is a global solution of (NLP_d) if and only if the condition*

$$\kappa(\boldsymbol{L}(\boldsymbol{\lambda}^*)) \geq 1 \tag{6.15}$$

is satisfied, where

$$\boldsymbol{L}(\boldsymbol{\lambda}) = \sum_{i=1}^{n_E} \lambda_i \boldsymbol{A}_i$$

and the matrices $\boldsymbol{A}_i$ are defined according to (6.11) and (6.12).

Testing condition (6.15) can be implemented efficiently. When the matrix $\boldsymbol{L}(\boldsymbol{\lambda}^*)$ appearing in condition (6.15) is sparse (which is for instance the case for k-neighbourhood graphs used in the MVU method), a Lanczos-type algorithm (e.g. [88]) can be used to compute its second smallest eigenvalue.

When a solution to problem (NLP_d) yields a vector $\boldsymbol{y}^* \in \mathbb{R}^{nd}$ for which the matrix $\boldsymbol{K}^* = \boldsymbol{Y}^* \boldsymbol{Y}^{*^T}$ with $\boldsymbol{Y}^* = \text{mat}(\boldsymbol{y}^*)$ is not a solution of the SDP (6.3) by condition (6.15), the algorithm increases the dimension d by one and solves problem (NLP_{d+1}). For solving this problem, the starting point is chosen as an augmented solution $\tilde{\boldsymbol{y}}^* = [\boldsymbol{y}^{*^T} \ \boldsymbol{0}_n^T]^T$.

The following theorem motivates the choice of $\tilde{\boldsymbol{y}}^*$ as the starting point for the solution of (NLP_{d+1}). Namely, it states that when condition (6.15) is not satisfied, the augmented solution $\tilde{\boldsymbol{y}}^*$ is a saddle point of problem (NLP$_{d+1}$). In this case, the theorem gives a descent direction from $\tilde{\boldsymbol{y}}^*$. This direction is also feasible. That is, it is orthogonal to the constraint gradients at $\tilde{\boldsymbol{y}}^*$.

Theorem 6.4.2 ([V]) *Let $\boldsymbol{y}^* \in \mathbb{R}^{nd}$ be a first-order KKT point of problem (NLP$_d$) with Lagrange multipliers $\boldsymbol{\lambda}^* \in \mathbb{R}^{n_E}$ such that $\boldsymbol{\lambda}^* \geq \boldsymbol{0}$ and $\boldsymbol{\mu}^* \in \mathbb{R}^d$ corresponding to the constraints $g_i^d(\boldsymbol{y}^*) \leq b_i$ and $h_i^d(\boldsymbol{y}) = 0$, respectively. If condition (6.15) is not satisfied, then the vectors*

$$\tilde{\boldsymbol{y}}^* = \begin{bmatrix} \boldsymbol{y}^* \\ \boldsymbol{0}_n \end{bmatrix}, \quad \boldsymbol{\lambda}^* \quad \textit{and} \quad \tilde{\boldsymbol{\mu}}^* = \begin{bmatrix} \boldsymbol{\mu}^* \\ 0 \end{bmatrix}$$

satisfy the first-order KKT conditions of problem (NLP$_{d+1}$). Furthermore, the eigenspace corresponding to the eigenvalue $\kappa(\boldsymbol{L}(\boldsymbol{\lambda}^*))$ contains an eigenvector $\boldsymbol{v}^*$ such that $\boldsymbol{1}_n^T \boldsymbol{v}^* = 0$. With such a vector $\boldsymbol{v}^*$, the Hessian of the Lagrangian $\mathcal{L}(\boldsymbol{y}; \boldsymbol{\lambda}; \boldsymbol{\mu})$ of problem (NLP$_{d+1}$) satisfies the condition

$$\boldsymbol{d}^T \nabla_{\boldsymbol{y}}^2 \mathcal{L}(\tilde{\boldsymbol{y}}^*; \boldsymbol{\lambda}^*; \tilde{\boldsymbol{\mu}}^*) \boldsymbol{d} < 0$$

with

$$\boldsymbol{d} = \begin{bmatrix} \boldsymbol{0}_{nd} \\ \boldsymbol{v}^* \end{bmatrix}.$$

In addition, the direction $\boldsymbol{d}$ satisfies the conditions

$$\begin{aligned} \nabla g_i^{d+1}(\tilde{\boldsymbol{y}}^*)^T \boldsymbol{d} &\leq 0, \quad i = 1, 2, \ldots, n_E, \\ \nabla h_i^{d+1}(\tilde{\boldsymbol{y}}^*)^T \boldsymbol{d} &= 0, \quad i = 1, 2, \ldots, d+1. \end{aligned}$$

When the augmented vector $\tilde{\boldsymbol{y}}^*$ obtained from a solution to problem (NLP$_d$) is a saddle point of (NLP$_{d+1}$), Theorem 6.4.2 suggests a strategy for escaping from the saddle point by moving along the descent direction $\boldsymbol{d}$. As a practical implementation of this idea, the algorithm obtains a solution to problem (NLP$_{d+1}$) by perturbing $\tilde{\boldsymbol{y}}^*$ along $\boldsymbol{d}$ and using a descent method from the starting point

$$\boldsymbol{y}_0 = \tilde{\boldsymbol{y}}^* + \varepsilon \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|}$$

with some small $\varepsilon > 0$ and the vectors $\tilde{\boldsymbol{y}}^*$ and $\boldsymbol{d}$ defined as in Theorem 6.4.2.

Based on the above considerations, the incremental low-rank algorithm for solving the SDP (6.3) is listed as Algorithm 6.1.

6.5 Numerical experiments

An extensive numerical comparison between the standard SDP solvers and the `ILR` algorithm (Algorithm 6.1) with different NLP solvers is given in

Algorithm 6.1: ILR (incremental low-rank graph embedding)

```
input  : graph matrices $\{\boldsymbol{A}_i\}_{i=1}^{n_E} \subset \mathbb{R}^{n \times n}$
         squared edge weights $\boldsymbol{b} \in \mathbb{R}^{n_E}$ such that $\boldsymbol{b} > \boldsymbol{0}$
         initial solution dimension $d_0 \geq 1$
         initial solution $\boldsymbol{Y}_0 \in \mathbb{R}^{n \times d_0}$
         perturbation parameter $\varepsilon > 0$
output: embedding $\boldsymbol{Y}^* \in \mathbb{R}^{n \times d^*}$ with $d^* \geq d_0$
$d \leftarrow d_0$
$\boldsymbol{y}_0 \leftarrow \mathrm{vec}(\boldsymbol{Y}_0)$
$\boldsymbol{\lambda}_0 \leftarrow \boldsymbol{0}$
while $d \leq \bar{d}$, where $\bar{d}$ is defined according to (6.9) do
    Starting from $\boldsymbol{y}_0 \in \mathbb{R}^{nd}$ and $\boldsymbol{\lambda}_0 \in \mathbb{R}^{n_E}$, obtain $\boldsymbol{y}^*$, a solution to
    (NLP$_d$) with Lagrange multipliers $\boldsymbol{\lambda}^*$.
    $\boldsymbol{L}^* \leftarrow \sum_{i=1}^{n_E} \lambda_i^* \boldsymbol{A}_i$
    Compute the eigenvalue $\kappa(\boldsymbol{L}^*)$.
    if $\kappa(\boldsymbol{L}^*) \geq 1$ then
        Terminate with $\boldsymbol{Y}^* = \mathrm{mat}(\boldsymbol{y}^*)$.
    else
        Compute the eigenvector $\boldsymbol{v}^*$ according to Theorem 6.4.2.
        $\boldsymbol{d} \leftarrow [\boldsymbol{0}_{nd}^T \quad \boldsymbol{v}^{*T}]^T$
        $\boldsymbol{y}_0 \leftarrow [\boldsymbol{y}^{*T} \quad \boldsymbol{0}_n^T]^T + \varepsilon \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|}$
        $\boldsymbol{\lambda}_0 \leftarrow \boldsymbol{\lambda}^*$
        $d \leftarrow d + 1$
```

[V]. The tests are carried out on two different problem sets: synthetically generated random graphs with different parameters and nearest neighbour graphs. In all tests, the graphs are constructed so that Assumptions 6.3.1 and 6.3.2 are satisfied.

The neighbour graphs are constructed as in the MVU method, and thus the incremental low-rank method can be viewed as a variant of MVU. For constructing the neighbour graphs, synthetic datasets having a low-dimensional embedding as well as benchmark datasets for machine learning are used.

Performance of the following algorithms is evaluated (see [V] for a detailed description of the solvers and the used parameters).

- Interior-point SDP solvers CSDP by Borchers [21] and SDPA by Yamashita et al. [137].
- The `ILR` algorithm combined with augmented Lagrangian methods `ALGENCAN-TN` and `ALGENCAN-NW` by Andreani et al. and Birgin et al. [3, 4, 14–17]. The former uses a truncated conjugate gradient method

for solving the linear systems to obtain the search directions. The latter solves the linear systems by using Harwell MA57 [1].

- The ILR algorithm combined with Ipopt by Wächter and Biegler [134, 135]. Ipopt is an interior point NLP solver using a filter-based line search method. The linear systems for obtaining the search directions are solved by using Harwell MA57.

The motivation for using ALGENCAN and Ipopt is their ability to exploit second-order information and sparsity in the highly structured graph embedding problem. As we will see in the following, they scale well to large problems. Recently, the L-BFGS -based augmented Lagrangian method by Burer et al. [23, 24] has also been proposed for solving graph embedding problems [75, 124]. However, as a first-order method it has slow convergence.

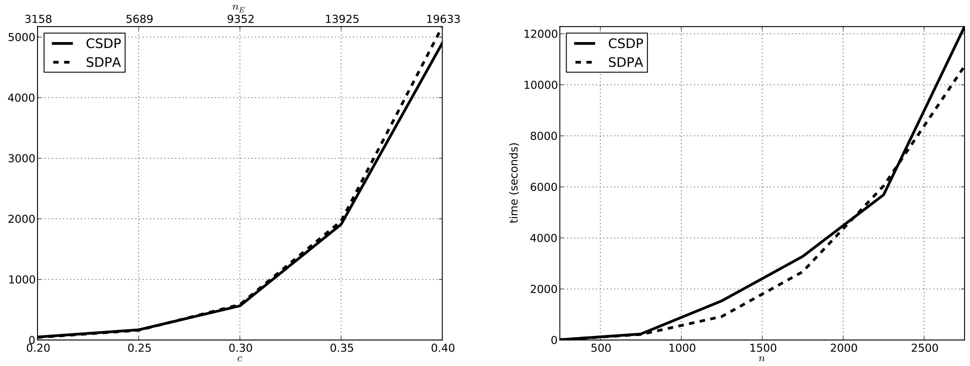
6.5.1 Random graphs

In the first tests carried out in [V], the solvers are run on random graphs with different parameters to isolate the effect of each parameter. The key parameters affecting to the performance of the solvers are the number of edges n_E, the embedding dimension d^* and the number of nodes n.

In each test, the performance is measured as a function of one parameter while keeping the other parameters fixed. The test problems are generated by first sampling a set of points $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^{d^*}$ uniformly from a d^*-dimensional unit cube. The edge set is then constructed by connecting the points $\boldsymbol{x}_i$ within a given distance c from each other according to

$$E_c = \{\{i, j\} \subset \{1, 2, \ldots, n\} \mid \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq c, \quad i \neq j\}.$$

The number of edges (among all possible combinations) in the graph is an increasing function of c , and thus we call it as the graph density parameter.



(a) as a function of c with $n = 500$ and $d^* = 3$ **(b)** as a function of n with $n_E \approx 8n$ and $d^* = 3$

Figure 6.1: Computation times of the CSDP and SDPA solvers on random graphs.

The results shown in Figure 6.1 reveal the limitations of the SDP solvers. The computation times used by `CSDP` and `SDPA` grow rapidly with graph density parameter c and the number of graph nodes n. The SDP solvers also have high memory requirements. As a result, the tests could not be carried out with larger values of c and n than those shown in Figure 6.1.

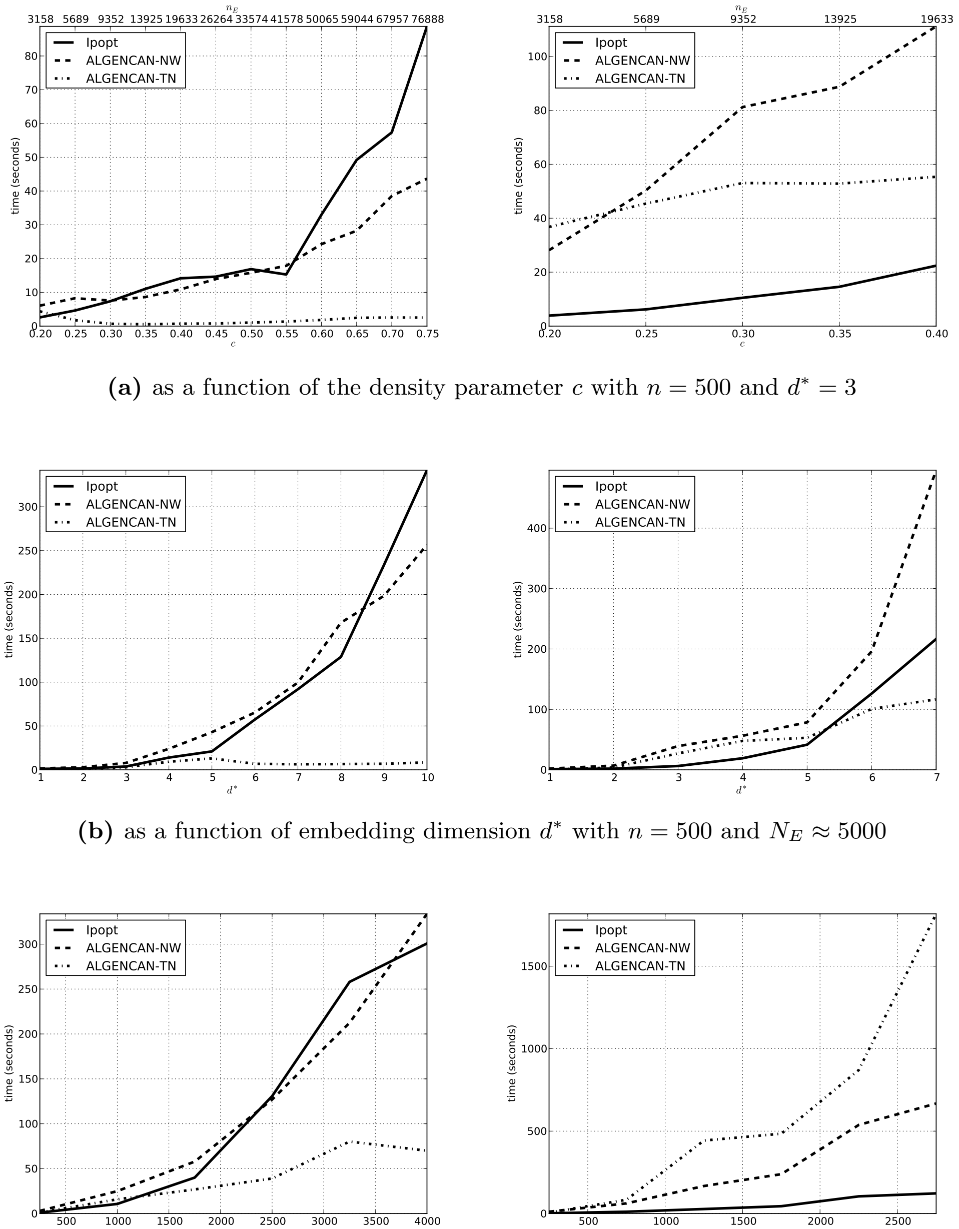
Two different tests are carried out for the NLP solvers combined with the `ILR` algorithm. In the first tests, the initial dimension d_0 is set to the dimension d^* of the optimal solution. In the second tests, the dimension d is successively incremented one by one by starting from $d_0 = 1$. Whereas the first case is of theoretical interest, the latter one is more relevant for practical applications. This is because d^* is usually not known a priori, which necessitates the use of an adaptive dimension update strategy such as the `ILR` algorithm. In all tests, `ILR` terminated with $d = d^*$.

Concerning the first tests where $d_0 = d^*$, the main observations from the left side of Figure 6.2 are listed below.

- Sharing the same `MA57` linear solver, `ALGENCAN-NW` and `Ipopt` have virtually identical performance when $c < 0.55$. However, the computation time used by `Ipopt` rises sharply when c is increased above this limit. This is probably explained by the fact that `Ipopt` uses n_E additional slack variables to convert the inequality constraints to equality constraits. This is not an issue for `ALGENCAN` that incorporates inequality constraints directly into the augmented Lagrangian [3].
- `ALGENCAN-NW` and `Ipopt` scale similarly with respect to the embedding dimension d^* and the graph size n, though `ALGENCAN-NW` is slightly faster for $d^* > 8$. This is expected since both use the same linear solver.
- `ALGENCAN-TN` has superior scalability with respect to c and d^* and also scales significantly better with respect to n than the other solvers. Instead of solving a full linear system at each iteration, using a truncated method (that typically takes only a small number of iterations) seems to give a decisive performance advantage here.

When using `ILR` with $d_0 = 1$ and incrementally updating d, we can make the following observations from the right side of Figure 6.2.

- For `Ipopt`, there is only a relatively small increase in computation times compared to running it with initial dimension $d_0 = d^*$. Rapid convergence of `Ipopt` to solutions of (NLP$_d$) and its effective warm-starting strategy give it a decisive advantage.
- Using `ALGENCAN`-based methods incrementally incurs a major performance penalty in all tests. A possible explanation is that warm-starting from a previous solution of (NLP$_d$) seems to lead to slow convergence.



(a) as a function of the density parameter c with $n = 500$ and $d^* = 3$

(b) as a function of embedding dimension d^* with $n = 500$ and $N_E \approx 5000$

(c) as a function of node count n with $n_E \approx 8n$ and $d^* = 3$

Figure 6.2: Computation times of the `Ipopt`, `ALGENCAN-NW` and `ALGENCAN-TN` solvers combined with `ILR` on random graphs with $d_0 = d^*$ (left) and with $d_0 = 1$ (right).

The above results suggest the following guidelines for choosing the solver (SDP or `ILR`) and using the NLP solvers with `ILR`.

- Using interior-point SDP solvers is only advisable for small and sparse graphs (say $n < 2000$ and $n_E < 30n$) due to high computational cost and memory requirements. However, their performance does not depend on dimension d^* of the optimal solution.
- `Ipopt` scales poorly with respect to d. Thus, it is not advisable to choose initial dimension d_0 larger than the solution dimension d^* (if d^* is known or some estimate is available). On the other hand, using `Ipopt` with $d_0 = 1$ leads to only a moderate increase in computational cost compared to the ideal choice $d_0 = d^*$. This strongly suggests that `Ipopt` should be used incrementally with $d_0 = 1$.
- As for `Ipopt`, choosing a large d_0 for `ALGENCAN-NW` can be computationally very costly. On the other hand, `ALGENCAN-NW` performs adequately when the embedding dimension is small (say $d^* < 7$). Therefore, the same arguments that justify using `Ipopt` with $d_0 = 1$ also apply to `ALGENCAN-NW`.
- When $d_0 = d^*$, the performance of `ALGENCAN-TN` is largely unaffected when c or d^* is increased. This is clearly not the case when $d_0 = 1$ and d is updated incrementally. The problem seems to lie in warm-starting the solver from a previous solution. This suggests that `ALGENCAN-TN` should be used with some large initial dimension (say $d_0 = 10$) and increment d when necessary. When d_0 is sufficiently large, this potentially avoids costly restarts of the solver at a small additional cost.

6.5.2 Synthetic and machine learning datasets

In the second set of experiments carried out in [V], the interior point SDP solvers and the `ILR` algorithm combined with different NLP solvers are applied to k-neighbourhood graphs. The graphs are constructed from a set of synthetic datasets that are known to have a low-dimensional embedding and also from benchmark datasets for machine learning.

The k-neighbourhood graphs are constructed as in the MVU method [131]. Such a neighbourhood of a given point belonging to a point set in $\mathbb{R}^d$ contains the point itself and its k nearest neighbours.

Definition 6.5.1 *Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n\} \subset \mathbb{R}^d$. The k-neighbourhood of a point $\boldsymbol{x} \in \boldsymbol{X}$ is*

$$\begin{aligned}\mathcal{N}_{\boldsymbol{x},k} = \{\{\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \dots, \boldsymbol{x}_{i_{k+1}}\} \subset \boldsymbol{X} \mid i_j \neq i_l \quad \forall j \neq l,\\ \|\boldsymbol{y} - \boldsymbol{x}\| \geq \|\boldsymbol{x}_{i_j} - \boldsymbol{x}\| \quad \forall \boldsymbol{y} \in \boldsymbol{X} \setminus \mathcal{N}_{\boldsymbol{x},k}, \quad j = 1, 2, \dots, k+1\}.\end{aligned}$$

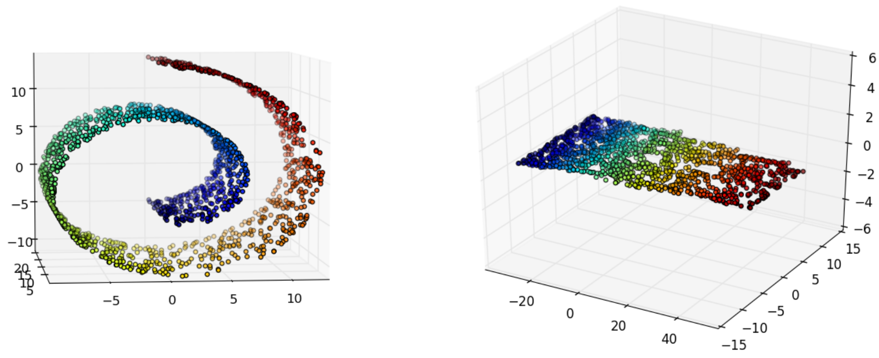
In the k-neighbourhood graph of a given point set in $\mathbb{R}^d$, two points are connected by an edge if the other is a k-nearest neighbour of the other one, or if they lie in the k-neighbourhood of another point.

Definition 6.5.2 *Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^d$. The k-neighbourhood graph of $\boldsymbol{X}$ is the graph $G_{\boldsymbol{X},k} = (V, E)$ with $V = \{1, 2, \ldots, n\}$ and*

$$E = \{\{i, j\} \subset V \mid \exists \boldsymbol{z} \in \boldsymbol{X} : \{\boldsymbol{x}_i, \boldsymbol{x}_j\} \in \mathcal{N}_{\boldsymbol{z},k}\}.$$

The edge weights are chosen as $D_{ij} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ for $\{i, j\} \in E$.

An embedding obtained by solving problem (6.3) for a k-neighbourhood graph of the Swiss roll dataset is shown in Figure 6.3. As in the MVU method, the two-dimensional embedding is obtained from the eigenvectors of the solution matrix $\boldsymbol{K}^*$ corresponding to the two largest eigenvalues.



(a) the original dataset

(b) two-dimensional embedding

Figure 6.3: The Swiss roll dataset and the embedding obtained by solving the MVU problem ($n = 1600$ and $k = 6$).

A detailed listing of the test problems is given in Table 6.1. The optimal embedding dimensions d^* are obtained by `Ipopt` (or by `ALGENCAN-TN` for test problems where `Ipopt` fails). Again, it should be noted here that d^* is in most applications not known a priori. Thus, the most viable choice is to use `ILR` with preferably small initial dimension d_0. Another point of interest is that due to finite sample size, d^* is not in all cases the underlying dimension of the data (two for the first eight datasets).

For each NLP solver used with `ILR`, the initial dimension d_0 is chosen according to the guidelines given in Subsection 6.5.1. That is, $d_0 = 1$ is used for `Ipopt`. This choice is also made for `ALGENCAN-NW` in order to allow a direct comparison with `Ipopt`. For the first eight datasets listed in Table 6.1, `ALGENCAN-TN` is used with $d_0 = 5$ and for the remaining datasets with $d_0 = 10$. As in Subsection 6.5.1, all runs terminated when $d = \max\{d_0, d^*\}$,

	k	n	n_E	d^*	D
Helix	6	800	4 800	4	3
		1 600	9 600	4	3
		2 500	15 000	4	3
		4 000	24 000	4	3
Incomplete tire	6	800	5 702	2	3
		1 600	11 548	3	3
		2 500	18 457	3	3
		4 000	29 764	2	3
S-roll	6	800	5 933	2	3
		1 600	11 839	2	3
		2 500	18 741	3	3
		4 000	29 690	3	3
Spiral	15	800	14 899	3	3
		1 600	32 354	3	3
		2 500	52 408	3	3
		4 000	65 605	2	3
Swiss roll	6	800	5 807	3	3
		1 600	11 653	3	3
		2 500	18 451	3	3
		4 000	29 560	3	3
Trefoil knot	9	800	6 611	2	3
		1 600	13 083	2	3
		2 500	20 607	2	3
		4 000	32 695	2	3
Trefoil ribbon	9	800	8 065	2	3
		1 600	17 665	2	3
		2 500	28 137	2	3
		4 000	46 001	2	3
Twin peaks	6	800	5 889	2	3
		1 600	11 688	3	3
		2 500	18 582	3	3
		4 000	29 666	2	3
Corel color histogram	6	5 000	55 239	9	32
Corel color moments	4	5 000	32 087	6	9
Corel co-occurrence texture	5	5 000	42 321	5	16
Frey faces	10	1 965	39 263	7	560
MNIST	5	6 131	64 802	8	784
USPS	5	1 100	10 748	10	256

Table 6.1: Neighbourhood sizes k, number of nodes n, number of edges n_E and embedding dimensions d^* of the k-neighbourhood graphs constructed from the test datasets. The input dimensions of the datasets are denoted by D.

		Algorithm				
Dataset	n	CSDP	SDPA	ALGENCAN-TN	ALGENCAN-NW	Ipopt
Helix	800	70.70	73.92	50.68	28.44	**6.10**
	1600	1022.38*[1]	499.43*	112.39	193.45	**15.12**
	2500	6003.98	3067.02*	54.98*	1242.09*	**19.93**
	4000	-[2]	-	184.05*	1324.51*	**91.38**
Incomplete tire	800	233.28	196.37	412.75	22.70	**2.62**
	1600	2086.04	1575.87	1497.68	78.16	**8.89**
	2500	9030.10	7794.99	4296.14	187.24	**15.14**
	4000	-	-	10633.47	529.23	**18.57**
S-roll	800	293.89	213.85	232.47	29.34	**2.72**
	1600	2175.45	1675.20	948.79	91.89	**7.63**
	2500	9605.56	6925.95	3519.97	238.89	**21.23**
	4000	-	-	6509.31	789.42	**42.94**
Spiral	800	5060.78	2217.75*	678.45	16.31	**11.54**
	1600	-	-	2140.09	46.58	**44.23**
	2500	-	-	5798.27	**91.48**	125.93
	4000	-	-	2368.59	408.06	**69.88**
Swiss roll	800	279.52	209.26	494.76	75.32	**4.03**
	1600	2262.69	1514.16*	2536.08	354.25	**10.36**
	2500	10898.32	6053.26*	3771.10	500.03	**18.35**
	4000	-	-	9086.93	3153.98	**32.98**
Trefoil knot	800	726.50*	203.20*	157.31	41.46	**3.61**
	1600	5298.22*	1421.72*	773.17	89.37	**9.31**
	2500	-	5255.61*	654.99	67.99	**20.75**
	4000	-	-	1449.84	235.25	**45.98**
Trefoil ribbon	800	816.47	483.68	303.23	28.42	**3.60**
	1600	6866.26	5970.95*	1045.04	117.98	**11.12**
	2500	-	-	2708.00	194.44	**20.04**
	4000	-	-	8119.87	630.77	**44.81**
Twin peaks	800	242.64	207.46	125.52	18.84	**2.76**
	1600	1886.06	1774.86	432.58	57.04	**11.13**
	2500	8783.87	6771.08	3661.15	138.91	**19.85**
	4000	-	-	6346.20	693.31	**26.43**
USPS	1100	1250.46	1192.97	**112.97**	12611.70	1089.52
MNIST	6131	-	-	**2095.75**	-	-
Frey faces	1965	-	-	899.95	7224.71	**845.01**
Color moments	5000	-	-	**991.77**	-	5415.14
Color histogram	5000	-	-	**9406.58**	-	17744.69
Co-occurrence texture	5000	-	-	-	8087.31	**1042.36**

[1] ”*” means that the solution was obtained with reduced accuracy due to premature termination.

[2] ”-” means that the solver either ran out of memory or failed to reach the stopping criterion in five hours.

Table 6.2: Computation times for the test datasets.

although this cannot be guaranteed by the theoretical results. The computation times are shown in Table 6.2. The cases where a solver did not converge to the SDP solution in five hours are considered as failures.

The results shown in Table 6.2 further highlight the limitations of the SDP approach. Excluding the smallest test problems, the interior-point SDP solvers `CSDP` and `SDPA` are clearly the slowest. Moreover, the tests with these solvers could not be carried out with $n = 4000$ or with the last five test problems because the test system ran out of memory. We can also observe that `SDPA` is generally slightly faster than `CSDP`. However, `SDPA` had numerical difficulties, which led to premature termination in several cases. This problem did not appear as often with `CSDP`, which suggests that it is more robust.

Interestingly, the SDP solvers are competitive with the NLP solvers on the USPS dataset having a ten-dimensional embedding. This example highlights the fact that when the input graph is small and has a high-dimensional embedding, the computational cost of solving a large number of quadratic problems can be higher than the cost of solving a single semidefinite relaxation.

`Ipopt` combined with the incremental rank approach performs excellently on test problems where the neighbourhood graph is sparse, highly structured and the embedding dimension is small. On the first eight test problems, it mostly outperforms the other NLP solvers. On the other hand, `Ipopt` performs less well on the last six test problems, where the neighbourhood graph is either large or dense or has a high-dimensional embedding. For the MNIST test problem, it even failed to converge within the five hour time limit.

The computation times of `ALGENCAN-TN` for the first eight test problems are relatively long, and it is even outperformed by the SDP solvers on some test problems with $n = 800$ and $n = 1600$. However, as observed in Subsection 6.5.1, `ALGENCAN-TN` scales well to large or dense graphs or graphs having a high-dimensional embedding. This can clearly be seen with the last six test problems, where `ALGENCAN-TN` is very competitive with `Ipopt` outperforming it on the USPS, MNIST, Color moments and Color histogram test problems. `ALGENCAN-TN` is also competitive with `Ipopt` on the Frey faces test problem, where the embedding is seven-dimensional and the k-neighbourhood graph is relatively dense due to the choice $k = 10$.

`ALGENCAN-NW` is outperformed in nearly all tests by `Ipopt`. Though the solver itself seems to have comparable performance with `Ipopt` (cf. Figure 6.2), the effect of its poor warm-starting ability can also be seen here. This gives a particularly large performance penalty on the USPS and Frey faces test problems, where the high embedding dimension requires solution of a large number of problems (NLP_d). Probably due to this reason `ALGENCAN-NW` failed on the MNIST and Color histogram test problems.

Chapter 7

Conclusions and discussion

Nonlinear dimensionality reduction, identification of curvilinear features from noisy data and finding modes of multivariate probability densities are fundamental tasks in modern data analysis. This thesis focuses on selected topics from these research areas. The emphasis is on algorithmic development and numerical comparison of algorithms. The common theme between the developed algorithms is that they are aimed at solving some optimization problem arising in the above tasks, thus placing the research at the crossroads between optimization and statistics.

Estimation of underlying structure from point sets by using ridges of density functions is nowadays an actively studied research field. Though this research field has gained popularity among statisticians, the development of numerical algorithms has gained surprisingly small amount of research interest. Therefore this thesis makes a threefold contribution to the algorithmic development of ridge-based methods.

The first contribution made in **Paper I** was summarized in Chapter 3. The contribution of this paper is extension of the classical trust region Newton method to finding ridges that are generalized maxima. The proposed method (Algorithm 3.1) has two important advantages over the earlier mean shift method and its subspace-constrained variant for finding modes and ridges, respectively. First, it was shown to consistently outperform the mean shift-based methods on all test problems. Second, it is provably convergent for a very general class of objective functions. Such convergence results can be obtained for the mean shift-based methods only in restricted special cases, and second-order optimality conditions cannot be guaranteed. These findings are of great significance, as the mean shift-based methods have been the standard tools for finding modes and ridges of Gaussian mixtures and kernel densities. However, no theoretical analysis was given for the convergence rate of the Newton-based method for ridge projection, which could be a worthy topic of future research.

Whereas the focus of **Paper I** is mostly theoretical, **Paper III** is more aimed at practical applications. The contribution of this paper summarized in Chapter 4 is the development of a highly efficient method for finding curvilinear structures from noisy data. The method has a wide range of applications such as identification of faults from seismological data and identification of filamentary structures from galaxy clusters. It is based on the statistical model and kernel density estimation methods presented in Chapter 2. For a given point set, the method obtains estimates for curvilinear structures by tracing the ridge curve set of its Gaussian kernel density estimate.

The ridge curve tracing method (Algorithms 4.1–4.3) has two novel features. The first one is definition of a ridge curve as a solution to a differential equation whose solution curves are traced by using a predictor-corrector method. The second one is implementation of rigorous stopping criteria based on the theory of ridge curves. The Newton-based method described in Chapter 3 is utilized in the predictor-corrector method and also in the mode finding step to obtain the starting points. An important computational result is that using the Newton method instead of the mean shift -based methods yields a significant performance improvement.

The final contribution to ridge-based methods made in **Paper IV** was summarized in Chapter 5. In this paper, the structure of ridge sets is utilized in development of a novel nonlinear generalization of principal component analysis (PCA). The so-called kernel density principal component analysis (KDPCA) constructs a nonlinear coordinate system from a ridge set of a Gaussian kernel density. A key result is that the principal component coordinates of a point set can be obtained one by one by successively projecting the data points onto lower-dimensional ridge sets of such a density. Another important result is that the kernel bandwidth has a natural interpretation as a scale parameter. As the bandwidth approaches infinity, the linear PCA is obtained as a special case of KDPCA.

A numerical algorithm (Algorithm 5.1) was developed for tracing the solution curve of a differential equation defining a projection curve onto a lower-dimensional ridge set. The algorithm utilizes the ridge projection method described in Chapter 3. To the knowledge of the author, obtaining principal component coordinates from r-dimensional ridge sets with $r > 1$ has not been previously studied, and the proposed algorithm appears to be the first one developed for this purpose. The applicability of KDPCA was demonstrated on climate model output and time series data. These test cases highlight the main advantages of KDPCA over its linear counterpart. They are the ability to produce a low-dimensional representation from highly nonlinear data and the ability to describe closed loops that occur in analysis of time series having periodic patterns.

An extensive comparison of KDPCA with other nonlinear dimensionality reduction methods would be of great interest. Another interesting topic would be a probabilistic extension of KDPCA, as in its present form it is not based on any statistical model. This could be done by giving a statistical interpretation to the kernel density and applying a bandwidth chooser. Based on the theory presented in Chapter 2, this was done in **Paper III** for curve estimation. An alternative approach could be adaptation of the ideas from the probabilistic PCA by Tipping and Bishop [121]. This method is a variant of the linear PCA for which a probabilistic interpretation is given via a latent variable model. The above aspects are partially covered in the final published version of **Paper IV**.

A major shortcoming of the proposed ridge-based methods is their high computational cost. The most important factor contributing to this was identified to be the evaluation of the Gaussian kernel density and its derivatives. Advanced methods have been developed for this purpose such as the fast Gauss transform by Greengard and Strain [54]. This approximate evaluation method is most appropriate for large datasets having a low dimension (say $d \leq 3$). On the other hand, Shaker et al. [111] describe an exact method for efficient evaluation of Gaussian kernels and their derivatives and show that the method is also applicable high-dimensional data. Determining the applicability of these methods is definitely an important research topic.

Finding significant modes of Gaussian mixtures and kernel densities is another important application area of trust region Newton methods. The contribution of **Paper II** summarized in Chapter 3 is the development of a homotopy continuation method (Algorithms 3.2 and 3.3) for this purpose. The idea is to apply the Gaussian convolution to gradually deform the original density into a unimodal one. Applying this idea reversely and utilizing a trust region Newton method, the global mode of the transformed density is traced back to a significant mode of the original density along the transformation. A computable estimate for a convolution parameter guaranteeing unimodality was derived. The proposed method is computationally efficient and finds global modes with a high probability, but is not guaranteed to do so in all cases. In addition, a precise definition for the significance of a mode found by the method still remains as a topic of future research.

The final contribution of the thesis is the development of an efficient approach for solving the optimization problem arising in the maximum variance unfolding (MVU) method for dimensionality reduction. This problem can either be formulated as a quadratic nonlinear problem (NLP) or a semidefinite problem (SDP). **Paper V** dealing with this topic was summarized in Chapter 6. The incremental low-rank method (Algorithm 6.1) applied to the quadratic formulation together with an efficient NLP solver was shown to give a drastic performance improvement compared to the standard SDP solvers. To the knowledge of the author, neither an incremental low-rank method nor the ALGENCAN and Ipopt solvers have been applied to the MVU problem (or the more general graph embedding problem) before. Thus, the research done in **Paper V** provides valuable knowledge on the behaviour of different NLP solvers when applied to such problems.

It was briefly noted in Chapter 6 that by solving the graph embedding problem, one also obtains a solution to its dual problem due to the zero duality gap. The dual problem has important applications in graph theory [46, 57]. Other applications such as determining the fastest mixing Markov process on a graph are described in [118]. A further study could be devoted to the optimization problem (6.2) when lower bounds for distances between neighbouring points are imposed. In this case, the feasible set becomes nonconvex, but a solution could possibly be obtained under more restrictive assumptions. Another interesting topic is a theoretical verification of the property that when the dimension of the quadratic problem (6.2) is the rank of the solution of the SDP (6.3) or larger, the NLP solvers yield a global solution that is also the SDP solution. This was indeed observed to be the case in all tests.

Bibliography

[1] HSL (2011). A collection of Fortran codes for large scale scientific computation. http://www.hsl.rl.ac.uk. visited on 9/9/2014.

[2] M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions*. Dover, New York, 1964.

[3] R. Andreani, E. Birgin, J. Martínez, and M. Schuverdt. On augmented Lagrangian methods with general lower-level constraints. *SIAM Journal on Optimization*, 18(4):1286–1309, 2008.

[4] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt. Second-order negative-curvature methods for box-constrained and general constrained optimization. *Computational Optimization and Applications*, 45(2):209–236, 2010.

[5] E. Arias-Castro and B. Pelletier. On the convergence of maximum variance unfolding. *Journal of Machine Learning Research*, 14:1747–1770, January 2013.

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

[7] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming - Theory and Algorithms*. John Wiley & Sons, Inc., New York, third edition, 2006.

[8] E. Baş. *Extracting structural information on manifolds from high dimensional data and connectivity analysis of curvilinear structures in 3D biomedical images*. PhD thesis, Northeastern University, Boston, MA, 2011.

[9] E. Baş and D. Erdogmus. Connectivity of projected high dimensional data charts on one-dimensional curves. *Signal Processing*, 91(10):2404–2409, 2011.

[10] E. Baş and D. Erdogmus. Principal curves as skeletons of tubular objects. *Neuroinformatics*, 9(2-3):181–191, 2011.

[11] E. Baş, D. Erdogmus, R. W. Draft, and J. W. Lichtman. Local tracing of curvilinear structures in volumetric color images: Application to the brainbow analysis. *Journal of Visual Communication and Image Representation*, 23(8):1260–1271, 2012.

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

[13] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25:539–575, 1993.

[14] E. G. Birgin and J. M. Martínez. Large-scale active-set box-constrained optimization method with spectral projected gradients. *Computational Optimization and Applications*, 23(1):101–125, 2002.

[15] E. G. Birgin and J. M. Martínez. Improving ultimate convergence of an augmented Lagrangian method. *Optimization Methods and Software*, 23(2):177–195, 2008.

[16] E. G. Birgin and J. M. Martínez. Structured minimal-memory inexact quasi-Newton method and secant preconditioners for augmented Lagrangian optimization. *Computational Optimization and Applications*, 39(1):1–16, 2008.

[17] E. G. Birgin and J. M. Martínez. Augmented Lagrangian method with nonmonotone penalty parameters for constrained optimization. *Computational Optimization and Applications*, 51(3):941–965, 2012.

[18] C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[19] J. M. Bofill, W. Quapp, and M. Caballero. The variational structure of gradient extremals. *Journal of Chemical Theory and Computation*, 8(3):927–935, 2012.

[20] N. A. Bond, , M. A. Strauss, and R. Cen. Crawling the cosmic network: identifying and quantifying filamentary structure. *Monthly Notices of the Royal Astronomical Society*, 409(1):156–168, 2010.

[21] B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11(1–4):613–623, 1999.

[22] C. Brunsdon. Path estimation from GPS tracks. In *9th International Conference on Geocomputation*, National Centre for Geocomputation, National University of Ireland, Maynooth, Eire, 2007.

[23] S. Burer and C. Choi. Computational enhancements in low-rank semidefinite programming. *Optimization Methods and Software*, 21(3):493–512, 2006.

[24] S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

[25] S. Burer and R. D. C. Monteiro. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005.

[26] M. Á. Carreira-Perpiñán. Mode-finding for mixtures of Gaussian distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1318–1323, 2000.

[27] M. Á. Carreira-Perpiñán. Gaussian mean-shift is an EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(5):767–776, 2007.

[28] M. Á. Carreira-Perpiñán and C. K. I. Williams. On the number of modes of a Gaussian mixture. In L. D. Griffin and M. Lillholm, editors, *Scale Space Methods in Computer Vision*, volume 2695 of *Lecture Notes in Computer Science*, pages 625–640. Springer, Berlin, Heidelberg, 2003.

[29] J. E. Chacón and T. Duong. Multivariate plug-in bandwidth selection with unconstrained pilot bandwidth matrices. *TEST*, 19(2):375–398, 2010.

[30] J. E. Chacón, T. Duong, and M. P. Wand. Asymptotics for general multivariate kernel density derivative estimators. *Statistica Sinica*, 21:807–840, 2011.

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

[32] B. Christiansen. The shortcomings of nonlinear principal component analysis in identifying circulation regimes. *Journal of Climate*, 18(22):4814–4823, 2005.

[33] R.T. Collins. Mean-shift blob tracking through scale space. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 234–240, 2003.

[34] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[35] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003.

[36] J. Damon. Generic structure of two-dimensional images under Gaussian blurring. *SIAM Journal on Applied Mathematics*, 59(1):97–138, 1998.

[37] P. Delicado. Another look at principal curves and surfaces. *Journal of Multivariate Analysis*, 77(1):84–116, 2001.

[38] P. Delicado and M. Huerta. Principal curves of oriented points: theoretical and computational improvements. *Computational Statistics*, 18(2):293–315, 2003.

[39] T. L. Delworth, A. J. Broccoli, A. Rosati, R. J. Stouffer, and V. Balaji. GFDL's CM2 global coupled climate models. part I: Formulation and simulation characteristics. *Journal of Climate*, 19(5):643–674, 2006.

[40] M. J. Drinkwater, Q. A. Parker, D. Proust, E. Slezak, and H. Quintana. The large scale distribution of galaxies in the Shapley supercluster. *Publications of the Astronomical Society of Australia*, 21(1):89–96, 2004.

[41] T. Duong. ks: Kernel density estimation and kernel discriminant analysis for multivariate data in R. *Journal of Statistical Software*, 21(7):1–16, 2007.

[42] T. Duong and M. L. Hazelton. Cross-validation bandwidth matrices for multivariate kernel density estimation. *Scandinavian Journal of Statistics*, 32(3):485–506, 2005.

[43] D. Eberly. *Ridges in Image and Data Analysis.* Kluwer Academic Publishers, Dordrecht, Netherlands, 1996.

[44] J. Einbeck, G. Tutz, and L. Evers. Local principal curves. *Statistics and Computing*, 15(4):301–313, 2005.

[45] M. Fashing and C. Tomasi. Mean shift is a bound optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):471–474, 2005.

[46] M. Fiedler. Absolute algebraic connectivity of trees. *Linear and Multilinear Algebra*, 26(1–2):85–106, 1990.

[47] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

[48] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, Boca Raton, Florida, 2004.

[49] C. R. Genovese, M. Perone-Pacifico, I. Verdinelli, and L. Wasserman. Nonparametric ridge estimation. *Annals of Statistics*, 42(4):1511–1545, 2014.

[50] Y. A. Ghassabeh. *On the Convergence and Applications of Mean Shift Type Algorithms*. PhD thesis, Queen's University, Kingston, Ontario, Canada, 2013.

[51] Y. A. Ghassabeh, T. Linder, and G. Takahara. On some convergence properties of the subspace constrained mean shift. *Pattern Recognition*, 46(11):3140–3147, 2013.

[52] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, third edition, 1996.

[53] N. Golyandina, V. Nekrutkin, and A. A. Zhigljavsky. *Analysis of Time Series Structure: SSA and related techniques*. Chapman and Hall/CRC Press, 2001.

[54] L. Greengard and J. Strain. The fast Gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.

[55] C. Grillenzoni. Detection of tectonic faults by spatial clustering of earthquake hypocenters. *Spatial Statistics*, 7:62–78, February 2014.

[56] L. Grippo, L. Palagi, and V. Piccialli. Necessary and sufficient global optimality conditions for NLP reformulations of linear SDP problems. *Journal of Global Optimization*, 44(3):339–348, 2009.

[57] F. Göring, C. Helmberg, and M. Wappler. Embedded in the shadow of the separator. *SIAM Journal on Optimization*, 19(1):472–501, 2008.

[58] P. Hall, M. C. Minnotte, and C. Zhang. Bump hunting with non-Gaussian kernels. *The Annals of Statistics*, 32(5):2124–2141, 2004.

[59] B. Han, Y. Zhu, D. Comaniciu, and L. S. Davis. Visual tracking by continuous density propagation in sequential bayesian filtering framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):919–930, 2009.

[60] R. M. Haralick. Ridges and valleys on digital images. *Computer Vision, Graphics, and Image Processing*, 22(1):28–38, 1983.

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

[62] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* Springer New York, second edition, 2009.

[63] G. E. Hinton and S. T. Roweis. Stochastic neighbor embedding. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15 (NIPS)*, pages 857–864. MIT Press, 2003.

[64] D. K. Hoffman, R. S. Nord, and K. Ruedenberg. Gradient extremals. *Theoretica Chimica Acta*, 69(4):265–279, 1986.

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

[66] W. W. Hsieh. Nonlinear multivariate and time series analysis by neural network methods. *Reviews of Geophysics*, 42(1):1–25, 2004.

[67] W. W. Hsieh and K. Hamilton. Nonlinear singular spectrum analysis of the tropical stratospheric wind. *Quarterly Journal of the Royal Meteorological Society*, 129(592):2367–2382, 2003.

[68] B. Inhester, L. Feng, and T. Wiegelmann. Segmentation of loops from coronal EUV images. *Solar Physics*, 248(2):379–393, 2008.

[69] I. T. Jolliffe. *Principal Component Analysis.* Springer-Verlag, Berlin, 1986.

[70] M. Journée, F. Bach, P. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

[71] N. Kambhatla and K. T. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

[72] M. Kirby and R. Miranda. Circular nodes in neural networks. *Neural Computation*, 8(2):390–402, 1996.

[73] W. Koch. On exploiting 'negative' sensor evidence for target tracking and sensor data fusion. *Information Fusion*, 8(1):28–39, 2007.

[74] M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2):233–243, 1991.

[75] B. Kulis, A.C. Surendran, and J.C. Platt. Fast low-rank semidefinite programming for embedding and clustering. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2007*, pages 512–521, 2007.

[76] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *SIAM Journal on Computing*, 30(2):457–474, 2000.

[77] B. Kégl and A. Krzyzak. Piecewise linear skeletonization using principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):59–74, 2002.

[78] B. Kégl, A. Krzyzak, T. Linder, and K. Zeger. Learning and design of principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):281–297, 2000.

[79] G. Lee, C. Rodriguez, and A. Madabhushi. Investigating the efficacy of nonlinear dimensionality reduction schemes in classifying gene and protein expression studies. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(3):368–384, 2008.

[80] X. Li, Z. Hu, and F. Wu. A note on the convergence of the mean shift. *Pattern Recognition*, 40(6):1756–1762, 2007.

[81] T. Lindeberg. *Scale-space theory in computer vision.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.

[82] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[83] M. Loève. *Probability theory: foundations, random sequences.* van Nostrand, Princeton, New Jersey, USA, 1955.

[84] A. Lucia, P. A. DiMaggio, and P. Depa. A geometric terrain methodology for global optimization. *Journal of Global Optimization*, 29(3):297–314, 2004.

[85] A. Lucia and Y. Feng. Global terrain methods. *Computers & Chemical Engineering*, 26(4–5):529–546, 2002.

[86] J. R. Magnus. On differentiating eigenvalues and eigenvectors. *Economic Theory*, 1(2):179–191, 1985.

[87] J. S. Marron and M. P. Wand. Exact mean integrated squared error. *The Annals of Statistics*, 20(2):712–736, 1992.

[88] G. Meurant. *The Lanczos and Conjugate Gradient Algorithms - From Theory to Finite Precision Computations.* SIAM, Philadelphia, 2006.

[89] J. Miller. *Relative Critical Sets in R^n and Applications to Image Analysis.* PhD thesis, University of North Carolina, 1998.

[90] A. H. Monahan. Nonlinear principal component analysis: Tropical indo–pacific sea surface temperature and sea level pressure. *Journal of Climate*, 14(2):219–233, 2001.

[91] J. J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing*, 4(3):553–572, 1983.

[92] J. J. Moré and Z. Wu. Global continuation for distance geometry problems. *SIAM Journal on Optimization*, 7(3):814–836, 1997.

[93] S. C. Newbigging, L. A. Mysak, and W. W. Hsieh. Improvements to the non-linear principal component analysis method, with applications to ENSO and QBO. *Atmosphere-Ocean*, 41(4):291–299, 2003.

[94] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer, New York, second edition, 2006.

[95] J. M. Ortega. *Numerical Analysis: A Second Course.* SIAM, Philadelphia, 1990.

[96] U. Ozertem and D. Erdogmus. Locally defined principal curves and surfaces. *Journal of Machine Learning Research*, 12:1249–1286, April 2011.

[97] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901.

[98] M. Piacentini. *Nonlinear formulation of Semidefinite Programming and Eigenvalue Optimization – Application to Integer Quadratic Problems.* PhD thesis, Sapienza - Università di Roma, 2012.

[99] S. Pulkkinen. Finding graph embeddings by incremental low-rank semidefinite programming. submitted to Optimization Methods and Software, conditionally accepted (based on TUCS Technical Report 1069).

[100] S. Pulkkinen. Ridge curve approach to extraction of curvilinear structures from noisy data. TUCS Technical Report TR1082, 2013.

[101] S. Pulkkinen. Nonlinear kernel density principal component analysis with application to climate data. *Statistics and Computing*, 2014. accepted (based on TUCS Technical Report 1091), doi:10.1007/s11222-014-9539-0.

[102] S. Pulkkinen. Ridge-based method for finding curvilinear structures from noisy data. *Computational Statistics and Data Analysis*, 82:89–109, February 2015.

[103] S. Pulkkinen, M. M. Mäkelä, and N. Karmitsa. A continuation approach to mode-finding of multivariate Gaussian mixtures and kernel density estimates. *Journal of Global Optimization*, 56(2):459–487, 2013.

[104] S. Pulkkinen, M. M. Mäkelä, and N. Karmitsa. A generative model and a generalized trust region Newton method for noise reduction. *Computational Optimization and Applications*, 57(1):129–165, 2014.

[105] I. Ross. *Nonlinear Dimensionality Reduction Methods in Climate Data Analysis.* PhD thesis, University of Bristol, United Kingdom, 2008.

[106] I. Ross, P. J. Valdes, and S. Wiggins. ENSO dynamics in current climate models: an investigation using nonlinear dimensionality reduction. *Nonlinear Processes in Geophysics*, 15:339–363, 2008.

[107] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[108] M. Scholz, M. Fraunholz, and J. Selbig. Nonlinear principal component analysis: Neural network models and applications. In A. N. Gorban, B. Kégl, D. C. Wunsch, and A. Y. Zinovyev, editors, *Principal Manifolds for Data Visualization and Dimension Reduction*, volume 58 of *Lecture Notes in Computational Science and Engineering*, pages 44–67. Springer Berlin Heidelberg, 2008.

[109] B. Schölkopf, A. Smola, and K.-R. Müller. Kernel principal component analysis. In W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, editors, *Artificial Neural Networks — ICANN'97*, volume 1327 of *Lecture Notes in Computer Science*, pages 583–588. Springer Berlin Heidelberg, 1997.

[110] D. W. Scott. *Multivariate Density Estimation: Theory Practice and Visualization.* John Wiley and Sons, New York, 1992.

[111] M. Shaker, J. N. Myhre, and D. Erdogmus. Computationally efficient exact calculation of kernel density derivatives. *Journal of Signal Processing Systems*. to appear, doi: 10.1007/s11265-014-0904-1.

[112] C. Shen, M. J. Brooks, and A. van den Hengel. Fast global kernel density mode seeking: Applications to localization and tracking. *IEEE Transactions on Image Processing*, 16(5):1457–1469, 2007.

[113] C. Sminchisescu and B. Triggs. Building roadmaps of minima and transitions in visual models. *International Journal of Computer Vision*, 61(1):81–101, 2005.

[114] J. Staal, M. D. Abramoff, M. Niemeijer, M. A. Viergever, and B. van Ginneken. Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging*, 23(4):501–509, 2004.

[115] D. C. Stanford and A. E. Raftery. Finding curvilinear features in spatial point patterns: Principal curve clustering with noise. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(6):601–609, 2000.

[116] T. Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on Numerical Analysis*, 20(3):626–637, 1983.

[117] J. Su, A. Srivastava, and F. W. Huffer. Detection, classification and estimation of individual shapes in 2D and 3D point clouds. *Computational Statistics and Data Analysis*, 58:227–241, 2013.

[118] J. Sun, S. Boyd, L. Xiao, and P. Diaconis. The fastest mixing Markov process on a graph and a connection to a maximum variance unfolding problem. *SIAM Review*, 48(4):681–699, 2006.

[119] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[120] R. Tibshirani. Principal curves revisited. *Statistics and Computation*, 2(4):183–190, 1992.

[121] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[122] L. van der Maaten and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, November 2008.

[123] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[124] N. Vasiloglou, A.G. Gray, and D.V. Anderson. Scalable semidefinite manifold learning. In *IEEE Workshop on Machine Learning for Signal Processing, MLSP 2008*, pages 368–373, 2008.

[125] R. Vautard, P. Yiou, and M. Ghil. Singular-spectrum analysis: A toolkit for short, noisy chaotic signals. *Physica D: Nonlinear Phenomena*, 58(1-4):95–126, 1992.

[126] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski. Information retrieval perspective to nonlinear dimensionality reduction for data visualization. *Journal of Machine Learning Research*, 11:451–490, March 2010.

[127] J.J. Verbeek, N. Vlassis, and B. Kröse. A k-segments algorithm for finding principal curves. *Pattern Recognition Letters*, 23(8):1009–1017, 2002.

[128] M. P. Wand and M. C. Jones. *Kernel Smoothing.* Chapman and Hall/CRC, London, 1995.

[129] X. Wang and K. K. Paliwal. Feature extraction and dimensionality reduction algorithms and their applications in vowel recognition. *Pattern Recognition*, 36(10):2429–2439, 2003.

[130] B. C. Weare, A. R. Navato, and E. R. Newell. Empirical orthogonal analysis of pacific sea surface temperatures. *Journal of Physical Oceanography*, 6(5):671–678, 1976.

[131] K. Weinberger and L. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, 70(1):77–90, 2006.

[132] O. Wirjadi and T. Breuel. A branch and bound algorithm for finding the modes in kernel density estimates. *International Journal of Computational Intelligence and Applications*, 8(1):17–35, 2009.

[133] Z. Wu. The effective energy transformation scheme as a special continuation approach to global optimization with application to molecular conformation. *SIAM Journal on Optimization*, 6(3):748–768, 1996.

[134] A. Wächter and L. T. Biegler. Line search filter methods for nonlinear programming: motivation and global convergence. *SIAM Journal on Optimization*, 16(1):1–31, 2005.

[135] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[136] L. Xiao, J. Sun, and S. Boyd. A duality view of spectral methods for dimensionality reduction. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 1041–1048, 2006.

[137] M. Yamashita, K. Fujisawa, and M. Kojima. Implementation and evaluation of SDPA 6.0 (Semidefinite Programming Algorithm 6.0). *Optimization Methods and Software*, 18(4):491–505, 2003.

[138] Y. Zhan and J. Yin. Robust local tangent space alignment via iterative weighted PCA. *Neurocomputing*, 74(11):1985–1993, 2011.

[139] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM Journal on Scientific Computing*, 26(1):313–338, 2004.

Turku Centre for Computer Science
TUCS Dissertations

1. **Marjo Lipponen**, On Primitive Solutions of the Post Correspondence Problem
2. **Timo Käkölä**, Dual Information Systems in Hyperknowledge Organizations
3. **Ville Leppänen**, Studies on the Realization of PRAM
4. **Cunsheng Ding**, Cryptographic Counter Generators
5. **Sami Viitanen**, Some New Global Optimization Algorithms
6. **Tapio Salakoski**, Representative Classification of Protein Structures
7. **Thomas Långbacka**, An Interactive Environment Supporting the Development of Formally Correct Programs
8. **Thomas Finne**, A Decision Support System for Improving Information Security
9. **Valeria Mihalache**, Cooperation, Communication, Control. Investigations on Grammar Systems.
10. **Marina Waldén**, Formal Reasoning About Distributed Algorithms
11. **Tero Laihonen**, Estimates on the Covering Radius When the Dual Distance is Known
12. **Lucian Ilie**, Decision Problems on Orders of Words
13. **Jukkapekka Hekanaho**, An Evolutionary Approach to Concept Learning
14. **Jouni Järvinen**, Knowledge Representation and Rough Sets
15. **Tomi Pasanen**, In-Place Algorithms for Sorting Problems
16. **Mika Johnsson**, Operational and Tactical Level Optimization in Printed Circuit Board Assembly
17. **Mats Aspnäs**, Multiprocessor Architecture and Programming: The Hathi-2 System
18. **Anna Mikhajlova**, Ensuring Correctness of Object and Component Systems
19. **Vesa Torvinen**, Construction and Evaluation of the Labour Game Method
20. **Jorma Boberg**, Cluster Analysis. A Mathematical Approach with Applications to Protein Structures
21. **Leonid Mikhajlov**, Software Reuse Mechanisms and Techniques: Safety Versus Flexibility
22. **Timo Kaukoranta**, Iterative and Hierarchical Methods for Codebook Generation in Vector Quantization
23. **Gábor Magyar**, On Solution Approaches for Some Industrially Motivated Combinatorial Optimization Problems
24. **Linas Laibinis**, Mechanised Formal Reasoning About Modular Programs
25. **Shuhua Liu**, Improving Executive Support in Strategic Scanning with Software Agent Systems
26. **Jaakko Järvi**, New Techniques in Generic Programming – C++ is more Intentional than Intended
27. **Jan-Christian Lehtinen**, Reproducing Kernel Splines in the Analysis of Medical Data
28. **Martin Büchi**, Safe Language Mechanisms for Modularization and Concurrency
29. **Elena Troubitsyna**, Stepwise Development of Dependable Systems
30. **Janne Näppi**, Computer-Assisted Diagnosis of Breast Calcifications
31. **Jianming Liang**, Dynamic Chest Images Analysis
32. **Tiberiu Seceleanu**, Systematic Design of Synchronous Digital Circuits
33. **Tero Aittokallio**, Characterization and Modelling of the Cardiorespiratory System in Sleep-Disordered Breathing
34. **Ivan Porres**, Modeling and Analyzing Software Behavior in UML
35. **Mauno Rönkkö**, Stepwise Development of Hybrid Systems
36. **Jouni Smed**, Production Planning in Printed Circuit Board Assembly
37. **Vesa Halava**, The Post Correspondence Problem for Market Morphisms
38. **Ion Petre**, Commutation Problems on Sets of Words and Formal Power Series
39. **Vladimir Kvassov**, Information Technology and the Productivity of Managerial Work
40. **Frank Tétard**, Managers, Fragmentation of Working Time, and Information Systems

41. **Jan Manuch**, Defect Theorems and Infinite Words
42. **Kalle Ranto**, Z_4-Goethals Codes, Decoding and Designs
43. **Arto Lepistö**, On Relations Between Local and Global Periodicity
44. **Mika Hirvensalo**, Studies on Boolean Functions Related to Quantum Computing
45. **Pentti Virtanen**, Measuring and Improving Component-Based Software Development
46. **Adekunle Okunoye**, Knowledge Management and Global Diversity – A Framework to Support Organisations in Developing Countries
47. **Antonina Kloptchenko**, Text Mining Based on the Prototype Matching Method
48. **Juha Kivijärvi**, Optimization Methods for Clustering
49. **Rimvydas Rukšėnas**, Formal Development of Concurrent Components
50. **Dirk Nowotka**, Periodicity and Unbordered Factors of Words
51. **Attila Gyenesei**, Discovering Frequent Fuzzy Patterns in Relations of Quantitative Attributes
52. **Petteri Kaitovaara**, Packaging of IT Services – Conceptual and Empirical Studies
53. **Petri Rosendahl**, Niho Type Cross-Correlation Functions and Related Equations
54. **Péter Majlender**, A Normative Approach to Possibility Theory and Soft Decision Support
55. **Seppo Virtanen**, A Framework for Rapid Design and Evaluation of Protocol Processors
56. **Tomas Eklund**, The Self-Organizing Map in Financial Benchmarking
57. **Mikael Collan**, Giga-Investments: Modelling the Valuation of Very Large Industrial Real Investments
58. **Dag Björklund**, A Kernel Language for Unified Code Synthesis
59. **Shengnan Han**, Understanding User Adoption of Mobile Technology: Focusing on Physicians in Finland
60. **Irina Georgescu**, Rational Choice and Revealed Preference: A Fuzzy Approach
61. **Ping Yan**, Limit Cycles for Generalized Liénard-Type and Lotka-Volterra Systems
62. **Joonas Lehtinen**, Coding of Wavelet-Transformed Images
63. **Tommi Meskanen,** On the NTRU Cryptosystem
64. **Saeed Salehi**, Varieties of Tree Languages
65. **Jukka Arvo**, Efficient Algorithms for Hardware-Accelerated Shadow Computation
66. **Mika Hirvikorpi**, On the Tactical Level Production Planning in Flexible Manufacturing Systems
67. **Adrian Costea**, Computational Intelligence Methods for Quantitative Data Mining
68. **Cristina Seceleanu**, A Methodology for Constructing Correct Reactive Systems
69. **Luigia Petre**, Modeling with Action Systems
70. **Lu Yan**, Systematic Design of Ubiquitous Systems
71. **Mehran Gomari**, On the Generalization Ability of Bayesian Neural Networks
72. **Ville Harkke**, Knowledge Freedom for Medical Professionals – An Evaluation Study of a Mobile Information System for Physicians in Finland
73. **Marius Cosmin Codrea**, Pattern Analysis of Chlorophyll Fluorescence Signals
74. **Aiying Rong**, Cogeneration Planning Under the Deregulated Power Market and Emissions Trading Scheme
75. **Chihab BenMoussa**, Supporting the Sales Force through Mobile Information and Communication Technologies: Focusing on the Pharmaceutical Sales Force
76. **Jussi Salmi**, Improving Data Analysis in Proteomics
77. **Orieta Celiku**, Mechanized Reasoning for Dually-Nondeterministic and Probabilistic Programs
78. **Kaj-Mikael Björk**, Supply Chain Efficiency with Some Forest Industry Improvements
79. **Viorel Preoteasa**, Program Variables – The Core of Mechanical Reasoning about Imperative Programs
80. **Jonne Poikonen**, Absolute Value Extraction and Order Statistic Filtering for a Mixed-Mode Array Image Processor
81. **Luka Milovanov**, Agile Software Development in an Academic Environment
82. **Francisco Augusto Alcaraz Garcia**, Real Options, Default Risk and Soft Applications
83. **Kai K. Kimppa**, Problems with the Justification of Intellectual Property Rights in Relation to Software and Other Digitally Distributable Media
84. **Dragoş Truşcan**, Model Driven Development of Programmable Architectures
85. **Eugen Czeizler**, The Inverse Neighborhood Problem and Applications of Welch Sets in Automata Theory

86. **Sanna Ranto**, Identifying and Locating-Dominating Codes in Binary Hamming Spaces
87. **Tuomas Hakkarainen**, On the Computation of the Class Numbers of Real Abelian Fields
88. **Elena Czeizler**, Intricacies of Word Equations
89. **Marcus Alanen**, A Metamodeling Framework for Software Engineering
90. **Filip Ginter**, Towards Information Extraction in the Biomedical Domain: Methods and Resources
91. **Jarkko Paavola**, Signature Ensembles and Receiver Structures for Oversaturated Synchronous DS-CDMA Systems
92. **Arho Virkki**, The Human Respiratory System: Modelling, Analysis and Control
93. **Olli Luoma**, Efficient Methods for Storing and Querying XML Data with Relational Databases
94. **Dubravka Ilić**, Formal Reasoning about Dependability in Model-Driven Development
95. **Kim Solin**, Abstract Algebra of Program Refinement
96. **Tomi Westerlund**, Time Aware Modelling and Analysis of Systems-on-Chip
97. **Kalle Saari**, On the Frequency and Periodicity of Infinite Words
98. **Tomi Kärki**, Similarity Relations on Words: Relational Codes and Periods
99. **Markus M. Mäkelä**, Essays on Software Product Development: A Strategic Management Viewpoint
100. **Roope Vehkalahti**, Class Field Theoretic Methods in the Design of Lattice Signal Constellations
101. **Anne-Maria Ernvall-Hytönen**, On Short Exponential Sums Involving Fourier Coefficients of Holomorphic Cusp Forms
102. **Chang Li**, Parallelism and Complexity in Gene Assembly
103. **Tapio Pahikkala**, New Kernel Functions and Learning Methods for Text and Data Mining
104. **Denis Shestakov**, Search Interfaces on the Web: Querying and Characterizing
105. **Sampo Pyysalo**, A Dependency Parsing Approach to Biomedical Text Mining
106. **Anna Sell**, Mobile Digital Calendars in Knowledge Work
107. **Dorina Marghescu**, Evaluating Multidimensional Visualization Techniques in Data Mining Tasks
108. **Tero Säntti**, A Co-Processor Approach for Efficient Java Execution in Embedded Systems
109. **Kari Salonen**, Setup Optimization in High-Mix Surface Mount PCB Assembly
110. **Pontus Boström**, Formal Design and Verification of Systems Using Domain-Specific Languages
111. **Camilla J. Hollanti**, Order-Theoretic Mehtods for Space-Time Coding: Symmetric and Asymmetric Designs
112. **Heidi Himmanen**, On Transmission System Design for Wireless Broadcasting
113. **Sébastien Lafond**, Simulation of Embedded Systems for Energy Consumption Estimation
114. **Evgeni Tsivtsivadze**, Learning Preferences with Kernel-Based Methods
115. **Petri Salmela**, On Commutation and Conjugacy of Rational Languages and the Fixed Point Method
116. **Siamak Taati**, Conservation Laws in Cellular Automata
117. **Vladimir Rogojin**, Gene Assembly in Stichotrichous Ciliates: Elementary Operations, Parallelism and Computation
118. **Alexey Dudkov**, Chip and Signature Interleaving in DS CDMA Systems
119. **Janne Savela**, Role of Selected Spectral Attributes in the Perception of Synthetic Vowels
120. **Kristian Nybom**, Low-Density Parity-Check Codes for Wireless Datacast Networks
121. **Johanna Tuominen**, Formal Power Analysis of Systems-on-Chip
122. **Teijo Lehtonen**, On Fault Tolerance Methods for Networks-on-Chip
123. **Eeva Suvitie**, On Inner Products Involving Holomorphic Cusp Forms and Maass Forms
124. **Linda Mannila**, Teaching Mathematics and Programming – New Approaches with Empirical Evaluation
125. **Hanna Suominen**, Machine Learning and Clinical Text: Supporting Health Information Flow
126. **Tuomo Saarni**, Segmental Durations of Speech
127. **Johannes Eriksson**, Tool-Supported Invariant-Based Programming

128. **Tero Jokela**, Design and Analysis of Forward Error Control Coding and Signaling for Guaranteeing QoS in Wireless Broadcast Systems
129. **Ville Lukkarila**, On Undecidable Dynamical Properties of Reversible One-Dimensional Cellular Automata
130. **Qaisar Ahmad Malik**, Combining Model-Based Testing and Stepwise Formal Development
131. **Mikko-Jussi Laakso**, Promoting Programming Learning: Engagement, Automatic Assessment with Immediate Feedback in Visualizations
132. **Riikka Vuokko**, A Practice Perspective on Organizational Implementation of Information Technology
133. **Jeanette Heidenberg**, Towards Increased Productivity and Quality in Software Development Using Agile, Lean and Collaborative Approaches
134. **Yong Liu**, Solving the Puzzle of Mobile Learning Adoption
135. **Stina Ojala**, Towards an Integrative Information Society: Studies on Individuality in Speech and Sign
136. **Matteo Brunelli**, Some Advances in Mathematical Models for Preference Relations
137. **Ville Junnila**, On Identifying and Locating-Dominating Codes
138. **Andrzej Mizera**, Methods for Construction and Analysis of Computational Models in Systems Biology. Applications to the Modelling of the Heat Shock Response and the Self-Assembly of Intermediate Filaments.
139. **Csaba Ráduly-Baka**, Algorithmic Solutions for Combinatorial Problems in Resource Management of Manufacturing Environments
140. **Jari Kyngäs**, Solving Challenging Real-World Scheduling Problems
141. **Arho Suominen**, Notes on Emerging Technologies
142. **József Mezei**, A Quantitative View on Fuzzy Numbers
143. **Marta Olszewska**, On the Impact of Rigorous Approaches on the Quality of Development
144. **Antti Airola**, Kernel-Based Ranking: Methods for Learning and Performace Estimation
145. **Aleksi Saarela**, Word Equations and Related Topics: Independence, Decidability and Characterizations
146. **Lasse Bergroth**, Kahden merkkijonon pisimmän yhteisen alijonon ongelma ja sen ratkaiseminen
147. **Thomas Canhao Xu**, Hardware/Software Co-Design for Multicore Architectures
148. **Tuomas Mäkilä**, Software Development Process Modeling – Developers Perspective to Contemporary Modeling Techniques
149. **Shahrokh Nikou**, Opening the Black-Box of IT Artifacts: Looking into Mobile Service Characteristics and Individual Perception
150. **Alessandro Buoni**, Fraud Detection in the Banking Sector: A Multi-Agent Approach
151. **Mats Neovius**, Trustworthy Context Dependency in Ubiquitous Systems
152. **Fredrik Degerlund**, Scheduling of Guarded Command Based Models
153. **Amir-Mohammad Rahmani-Sane**, Exploration and Design of Power-Efficient Networked Many-Core Systems
154. **Ville Rantala**, On Dynamic Monitoring Methods for Networks-on-Chip
155. **Mikko Pelto**, On Identifying and Locating-Dominating Codes in the Infinite King Grid
156. **Anton Tarasyuk**, Formal Development and Quantitative Verification of Dependable Systems
157. **Muhammad Mohsin Saleemi**, Towards Combining Interactive Mobile TV and Smart Spaces: Architectures, Tools and Application Development
158. **Tommi J. M. Lehtinen**, Numbers and Languages
159. **Peter Sarlin**, Mapping Financial Stability
160. **Alexander Wei Yin**, On Energy Efficient Computing Platforms
161. **Mikołaj Olszewski**, Scaling Up Stepwise Feature Introduction to Construction of Large Software Systems
162. **Maryam Kamali**, Reusable Formal Architectures for Networked Systems
163. **Zhiyuan Yao**, Visual Customer Segmentation and Behavior Analysis – A SOM-Based Approach
164. **Timo Jolivet**, Combinatorics of Pisot Substitutions
165. **Rajeev Kumar Kanth**, Analysis and Life Cycle Assessment of Printed Antennas for Sustainable Wireless Systems
166. **Khalid Latif**, Design Space Exploration for MPSoC Architectures

167. **Bo Yang**, Towards Optimal Application Mapping for Energy-Efficient Many-Core Platforms
168. **Ali Hanzala Khan**, Consistency of UML Based Designs Using Ontology Reasoners
169. **Sonja Leskinen**, m-Equine: IS Support for the Horse Industry
170. **Fareed Ahmed Jokhio**, Video Transcoding in a Distributed Cloud Computing Environment
171. **Moazzam Fareed Niazi**, A Model-Based Development and Verification Framework for Distributed System-on-Chip Architecture
172. **Mari Huova**, Combinatorics on Words: New Aspects on Avoidability, Defect Effect, Equations and Palindromes
173. **Ville Timonen**, Scalable Algorithms for Height Field Illumination
174. **Henri Korvela**, Virtual Communities – A Virtual Treasure Trove for End-User Developers
175. **Kameswar Rao Vaddina**, Thermal-Aware Networked Many-Core Systems
176. **Janne Lahtiranta**, New and Emerging Challenges of the ICT-Mediated Health and Well-Being Services
177. **Irum Rauf**, Design and Validation of Stateful Composite RESTful Web Services
178. **Jari Björne**, Biomedical Event Extraction with Machine Learning
179. **Katri Haverinen**, Natural Language Processing Resources for Finnish: Corpus Development in the General and Clinical Domains
180. **Ville Salo**, Subshifts with Simple Cellular Automata
181. **Johan Ersfolk**, Scheduling Dynamic Dataflow Graphs
182. **Hongyan Liu**, On Advancing Business Intelligence in the Electricity Retail Market
183. **Adnan Ashraf**, Cost-Efficient Virtual Machine Management: Provisioning, Admission Control, and Consolidation
184. **Muhammad Nazrul Islam**, Design and Evaluation of Web Interface Signs to Improve Web Usability: A Semiotic Framework
185. **Johannes Tuikkala**, Algorithmic Techniques in Gene Expression Processing: From Imputation to Visualization
186. **Natalia Díaz Rodríguez**, Semantic and Fuzzy Modelling for Human Behaviour Recognition in Smart Spaces. A Case Study on Ambient Assisted Living
187. **Mikko Pänkäälä**, Potential and Challenges of Analog Reconfigurable Computation in Modern and Future CMOS
188. **Sami Hyrynsalmi**, Letters from the War of Ecosystems – An Analysis of Mobile Application Marketplaces
189. **Seppo Pulkkinen**, Efficient Optimization Algorithms for Nonlinear Data Analysis

Turku Centre *for* Computer Science

Joukahaisenkatu 3-5 B, 20520 Turku, Finland | www. tucs.fi

University of Turku
Faculty of Mathematics and Natural Sciences
- Department of Information Technology
- Department of Mathematics and Statistics

Turku School of Economics
- Institute of Information Systems Science

Åbo Akademi University
Division for Natural Sciences and Technology
- Department of Information Technologies

ISBN 978-952-12-3152-0
ISSN 1239-1883

Seppo Pulkkinen

Efficient Optimization Algorithms for Nonlinear Data Analysis