ON THE CONVERGENCE AND APPLICATIONS OF MEAN SHIFT TYPE ALGORITHMS

by

YOUNESS ALIYARI GHASSABEH

A thesis submitted to the
Department of Mathematics and Statistics
in conformity with the requirements for
the degree of Doctor of Philosophy

Queen's University
Kingston, Ontario, Canada
September 2013

Abstract

Mean shift (MS) and subspace constrained mean shift (SCMS) algorithms are non-parametric, iterative methods to find a representation of a high dimensional data set on a principal curve or surface embedded in a high dimensional space. The representation of high dimensional data on a principal curve or surface, the class of mean shift type algorithms and their properties, and applications of these algorithms are the main focus of this dissertation.

Although MS and SCMS algorithms have been used in many applications, a rigorous study of their convergence is still missing. This dissertation aims to fill some of the gaps between theory and practice by investigating some convergence properties of these algorithms. In particular, we propose a sufficient condition for a kernel density estimate with a Gaussian kernel to have isolated stationary points to guarantee the convergence of the MS algorithm. We also show that the SCMS algorithm inherits some of the important convergence properties of the MS algorithm. In particular, the monotonicity and convergence of the density estimate values along the sequence of output values of the algorithm are shown. We also show that the distance between consecutive points of the output sequence converges to zero, as does the projection of the gradient vector onto the subspace spanned by the $D-d$ eigenvectors corresponding to the $D-d$ largest eigenvalues of the local inverse covariance matrix.

Furthermore, three new variations of the SCMS algorithm are proposed and the running times and performance of the resulting algorithms are compared with original SCMS algorithm. We also propose an adaptive version of the SCMS algorithm to consider the effect of new incoming samples without running the algorithm on the whole data set.

As well, we develop some new potential applications of the MS and SCMS algorithm. These applications involve finding straight lines in digital images; pre-processing data before applying locally linear embedding (LLE) and ISOMAP for dimensionality reduction; noisy source vector quantization where the clean data need to be estimated before the quanization step; improving the performance of kernel regression in certain situations; and skeletonization of digitally stored handwritten characters.

Acknowledgments

I would like to express my sincere gratitude to my supervisors, Professor Tamás Linder and Professor Glen Takahara for their insightful guidance, continuous support, and thoughtful suggestions and comments throughout the completion of this work. I was lucky to have had the opportunity to work under their supervision. I am also grateful to my committee members Professor Adam Krzyzak, Professor Wai-Yip Geoffrey Chan, Professor Bahman Gharesifard, and Professor Serban Belinschi, for spending their valuable time to read my thesis, attending my defense, and providing detailed and constructive comments. I am also thankful to Professor Mohamed Ibnkahla for chairing my defense session.

I would also like to thank my parents, my sister, and my brother for their unconditional support and encouragement.

Finally, I would like to thank my wife, Nasim, and my lovely son, Ryan, for their patience and support. This dissertation could not have been finished, without their love and encouragement. I have been fortunate to have such a wonderful family, I thank them all for their everlasting love and encouragement.

Contents

List of Figures

List of Tables

List of Abbreviations

ASCMS algorithm Adaptive subspace constrained mean shift algorithm

GBMS algorithm Gaussian blurring mean shift algorithm

HSPC Principal curves defined by Hastie and Stuetzle

ICA Independent component analysis

KPCA Kernel Principal component analysis

LDA Linear discriminant analysis

LLE Locally linear embedding

MDS Multidimensional scaling

MMS algorithm Modified mean shift algorithm

MS algorithm Mean shift algorithm

MVU algorithm Maximum variance unfolding algorithm

PCA Principal component analysis

PCOP Principal curve of oriented points

POP Principal oriented points

SCMS algorithm Subspace constrained mean shift algorithm

Chapter 1

Introduction

Dimensionality reduction and manifold-learning are two important problems in many information processing fields including statistical pattern recognition, machine learning, artificial intelligence, information retrieval, statistics, data mining, and data compression. Real world data, such as digital images, genomic data, fMRI scans, and speech signals, often have high dimensionality, which makes their processing difficult and time consuming. It is often desirable that the observed high-dimensional data be represented in a lower dimensional space while preserving the original information as much as possible.

Dimensionality reduction and manifold-learning techniques provide compact and meaningful representations, which facilitate compression, classification, and visualization of high-dimensional data. Using these techniques, one can tackle practical issues, such as limited computational power and memory, which arise during the processing of a high-dimensional data set. In many applications, it is a realistic assumption that the observed high-dimensional data will have an intrinsically low-dimensional structure, so that the data points lie on or near a low-dimensional manifold embedded in the high-dimensional space.

Learning the underlying manifold from the high-dimensional data can help to reduce the dimensionality of the observed data. A multitude of different algorithms have been introduced to find or approximate the underlying low-dimensional manifold. These algorithms can be classified as either linear or nonlinear dimensionality reduction techniques.

Linear dimensionality reduction techniques, such as principal component analysis (PCA) [67], multidimensional scaling (MDS) [23], independent component analysis (ICA) [63], factor analysis [52], and linear discriminant analysis (LDA) [91], provide a low-dimensional representation of the high-dimensional data in a linear subspace. These linear techniques are simple to implement and if the observed data lie on or near a linear subspace, they guarantee to find the low-dimensional linear structure. However, in real world problems the underlying low-dimensional manifold often has a nonlinear structure that cannot be revealed using the linear techniques. For a nonlinear underlying manifold, different techniques including locally linear embedding (LLE) [108], ISOMAP [116], kernel PCA [110], maximum variance unfolding (MVU) [125], and Hessian LLE [31], among others, have been proposed.

The problem becomes more complicated when the high-dimensional input data is corrupted by noise. In this case, the observed data can be modeled as low-dimensional "clean" data corrupted by high-dimensional noise. In this case, applying common linear/nonlinear dimensionality reduction techniques [66] on the noisy observations may not lead to a meaningful low-dimensional representation of the observed data. Partly to overcome this problem, nonlinear generalizations of principal components, called principal curves (and surfaces) have been proposed. The first formal definition of a principal curve was given by Hastie and Stuetzle [57]. According to their definition, a principal curve is a smooth (one-dimensional) curve that passes through the middle of a data set to provide a nonlinear summary of the data. Several definitions of principal curves and algorithms to construct them have been proposed based on, or inspired by, Hastie and Stuetzle's original definition (see [6], [117], [15], [28], [74], and [107], among others). The aim of these new definitions and algorithms was to address some of the shortcomings of the original (and subsequent) definition(s) and to extend the range of potential applications.

1.1 Problem Statement

As mentioned before, since Hastie and Stuetzle's groundbreaking work, many different definitions and algorithms have been proposed to estimate a principal curve or surface. In fact, one difficulty with principal curves and surfaces is that there are several different notions of them in the literature. Recently, an interesting new definition of principal curves and surfaces has been proposed by Ozertem and Erdogmus [96]. According to this definition, given a smooth (at least twice continuously differentiable) probability density function (pdf) f on $\mathbb{R}^D$, a d-dimensional principal surface ($d < D$) is the collection of all points where the gradient of f is orthogonal to exactly $D - d$ eigenvectors of the Hessian of f, and the eigenvalues corresponding to these eigenvectors are negative. An attractive property of this new definition is that the smoothness of the principal curves and surfaces is not stipulated by their definition, but rather it is inherited from the smoothness of the underlying pdf or its estimate.

To estimate principal curves/surfaces based on the new definition, [96] proposed the so-called subspace constrained mean shift (SCMS) algorithm. It is a generalization of the well-known mean shift (MS) algorithm ([44], [16], and [18]), which iteratively tries to find modes of a pdf (estimated from data samples) in a local subspace. On synthetic data sets, the performance of the SCMS algorithm is comparable to (and in some situations better than) the principal curve algorithms of Hastie and Stuetzle [57] and Kégl *et al.* [74], and it is computationally less demanding. Moreover, in contrast to most previous principal curve algorithms, the SCMS algorithm can naturally handle loops and self-intersections, and it easily generalizes from principal curves to surfaces. The SCMS algorithm has been successfully used for applications such as time-series denoising [95], independent component analysis [96], nonlinear dimensionality reduction in the presence of noise [48], and vector quantization of noisy sources [47].

Based on an assertion in [18] that the so-called mean shift (MS) algorithm converges, Ozertem and Erdogmus claimed that their SCMS algorithm converges to a principal curve/surface. However, there is a fundamental mistake in the proof of the convergence of the MS algorithm in [18]. The authors in [18] claimed that the sequence generated by the MS algorithm is a Cauchy sequence, which is not true in general. Therefore, the convergence of the MS algorithm does not follow. The authors in [96] claimed that the SCMS algorithm will converge to a point on the principal surface with appropriate dimensionality. This claim was based on the assumption that the MS algorithm always converges, which, as we discussed, has so far been unproven. In addition, it does not seem at all clear that the convergence of the MS algorithm actually implies the convergence of the SCMS algorithm, let alone its convergence to the principal surface. Furthermore, the authors in [96] provided two stopping criteria for the SCMS algorithm, but they did not prove that the algorithm stops after a finite number of iterations when using these two criteria. In other words, although the MS and SCMS algorithms have been widely used in many applications related to information and signal processing, a rigorous study of their convergence properties is still missing. Thus it seems that, similar to most previous principal curve algorithms (with the exception of [74]), no optimality properties for the SCMS algorithm have been proved.

We are interested in investigating the convergence properties of the MS and SCMS algorithms in order to fill some of the gaps between theory and practice. Our goal is to take initial steps to show that the SCMS algorithm inherits some important convergence properties of the MS algorithm. We are also interested in using the MS and the SCMS algorithms for new applications.

1.2 Thesis Organization and Contributions

This dissertation is divided into seven chapters. We give a short survey of some of the existing definitions of principal curves and surfaces in Chapter 2. We start with the original definition of a principal curve given by Hastie and Stuetzle and continue to more recent definitions. A brief review of the proposed algorithms used to find a principal curve based on a given definition is also provided in Chapter 2. In Chapter 3, we review a recent definition of a principal curve/surface given by Ozertem and Erdogmus, which is the main subject of this thesis. The MS and the SCMS algorithm are also briefly reviewed in Chapter 3.

In Chapter 4, we first show that the MS algorithm with isolated stationary points generates a convergent sequence. Then, we provide a sufficient condition for the MS algorithm with the Gaussian kernel to have isolated stationary points. A convergence proof for the MS algorithm for the one-dimensional case is also given in Chapter 4. We show that by slightly modifying the MS algorithm, the convergence can be guaranteed. For the SCMS algorithm, we first provide an alternative interpretation for the points on a principal curve. Then the monotonicity and convergence of the density estimate values along the sequence of output values of the SCMS algorithm are shown. Also, it is shown that the distance between consecutive points of the output sequence converges to zero, as does the projection of the gradient vector onto the subspace spanned by the $D - d$ largest eigenvectors of the local inverse covariance matrix. These last two properties provide theoretical guarantees for the stopping criteria. By modifying the projection step, three variations of the SCMS algorithm are proposed and the running times and performance of the resulting algorithms are compared. Finally, we propose an adaptive version of the SCMS algorithm to consider the effect of new samples.

Nonlinear dimension reduction in the presence of noise is discussed in Chapter 5. In this chapter, we first give a brief review of some of the popular nonlinear dimensionality reduction techniques. Then we show how using the SCMS algorithm as a pre-processing step before LLE and ISOMAP can improve the performance of these techniques in regards to finding a low-dimensional representation of the data points.

Some new applications for the MS and the SCMS algorithms are presented in Chapter 6. We first show that the MS algorithm can be used to accurately find straight lines in digital images. The performance of this method is compared with the Hough transform. We then investigate the application of the SCMS algorithm to the problem of noisy source vector quantization where the clean source needs to be estimated from its noisy observation before quantizing with an optimal vector quantizer. We demonstrate that an SCMS-based preprocessing step can be effective for sources that have intrinsically low dimensionality in situations where clean source samples are unavailable and the system design relies only on noisy source samples for training. We also show how the SCMS algorithm can improve the performance of the kernel regression technique when the explanatory variables are corrupted by noise. Finally, we propose a weighted version of the SCMS algorithm that can be used for skeletonization.

Chapter 7 presents a summary of the thesis.

Chapter 2

Literature Review

Principal Curves and Surfaces

Principal curves and surfaces can be interpreted as a nonlinear generalization of principal component analysis (PCA) [69]. By mapping the high-dimensional observations onto a low-dimensional manifold, embedded in the high-dimensional space, they provide a new representation of the input data that makes tasks such as visualization and dimensionality reduction much easier and more accurate. There are different approaches to defining a principal curve or surface in the literature. In this chapter we give a short survey of some of the existing definitions for the principal curves and surfaces. We also briefly review the proposed algorithms used to find a principal curve or surface based on a given definition.

2.1 The Hastie-Stuetzle Definition

The first formal definition of a principal curve was given by Hastie and Stuetzle [57] (hereafter HSPC). According to their definition, a principal curve is a smooth (infinitely differentiable) one-dimensional curve that passes through the middle of a data set. More formally, a principal curve of a probability distribution is a smooth, self-consistent, parameterized curve that does not intersect itself and has finite length inside any bounded ball. Hastie and Stuetzle's definition of a principal curve relies on the concept of self-consistency. The self-consistency property means that for every point selected on the principal curve, the average of the collection of data points that project onto that point will coincide with the selected point on the curve. This intuitive concept can be made mathematically rigorous as follows.

Definition 2.1. *Let $\boldsymbol{x} \in \mathbb{R}^D$ denote a random vector with probability density function (pdf) f and finite second moments. A smooth parameterized curve $\boldsymbol{h}(l)$ that does not intersect itself and has finite length inside any finite ball in $\mathbb{R}^D$ is a principal curve for the distribution $f(\boldsymbol{x})$ if $\boldsymbol{h}$ is self-consistent. The curve $\boldsymbol{h}$ is called self-consistent if*

$$E(\boldsymbol{X}|\lambda_{\boldsymbol{h}}(\boldsymbol{X}) = \lambda) = \boldsymbol{h}(\lambda), \tag{2.1}$$

where the projection index $\lambda_{\boldsymbol{h}} : \mathbb{R}^D \rightarrow \mathbb{R}$ is defined as $\lambda_{\boldsymbol{h}}(\boldsymbol{x}) = \sup_{\lambda}\{\lambda : \|\boldsymbol{x} - \boldsymbol{h}(\lambda)\| = \inf_{\mu} \|\boldsymbol{x} - \boldsymbol{h}(\mu)\|\}$.

The projection index $\lambda_{\boldsymbol{h}}(\boldsymbol{x})$ of $\boldsymbol{x}$ in Definition 2.1 represents the value of λ for which $\boldsymbol{h}(\lambda)$ is the closest to $\boldsymbol{x}$. If the value of λ is not unique, the $\sup$ operator simply selects the largest value among the candidates. It was shown that the projection index $\lambda_{\boldsymbol{h}}(\boldsymbol{x})$ is well-defined and measurable [57]. Based on Definition 2.1, it can be proved that if a straight line is self-consistent then it is a principal component and as a result for ellipsoidal distributions, the lines determined by principal components are principal curves. Let $d(\boldsymbol{x}, \boldsymbol{h})$ denote the Euclidean distance of point $\boldsymbol{x}$ to its projection on $\boldsymbol{h}$, i.e., $d(\boldsymbol{x}, \boldsymbol{h}) = \|\boldsymbol{x} - \boldsymbol{h}(\lambda_{\boldsymbol{h}}(\boldsymbol{x}))\|$. The mean squared Euclidean distance of points to their projection on the curve is defined by

$$D(f, \boldsymbol{h}) = E_f d^2(\boldsymbol{X}, \boldsymbol{h}) = \int d^2(\boldsymbol{X}, \boldsymbol{h}) f(\boldsymbol{X}) d\boldsymbol{X}, \tag{2.2}$$

where the expected value is computed with respect to the density function f of the data points. By generalizing the minimum distance property of linear principal component, Hastie and Stuetzle proved that a principal curve is a critical point of the mean squared Euclidean distance $D(f, \boldsymbol{h})$. Therefore, we have the following result [56]:

Theorem 2.1. *Let $\boldsymbol{h}$ be a principal curve, and let $\boldsymbol{h}_t$ be a family of smooth curves with $\boldsymbol{h}_0 = \boldsymbol{h}$, then*

$$\frac{d}{dt} D(f, \boldsymbol{h}_t)\bigg|_{t=0} = 0.$$

The authors in [57] observed that if $\boldsymbol{h}_t$ is restricted to be a straight line, then the eigenvectors of the covariance matrix $\boldsymbol{\Sigma}$ satisfy the required conditions in Theorem 2.1 to be a principal curve. Two major theoretical contributions of Hastie and Stuetzle are as follows

1. If a straight line is self-consistent, then it is a principal component (see Chapter 5.2).

2. Based on the minimum mean squared error criterion, principal curves are the stationary points of the mean squared Euclidean distance between the data points and their projection on the curve.

The second property is used to design an algorithm that starts from the principal line and iteratively finds the HSPC by minimizing the averaged squared distance of the data points and the curve. The proposed algorithm to find the principal curve for a data set with density function f in D dimensional space is given as follows [57].

1. Initialization: set $\boldsymbol{h}^0(\lambda) = \bar{\boldsymbol{x}} + \mathbf{a}\lambda$, where $\bar{\boldsymbol{x}}$ is the average of the data set and $\mathbf{a}$ is the first principal component of the covariance matrix of the density f. Set $\lambda^0(\boldsymbol{x}) = \lambda_{\boldsymbol{h}^0}(\boldsymbol{x})$, where $\lambda_{\boldsymbol{h}^0}(\boldsymbol{x})$ is the value of λ for which $\boldsymbol{h}^0(\lambda)$ is closest to $\boldsymbol{x}$.

2. Repeat the following steps for $j \in \mathbb{Z} \geq 1$:

   - Set $\boldsymbol{h}^j(\cdot) = E[\boldsymbol{X}|\lambda_{\boldsymbol{h}^{j-1}}(\boldsymbol{X}) = \cdot]$.
   - Define $\lambda^j(\boldsymbol{x}) = \lambda_{\boldsymbol{h}^j}(\boldsymbol{x})$ for every $\boldsymbol{x} \in f$. Transform λ^j so that $\boldsymbol{h}^j$ is unit speed (a curve with $\|\boldsymbol{h}'\| = 1$ is called a unit speed parameterized curve).
   - Evaluate $D(f, \boldsymbol{h}^j) = E_f[\|\boldsymbol{X} - \boldsymbol{h}(\lambda^j(\boldsymbol{X}))\|^2]$.

     Until: the change in $D(f, \boldsymbol{h}^j)$ is under some threshold.

There are certain problems with the HSPC algorithm. By definition the HSPC are differentiable curves, but there is no guarantee that the curves generated by the conditional expectation will also be differentiable. Discontinuities can happen at the end points of a curve where the mean of the points with the closest distance to an endpoint can be disjointed from the updated new curve. Convergence of the HSPC algorithm is not proved, and therefore existence of the principal curves could be proven only for special cases, such as ellipsoidal distributions. For ellipsoidal distributions (e.g., Gaussian distribution), the principal components are principal curves. Also any subspace spanned by the d largest principal components is a d dimensional principal surface. For spherical symmetric distributions, any straight line passing through the center is a principal curve. The authors in [57] showed that if each iteration generates a differentiable curve, then the expected squared distance $D(f, \boldsymbol{h}^j)$ converges.

The HSPC algorithm generates the principal curve under the assumption that the pdf of the observations is known. In real world applications, we usually do not have access to the probability distribution and we are only provided with a finite data set. In the absence of prior information about the probability distribution, the HSPC algorithm represents a curve $\boldsymbol{h}(\lambda)$ by n tuples $(\lambda_i, \boldsymbol{h}_i)$, where n is the size of the data set. The n tuples are connected together in an increasing order of λ to form a polygon. The projection index λ_i is the arc length along the polygon from $\boldsymbol{h}_1$ to $\boldsymbol{h}_i$ and $\lambda_1 = 0$. Similar to the previous case, the algorithm is initialized to the first principal component line and alternatively repeats the projection and the expectation steps. The iterations stop when the relative change in the average distance $\|D(f, \boldsymbol{h}^j) - D(f, \boldsymbol{h}^{j-1})\| / \|D(f, \boldsymbol{h}^{j-1})\|$ becomes less than some predefined threshold. The average distance $D(f, \boldsymbol{h})$ is estimated by averaging over the squared distances of the points in the sample to their closest points on the current curve. Although the authors in [57] could not prove the convergence of the algorithm and did not show that each step guarantees a decrease in the given criterion, they reported that they have not had any convergence issues in their simulations.

2.2 Principal Curves with a Length Constraint

The existence of the HSPC for special cases, such as the uniform distribution on rectangles and annuli, was investigated in [34][33]. The authors in [34][33] showed that the HSPC are a solution of a differential equation and by solving the differential equation they found principal curves for the uniform distribution on rectangles and annuli. Unfortunately, it is not known if the HSPC exists for a large class of distribution functions in general. To address this problem, Kégl *et al.* redefined the principal curves. They observed that the first principal component minimizes the expected squared distance among all straight lines. Therefore, in contrast to the HSPC, which defined principal curves as critical points of the distance function, Kégl *et al.* [74] defined principal curves as a minimizer of the expected squared distance over a class of curves. The authors in [74] put a constraint on the length of the curve to avoid a principal curve having an infinite length. They also relaxed the requirement of differentiability of a principal curve [70]. The new definition is given as follows. [74]

Definition 2.2. *A curve $\boldsymbol{h}^*$ is called a principal curve with length l for a density f, if $\boldsymbol{h}^*$ minimizes $D(f, \boldsymbol{h})$ over all curves of length less than or equal to l.*

Finding a principal curve based on Definition 2.2 is similar to finding k principal points of a D variate random variable. The k principal points of a random variable $\boldsymbol{x} \in \mathbb{R}^D$ are defined as those points $\boldsymbol{y}_i^* \in \mathbb{R}^D, i = 1, 2, \ldots, k$ that minimize the expected squared distance of $\boldsymbol{X}$ from the nearest of the $\boldsymbol{y}_i^*$ [42]. Therefore, both definitions try to minimize the same expected squared distance criterion. There is a constraint on the number of principal points, whereas there is a length constraint for a principal curve based on Definition 2.2. A further discussion on the principal points and how to estimate them is given in [114][43]. According to the terminology used in [114], a principal curve based on Definition 2.2 is a set of principal points, while a principal curve based on Definition 2.1 is a set of self-consistent points [70]. A set of points $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_k\}$ is called self consistent if [114]

$$\boldsymbol{y}_i = E[\boldsymbol{X} | \boldsymbol{X} \in D_i] \text{ for } i = 1, 2, \ldots, k, \tag{2.3}$$

where $D_i, i = 1, 2, \ldots, k$ form a partition of $\mathbb{R}^D$, the so-called Voronoi partitions with respect to the points $\{\boldsymbol{y}_i, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_k\}$ [17]. In this sense, the set $\mathcal{C} = \{\boldsymbol{y}_1, \ldots \boldsymbol{y}_k\}$ represents a codebook corresponding to a k-point vector quantizer satisfying the necessary optimality conditions [86].

The authors in [74] proved the existence of principal curves based on this definition and proposed an algorithm for constructing them. The proposed algorithm is initialized with the shortest segment of the first principal component line, which contains the projected data set [73]. The data set is then partitioned into disjoint sets based on the Euclidean distance between the data points and vertices and segments. The projection and vertex optimization steps are done iteratively until convergence occurs, and then a new vertex is added to the curve. The algorithm stops when the total number of vertices k exceeds a predefined threshold [74]. Selecting the number of segments k and the curve length l are an essential issue, since an inappropriate choice of them may result in a poor estimation of the principal curve. In a recent work, the parameter selection issue using the point of view of model selection via penalization is addressed [7].

2.3 Alternative Definitions

Hastie and Stuetzle showed that if each iteration in the proposed algorithm is well defined and generates a differentiable curve, then the expected value of the squared Euclidean distance $D(f, \boldsymbol{h})$ converges. Unfortunately, convergence of $D(f, \boldsymbol{h})$ does not imply that the estimated curve $\boldsymbol{h}$ is a meaningful solution to the problem. Duchamp and Stuetzle showed that while HSPCs are critical points of $D(f, \boldsymbol{h})$, they are not local minima [34]. In fact, the authors in [34] proved that the HSPC are saddle points of the expected value of the squared distance function, therefore any algorithm that tries to estimate a principal curve by minimizing the distance function will fail to converge to a stable solution. The HSPCs are biased at points of large curvature, and the authors in [57] exclude the handling of crossings or closed curves. There are also two sources of bias during the estimation of a principal curve based on Definition 2.1: model bias and estimation bias. The problems with the HSPC motivated the researchers to modify the HSPC definition or give new definitions to overcome the aforementioned difficulties.

2.3.1 Modified HSPC

The estimation bias occurs because of averaging over neighborhoods in order to estimate the conditional expectations [57]. As a result, the generated curve is biased toward the center of the curvature. The estimation bias increases as the size of the local neighborhood that is used for the averaging increases. In other words, we get a smoother curve at the price of having more estimation bias. The estimation bias problem was addressed by Banfield and Raftery [6]. They used the HSPC definition and extended it to closed curves to model the outlines of ice floes in satellite images. They modified the algorithm in [57] and reduced the estimation bias by estimating the error residual instead of the actual curve during the computation. According to [6], the estimated curve $\boldsymbol{h}$ is updated at the j-th iteration as

$$\boldsymbol{h}^{j+1}(\lambda) = \boldsymbol{h}^{j}(\lambda) + \boldsymbol{b}^{j}(\boldsymbol{X}, \lambda), \tag{2.4}$$

where $\boldsymbol{b}^{j}(\boldsymbol{X}, \lambda) = E\Big(\boldsymbol{X} - \boldsymbol{h}^{j}(\lambda) | \lambda_{\boldsymbol{h}^{j}}(\boldsymbol{X}) = \lambda\Big)$ can be thought of as the measure of the estimation bias at the $(j+1)$-th iteration at $\boldsymbol{h}^{j}(\lambda)$. The projection residual of the data point $\boldsymbol{x}_i$ projected onto $\boldsymbol{h}^{j}$ is defined by $\boldsymbol{p}_i^{j} = \boldsymbol{x}_i - \boldsymbol{h}^{j}(\lambda_{\boldsymbol{h}^{j}}(\boldsymbol{x}_i))$. Therefore, the expected value of the projection residuals of the data points that project onto $\boldsymbol{h}^j$ at λ is the estimation bias measure $\boldsymbol{b}^j(\boldsymbol{x}, \lambda)$. In situations where the distribution of the data is unknown, the authors in [6] suggested that a weighted average of the projection residuals of the data points, rather than data points, should be used to update $\boldsymbol{h}^{j+1}(\lambda)$. So, when the distribution of data is not available, $\boldsymbol{h}^{j+1}(\lambda)$ is given by

$$\boldsymbol{h}^{j+1}(\lambda^j_{\boldsymbol{h}}(\boldsymbol{x}_i)) = \boldsymbol{h}^j(\lambda^j_{\boldsymbol{h}}(\boldsymbol{x}_i)) + \bar{\boldsymbol{p}}_i^{\,j}, \tag{2.5}$$

where $\bar{\boldsymbol{p}}_i^{\,j}$ is the weighted average of the projection residuals of the data points. Similar to the HSPC algorithm, there is no formal proof of the convergence for the modified version proposed in [6]. Furthermore, the modified HSPC algorithm in [6] introduces numerical instabilities that may lead to a smooth but incorrect principal curve in practice.

2.3.2 Tibshirani's Definition

Model bias occurs when the data has an additive form. Let $\epsilon = (\epsilon_1, \epsilon_2, \ldots, \epsilon_D)$ and $E(\epsilon) = \mathbf{0}$ for random variables $\epsilon_1, \epsilon_2, \ldots, \epsilon_D$. Hastie and Stuetzle observed that if $\boldsymbol{x} = \boldsymbol{h}(\lambda) + \epsilon$, where λ and $\epsilon_i, i = 1, \ldots, D$ are independent, then the HSPC may not generate $\boldsymbol{h}$ as the principal curve of the distribution of $\boldsymbol{x}$. Instead, the HSPC generates a biased version of $\boldsymbol{h}$ even if it is initialized at the generating curve. The model bias is proportional to the ratio of the noise [1] variance to the radius of curvature. By relaxing the self-consistency property, Tibshirani gave a different definition of a principal curve to address the this problem. Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_D)$ be a D-dimensional random vector with density $g_{\boldsymbol{x}}$. Assume that the random vector $\boldsymbol{x}$ is generated by an additive model in two stages as follows

[1] In the additive model $\boldsymbol{x} = \boldsymbol{h}(\lambda) + \epsilon$, the second term, ϵ, can be considered as noise.

1. A point on a parameterized curve $\boldsymbol{h}(\lambda)$ is generated according to some distribution g_λ.

2. The random vector $\boldsymbol{x}$ is generated from a conditional distribution $g_{\boldsymbol{x}|\lambda}$ such that $E(\boldsymbol{x}|\lambda) = \boldsymbol{h}(\lambda)$ and $x_1, \dots, x_D$ are conditionally independent given λ.

Using the above model, Tibshirani defined the principal curve for density $g_{\boldsymbol{x}}$ as follows

Definition 2.3. *The principal curve for a random variable $\boldsymbol{x}$ with distribution $g_{\boldsymbol{x}}$ is a triplet $(g_\lambda, g_{\boldsymbol{x}|\lambda}, \boldsymbol{h})$ that satisfies the following conditions:*

1. *Two distributions g_λ and $g_{\boldsymbol{X}|\lambda}$ are consistent with $g_{\boldsymbol{X}}$, that is $g_{\boldsymbol{X}}(\boldsymbol{x}) = \int g_{\boldsymbol{X}|\lambda}(\boldsymbol{x}|\mu)g_\lambda(\mu)d\mu$.*

2. *$x_1, x_2, \dots, x_D$ are conditionally independent given λ.*

3. *$\boldsymbol{h}(t)$ is a parameterized curve in $\mathbb{R}^D$, where t is on a closed interval in $\mathbb{R}$ and $\boldsymbol{h}(t) = E(\boldsymbol{X}|\lambda = t)$.*

The main advantage of Definition 2.3 is that it solves the model bias problem. According to Definition 2.3, if a random vector $\boldsymbol{x}$ is the result of adding noise to a random point over a one-dimensional curve $\boldsymbol{h}$, then the generative $\boldsymbol{h}$ is the principal curve of $\boldsymbol{x}$ (The principal surface is defined in the same way.) Based on Definition 2.3, Tibshirani proposed a semi-parametric model that estimates a principal curve via the expectation maximization (EM) algorithm. Although Tibshirani's definition solved the model bias issue, there is no evidence that the proposed procedure works any better than that given by Hastie and Stuetzle and the self-consistency property no longer holds. As well, Tibshirani's approach does not seem to be flexible enough to recover curves with high curvature. This problem has been addressed in [120] by using polygonal lines to estimate the principal curve that generate unsmooth curves. In a later work, LeBlanc and Tibshirani used multivariate adaptive regression splines to develop the estimation procedure of the principal curves and surfaces [80].

2.3.3 Principal oriented points

Let $\boldsymbol{X} \in \mathbb{R}^D$ be a multivariate normal random variable with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, i.e., $\boldsymbol{X} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. A well-known property of the first principal component of multivariate normal distribution states that the total variance [1] of the conditional distribution of $\boldsymbol{X}$, given that $\boldsymbol{X}$ belongs to a hyperplane, is minimal when the hyperplane is orthogonal to the first principal component [27], i.e., the normal vector $\boldsymbol{b}$ is in the direction of the first eigenvector of $\boldsymbol{\Sigma}$. By generalizing the above observation, Delicado [28] introduced the notion of principal oriented points (POP) and defined a principal curve as a one-dimensional curve that passes through POPs. Let

$$\mu(\boldsymbol{x}_0, \boldsymbol{b}) = E(\boldsymbol{x} | \boldsymbol{x} \in H(\boldsymbol{x}_0, \boldsymbol{b})),$$
$$\phi(\boldsymbol{x}_0, \boldsymbol{b}) = TV(\boldsymbol{x} | \boldsymbol{x} \in H(\boldsymbol{x}_0, \boldsymbol{b})),$$

where TV is the total variance. A vector $\boldsymbol{b}^*$ is called the principal direction for $\boldsymbol{x}_0$ if $\boldsymbol{b}^*(\boldsymbol{x}_0) = \arg\min_{\|\boldsymbol{b}\|=1} \phi(\boldsymbol{x}_0, \boldsymbol{b})$, and $\mu^*(\boldsymbol{x}_0) = \mu(\boldsymbol{x}_0, \boldsymbol{b}^*(\boldsymbol{x}_0))$ is called a POP [27]. Delicado gave a new definition of a principal curve as follows [28]

Definition 2.4. *Let $\boldsymbol{h} : [a, b] \to \mathbb{R}^D$ be a continuous curve that is parameterized by the arc*

[1] The total variance for a vector is defined as the sum of the variances of all elements

length. It is a principal curve of oriented points (PCOP) if

$$\{\boldsymbol{h}(\lambda) : \lambda \in [a, b]\} \subset \mathbf{X},$$

where $\mathbf{X}$ is the set of all POPs.

According to Definition 2.4, for a multivariate normal distribution only the first principal component can be considered as a principal curve; however, based on Definition 2.1 every principal component satisfies the self-consistency property and can be a HSPC. Delicado proved that POPs exist for theoretical distributions [1][27]. In contrast to the self-consistency property, which is defined for a curve (or for a set of points), the POP property is a point property, which means that regardless of knowing the underlying principal curve it can be verified if an arbitrary point $\boldsymbol{x}_0$ is a POP, i.e., $\boldsymbol{x}_0 = \mu^*(\boldsymbol{x}_0)$.

Delicado also proposed an algorithm to find principal oriented points for a given data set and by using smoothing techniques tried to find a smoother version of the polygonal curve (PCOP) passing through these points [28]. Similar to the HSPC, there is a bias for the PCOP when the data are generated in the form $\boldsymbol{x} = \boldsymbol{h}(\lambda) + \boldsymbol{e}$ and the PCOP is unable to recover the generative curve $\boldsymbol{h}$ [28]. The simulation results in [28] and [27] indicate that the PCOPs are less smooth than HSPCs. One reason for this observation could be the small size of the set of the POPs (compared with a higher number of points generated by the HSPC algorithm), which causes the smoothing techniques to generate less smooth curves compared to the HSPC algorithm.

[1] Distributions that describe or define a probability model are called theoretical distributions, e.g., normal distribution.

Chapter 3

Locally Defined Principal Curves and Surfaces

3.1 Introduction

An interesting new definition of principal curves and surfaces has recently been proposed by Ozertem and Erdogmus [96]. The principal curves generated based on the new definition correspond to the ridge of the probability density function (pdf). Based on the new definition, every point on a principal curve/surface is the local maximum of the pdf in the orthogonal subspace. This contrasts with the self-consistency property, which states that every point on a principal curve coincides with the expected value of the points in the orthogonal subspace of the principal curve at that point. According to the definition in [96], given a smooth pdf f on $\mathbb{R}^D$, a d-dimensional principal surface ($d < D$) is the collection of all points where the gradient of f is orthogonal to exactly $D - d$ eigenvectors of the Hessian of f and the eigenvalues corresponding to these eigenvectors are negative [95]. Thus each point on the principal surface is a local maximum of the pdf in a $(D - d)$-dimensional affine subspace and the principal surface is a d-dimensional ridge of the pdf. An attractive property of this new definition is that the smoothness of the principal curves and surfaces is not stipulated by their definition, but rather it is inherited from the smoothness of the underlying pdf or its estimate.

To estimate principal curves/surfaces based on the new definition, [96] proposed the so-called subspace constrained mean shift (SCMS) algorithm. It is a generalization of the well-known mean shift (MS) algorithm [44] that iteratively tries to find modes of a pdf (estimated from data samples) in a local subspace. On synthetic data sets, the performance of the SCMS algorithm is comparable to (and in some situations better than) the principal curve algorithms of Hastie and Stuetzle [57] and Kégl *et al.* [74], and it is computationally less demanding. Moreover, in contrast to most previous principal curve algorithms, the SCMS algorithm can naturally handle loops and self intersections, and it easily generalizes from principal curves to surfaces.

3.2 Locally Defined Principal Curves and Surfaces

Let f be a smooth (at least twice continuously differentiable) pdf on $\mathbb{R}^D$ with gradient ∇f and Hessian $\boldsymbol{H}$. The d-dimensional critical set $\mathcal{C}^d$ is defined as the set of all points $\boldsymbol{x}$ such that the gradient vector $\nabla f(\boldsymbol{x})$ is orthogonal to at least $D - d$ eigenvectors of the Hessian matrix $\boldsymbol{H}(\boldsymbol{x})$ [96]. Therefore, $\mathcal{C}^0$ consists of all critical points of f, and it is trivial to show $\mathcal{C}^d \subset \mathcal{C}^{d+1}$. A point $\boldsymbol{x}$ is called regular if $\boldsymbol{x} \in \mathcal{C}^d - \mathcal{C}^{d-1}$, otherwise it is called an irregular point. Thus, for a regular point $\boldsymbol{x}$, the gradient vector $\nabla f(\boldsymbol{x})$ is orthogonal to exactly $D-d$ eigenvectors of the Hessian matrix $\boldsymbol{H}(\boldsymbol{x})$ and the subspace spanned by these eigenvectors is called the normal subspace of $\mathcal{C}^d(\boldsymbol{x})$, denoted by $\mathcal{C}^d_{\perp}$. The orthogonal subspace is called the tangent space and denoted by $\mathcal{C}^d_{\parallel}(\boldsymbol{x})$, i.e., $\mathbb{R}^d = \mathcal{C}^d_{\perp} \cup \mathcal{C}^d_{\parallel}(\boldsymbol{x})(\boldsymbol{x})$. Ozertem proved the following result [95]

Theorem 3.1. *Let $\boldsymbol{x}$ be a regular point of C^d and I be an index set with cardinality $|I| = (D-d)$ such that $\nabla f(\boldsymbol{x})^T v_i(\boldsymbol{x}) = 0$ if and only if $i \in I$, where v_i are the eigenvectors of the Hessian matrix at $\boldsymbol{x}$. The following statements hold:*

1. *$\boldsymbol{x}$ is a local maximum in $\mathcal{C}^d_{\perp}(\boldsymbol{x})$ if and only if $\lambda_i(\boldsymbol{x}) < 0,\ \forall i \in I$.*

2. *$\boldsymbol{x}$ is a local minimum in $\mathcal{C}^d_{\perp}(\boldsymbol{x})$ if and only if $\lambda_i(\boldsymbol{x}) > 0,\ \forall i \in I$.*

3. *$\boldsymbol{x}$ is a saddle point in $\mathcal{C}^d_{\perp}(\boldsymbol{x})$ if and only if $\exists \lambda_i(\boldsymbol{x}) < 0$ and $\exists \lambda_j(\mathrm{x}) > 0$ for $i, j \in I$,*

where $\lambda_i(\boldsymbol{x}), i \in I$ is an eigenvalues of the Hessian matrix of f at $\boldsymbol{x}$ corresponding to $v_i(\boldsymbol{x}), i \in I$. Let the principal set $\mathcal{P}^d$ consist of all the local maxima of $\mathcal{C}^d_{\perp}(\boldsymbol{x})$. Then, P^0 consists of the local maxima of $f(\boldsymbol{x})$ and we have $\mathcal{P}^d \subset \mathcal{P}^{d+1}$. According to the new definition, members of the principal set $\mathcal{P}^d$ are points on a d-dimensional principal surface. In other words, for $d \in \{0, 1, \ldots, D-1\}$, Ozertem and Ergodmus defined the d-dimensional principal surfaces associated with the pdf f as follows [96]

Definition 3.1. *The d-dimensional principal surface associated with pdf f is the collection of all points $\boldsymbol{x} \in \mathbb{R}^D$ such that the gradient $\nabla f(\boldsymbol{x})$ is orthogonal to exactly $D-d$ eigenvectors of the Hessian $\boldsymbol{H}(\boldsymbol{x})$ and the eigenvalues of $\boldsymbol{H}(\boldsymbol{x})$ corresponding to these $D-d$ orthogonal eigenvectors are negative.*

For the one-dimensional ($d = 1$) case, this definition simplifies to the following: the one-dimensional principal surface (principal curve) is the collection of all points $\boldsymbol{x} \in \mathbb{R}^D$

at which the gradient of the pdf is an eigenvector of the Hessian of the pdf and the rest of the eigenvectors of the Hessian have negative eigenvalues. Clearly, all points on a d-dimensional principal surface in Definition 3.1 are local maxima of the pdf in a local affine orthogonal $D - d$-dimensional subspace. In other words, a principal curve is a ridge of the pdf, and every point on the principal curve is a local maximum of the pdf in the affine subspace orthogonal to the curve. Thus Ozertem and Ergodmus' definition replaces Hastie and Stuetzle's requirement that every point on the principal curve be the conditional expectation of the pdf in a local orthogonal subspace with the requirement that the pdf have a local maximum in a local orthogonal subspace.

For the Gaussian distribution, the principal surfaces of Definition 3.1 coincide with the subspaces spanned by the eigenvectors of the covariance matrix, which reveals the connection with principal component analysis [96]. According to Definition 3.1, a principal curve always exists as long as the pdf is twice differentiable such that the Hessian matrix is nonzero. In practice, the underlying pdf is usually unknown. Hence, if the pdf is estimated using methods such as kernel density estimation (KDE) with a Gaussian kernel then it is guaranteed to have non zero Hessian matrix. Further existence issues and properties of the new definition for principal surfaces were not treated in detail in [96], but an effective iterative algorithm was given. This algorithm is based on the well-known mean shift (MS) procedure, which we review before turning to the subspace constrained mean shift algorithm (SCMS) of [96].

3.3 The Mean Shift Algorithm

The modes of a probability density function play an important role in many pattern recognition applications, including classification [1], clustering [83], multi-valued regression [11], image segmentation [18], and object tracking [20]. The mean shift (MS) algorithm is a simple non-parametric iterative method introduced by Fukunaga and Hostetler [44] for locating modes of a pdf obtained via a kernel density estimate (see, e.g., [111]) from a given data set. The algorithm was generalized by Cheng [16] in order to show that the MS algorithm is a mode seeking process on a surface constructed with a shadow kernel. Later, the algorithm became popular in the machine learning community when its potential usage for feature space analysis was studied [18]. The MS algorithm iteratively shifts each data point to a weighted average of neighboring points to find stationary points of the estimated pdf. The MS algorithm can be used as a clustering tool, where each mode represents a cluster. In contrast to the k-mean clustering approach, the MS algorithm does not require any prior knowledge of the number of clusters and there is no assumption for the shape of the clusters. In recent years, the algorithm has been successfully used for applications such as image segmentation [131][122], edge detection [132][54], object tracking [68][130], and information fusion [83].

A D-variate kernel $K : \mathbb{R}^D \to \mathbb{R}$ is a non-negative real-valued function that satisfies the following conditions [121]

$$\int_{\mathbb{R}^D} K(\boldsymbol{x})d\boldsymbol{x} = 1, \quad \lim_{\|\boldsymbol{x}\|\to\infty} \|\boldsymbol{x}\|^D K(\boldsymbol{x}) = 0,$$
$$\int_{\mathbb{R}^D} \boldsymbol{x}K(\boldsymbol{x})d\boldsymbol{x} = 0, \quad \int_{\mathbb{R}^D} \boldsymbol{x}\boldsymbol{x}^T K(\boldsymbol{x})d\boldsymbol{x} = c_K\boldsymbol{I},$$

where c_K is a constant and $\boldsymbol{I}$ is the identity matrix. Let $\boldsymbol{x}_i \in \mathbb{R}^D, i = 1, \ldots, n$ be a sequence of n independent and identically distributed (iid) random variables. The kernel density estimate $\hat{f}$ at an arbitrary point $\boldsymbol{x}$ using a kernel $K(\boldsymbol{x})$ is given by

$$\hat{f}(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} K_{\mathbf{H}}(\boldsymbol{x} - \boldsymbol{x}_i), \tag{3.1}$$

where $K_{\mathbf{H}}(\boldsymbol{x}) = |\mathbf{H}|^{-1/2} K(\mathbf{H}^{-1/2}\boldsymbol{x})$, $\mathbf{H}$ is a symmetric positive definite $D \times D$ matrix called the bandwidth matrix, and $|\mathbf{H}|$ denotes the determinant of $\mathbf{H}$. A special class of kernels, called radially symmetric kernels, has been widely used for pdf estimation. Radially symmetric kernels are defined by $K(\boldsymbol{x}) = c_{k,D} k(\|\boldsymbol{x}\|^2)$, where $c_{k,D}$ is a normalization factor that causes $K(\boldsymbol{x})$ to integrate to one and $k : [0, \infty) \to [0, \infty)$ is called the profile of the kernel. The profile of a kernel is assumed to be a non-negative, non-increasing, and piecewise continuous function that satisfies $\int_0^\infty k(x)dx < \infty$. Two widely used kernel functions are the Epanechnikov kernel and the Gaussian kernel, defined by

1. Epanechnikov kernel

$$K_E(\boldsymbol{x}) = \begin{cases} \frac{1}{2} c_D^{-1} (D+2)(1 - \|\boldsymbol{x}\|^2) & \text{if } \|\boldsymbol{x}\| \leq 1 \\ 0 & \text{if } \|\boldsymbol{x}\| > 1. \end{cases}$$

   where c_D is the volume of the unit D-dimensional sphere.

2. Gaussian kernel

$$K_N(\boldsymbol{x}) = (2\pi)^{-D/2} \exp\Big(- \frac{\|\boldsymbol{x}\|^2}{2} \Big).$$

The probability density estimation that results from this technique is asymptotically unbiased and consistent in the mean square sense [98]. For the sake of simplicity, the bandwidth matrix $\mathbf{H}$ is chosen to be proportional to the identity matrix, i.e., $\mathbf{H} = h\boldsymbol{I}$. Then, by using the profile k and the bandwidth h, the estimated pdf changes to the following well-known form [36]

$$\hat{f}_{h,k}(\boldsymbol{x}) = \frac{c_{k,D}}{nh^D} \sum_{i=1}^{n} k(\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\|^2). \tag{3.2}$$

Assuming that k is differentiable with derivative k', taking the gradient of (3.2) yields

$$\nabla \hat{f}_{h,k}(\boldsymbol{x}) = \frac{2c_{k,D}}{nh^{D+2}} \Big[\sum_{i=1}^{n} g\big(\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\|^2\big) \Big] \Big[\frac{\sum_{i=1}^{n} \boldsymbol{x}_i g(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2)}{\sum_{i=1}^{n} g(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2)} - \boldsymbol{x} \Big], \tag{3.3}$$

where $g(x) = -k'(x)$. The first term in the above equation is proportional to the density estimate at $\boldsymbol{x}$ using kernel $G(\boldsymbol{x}) = c_{g,D} g(\|\boldsymbol{x}\|^2)$. The second term is called the mean shift (MS) vector, $\boldsymbol{m}_{h,g}(\boldsymbol{x})$, and (3.3) can be rewritten in the following form

$$\nabla \hat{f}_{h,k}(\boldsymbol{x}) = \hat{f}_{h,g}(\boldsymbol{x}) \frac{2c_{k,D}}{h^2 c_{g,D}} \boldsymbol{m}_{h,g}(\boldsymbol{x}). \tag{3.4}$$

The above expression indicates that the MS vector computed with bandwidth h and profile g is proportional to the normalized gradient density estimate obtained with the profile k (normalization is done by density estimate with profile g). Therefore, the MS vector always points toward the direction of the maximum increase in the density function. In fact, the MS algorithm is an instance of the gradient ascent algorithm with an adaptive step size [39].

The modes of the estimated density function are located at the zeros of the gradient function, i.e., $\nabla \hat{f}(\boldsymbol{x}) = 0$. Equating (3.3) to zero, reveals that the modes of the estimated

Table 3.1: Mean shift algorithm

$a)$ Initialize the mode estimate $\boldsymbol{y}_0$ to be one of the observed data. Set $j = 0$.

$b)$ Compute the mean shift vector $\boldsymbol{m}_{h,g}(\boldsymbol{y}_j) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g\left(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|\right)}{\sum_{i=1}^{n} g\left(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|\right)} - \boldsymbol{y}_j$.

$c)$ Update the mode estimate $\boldsymbol{y}_{j+1} = \boldsymbol{y}_j + \boldsymbol{m}(\boldsymbol{y}_j)$. Increment j.

$d)$ Iterate (b) and (c) until $|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j| < \epsilon$, where ϵ is a predefined threshold.

pdf are fixed points of the function

$$\mathbf{m}_{h,g}(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g\left(\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\|^2\right)}{\sum_{i=1}^{n} g\left(\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\|^2\right)} - \boldsymbol{x}, \tag{3.5}$$

The MS algorithm initializes the mode estimate sequence to be one of the observed data. The mode estimate $\boldsymbol{y}_j$ in the jth iteration is updated as

$$\begin{aligned} \boldsymbol{y}_{j+1} &= \boldsymbol{y}_j + \boldsymbol{m}(\boldsymbol{y}_j) \\ &= \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g\left(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2\right)}{\sum_{i=1}^{n} g\left(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2\right)}. \end{aligned} \tag{3.6}$$

The MS algorithm iterates this step until the norm of the difference between two consecutive mode estimates becomes less than some predefined threshold. The MS algorithm is summarized in Table 3.1. Typically n instances of the MS algorithm are run in parallel, with the ith instance initialized to the ith data point.

Although the MS algorithm has been used in different applications, a rigorous proof for the convergence of the algorithm has not been given. The following statement is claimed to be true about the MS algorithm [18]: if the kernel K has a convex, monotonically decreasing, and bounded profile, the mode estimate sequence $\{\boldsymbol{y}_j\}_{j=1,2,...}$ and the sequence $\{\hat{f}_{h,k}(\boldsymbol{y}_j)\}$ converge. The authors in [18] successfully showed that the sequence $\{\hat{f}_{h,k}(\boldsymbol{y}_j)\}_{j=1,2,...}$ is an increasing and convergent sequence. However, an error was pointed out in [84] in the proof of the main statement of Theorem 1 in [18], which claims that the sequence $\{\boldsymbol{y}_i; i = 1, 2, \ldots\}$ converges. Through further manipulation of the proof in [18], it can be shown that $\lim_{k\to\infty} \|\boldsymbol{y}_{k+1} - \boldsymbol{y}_k\| \to 0$, which does not imply convergence of the mode estimate sequence $\{\boldsymbol{y}_j\}$. Carreira-Perpiñán [14] showed that the MS algorithm with the Gaussian kernel $K(\boldsymbol{x}) = c\, e^{-\|\boldsymbol{x}\|^2}$ is an instance of the EM algorithm and claimed that this fact implies the convergence of $\{\boldsymbol{y}_j\}$. However, without additional conditions the EM algorithm may not converge (see [9] or [127]), and so it appears that the convergence of the MS algorithm has not yet been proved. Incidentally, the error in the original proof for the convergence of the EM algorithm in [29] and the error in the proof of the convergence of the MS algorithm in [18] are both due to the same incorrect use of the triangle inequality.

3.4 Gaussian Blurring Mean Shift Algorithm

Let κ be the average number of iterations in the mean shift algorithm. The computational cost of each iteration is $O(Dn)$ (D is the dimensionality of the space), so applying the mean shift algorithm to the entire data set has cost $O(\kappa D n^2)$. This is particularly expensive for applications like image segmentation where the total number of pixels is large. By only considering k nearest neighbors for each data point, the computational cost for each iteration can be reduced. Decreasing the average number of iterations is an alternative way to reduce the computational cost. Gaussian blurring mean shift (GBMS) is a technique to reduce the average number of iterations [13]. It uses the Gaussian kernel and in every iteration an updated data set is used. It is expected that the GBMS algorithm leads to a

Table 3.2: Gaussian blurring mean shift algorithm

$a)$ For $m = 1, \ldots, n$ compute $\boldsymbol{y}_m = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i e^{\frac{-1}{2}\|\frac{\boldsymbol{x}_i - \mathbf{x}_m}{\sigma}\|^2}}{\sum_{i=1}^{n} e^{\frac{-1}{2}\|\frac{\boldsymbol{x}_i - \mathbf{x}_m}{\sigma}\|^2}}$.

$b)$ $\forall m : \boldsymbol{x}_m \leftarrow \boldsymbol{y}_m$ and stop when the average update drops below some small tolerance, otherwise go to step (a).

data set where all points coincide. However, if GBMS stops before the clusters start to move toward each other then it can be used as a clustering tool. The GBMS algorithm is summarized in Table 3.2 [12]. The typical behavior of the GBMS algorithm has two phases. First, points merge into compact clusters, which takes a few iterations. In the second phase, which may take several hundred iterations, the clusters move toward each other and finally merge into a single point. It is desirable to stop the GBMS algorithm right after the first phase where points merge into clusters. The following stopping criteria can be used to terminate the iterations

$$\frac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{x}_i^j - \boldsymbol{x}_i^{j-1}\| < \epsilon.$$

Simulation results show that the average number of iterations for each data point is dramatically reduced using the GBMS algorithm. The computational cost for the GBMS algorithm is $O(\eta\kappa D n^2)$, where $\eta < 1$. Let $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ be a $D \times n$ matrix of data points and $\boldsymbol{W}$ be a $n \times n$ symmetric matrix whose $(i, j)th$ component is $w_{ij} = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/(2h^2))$. Then each iteration of the GBMS algorithm can be written in the following matrix form [13]

$$\boldsymbol{X}_{update} = \boldsymbol{X}\boldsymbol{W}\boldsymbol{D}^{-1},$$

where $\boldsymbol{D}$ is a diagonal matrix, whose diagonal elements are $\sum_{i=1}^{n} w_{im}$. Although the GBMS algorithm is faster than the mean shift algorithm, clustering large data sets is still computationally expensive because the computational cost of the GBMS algorithm is proportional to $O(n^2)$.

3.5 Subspace Constrained Mean Shift Algorithm

Under some regularity conditions, the set of local maxima of a pdf is exactly the zero-dimensional principal manifold, $\mathcal{P}^0$, resulting from Definition 3.1 for $d = 0$. The SCMS algorithm [96] generalizes the MS algorithm to estimate higher order principal curves and surfaces ($d \geq 1$). Similar to the MS algorithm, the SCMS algorithm starts from a finite data set sampled from the probability distribution and forms a kernel density estimate $\hat{f}$ based on the data, and it evaluates the MS vector in each iteration. However, the SCMS algorithm projects the mean shift vector to the local (affine) subspace spanned by the $D-d$ eigenvectors corresponding to the $D-d$ largest eigenvalues of the so-called local inverse covariance matrix of the pdf estimate at that point, given by

$$\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}) = -\hat{\boldsymbol{H}}(\boldsymbol{x})\hat{f}(\boldsymbol{x})^{-1} + \nabla\hat{f}(\boldsymbol{x})\nabla\hat{f}(\boldsymbol{x})^{T}\hat{f}(\boldsymbol{x})^{-2}, \tag{3.7}$$

where $\hat{\boldsymbol{H}}(\boldsymbol{x})$ and $\nabla\hat{f}(\boldsymbol{x})$ are the Hessian and gradient of the pdf estimate at $\boldsymbol{x}$. Note that $\hat{\boldsymbol{\Sigma}}^{-1}$ is the negative Hessian of the logarithm of $\hat{f}$. The main reason that the above definition is attractive is its connection to the Gaussian distribution. If the underlying density has the Gaussian distribution then the projection subspace coincides with the subspace spanned by the principal components.

For the special case of the Gaussian distribution $\hat{f} \sim N(\mu, \boldsymbol{\Sigma})$, we have

$$\begin{aligned}\hat{f}(\boldsymbol{x}) &= C_{\boldsymbol{\Sigma}} \exp\Big(-\frac{\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}}{2}\Big),\\ \nabla\hat{f}(\boldsymbol{x}) &= -\hat{f}(\boldsymbol{x})\boldsymbol{\Sigma}^{-1}\boldsymbol{x},\\ \hat{\boldsymbol{H}}(\boldsymbol{x}) &= \hat{f}(\boldsymbol{x})(\boldsymbol{\Sigma}^{-1}\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1} - \boldsymbol{\Sigma}^{-1}),\end{aligned}$$

where $C_{\boldsymbol{\Sigma}}$ is the normalization factor. Then the local inverse covariance matrix in (3.7) becomes constant and equal to the inverse covariance matrix, i.e., $\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}) = \boldsymbol{\Sigma}^{-1}$. The subspace spanned by the $D-d$ eigenvectors corresponding to the $D-d$ largest eigenvalues of $\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x})$ coincides with the subspace spanned by the last $D-d$ principal components. Therefore, through projection into this subspace in each iteration, the SCMS algorithm tries to find local maxima in the d-dimensional space (d-dimensional principal surface) spanned by the first d principal components. The SCMS algorithm can be summarized as follows

1. Set $\epsilon > 0$, $j = 1$, and initialize the SCMS algorithm to an arbitrary point $\boldsymbol{y}_1$.

2. Evaluate the mean shift vector $\boldsymbol{m}(\boldsymbol{y}_j)$ using (3.5).

3. Evaluate the gradient, the Hessian matrix, and the local inverse covariance matrix $\hat{\boldsymbol{\Sigma}}^{-1}$ given in (3.7) at $\boldsymbol{y}_j$. Perform the eigendecomposition of $\hat{\boldsymbol{\Sigma}}_j^{-1} = \hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{y}_j)$ and find its eigenvalues and eigenvectors.

4. Let $\boldsymbol{V}_j = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{D-d}]$ be the $D \times (D-d)$ matrix whose columns are the $D-d$ orthonormal eigenvectors corresponding to the $D-d$ largest eigenvalues of $\hat{\boldsymbol{\Sigma}}_j^{-1}$.

5. Compute $\boldsymbol{y}_{j+1} = \boldsymbol{V}_j\boldsymbol{V}_j^T\boldsymbol{m}(\boldsymbol{y}_j) + \boldsymbol{y}_j$.

6. Stop if $\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| < \epsilon$; otherwise increment j by 1 and go to step 2.

Remark. In [96], the stopping rule $\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| < \epsilon$ was suggested as an alternative to the recommended rule,

$$\frac{\|\boldsymbol{V}_{j+1}^T \nabla \hat{f}(\boldsymbol{y}_{j+1})\|}{\|\nabla \hat{f}(\boldsymbol{y}_{j+1})\|} < \epsilon$$

which is meant to check if the gradient is (nearly) orthogonal to the subspace spanned by the columns of $\boldsymbol{V}_j$. However, this criterion seems to be problematic (e.g., the denominator is zero if the algorithm starts at a stationary point). We will later consider the following simpler stopping rule of a similar flavor:

6'. Stop if $\|\boldsymbol{V}_{j+1}^T \nabla \hat{f}(\boldsymbol{y}_{j+1})\| < \epsilon$; otherwise increment j by 1 and go to step 2.

Typically, n instances of the SCMS algorithm are run, each time initialized to one of the n data points. The resulting n output points are considered as a discrete approximation to the underlying principal curve or surface; see the illustrative example in Figure 3.1. In both the MS and the SCMS algorithms, the stopping threshold ϵ is set manually so that a good tradeoff between running time and approximation accuracy is achieved. The problem of selecting the bandwidth h for the MS algorithm is discussed in detail in [18], and variable-bandwidth, locally-adaptive MS algorithms are introduced and investigated in [19]. The bandwidth selection problem for the SCMS algorithm is discussed in detail in [96], but it is not clear that the automatic rules suggested from the literature of kernel density estimation are in any way optimal when applied in conjunction with the SCMS algorithm.

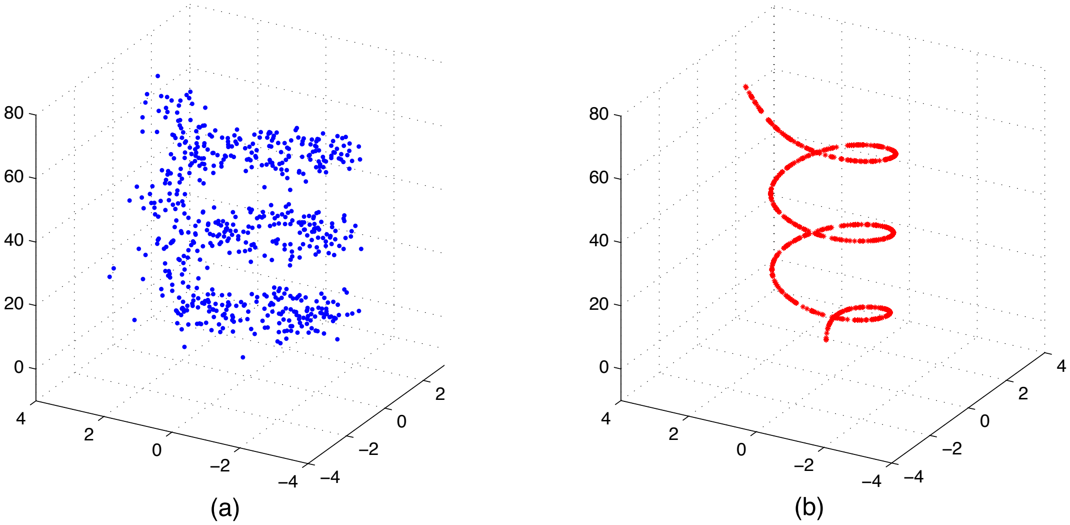

Figure 3.1: (a) $n = 600$ data points were generated by adding 3-dimensional standard Gaussian noise samples to 600 points uniformly sampled on a spiral in $\mathbb{R}^3$. (b) The output of the SCMS algorithm using $D = 3$, $d = 1$, the Gaussian kernel with bandwidth $h = 3$, and stopping threshold $\epsilon = 0.005$.

Chapter 4

Theoretical Results

4.1 Preliminary Results

Let $\{\boldsymbol{y}_j\}_{j=1,2,\dots}$ denote the mode estimate sequence generated by the MS algorithm. As mentioned before, it was proved in [18] that if the kernel K has a convex, differentiable, and strictly decreasing profile k [1], then $\{\hat{f}_{h,k}(\boldsymbol{y}_j)\}_{j=1,2,\dots}$ is a monotonically nondecreasing and convergent sequence. To prove the monotonicity of the sequence, the authors in [18] assumed that $\boldsymbol{y}_j = \mathbf{0}$ and based on this assumption, they showed $\hat{f}_{h,k}(\boldsymbol{y}_{j+1}) > \hat{f}_{h,k}(\boldsymbol{y}_j)$ for an arbitrary j. In the following theorem, we relax the assumption $\boldsymbol{y}_j = \mathbf{0}$ and prove the monotonicity and convergence of $\{\hat{f}_{h,k}(\boldsymbol{y}_j)\}$.

Theorem 4.1. *If the kernel K has a convex, differentiable, and strictly decreasing profile k, then the sequence $\{\hat{f}_{h,k}(\boldsymbol{y}_j)\}_{j=1,2,\dots}$ is monotonically increasing and convergent.*

[1] Recall that profile $k : [0, \infty) \to [0, \infty)$ is a non-negative, non-increasing, and piecewise continuous function that satisfies $\int_0^\infty k(x)dx < \infty$ and $K(\boldsymbol{x}) = c_k k(\|x\|^2)$, where c_k is a normalization factor that causes $K(\boldsymbol{x})$ to integrate to one.

Proof. The proof is a reproduction of the proof in [18], except $\boldsymbol{y}_j \neq \mathbf{0}$. Let $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ denote the data set. Let $\boldsymbol{y}_j \neq \boldsymbol{y}_{j+1}$, we show $\hat{f}_{h,k}(\boldsymbol{y}_{j+1}) > \hat{f}_{h,k}(\boldsymbol{y}_j)$. From (3.2), we have

$$\begin{aligned}\hat{f}_{h,k}(\boldsymbol{y}_{j+1}) - \hat{f}_{h,k}(\boldsymbol{y}_j) &= \frac{c_{k,D}}{nh^D}\Big[\sum_{i=1}^{n} k\big(\|\frac{\boldsymbol{y}_{j+1}-\boldsymbol{x}_i}{h}\|^2\big) - \sum_{i=1}^{n} k\big(\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\|^2\big)\Big] \\ &\geq \frac{c_{k,D}}{nh^D}\sum_{i=1}^{n} k'\Big(\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\|^2\Big)\Big(\|\frac{\boldsymbol{y}_{j+1}-\boldsymbol{x}_i}{h}\|^2 - \|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\|^2\Big),\end{aligned} \tag{4.1}$$

where the last inequality comes from the convexity of the profile function k, i.e., $k(x_2) - k(x_1) \geq k'(x_1)(x_2 - x_1)$. By expanding the terms in the right side of (4.1) and using (3.6), we have

$$\begin{aligned}\hat{f}_{h,k}(\boldsymbol{y}_{j+1}) - \hat{f}_{h,k}(\boldsymbol{y}_j) &\geq \frac{c_{k,D}}{nh^{D+2}}\sum_{i=1}^{n} k'\Big(\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\|^2\Big)\big(\|\boldsymbol{y}_{j+1}\|^2 - \|\boldsymbol{y}_j\|^2 - 2(\boldsymbol{y}_{j+1}-\boldsymbol{y}_j)^T\boldsymbol{x}_i\big), \\ &= \frac{c_{k,D}}{nh^{D+2}}\sum_{i=1}^{n} k'\Big(\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\|^2\Big)\big(\|\boldsymbol{y}_{j+1}\|^2 - \|\boldsymbol{y}_j\|^2 - 2(\boldsymbol{y}_{j+1}-\boldsymbol{y}_j)^T\boldsymbol{y}_{j+1}\big), \\ &= -\frac{c_{k,D}}{nh^{D+2}}\sum_{i=1}^{n} k'\Big(\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\|^2\Big)\big(\|\boldsymbol{y}_{j+1}-\boldsymbol{y}_j\|^2\big),\end{aligned} \tag{4.2}$$

where $\boldsymbol{y}^T$ denotes the transpose of $\boldsymbol{y}$. Since the profile function k is strictly decreasing, the right side of (4.2) is strictly greater than zero. Therefore, the sequence $\{\hat{f}(\boldsymbol{y}_j)\}_{j=1,2,\ldots}$ is strictly increasing and for an arbitrary j we have $\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) > 0$.

The mode estimate sequence $\{\boldsymbol{y}_j\}$ is bounded sequence. The boundedness and monotonicity of $\{\hat{f}(\boldsymbol{y}_j)\}$ implies the convergence. □

The authors of [18] claimed that the mode estimate sequence $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$ is a Cauchy sequence, which is not true in general. This error was also pointed out in [84]. From (4.2), we have

$$\hat{f}_{h,k}(\boldsymbol{y}_{j+1}) - \hat{f}_{h,k}(\boldsymbol{y}_j) \geq \frac{c_{k,D}}{nh^{D+2}}\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\|^2 \sum_{i=1}^{n} g(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2), \tag{4.3}$$

where $g(x) = -k'(x)$. If $k(x)$ is a convex and strictly decreasing function such that $0 < |k'(x)|$ for all $x \geq 0$, then $g(x)$ is positive and decreasing on $[0, \infty)$. Let $M(j) = \min\{g\Big(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2\Big), i = 1, \ldots, n\}$. Since $\boldsymbol{y}_j$ lies in the convex hull $\mathcal{C}$ of the data set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, we have $M(j) \geq g((\frac{a}{h})^2)$, where a is the diameter of $\mathcal{C}$. Let $\varphi = g((\frac{a}{h})^2)$. Hence, the above equality implies

$$\hat{f}_{h,k}(\boldsymbol{y}_{j+1}) - \hat{f}_{h,k}(\boldsymbol{y}_j) \geq \frac{c_{k,D}}{h^{D+2}}\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\|^2 \varphi. \tag{4.4}$$

Therefore, we have

$$\big(\hat{f}_{h,k}(\boldsymbol{y}_{j+1}) - \hat{f}_{h,k}(\boldsymbol{y}_j)\big)\frac{h^{D+2}}{\varphi c_{k,D}} \geq \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\|^2 \geq 0. \tag{4.5}$$

Since $\hat{f}_{h,k}(\boldsymbol{y}_{j+1})$ is a convergent sequence, the limit of the left side of the above inequality converges to zero. Therefore, the following limit relation holds

$$\lim_{j\to\infty} \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| = 0. \tag{4.6}$$

Combining the above result with (3.4) and (3.6) gives

$$\lim_{j\to\infty} \nabla \hat{f}_{h,k}(\boldsymbol{y}_j) = \boldsymbol{0}. \tag{4.7}$$

According to the definition of the mean shift vectors, the mode estimate sequence

$\{\boldsymbol{y}_j\}_{j=1,2,\dots}$ is always in the convex hull of the data set, i.e., $\boldsymbol{y}_j \in \mathcal{C}, j = 1, 2, \dots$. Therefore, $\{\boldsymbol{y}_j\}$ is a bounded sequence satisfying the above limit. Despite the claim in [18], the last two properties are not enough to prove the convergence of $\{\boldsymbol{y}_j\}_{j=1,2\dots,}$. For example, consider the sequence $\{\boldsymbol{z}_j\}_{j=1,2,\dots} \in \mathbb{R}^2$ defined as follows

$$\boldsymbol{z}_j = \Big(\sin(2\pi\sum_{k=1}^{j}\frac{1}{k}), \cos(2\pi\sum_{k=1}^{j}\frac{1}{k})\Big),\ j = 1, 2, \dots$$

The above sequence is bounded and satisfies the inequality

$$\|\boldsymbol{z}_j - \boldsymbol{z}_{j+1}\| \leq 2\pi\frac{1}{j+1}.$$

The left side is the length of the chord connecting two consecutive members of the sequence, and the right side is the geodesic distance along the unit circle between those two members. It can be observed that the right side of the above inequality goes to zero as $j \to \infty$, but $\{\boldsymbol{z}_j\}$ is not a convergent sequence.

4.2 The MS Algorithm with the Gaussian Kernel

In this section, we consider the MS algorithm with the Gaussian kernel. The Gaussian kernel has been widely used in various applications, and its properties have been extensively studied in the literature. We show that for the MS algorithm with the Gaussian kernel, all the stationary points of the estimated pdf are inside the convex hull of the data set. We also find a sufficient condition to have isolated stationary points. Later in this chapter we prove that if the stationary points of the estimated pdf are isolated then the mode estimate sequence generated by the MS algorithm converges.

4.2.1 Stationary points are inside the convex hull of the data set

Let $\boldsymbol{x}_i \in \mathbb{R}^D, i = 1, \ldots, n$ be the input data. From (3.2), the estimated pdf using the Gaussian kernel is given by $\hat{f}(\boldsymbol{x}) = c \sum_{i=1}^{n} k(\|(\boldsymbol{x} - \boldsymbol{x}_i)/h\|^2)$, where $k(x) = \exp(-x/2)$ and $c = (2\pi)^{-D/2}/(nh^D)$. Let $\mathcal{C}$ denote the convex hull of the data set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. The authors in [101] showed that all the stationary points of the estimated pdf using the Gaussian kernel are inside the convex hull of the data set. In the following lemma, we prove the same result for a wide class of kernels K with a strictly decreasing and differentiable profile k.

Lemma 4.1. *If a kernel function K has a strictly decreasing differentiable profile k, such that $|k'(x)| > 0$ for all $x > 0$, then the gradient of the estimated pdf using the kernel K and bandwidth h is nonzero outside the convex hull of the data set.*

Proof. Let $\boldsymbol{t} \notin \mathcal{C}$ be an arbitrary point outside the convex hull $\mathcal{C}$. Since the input data is a finite set, $\mathcal{C}$ is a bounded closed set. Therefore, there exists $\boldsymbol{x}_0 \in \mathcal{C}$ such that $\boldsymbol{x}_0$ has the smallest distance to $\boldsymbol{t}$

$$d(\boldsymbol{x}_0, \boldsymbol{t}) = \inf_{\boldsymbol{x} \in \mathcal{C}} d(\boldsymbol{x}, \boldsymbol{t}) > 0,$$

where $d(\boldsymbol{x}, \boldsymbol{t}) = \|\boldsymbol{x} - \boldsymbol{t}\|$. Since the profile function k is strictly decreasing and $|k'(x)| > 0$, we have $k'(x) < 0, x \in (0, \infty)$. The estimated pdf and the gradient of the estimated pdf at point $\boldsymbol{t} \notin \mathcal{C}$ are computed as follows

$$\begin{aligned} \hat{f}(\boldsymbol{t}) &= c \sum_{i=1}^{n} k(\|(\boldsymbol{t} - \boldsymbol{x}_i)/h\|^2) \\ \nabla \hat{f}(\boldsymbol{t}) &= \frac{c}{h^2} \sum_{i=1}^{n} 2(\boldsymbol{t} - \boldsymbol{x}_i) k'(\|(\boldsymbol{t} - \boldsymbol{x}_i)/h\|^2). \end{aligned} \tag{4.8}$$

The directional derivative $D_{\boldsymbol{u}}$ in the direction of the unit vector $\boldsymbol{u} = \frac{\boldsymbol{x}_0 - \boldsymbol{t}}{\|\boldsymbol{x}_0 - \boldsymbol{t}\|}$ at point $\boldsymbol{t}$ is given by

$$D_{\boldsymbol{u}}(\boldsymbol{t}) = \nabla \hat{f}(\boldsymbol{t}) \cdot \boldsymbol{u}, \tag{4.9}$$

where $\boldsymbol{x} \cdot \boldsymbol{y}$ denotes the inner product of $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^D$. We will show that $D_{\boldsymbol{u}}(\boldsymbol{t})$ is positive. Because the profile k is a strictly decreasing function, we have

$$k'(\|(\boldsymbol{t} - \boldsymbol{x}_i)/h\|^2) < 0.$$

It follows from (4.8) that it suffices to show that $(\boldsymbol{t} - \boldsymbol{x}_i) \cdot \boldsymbol{u} < 0, i = 1, \ldots, n$. According to the separating hyperplane theorem [94], there exists a hyperplane P with normal vector $\boldsymbol{u} = \frac{\boldsymbol{x}_0 - \boldsymbol{t}}{\|\boldsymbol{x}_0 - \boldsymbol{t}\|}$ that contains $\boldsymbol{x}_0$ and separates $\boldsymbol{t}$ and $\mathcal{C}$. The hyperplane P is defined by

$$\begin{aligned} P &= \{\boldsymbol{x} : (\boldsymbol{x} - \boldsymbol{x}_0) \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) = 0\} \\ &= \{\boldsymbol{x} : \boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) = c\}, \end{aligned}$$

where $c = \boldsymbol{x}_0 \cdot (\boldsymbol{x}_0 - \boldsymbol{t})$. Let P_- and P_+ be the half spaces separated by the hyperplane P such that $\mathcal{C} \subset P_+$ and $\boldsymbol{t} \in P_-$, i.e., $P_+ = \{\boldsymbol{x} : \boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) \geq c\}$ and $P_- = \{\boldsymbol{x} : \boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) \leq c\}$. Consider a new hyperplane $\hat{P}$ with the same normal vector $\boldsymbol{u}$ that contains $\boldsymbol{t}$. The new hyperplane $\hat{P}$ is parallel to P and is defined by

$$\begin{aligned} \hat{P} &= \{\boldsymbol{x} : (\boldsymbol{x} - \boldsymbol{t}) \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) = 0\} \\ &= \{\boldsymbol{x} : \boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) = \hat{c}\}, \end{aligned}$$

where $\hat{c} = \boldsymbol{t} \cdot (\boldsymbol{x}_0 - \boldsymbol{t})$. The half spaces $\hat{P}_-$ and $\hat{P}_+$ corresponding to $\hat{P}$ are $\hat{P}_+ = \{\boldsymbol{x} :$

$\boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) \geq \hat{c}\}$ and $\hat{P}_- = \{\boldsymbol{x} : \boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) \leq \hat{c}\}$. Since $\mathcal{C} \subset P_+$, we have $\boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) \geq c$ for $\boldsymbol{x} \in \mathcal{C}$. Since $\hat{c} + \|\boldsymbol{x}_0 - \boldsymbol{t}\|^2 = c$, we obtain $\boldsymbol{x} \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) > \hat{c}$ for all $\boldsymbol{x} \in \mathcal{C}$. The last inequality naturally holds for $\boldsymbol{x} = \boldsymbol{x}_i, i = 1, \ldots, n$, so that

$$\boldsymbol{x}_i \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) > c - (\boldsymbol{x}_0 - \boldsymbol{t}) \cdot (\boldsymbol{x}_0 - \boldsymbol{t}),$$

which is easily seen to be equivalent to

$$(\boldsymbol{x}_i - \boldsymbol{t}) \cdot (\boldsymbol{x}_0 - \boldsymbol{t}) > 0,\ i = 1, \ldots, n.$$

From the above inequality and equations (4.8) and (4.9), we conclude that $D_{\mathbf{u}}(\boldsymbol{t}) > 0$ for all $\boldsymbol{t} \notin \mathcal{C}$. Therefore, the gradient of the estimated pdf cannot be zero outside of the convex hull, so all stationary points of $\hat{f}(\boldsymbol{x})$ must lie in $\mathcal{C}$. □

Lemma 4.1 guarantees that for a certain class of kernel functions, e.g., Gaussian kernel, all the stationary points of the estimated pdf lie inside the convex hull $\mathcal{C}$.

4.2.2 Isolated stationary points using the Gaussian kernel

Now, we are in a position to introduce a sufficient condition for the stationary points of the estimated pdf using the Gaussian kernel to be isolated. The probability density estimate using the Gaussian kernel with the covariance matrix $\boldsymbol{\Sigma}$ is given by $\hat{f}(\boldsymbol{x}) = c_N \sum_{i=1}^{n} \exp(-\frac{(\boldsymbol{x}-\boldsymbol{x}_i)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{x}_i)}{2})$, where $c_N > 0$ is a normalization factor to ensure that

$\hat{f}(\boldsymbol{x})$ integrates to one. The gradient and Hessian matrix of the estimated pdf are given by

$$\begin{aligned}
\nabla \hat{f}(\boldsymbol{x}) &= c_N \sum_{i=1}^{n} \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_i - \boldsymbol{x}) \exp(-(\boldsymbol{x} - \boldsymbol{x}_i)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{x}_i)/2), \\
\boldsymbol{H}(\boldsymbol{x}) &= c_N \sum_{i=1}^{n} \boldsymbol{\Sigma}^{-1}(-\boldsymbol{I} + (\boldsymbol{x} - \boldsymbol{x}_i)(\boldsymbol{x} - \boldsymbol{x}_i)^T \boldsymbol{\Sigma}^{-1}) \exp(-(\boldsymbol{x} - \boldsymbol{x}_i)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{x}_i)/2).
\end{aligned}$$

Let

$$\begin{aligned}
C(\boldsymbol{x}) &= \sum_{i=1}^{n} \exp(-(\boldsymbol{x} - \boldsymbol{x}_i)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{x}_i)/2), \\
\boldsymbol{A}(\boldsymbol{x}) &= \sum_{i=1}^{n} (\boldsymbol{x} - \boldsymbol{x}_i)(\boldsymbol{x} - \boldsymbol{x}_i)^T \exp(-(\boldsymbol{x} - \boldsymbol{x}_i)^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{x}_i)/2).
\end{aligned}$$

Let S denote the set of stationary points of the estimated pdf, i.e., $S = \{\boldsymbol{x}^* : \nabla \hat{f}(\boldsymbol{x}^*) = \boldsymbol{0}\}$. Since $\hat{f}(\boldsymbol{x})$ has partial derivatives of arbitrarily high order, a well-known theorem of differential geometry states that if the Hessian matrix at the stationary points is of full rank, the stationary points are isolated [50][1]. We provide a sufficient condition for $\boldsymbol{\Sigma}$ such that the Hessian matrix at the stationary points has full rank. If the Hessian matrix $\boldsymbol{H}$ is not full rank, then there exists a vector $\boldsymbol{v} \neq \boldsymbol{0}$ such that $\boldsymbol{H}\boldsymbol{v} = \boldsymbol{0}$. This is equivalent to

[1] This result can also be deduced from the inverse function theorem. The inverse function theorem states that if $f : \mathbb{R}^n \to \mathbb{R}^n$ is a continuously differentiable function on some open set containing $\mathbf{a} \in \mathbb{R}$, such that $|Jf(\mathbf{a}) \neq \mathbf{0}|$, where J denotes the Jacobian of f, then there is some open set V containing $\mathbf{a}$ and an open W containing $f(\mathbf{a})$ such that $f : V \to W$ has a continuous inverse $f^{-1} : W \to V$ which is differentiable for all $\boldsymbol{y} \in W$. Therefore, if f denotes the pdf estimate, then the Hessian matrix is the Jacobian of the gradient of f. If the Hessian matrix is of full rank at some stationary point $\boldsymbol{x}^*$, then its determinant is nonzero and based on the inverse function theorem the stationary point $\boldsymbol{x}^*$ is isolated.

$\boldsymbol{A}(\boldsymbol{x})\boldsymbol{\Sigma}^{-1}\boldsymbol{v} = C(\boldsymbol{x})\boldsymbol{v}$. By expanding the last equality, we obtain

$$\Big(\boldsymbol{x}\boldsymbol{x}^T C(\boldsymbol{x}) - 2\boldsymbol{x}\sum_{i=1}^{n} \boldsymbol{x}_i^T \exp(-(\boldsymbol{x}-\boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{x}_i)/2) \\ + \sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^T \exp(-(\boldsymbol{x}-\boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{x}_i)/2)\Big)\boldsymbol{\Sigma}^{-1}\boldsymbol{v} = C(\boldsymbol{x})\boldsymbol{v}. \quad (4.10)$$

By definition, a stationary point $\boldsymbol{x}^*$, we have

$$\boldsymbol{x}^* = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i \exp(-\frac{(\boldsymbol{x}^*-\boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^*-\boldsymbol{x}_i)}{2})}{C(\boldsymbol{x}^*)}. \quad (4.11)$$

Then, equation (4.10) at a stationary point $\boldsymbol{x}^*$ can be simplified to

$$\overbrace{\Big(-\boldsymbol{x}^*\boldsymbol{x}^{*T}C(\boldsymbol{x}^*) + \sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^T \exp(-(\boldsymbol{x}^*-\boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^*-\boldsymbol{x}_i)/2)\Big)}^{\boldsymbol{B}(\boldsymbol{x}^*)}\boldsymbol{\Sigma}^{-1}\boldsymbol{v} = C(\boldsymbol{x}^*)\boldsymbol{v}. \quad (4.12)$$

The above equality implies that if the Hessian matrix is not of full rank at a stationary point $\boldsymbol{x}^*$, then $C(\boldsymbol{x}^*)$ is an eigenvalue of $\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}$.

Let $\boldsymbol{\Sigma}$ be a symmetric, positive definite matrix. We show that if $\boldsymbol{\Sigma}$ satisfies a certain condition, then $C(\boldsymbol{x}^*)$ can never be an eigenvalue of $\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}$. We need the following lemmas.

Lemma 4.2. *Let $\boldsymbol{\Sigma}$ be a nonsingular $D \times D$ matrix and $\boldsymbol{x} \in \mathbb{R}^D$. Then, for any $\boldsymbol{x} \in \mathbb{R}^D$, $\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}$ has rank one and its only nonzero eigenvalue $\hat{\lambda}$ is $\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}$.*

Lemma 4.3. *Let $\|.\|$ be any matrix norm on $\mathbb{C}^{D\times D}$. Let $\lambda_1, \lambda_2, \ldots, \lambda_D$ be the (real or* *complex) eigenvalues of $\boldsymbol{A} \in \mathbb{C}^{D\times D}$. Then, we have*

$$\rho(\boldsymbol{A}) \leq \|\boldsymbol{A}\|,$$

where $\rho(\boldsymbol{A})$ is the spectral radius of $\boldsymbol{A}$ and is defined as $\rho(\boldsymbol{A}) = \max_i |\lambda_i|$.

Lemma 4.4. *Let $\boldsymbol{A}$ be a $D \times D$ matrix. Let $\boldsymbol{A}^*$ denotes conjugate transpose of $\boldsymbol{A}$. Then $\boldsymbol{A}^*\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{A}^*$ have the same eigenvalues.*

Lemma 4.5. *Let $\boldsymbol{A}$ be a $D \times D$ Hermitian matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_D$. Then*

$$\max_{\boldsymbol{x}\neq\boldsymbol{0}} \frac{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}}{\|\boldsymbol{x}\|^2} = \lambda_1.$$

Lemma 4.6. *[113] Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be two $D \times D$ symmetric matrices. Then we have the following inequality*

$$\lambda_{max}(\boldsymbol{A}+\boldsymbol{B}) \leq \lambda_{max}(\boldsymbol{A}) + \lambda_{max}(\boldsymbol{B}),$$

where λ_{max} denotes the largest eigenvalue.

The spectral norm of a $D \times D$ matrix $\boldsymbol{A}$ induced by L_2 vector norm is given by [60]

$$\|\boldsymbol{A}\|_2 = \max_{\|\boldsymbol{x}\|=1} \|\boldsymbol{A}\boldsymbol{x}\| = \sqrt{\lambda_{max}(\boldsymbol{A}^*\boldsymbol{A})},$$

where λ_{max} denotes the largest eigenvalue of $\boldsymbol{A}^*\boldsymbol{A}$. Note that $\boldsymbol{A}^*\boldsymbol{A}$ is a positive semi-definite matrix, therefore $\lambda_{max} \geq 0$. Using the triangle inequality for norm of any two $D \times D$ matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, we have $\|\boldsymbol{A}+\boldsymbol{B}\| \leq \|\boldsymbol{A}\| + \|\boldsymbol{B}\|$ [60]. Using Lemma 4.3 and triangle inequality for spectral norm, we have

$$\begin{aligned}
\rho(\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}) &\leq \|\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}\|_2 \\
&= \| - \boldsymbol{x}^*\boldsymbol{x}^{*T}C(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1} + \sum_{i=1}^{n} \boldsymbol{x}_i\boldsymbol{x}_i^T \exp(-(\boldsymbol{x}^* - \boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^* - \boldsymbol{x}_i)/2)\boldsymbol{\Sigma}^{-1}\|_2 \\
&\leq \|\boldsymbol{x}^*\boldsymbol{x}^{*T}C(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}\|_2 + \sum_{i=1}^{n} \|\boldsymbol{x}_i\boldsymbol{x}_i^T \exp(-(\boldsymbol{x}^* - \boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^* - \boldsymbol{x}_i)/2)\boldsymbol{\Sigma}^{-1}\|_2 \\
&= C(\boldsymbol{x}^*)\|\boldsymbol{x}^*\boldsymbol{x}^{*T}\boldsymbol{\Sigma}^{-1}\|_2 + \sum_{i=1}^{n} \exp(-(\boldsymbol{x}^* - \boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^* - \boldsymbol{x}_i)/2)\|\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-1}\|_2.
\end{aligned} \tag{4.13}$$

Using Lemma 4.4 for $\|\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-1}\|_2, i = 1, 2, \ldots, n$, we have

$$\begin{aligned}
\|\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-1}\|_2 &= \sqrt{\lambda_{max}(\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-1})} \\
&= \sqrt{\lambda_{max}(\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}_i\boldsymbol{x}_i^T)} \\
&= \sqrt{\lambda_{max}(\boldsymbol{x}_i\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-2}\boldsymbol{x}_i\boldsymbol{x}_i^T)} \\
&= a_i\sqrt{\lambda_{max}(\boldsymbol{x}_i\boldsymbol{x}_i^T)} \\
&= a_i\sqrt{\|\boldsymbol{x}_i\|^2} \\
&= a_i\|\boldsymbol{x}_i\|,
\end{aligned} \tag{4.14}$$

where $a_i = \sqrt{\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-2}\boldsymbol{x}_i}$ and $\|\boldsymbol{x}_i\|^2$ is the largest eigenvalue of $\boldsymbol{x}_i\boldsymbol{x}_i^T$. Combining (4.13) and (4.14), we obtain

$$\begin{aligned}\rho(\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}) &\leq C(\boldsymbol{x}^*)a^*\|\boldsymbol{x}^*\| + \sum_{i=1}^{n}\exp(-(\boldsymbol{x}^*-\boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^*-\boldsymbol{x}_i)/2)a_i\|\boldsymbol{x}_i\| \\ &\leq C(\boldsymbol{x}^*)a^*\|\boldsymbol{x}^*\| + a_{max}\|\boldsymbol{x}_{max}\|\sum_{i=1}^{n}\exp(-(\boldsymbol{x}^*-\boldsymbol{x}_i)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}^*-\boldsymbol{x}_i)/2) \\ &= C(\boldsymbol{x}^*)a^*\|\boldsymbol{x}^*\| + a_{max}\|\boldsymbol{x}_{max}\|C(\boldsymbol{x}^*) \\ &\leq C(\boldsymbol{x}^*)a^*\|\boldsymbol{x}_{max}\| + a_{max}\|\boldsymbol{x}_{max}\|C(\boldsymbol{x}^*), \qquad (4.15)\end{aligned}$$

where $a^* = \sqrt{\boldsymbol{x}^{*T}\boldsymbol{\Sigma}^{-2}\boldsymbol{x}^*}$, $a_{max} = \max_i a_i$, and $\|\boldsymbol{x}_{max}\| = \max_i \|\boldsymbol{x}_i\|$. Let $\|\boldsymbol{x}^*\|^2 = b$ (b is unknown but less than $\|\boldsymbol{x}_{max}\|^2$), then from Lemma 4.5, $a^* \leq \sqrt{b\lambda_{max}(\boldsymbol{\Sigma}^{-2})} \leq \|\boldsymbol{x}_{max}\|\lambda_{max}(\boldsymbol{\Sigma}^{-1})$.

If $\|\boldsymbol{x}_{max}\|^2\lambda_{max}(\boldsymbol{\Sigma}^{-1}) + a_{max}\|\boldsymbol{x}_{max}\| < 1$, then we observe that $\rho(\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}) < C(\boldsymbol{x}^*)$. This means $C(\boldsymbol{x}^*)$ cannot be an eigenvalue of $\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}$, which contradicts (4.12). Therefore, we have the following result.

Lemma 4.7. *Let $\boldsymbol{x}_i \in \mathbb{R}^D, i = 1, \ldots, n$. Let $\|\boldsymbol{x}_{max}\|^2$ denote the largest norm among all $\boldsymbol{x}_i, i = 1, \ldots, n$. Let $a_{max} = \max_i \sqrt{\boldsymbol{x}_i^T\boldsymbol{\Sigma}^{-2}\boldsymbol{x}_i}$. Let $\hat{f}(\boldsymbol{x})$ denote the estimated pdf using the Gaussian kernel with the covariance matrix $\boldsymbol{\Sigma}$. If $\|\boldsymbol{x}_{max}\|^2\lambda_{max}(\boldsymbol{\Sigma}^{-1})+a_{max}\|\boldsymbol{x}_{max}\| < 1$, then the Hessian matrix of the estimated pdf at the stationary points is of full rank and the stationary points are isolated.*

Remark. Note that for the special case that $\boldsymbol{\Sigma} = h^2\boldsymbol{I}$, using Lemma 4.2 we know the only nonzero eigenvalue of $\boldsymbol{x}_i\boldsymbol{x}_i^T/h^2, i = 1, \ldots, n$ is equal to $\boldsymbol{x}_i^T\boldsymbol{x}_i/h^2$. Then using Lemma 4.6

we obtain

$$\begin{aligned}\lambda_1(\boldsymbol{B}(\boldsymbol{x}^*)/h^2) &\leq \lambda_1(-\boldsymbol{x}^*\boldsymbol{x}^{*T}C(\boldsymbol{x}^*)/h^2) + \sum_{i=1}^{n} \lambda_1(\boldsymbol{x}_i\boldsymbol{x}_i^T \exp(-(\boldsymbol{x}^* - \boldsymbol{x}_i)^T(\boldsymbol{x}^* - \boldsymbol{x}_i)/(2h^2))/h^2) \\ &\leq \sum_{i=1}^{n} \boldsymbol{x}_i^T\boldsymbol{x}_i/h^2 \exp(-(\boldsymbol{x}^* - \boldsymbol{x}_i)^T(\boldsymbol{x}^* - \boldsymbol{x}_i)/(2h^2)) \\ &\leq \|\boldsymbol{x}_{max}\|^2 C(\boldsymbol{x}^*)/h^2 \qquad (4.16)\end{aligned}$$

where $\lambda_1(\boldsymbol{A})$ denotes the largest eigenvalue of $\boldsymbol{A}$ and $\|\boldsymbol{x}_{max}\|^2 = \max_{i=1,\ldots,n} \|\boldsymbol{x}_i\|^2$.

If $\|\boldsymbol{x}_{max}\|^2/h^2 < 1$, then we observe from (4.16) that $\lambda_1(\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}) < C(\boldsymbol{x}^*)$. This means $C(\boldsymbol{x}^*)$ cannot be an eigenvalue of $\boldsymbol{B}(\boldsymbol{x}^*)\boldsymbol{\Sigma}^{-1}$, which contradicts equation (4.12). Therefore, we have the following result

Lemma 4.8. *Let $\boldsymbol{x}_i \in \mathbb{R}^D, i = 1, \ldots, n$. Let $\hat{f}(\boldsymbol{x})$ denote the estimated pdf using the Gaussian kernel with the covariance matrix $\boldsymbol{\Sigma} = h^2\boldsymbol{I}$. Let $\|\boldsymbol{x}_{max}\|^2 = \max_{i=1,\ldots,n} \|\boldsymbol{x}_i\|^2$. If $\|\boldsymbol{x}_{max}\|^2/h^2 < 1$, then the Hessian matrix of the estimated pdf at the stationary points is of full rank and the stationary points are isolated.*

Using a fully parameterized $\boldsymbol{\Sigma}$ increases the computational complexity of the Gaussian pdf estimate. Furthermore, finding a covariance matrix $\boldsymbol{\Sigma}$ that satisfies the sufficient condition in Lemma 4.7 is a challenging task, especially when the size of the input data set is large. Therefore, in practice in order to reduce the computational cost, the covariance matrix $\boldsymbol{\Sigma}$ is chosen either as a diagonal matrix $\boldsymbol{\Sigma} = diag(h_1^2, h_2^2, \ldots, h_D^2)$ or proportional to the identity matrix $\boldsymbol{\Sigma} = h\boldsymbol{I}$. The main advantage of the latter case it that only one parameter h, the bandwidth, needs to be set in advance. When the covariance matrix is chosen proportional to the identity matrix, Lemma 4.8 states that the modes of the Gaussian pdf estimate are isolated if $h^2 \geq \|\boldsymbol{x}_{max}\|^2$. Choosing a large value of the bandwidth h generates a smooth pdf estimate with low estimation variance, at the expense of introducing a large bias into the estimation [111]. The latter is not practically desirable, since a large bias will lead to a poor estimation of the pdf that results in an inaccurate mode estimate. Furthermore, it has been shown that conditions for the consistency[1] of such a Gaussian pdf estimate are $h_n \to 0$ and $nh_n \to \infty$, as $n \to \infty$ [111]. It is clear that the first consistency condition contradicts the sufficient condition given in Lemma 4.8.

Therefore, the theoretical conditions provided by Lemma 4.7 and Lemma 4.8 for a Gaussian pdf estimate to have isolated stationary points, are of limited use in practice.

Proof of Lemma 4.2. First, we show that $\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}$ has rank one. Since Rank$(\boldsymbol{\Sigma}) = D$ and Rank$(\boldsymbol{x}\boldsymbol{x}^T) = 1$, we have [89]

$$\text{Rank}(\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}) \leq \min\{\text{Rank}(\boldsymbol{x}\boldsymbol{x}^T), \text{Rank}(\boldsymbol{\Sigma}^{-1})\} = 1. \tag{4.17}$$

Also, according to the Sylvester's rank inequality [25], we have

$$\text{Rank}(\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}) \geq \text{Rank}(\boldsymbol{\Sigma}^{-1}) + \text{Rank}(\boldsymbol{x}\boldsymbol{x}^T) - D = 1. \tag{4.18}$$

Using (4.17) and (4.18), Rank$(\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}) = 1$.

Assume $\boldsymbol{y}$ is an eigenvector of $\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}$ so that $\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{y} = \lambda\boldsymbol{y}$. If $\lambda \neq 0$, then $\lambda\boldsymbol{y} = (\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{y})\boldsymbol{x}$, so $\boldsymbol{y}$ is a constant multiple of $\boldsymbol{x}$. Setting $\boldsymbol{y} = \boldsymbol{x}$, we obtain that $\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x}$ is the only nonzero eigenvalue of $\boldsymbol{x}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}$. □

Proof of Lemma 4.3. Let λ be an arbitrary eigenvalue of $\boldsymbol{A}$ with corresponding normalized

[1] A consistent pdf estimate $\hat{f}(\boldsymbol{x})$ is an estimator having the property that as the number of data points increases indefinitely, the resulting sequence of estimates converges in probability to $f(\boldsymbol{x})$.

eigenvector $\boldsymbol{v}$. i.e., $\boldsymbol{A}\boldsymbol{v} = \lambda\boldsymbol{v}$. Using definition of a matrix norm, we have

$$\|\boldsymbol{A}\| = \max_{\|\boldsymbol{x}\|=1} \|\boldsymbol{A}\boldsymbol{x}\| \geq \|\boldsymbol{A}\boldsymbol{v}\| = |\lambda|\|\boldsymbol{v}\| = |\lambda|.$$

Hence

$$\|\boldsymbol{A}\| \geq \rho(\boldsymbol{A}).$$

□

Proof of Lemma 4.4. The matrices $\boldsymbol{A}^*\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{A}^*$ are symmetric, therefore they have real eigenvalues. Let λ be an eigenvalue of $\boldsymbol{A}\boldsymbol{A}^*$ with corresponding eigenvector $\boldsymbol{v}$, i.e., $\boldsymbol{A}\boldsymbol{A}^*\boldsymbol{v} = \lambda\boldsymbol{v}$. We show λ is also an eigenvalue of $\boldsymbol{A}^*\boldsymbol{A}$. Let $\boldsymbol{u} = \boldsymbol{A}^*\boldsymbol{v}$, then we obtain

$$\boldsymbol{A}\boldsymbol{u} = \lambda\boldsymbol{v} \Rightarrow \boldsymbol{A}^*\boldsymbol{A}\boldsymbol{u} = \lambda\boldsymbol{A}^*\boldsymbol{v} = \lambda\boldsymbol{u}.$$

That means λ is also an eigenvalue of $\boldsymbol{A}^*\boldsymbol{A}$. Using the same argument we can show that if $\hat{\lambda}$ is an eigenvalue of $\boldsymbol{A}^*\boldsymbol{A}$, then it is also an eigenvalue of $\boldsymbol{A}\boldsymbol{A}^*$. Therefore $\boldsymbol{A}^*\boldsymbol{A}$ and $\boldsymbol{A}\boldsymbol{A}^*$ have the same eigenvalues. □

Proof of Lemma 4.5. From the spectral decomposition we obtain $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T$, where $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_D]$ and $\boldsymbol{\Lambda} = diag(\lambda_1, \ldots, \lambda_D)$. Let $\boldsymbol{u} = \boldsymbol{V}^T\boldsymbol{x}$. Then $\boldsymbol{x} \neq \boldsymbol{0}$ implies that $\boldsymbol{u} \neq \boldsymbol{0}$, and we obtain

$$\frac{\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x}}{\|\boldsymbol{x}\|^2} = \frac{\boldsymbol{u}^T\boldsymbol{\Lambda}\boldsymbol{u}}{\boldsymbol{x}^T\boldsymbol{x}} = \frac{\sum_{i=1}^{D}\lambda_i u_i^2}{\sum_{i=1}^{D} u_i^2} \leq \lambda_1$$

□

4.3 Convergence Proof when the Set of Stationary Points is Finite

Assuming that the stationary points are isolated, then the total number of stationary points of the estimated pdf inside the convex hull $\mathcal{C}$ cannot be infinite. Since the stationary points are inside the closed and bounded set $\mathcal{C}$, an infinite number of stationary points would have a convergent subsequence whose limit would not be isolated. By continuity, the limit point is also a stationary point and it is not isolated, which contradicts the fact that each stationary point is isolated. Hence, the number of stationary points is finite.

Next, we show that when the number of stationary points of the estimated pdf is finite, then the mode estimate sequence $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$ is a convergent sequence. We prove the following theorem

Theorem 4.2. *Let $\boldsymbol{x}_i \in \mathbb{R}^D, i = 1, \ldots, n$. Assume that the stationary points of the estimated pdf are isolated. Then the mode estimate sequence $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$ converges.*

Proof. Let $\mathcal{C}$ denote the convex hull of the data set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. Let S denote the set of stationary points of the estimated pdf $\hat{f}_{h,k}$, i.e., $S = \big\{\boldsymbol{x}_i^* : \ \|\nabla \hat{f}_{h,k}(\boldsymbol{x}_i^*)\| = 0\big\}$. Let ζ be the smallest distance between the points in S, i.e., $\zeta = \min\{\|\boldsymbol{x}_i^* - \boldsymbol{x}_j^*\| : \boldsymbol{x}_i^*, \boldsymbol{x}_j^* \in S, i \neq j\}$. Since S is finite we have $\zeta > 0$. Let $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$ be the mode estimate sequence generated by the MS algorithm. From the definition, it is clear that the mode estimate sequence $\{\boldsymbol{y}_j\}_{j=1,2,\ldots}$ is always inside the convex hull $\mathcal{C}$. From (4.7), we have

$$\lim_{j\to\infty} \nabla \hat{f}_{h,k}(\boldsymbol{y}_j) = \mathbf{0}. \tag{4.19}$$

This implies that the norm of the difference between two consecutive mode estimates converges to zero. Hence, there exists $N_1 > 0$ such that $\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| < \frac{\zeta}{3}$ for all $j \geq N_1$. Assume $S = \{\boldsymbol{x}_1^*, \ldots, \boldsymbol{x}_M^*\}$ and define $B(\boldsymbol{x}_i^*, \zeta/3)$ as the open ball of radius $\zeta/3$ centered at $\boldsymbol{x}_i^*$. Then the gradient of the estimated pdf outside of these balls is nonzero, i.e., $\nabla \hat{f}_{h,k}(\boldsymbol{y}_j) \neq 0,\ \boldsymbol{y}_j \notin B(\boldsymbol{x}_i^*, \zeta/3),\ i = 1, 2, \ldots, M$. If these open balls are removed from the convex hull of the data set, then the remaining set is compact. The norm of the gradient is a continuous function and attains its minimum value, say c, over this compact set. From (4.19), we can find N_2 such that $\|\nabla \hat{f}_{h,k}(\boldsymbol{y}_j)\| < c$ for all $j \geq N_2$. Thus for $j \geq N_2$, $\boldsymbol{y}_j$ cannot be outside $\bigcup_{i=1}^M B(\boldsymbol{x}_i^*, \zeta/3)$. Letting $N = \max\{N_1, N_2\}$, we will prove that for all $j > N$, if $\boldsymbol{y}_j \in B(\boldsymbol{x}_i, \zeta/3)$ then $\boldsymbol{y}_{j+1} \in B(\boldsymbol{x}_i, \zeta/3)$. We know that for $j \geq N$, $\boldsymbol{y}_{j+1} \in \bigcup_{i=1}^M B(\boldsymbol{x}_i^*, \zeta/3)$. Assume $\boldsymbol{y}_{j+1} \in B(\boldsymbol{x}_k^*, \zeta/3),\ \ k \neq i$. Then by the triangle inequality

$$\begin{aligned}\|\boldsymbol{x}_k^* - \boldsymbol{x}_i^*\| &= \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_{j+1} + \boldsymbol{x}_k^* - \boldsymbol{x}_i^* + \boldsymbol{y}_j - \boldsymbol{y}_j\| \\ &\leq \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| + \|\boldsymbol{y}_j - \boldsymbol{x}_i^*\| + \|\boldsymbol{x}_k^* - \boldsymbol{y}_{j+1}\|.\end{aligned}$$

Since by definition of ζ, $\|\boldsymbol{x}_k^* - \boldsymbol{x}_i^*\| \geq \zeta$, and by assumption $\|\boldsymbol{y}_j - \boldsymbol{x}_i^*\| \leq \zeta/3$, and $\|\boldsymbol{y}_{j+1} - \boldsymbol{x}_k^*\| \leq \zeta/3$, we have

$$\begin{aligned}\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| &\geq \|\boldsymbol{x}_k^* - \boldsymbol{x}_i^*\| - \|\boldsymbol{y}_j - \boldsymbol{x}_i^*\| - \|\boldsymbol{x}_k^* - \boldsymbol{y}_{j+1}\| \\ &\geq \zeta - \frac{\zeta}{3} - \frac{\zeta}{3} \\ &= \frac{\zeta}{3}.\end{aligned}$$

This contradicts that $\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| < \frac{\zeta}{3}$. Therefore if $\boldsymbol{y}_j \in B(\boldsymbol{x}_i^*, \zeta/3)$, then $\boldsymbol{y}_{j+1} \in B(\boldsymbol{x}_i^*, \zeta/3)$ for all $j \geq N$. Since $\boldsymbol{y}_j \in \bigcup_{i=1}^M B(\boldsymbol{x}_i^*, \zeta/3)$ for all $j \geq N$, we obtain that there is an index i such that $\boldsymbol{y}_j \in B(\boldsymbol{x}_i^*, \zeta/3)$ for all $j \geq N$. Since $\|\nabla \hat{f}_{h,K}(\boldsymbol{y}_j)\| \to 0$ and $\boldsymbol{x}_i^*$ is the unique zero of $\|\nabla \hat{f}_{h,K}\|$ in $B(\boldsymbol{x}_i^*, \zeta/3)$, by the continuity of $\|\nabla \hat{f}_{h,K}\|$ we have $\lim_{j\to\infty} \boldsymbol{y}_j = \boldsymbol{x}_i^*$. □

Theorem 4.2 guarantees the convergence of the mode estimate sequence when the number of modes of the estimated pdf is finite or, equivalently, when the stationary points of the estimated pdf are isolated. Lemma 4.8 also provides sufficient conditions to have isolated stationary points.

4.4 Convergence of the MS Algorithm in the One-Dimensional Case

The following result considers the special case $D = 1$, which may admittedly be of limited interest in applications.

Proposition 4.1. *For $D = 1$ the mode estimate sequence $\{y_j\}_{j=1,2,\ldots}$ generated by the MS algorithm using the profile $k(x) = e^{-x}$ associated with the Gaussian kernel converges to a stationary point of $\hat{f}(x)$.*

Proof. Since $K(x) = c\, e^{-x^2}$, the derivative of the kernel pdf estimate, $\hat{f}'(x)$, is proportional to

$$\sum_{i=1}^{n} (x_i - x) \exp\Big(-\frac{(x - x_i)^2}{h^2}\Big),$$

which is easily seen to be a real analytic function that is not constant on $\mathbb{R}$. Hence the set of stationary points $S = \{x \in \mathbb{R} : \hat{f}'(x) = 0\}$ has no limit points. However, S is a bounded set since one clearly has $S \subset [m, M]$, where $m = \min_{1 \le i \le n} x_i$ and $M = \max_{1 \le i \le n} x_i$, implying that S is finite. Thus $\{y_j\}$ converges to a point in S by Theorem 4.2. □

Remark. Note that the proof for Proposition 4.1 cannot be generalized for a high-dimensional case, since a real analytic non-constant function from $\mathbb{R}^D$ to $\mathbb{R}$ $(D > 1)$ can have infinity many stationary points inside a compact set.

Proposition 4.1 guarantees the convergence of the mode estimate sequence in the one-dimensional space when the MS algorithm uses the Gaussian kernel. When the kernel function is not Gaussian, the convergence result in the one-dimensional space still holds, but the convergence proof is longer than in the previous case. The following lemma shows the convergence of the MS algorithm in one dimension with a differentiable, convex, and strictly decreasing profile function.

Theorem 4.3. *Let $X = \{x_1, x_2, \ldots, x_n\}$ denote the input data. Let $\hat{f}_{h,k}(x)$ denote the estimated pdf using a kernel K with a differentiable, convex, and strictly decreasing profile k and a bandwidth h. Then the mode estimate sequence generated by the mean shift algorithm converges.*

Proof. From (4.6) and (4.7), we have

$$\lim_{j \to \infty} |y_{j+1} - y_j| = 0, \tag{4.20}$$

$$\lim_{j \to \infty} \hat{f}'_{h,k}(y_j) = 0. \tag{4.21}$$

Now we consider the case that $\hat{f}'_{h,k}(x_i) \neq 0, \forall x_i \in X$. Then there exists $\epsilon_i > 0$ such that $\hat{f}'_{h,k}(x)$ is nonzero in the closed interval centered at x_i with radius ϵ_i, denoted by

$I[x_i, \epsilon_i], i = 1, \ldots, n$. Let $\epsilon = \min\{\epsilon_i,\ i = 1, \ldots, n\}$. Since $\hat{f}'_{h,k}(x)$ is continuous, it achieves its minimum absolute value over the compact set $\bigcup_{i=1}^{n} I[x_i, \epsilon]$, so let $c = \min_{x \in \bigcup_{i=1}^{n} I[x_i,\epsilon]} |\hat{f}'_{h,k}(x)|$. By assumption, it is clear that $c > 0$. From (4.20) the sequence $\{|y_{j+1} - y_j|\}_{j=1,2,\ldots}$ converges to zero. Therefore, for every $\epsilon/2 > 0$, there exists a constant $N_1(\epsilon/2) > 0$ such that for all j greater than $N_1(\epsilon/2)$, the difference between two consecutive mode estimates becomes less than $\epsilon/2$, i.e., $|y_{j+1} - y_j| < \epsilon/2, \forall j > N_1(\epsilon/2)$. Furthermore, there exists N_2 such that for all j greater than N_2 the estimated derivative function along the mode estimates becomes less than c, i.e., $\hat{f}'_{h,k}(y_j) < c, \forall j > N_2$. Let $N = \max\{N_1(\epsilon/2), N_2\}$. Then, we have

$$\forall j > N : y_j \notin \bigcup_{i=1}^{n} I[x_i, \epsilon],\ y_j - \epsilon/2 < y_{j+1} < y_j + \epsilon/2. \tag{4.22}$$

Let $j > N$ and, without loss of generality, assume $y_{j+1} \geq y_j$. We show that $y_{j+2} \geq y_{j+1}$, and hence for $j > N$ the mode estimate sequence will be a non-decreasing sequence. We define sets D_1, D_2, and D_3 as follows

$$D_1 = \{x_i : y_j > x_i\},\ D_2 = \{x_i : y_{j+1} > x_i > y_j\},\ D_3 = \{x_i : x_i > y_{j+1}\}.$$

Let $g(x) = -k'(x)$. Since g is a decreasing function, then the following inequality holds

$$\sum_{x_i \in D_3} (x_i - y_{j+1}) g\big(|x_i - y_j|^2\big) \leq \sum_{x_i \in D_3} (x_i - y_{j+1}) g\big(|x_i - y_{j+1}|^2\big). \tag{4.23}$$

Using (3.6), we obtain (we may assume without loss of generality that $h = 1$)

$$\sum_{x_i \in D_3} (x_i - y_{j+1}) g\big(|x_i - y_j|^2\big) = \sum_{x_i \in D_1 \cup D_2} (y_{j+1} - x_i) g\big(|x_i - y_j|^2\big). \tag{4.24}$$

Replacing the left side of (4.23) with the right side of (4.24), we get

$$\sum_{x_i \in D_1 \cup D_2} (y_{j+1} - x_i) g\big(|x_i - y_j|^2\big) \leq \sum_{x_i \in D_3} (x_i - y_{j+1}) g\big(|x_i - y_{j+1}|^2\big). \tag{4.25}$$

Adding $\sum_{x_i \in D_1 \cup D_2} (x_i - y_{j+1}) g(|x_i - y_{j+1}|^2)$ to both sides of equation (4.25) gives

$$\begin{aligned} &\sum_{x_i \in D_1 \cup D_2} (y_{j+1} - x_i) g(|x_i - y_j|^2) + \sum_{x_i \in D_1 \cup D_2} (x_i - y_{j+1}) g(|x_i - y_{j+1}|^2) \\ &\leq \sum_{x_i \in D_3} (x_i - y_{j+1}) g(|x_i - y_{j+1}|^2) + \sum_{x \in D_1 \cup D_2} (x_i - y_{j+1}) g(|x_i - y_{j+1}|^2). \end{aligned} \tag{4.26}$$

From the properties given in (4.20) and (4.21), we observe that D_2 is an empty set. Therefore, the left side of the above inequality can be simplified to

$$\begin{aligned} &\sum_{x_i \in D_1 \cup D_2} (y_{j+1} - x_i) g(|x_i - y_j|^2) + \sum_{x_i \in D_1 \cup D_2} (x_i - y_{j+1}) (|x_i - y_{j+1}|^2) \\ &= \sum_{x_i \in D_1} (y_{j+1} - x_i) \Big(g(|x_i - y_j|^2) - g(|x_i - y_{j+1}|^2) \Big) \geq 0. \end{aligned}$$

Hence, the right side of (4.26) is non-negative and we have

$$0 \leq \sum_{x_i \in D_3 \cup D_2 \cup D_1} (x_i - y_{j+1}) g(|x_i - y_{j+1}|^2).$$

This is equivalent to $y_{j+2} \geq y_{j+1}$. Therefore, for all $j > N$ if $y_{j+1} \geq y_j$ then $y_{j+2} > y_{j+1}$. By induction, for all $j > N$ the sequence $\{y_j\}$ will be a monotonically increasing and hence convergent sequence.

For the case that $y_{j+1} \leq y_j$, we define sets D_1, D_2, and D_3 as follows

$$D_1 = \{x_i : x_i < y_{j+1}\},\ D_2 = \{x_i : y_{j+1} < x_i < y_j\},\ D_3 = \{x_i : x_i > y_j\}.$$

Then, similar to the previous case, it is straightforward to show that $y_{j+2} \leq y_{j+1}$. Therefore, the mode estimate sequence $\{y_j\}$ for all $j > N$ becomes a monotonically decreasing and convergent sequence.

It remains to prove the monotonicity of the mode estimate sequence for the case that for some $x_i \in X$, $\hat{f}^{'}_{h,k}(x_i) = 0$. Let $\hat{f}^{'}_{h,k}(x_i^*) = 0$ for some $x_i^* \in X$. If there exists N, such that for all $j > N$ there is not any x_i^* between y_j and y_{j+1}, then the previous results can be applied to show that the mode estimate sequence is a monotone sequence. Otherwise, we assume that such N does not exist. We need the following lemma.

Lemma 4.9. *Consider a fixed point iteration defined by $y_{j+1} = m(y_j)$, where m is a differentiable function. Let x^* denote a solution of the fixed point problem (if it exists), i.e., $x^* = m(x^*)$, and let e_j denote the distance between the fixed point x^* and y_j, i.e., $e_j = |x^* - y_j|$. Then there exists δ such that $e_{j+1} = e_j|m^{'}(\delta)|$ and $y_j < \delta < x^*$ if $y_j < x^*$ and $x^* < \delta < y_j$ if $x^* < y_j$.*

Proof. Using the mean-value theorem, there exists δ such that $y_j < \delta < x^*$ (without loss of generality, assume $y_j < x^*$) and $m(x^*) - m(y_j) = (x^* - y_j)m'(\delta)$. Then, we have

$$\begin{aligned} e_{j+1} = |x^* - y_{j+1}| &= |m(x^*) - m(y_j)| \\ &= |(x^* - y_j)m'(\delta)| \\ &= |(x^* - y_j)||m'(\delta)| \\ &= e_j|m'(\delta)|. \end{aligned}$$

This shows that $e_{j+1} = e_j|m'(\delta)|$. □

Using Lemma 4.9, there are three possibilities for $m'(x^*)$ that we check separately:

1. If $|m'(x^*)| < 1$, then there exists an interval $I = [x^* - \epsilon, x^* + \epsilon]$ such that for all $x \in I$, $|m'(x)| < 1$. Hence, if the sequence $\{y_j\}$ falls in I, then it converges to x^* using lemma 4.9 (since e_j becomes a decreasing sequence and finally converges to zero). If the sequence $\{y_j\}$ never falls in this interval, then there exists N large enough such that for all $j > N$, x^* is not between y_j and y_{j+1}, which contradicts the assumption we have made about the non-existence of such N.

2. If $|m'(x^*)| > 1$, then there is a closed interval $I = [x^* - \epsilon, x^* + \epsilon]$ such that for all $x \in I$, we have $|m'(x)| > 1$. For some j, let the sequence y_j fall in I. Otherwise, we can find large enough N such that for all $j > N$ there is no x_i^* between y_j and y_{j+1}, which contradicts our assumption about the non-existence of such N. We choose j large enough such that $|y_{j+1} - y_j| < \epsilon/2$. There are four possibilities as follows

   (a) $x^* - \epsilon \leq y_j < x^* - \frac{\epsilon}{2}$,

   (b) $x^* - \frac{\epsilon}{2} \leq y_j < x^*$,

   (c) $x^* \leq y_j < x^* + \frac{\epsilon}{2}$,

   (d) $x^* + \frac{\epsilon}{2} \leq y_j < x^* + \epsilon$.

   Let $x^* - \epsilon \leq y_j < x^* - \frac{\epsilon}{2}$. It is clear that in this case $e_{k+1} > e_k$, since for all $x \in I$, $m'(x) > 1$. This means that the Euclidean distance between y_{j+1} and x^* is greater than the Euclidean distance between y_j and x^* (y_{j+1} is also on the left side of the x^* because it is assumed that $|y_{j+1} - y_j| < \epsilon/2$). Therefore, in this case the sequence y_j can never fall in the interval $I' = [x^* - \frac{\epsilon}{2}, x^* + \frac{\epsilon}{2}]$. Hence, for all $j > N$, there is no

x_i^* between y_j and y_{j+1}, which contradicts our assumption about the non-existence of such N.(Case 4 can be treated in exactly the same way).

Let $x^* - \frac{\epsilon}{2} \leq y_j < x^*$. Also for all $x \in I$, $m'(x) > 1$. It is obvious that the Euclidean distance between y_{j+1} and x^* is greater than the Euclidean distance between y_j and x^* (y_{j+1} can be on the left or right side of the x^*). In this case, after some finite iterations (let us say M iterations), the cases 1 or 4 will happen and then it can be concluded for all $j > N + M$, the sequence $y_j \notin I' = [x^* - \frac{\epsilon}{2}, x^* + \frac{\epsilon}{2}]$, which contradicts our assumption about the non-existence of such N. The third case can be treated similar to the second case.

3. If $|m'(x^*)| = 1$, then there are three possibilities as follows:

   (a) $\exists I$ around x^* such that $\forall x \in I$, $m'(x) > 1$. This case was discussed before.

   (b) $\exists I$ around x^* such that $\forall x \in I$, $m'(x) < 1$. This case was discussed before.

   (c) $\exists I$ around x^* such that $\forall x \in I$ and $x < x^*$, $m'(x) < 1$. Also, $\forall x \in I$ and $x > x^*$, $m'(x) > 1$. In this case, the mode estimate sequence either converges to x^* or there is a closed interval I' around x^* such that y_j never falls in that interval. Convergence of the latter case is guaranteed according to the above discussion.

This completes the convergence proof of the sequence in one dimension.

□

4.5 Modified MS Algorithm

Lemma 4.8 states that the modes of the estimated pdf are isolated if $h^2 > \|\boldsymbol{x}_{max}\|^2$. Unfortunately, this condition is not practically useful. The bandwidth h, as a function of the sample size n, is chosen to satisfy $\lim_{n\to\infty} h(n) = 0$ to guarantee the asymptotic consistency of the pdf estimate [111]. Although choosing the bandwidth h based on the lower bound provided by Lemma 4.8 guarantees isolated stationary points, we get a poor estimation of the pdf that results in an inaccurate mode estimate. Unfortunately, a general and useful condition that leads to a set of isolated stationary points of the estimated pdf for commonly used kernels (such as the Gaussian kernel) still seems to be missing (although [14] makes the plausible claim, without proof, that the set of stationary points is always finite for the Gaussian kernel).

We slightly modify the MS algorithm to guarantee the convergence of the mode estimate sequence. The modified MS (MMS) algorithm is given as follows

(a) Initialize the mean shift vector to be one of the observed data.

(b) Compute the mean shift vector $\boldsymbol{m}(\boldsymbol{y}_j) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i g\left(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|\right)}{\sum_{i=1}^{n} g\left(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|\right)} - \boldsymbol{y}_j$.

(c) Update the mode estimate as $\hat{\boldsymbol{y}}_{j+1} = \boldsymbol{y}_j + \boldsymbol{m}(\boldsymbol{y}_j)$.

(d) Find the closest data point to the mode estimate $\boldsymbol{y}_{j+1} = \arg\min_{\boldsymbol{x}\in\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n\}} \|\hat{\boldsymbol{y}}_{j+1} - \boldsymbol{x}\|$.

(e) Iterate (b), (c), and (d) until the convergence occurs.

Similar to the MS algorithm, the sequence $\{\hat{f}(\boldsymbol{y}_j)\}_{j=1,2,\ldots}$ generated by the modified MS algorithm is an increasing and convergent sequence. In fact, we have

Lemma 4.10. *The density estimate values along the sequence of output values of the modified MS algorithm is a non-decreasing and convergent sequence.*

Proof. Let $\boldsymbol{y}_j \neq \boldsymbol{y}_{j+1}$. Then from (4.1) we have

$$\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) \geq \frac{c_{k,D}}{nh^{D+2}} \sum_{i=1}^{n} k'\Big(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2\Big)\Big(\|\boldsymbol{y}_{j+1} - \boldsymbol{x}_i\|^2 - \|\boldsymbol{y}_j - \boldsymbol{x}_i\|^2\Big). \quad (4.27)$$

By the triangle inequality, we have

$$\|\boldsymbol{y}_{j+1} - \hat{\boldsymbol{y}}_{j+1}\| \leq \|\hat{\boldsymbol{y}}_{j+1} - \boldsymbol{x}_i\| + \|\boldsymbol{y}_{j+1} - \boldsymbol{x}_i\|,\ i = 1, 2, \ldots, n. \quad (4.28)$$

Using (4.27) and (4.28), we obtain

$$\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) \geq \frac{c_{k,D}}{nh^{D+2}} \sum_{i=1}^{n} k'\Big(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2\Big) \times$$
$$\times \Big(\|\boldsymbol{y}_{j+1} - \hat{\boldsymbol{y}}_{j+1}\|^2 - \|\hat{\boldsymbol{y}}_{j+1} - \boldsymbol{x}_i\|^2 - 2\|\hat{\boldsymbol{y}}_{j+1} - \boldsymbol{x}_i\|\|\boldsymbol{y}_{j+1} - \boldsymbol{x}_i\| - \|\boldsymbol{y}_j - \boldsymbol{x}_i\|^2\Big). \quad (4.29)$$

From the definition of $\boldsymbol{y}_{j+1}$ we have $\|\boldsymbol{y}_{j+1} - \hat{\boldsymbol{y}}_{j+1}\|^2 - \|\hat{\boldsymbol{y}}_{j+1} - \boldsymbol{x}_i\|^2 < 0$, therefore the right side of (4.29) is always positive and we have $\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) > 0$. Since $\{\hat{f}(\boldsymbol{y}_j)\}$ is bounded, it is a convergent sequence. □

The modified MS algorithm starts from one of the observed data, and in each iteration the mode estimate is assigned to be one of the data points. The algorithm stops when two consecutive mode estimates become equal, i.e., $\boldsymbol{y}_{j+1} = \boldsymbol{y}_j$ for some $j \geq 1$. From Lemma 4.10, in each iteration each data point can be assigned to the mode estimate at most one time, otherwise $\hat{f}(\boldsymbol{y}_{j+k}) = \hat{f}(\boldsymbol{y}_j), k \geq 1$, which contradicts Lemma 4.10. Since the data set is finite, after a finite number of iterations the convergence occurs.

4.6 Theoretical Results for the SCMS Algorithm

In [96] extensive simulation results on artificial data demonstrated the ability of the algorithm to effectively approximate principal curves and surfaces. As well, promising applications of the SCMS algorithm to time-varying MIMO channel equalization and time series signal denoising were discussed. We note here that an algorithm for manifold denoising that is somewhat similar in spirit to SCMS but is based on the blurring version of the MS procedure was given by Wang and Carreira-Perpiñán [124]. On the theoretical side, [96] claimed that the SCMS algorithm will converge to a point on the principal surface with appropriate dimensionality. This claim was based on the assumption that the MS algorithm always converges, which as we discussed, has so far been unproven. In addition, it does not seem clear at all that the convergence of MS actually implies the convergence of SCMS, let alone its convergence to the principal surface. The next proposition states three convergence results relating to the density estimate values produced by the SCMS algorithm and the two stopping criteria presented earlier [49].

Proposition 4.2. *Assume the kernel pdf estimator $\hat{f}$ is defined as in* (3.2) *with a radially symmetric kernel K having profile k which is positive, non-increasing, and convex, such that the function $t \mapsto k(t^2)$ is twice continuously differentiable at all $t \in \mathbb{R}$. Let $\{\boldsymbol{y}_j\}$ denote the sequence of points generated by the SCMS algorithm with arbitrary initialization. Then the following hold:*

(i) *The sequence $\{\hat{f}(\boldsymbol{y}_j)\}$ is non-decreasing and convergent.*

(ii) $\lim\limits_{j\to\infty} \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| = 0.$

(iii) $\lim\limits_{j\to\infty} \|\boldsymbol{V}_j^T \nabla \hat{f}(\boldsymbol{y}_j)\| = 0.$

Proof. The subspace constrained mean shift sequence $\{\boldsymbol{y}_j\}$ is defined recursively by

$$\boldsymbol{y}_{j+1} = \boldsymbol{V}_j \boldsymbol{V}_j^T \boldsymbol{m}(\boldsymbol{y}_j) + \boldsymbol{y}_j, \tag{4.30}$$

where

$$\boldsymbol{m}(\boldsymbol{y}_j) = \frac{\sum_{i=1}^n \boldsymbol{x}_i g(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2)}{\sum_{i=1}^n g(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2)} - \boldsymbol{y}_j, \tag{4.31}$$

with $\boldsymbol{y}_1$ being an arbitrary starting point. Here $g(x) = -k'(x)$, where k is the profile of kernel K and $\boldsymbol{V}_j$ is the $D \times (D-d)$ matrix having orthonormal columns that are eigenvectors corresponding to the largest eigenvalues of the local inverse covariance matrix $\hat{\boldsymbol{\Sigma}}^{-1}$ evaluated at $\boldsymbol{y}_j$.

Since the profile k is bounded, the sequence $\{\hat{f}(\boldsymbol{y}_j)\}$ is bounded, so it suffices to show that the sequence is non-decreasing to prove convergence. The convexity of k implies that $k(t_2) - k(t_1) \geq g(t_1)(t_1 - t_2)$ for all $t_1, t_2 \geq 0$, where $g = -k'$. This and the definition of $\hat{f}$ yield

$$\begin{aligned}
\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) &= \frac{c}{nh^D} \sum_{i=1}^n \left(k\Big(\Big\| \frac{\boldsymbol{y}_{j+1} - \boldsymbol{x}_i}{h} \Big\|^2 \Big) - k\Big(\Big\| \frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h} \Big\|^2 \Big) \right) \\
&\geq \frac{c}{nh^{D+2}} \sum_{i=1}^n g\Big(\Big\| \frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h} \Big\|^2 \Big) \big(\|\boldsymbol{y}_j - \boldsymbol{x}_i\|^2 - \|\boldsymbol{y}_{j+1} - \boldsymbol{x}_i\|^2 \big) \\
&= C_j \sum_{i=1}^n p_j(i) \big(\|\boldsymbol{y}_j - \boldsymbol{x}_i\|^2 - \|\boldsymbol{y}_{j+1} - \boldsymbol{x}_i\|^2 \big),
\end{aligned} \tag{4.32}$$

where

$$p_j(i) = \frac{g\big(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h}\|^2\big)}{\sum_{k=1}^n g\big(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_k}{h}\|^2\big)}, \quad i = 1, \ldots, n$$

and

$$C_j = \frac{c}{nh^{D+2}} \sum_{i=1}^n g\Big(\Big\| \frac{\boldsymbol{y}_j - \boldsymbol{x}_i}{h} \Big\|^2 \Big).$$

Since $g(t) > 0$ for all $t \geq 0$, $p_j(1), \ldots, p_j(n)$ are well defined, positive, and sum to 1. In fact, the term "mean shift" derives from the fact that the mean shift of $\boldsymbol{y}_j$, given in (3.5), can be written in terms of an expectation, namely

$$\boldsymbol{m}(\boldsymbol{y}_j) = \sum_{i=1}^{n} p_j(i)(\boldsymbol{x}_i - \boldsymbol{y}_j) = E[\boldsymbol{Z}_j],$$

where $\boldsymbol{Z}_j$ is an $\mathbb{R}^D$-valued random vector with discrete distribution given by $\Pr(\boldsymbol{Z}_j = \boldsymbol{x}_i - \boldsymbol{y}_j) = p_j(i),\ i = 1, \ldots, n$. Thus, letting $\boldsymbol{T}_j = \boldsymbol{V}_j\boldsymbol{V}_j^T$, the SCMS update step can be rewritten as

$$\boldsymbol{y}_{j+1} - \boldsymbol{y}_j = \boldsymbol{T}_j\boldsymbol{m}(\boldsymbol{y}_j) = \boldsymbol{T}_jE[\boldsymbol{Z}_j]. \tag{4.33}$$

Let $\boldsymbol{W}_j$ be a $D \times D$ matrix representing any orthogonal projection onto the null space of $\boldsymbol{T}_j$. Then $\boldsymbol{x} = \boldsymbol{T}_j\boldsymbol{x} + \boldsymbol{W}_j\boldsymbol{x}$ for all $\boldsymbol{x} \in \mathbb{R}^D$, and $\boldsymbol{T}_j\boldsymbol{x}$ and $\boldsymbol{W}_j\boldsymbol{y}$ are orthogonal for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^D$. We can rewrite the last sum in (4.32) as follows

$$\begin{aligned}
&\sum_{i=1}^{n} p_j(i)\big(\|\boldsymbol{x}_i - \boldsymbol{y}_j\|^2 - \|\boldsymbol{x}_i - \boldsymbol{y}_{j+1}\|^2\big) \\
&\quad = \ E\big[\|\boldsymbol{Z}_j\|^2\big] - E\big[\big\|\boldsymbol{Z}_j - \boldsymbol{T}_jE[\boldsymbol{Z}_j]\big\|^2\big] \\
&\quad = \ E\big[\|\boldsymbol{W}_j\boldsymbol{Z}_j\|^2 + \|\boldsymbol{T}_j\boldsymbol{Z}_j\|^2\big] - E\big[\|\boldsymbol{W}_j\boldsymbol{Z}_j\|^2 + \big\|\boldsymbol{T}_j\boldsymbol{Z}_j - \boldsymbol{T}_jE[\boldsymbol{Z}_j]\big\|^2\big] \\
&\quad = \ E\big[\|\boldsymbol{T}_j\boldsymbol{Z}_j\|^2\big] - E\big[\big\|\boldsymbol{T}_j\boldsymbol{Z}_j - E[\boldsymbol{T}_j\boldsymbol{Z}_j]\big\|^2\big] \\
&\quad = \ \big\|E[\boldsymbol{T}_j\boldsymbol{Z}_j]\big\|^2 = \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\|^2,
\end{aligned}$$

where in the penultimate equality we applied the identity $E[Z^2] = \mathrm{Var}[Z] + (E[Z])^2$, which is valid for real random variables with finite variance, to the components of $\boldsymbol{T}_j\boldsymbol{Z}_j$.

Combining this with (4.32), we obtain

$$\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j) \geq C_j \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\|^2, \tag{4.34}$$

where $C_j > 0$, which implies that $\{\hat{f}(\boldsymbol{y}_j)\}$ is non-decreasing and thus convergent, proving part (i) of the proposition.

To prove part (ii), we note that $k(x) > 0$ for all $x \geq 0$ implies that $\hat{f}(\boldsymbol{y}_1) > 0$, so part (i) yields $\min\{\hat{f}(\boldsymbol{y}_j) : j \geq 1\} = \hat{f}(\boldsymbol{y}_1) > 0$. But this in turn implies that $\{\boldsymbol{y}_j\}$ is a bounded sequence, since otherwise it would have a subsequence $\{\boldsymbol{y}_{j_k}\}$ such that $\lim_{k\to\infty} \|\boldsymbol{y}_{j_k}\| = \infty$ which, in view of $\lim_{x\to\infty} k(x) = 0$, would give $\lim_{k\to\infty} \hat{f}(\boldsymbol{y}_{j_k}) = 0$, contradicting our uniform positive lower bound on the $\hat{f}(\boldsymbol{y}_j)$.

In view of the above, there exists $R > 0$ such that $\|\boldsymbol{y}_j - \boldsymbol{x}_i\| \leq R$ for all $j \geq 1$ and $i = 1, \ldots, n$. Since $g = -k'$ is non-increasing on $[0, \infty)$, we obtain

$$C_j = \frac{c}{nh^{D+2}} \sum_{k=1}^{n} g\Big(\Big\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_k}{h}\Big\|^2\Big) \geq \frac{c}{h^{D+2}} g\Big(\frac{R^2}{h^2}\Big) = C,$$

where $C > 0$ since $g(x) > 0$ for all $x \geq 0$. Thus (4.34) implies

$$\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\|^2 \leq C^{-1}\big(\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_j)\big),$$

and since $\lim\limits_{j\to\infty}\big(\hat{f}(\boldsymbol{y}_{j+1}) - \hat{f}(\boldsymbol{y}_{j+1})\big) = 0$ by part (i), we obtain $\lim\limits_{j\to\infty} \|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| = 0.$

Finally, to show (iii) we note that by definition (2.1) of $\hat{f}$,

$$\begin{aligned}\nabla\hat{f}(\boldsymbol{y}_j) &= \frac{2c}{nh^{D+2}}\sum_{i=1}^{n}(\boldsymbol{x}_i-\boldsymbol{y}_j)g\Big(\Big\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\Big\|^2\Big)\\ &= \frac{2c}{nh^{D+2}}\left[\sum_{i=1}^{n}g\Big(\Big\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\Big\|^2\Big)\right]\left[\frac{\sum_{i=1}^{n}\boldsymbol{x}_i g(\|\frac{\boldsymbol{x}_i-\boldsymbol{y}_j}{h}\|^2)}{\sum_{i=1}^{n}g(\|\frac{\boldsymbol{x}_i-\boldsymbol{y}_j}{h}\|^2)}-\boldsymbol{y}_j\right]\\ &= \frac{2c}{nh^{D+2}}\left[\sum_{i=1}^{n}g\Big(\Big\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\Big\|^2\Big)\right]\boldsymbol{m}(\boldsymbol{y}_j).\end{aligned}$$

Therefore,

$$\|\boldsymbol{V}_j^T\nabla\hat{f}(\boldsymbol{y}_j)\| = \frac{2c}{nh^{D+2}}\left[\sum_{i=1}^{n}g\Big(\Big\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\Big\|^2\Big)\right]\|\boldsymbol{V}_j^T\boldsymbol{m}(\boldsymbol{y}_j)\|.$$

Since $\boldsymbol{V}_j$ has orthonormal columns and $\boldsymbol{T}_j=\boldsymbol{V}_j\boldsymbol{V}_j^T$, we have $\|\boldsymbol{T}_j\boldsymbol{m}(\boldsymbol{y}_j)\|=\|\boldsymbol{V}_j^T\boldsymbol{m}(\boldsymbol{y}_j)\|$. This and (4.33) yield

$$\|\boldsymbol{V}_j^T\nabla\hat{f}(\boldsymbol{y}_j)\| = \frac{2c}{nh^{D+2}}\Big[\sum_{i=1}^{n}g\Big(\Big\|\frac{\boldsymbol{y}_j-\boldsymbol{x}_i}{h}\Big\|^2\Big)\Big]\|\boldsymbol{y}_{j+1}-\boldsymbol{y}_j\|$$

so part (iii) follows from part (ii) and the fact that the conditions on k ensure that $g=-k'$ is bounded.

□

Remarks

(a) Parts (i) and (ii) of the proposition are analogous to what is proved in Theorem 1 of [18] for the MS algorithm, with some proof ideas being similar. All three statements indicate (but by no means prove) the ability of the SCMS algorithm to converge to the principal surface of dimension d. In particular, (i) is related to the “ridge” property of locally defined principal curves and surfaces, (ii) and (iii) provide useful stopping criteria, while (iii) is related to the fact that at any point $\boldsymbol{y}$ of $\mathcal{P}^d$ one must have $\boldsymbol{V}(\boldsymbol{y})^T \nabla \hat{f}(\boldsymbol{y}_j) = \boldsymbol{0}$, where $\boldsymbol{V}(\boldsymbol{y})$ is the $D \times (D-d)$ matrix whose columns are the $D-d$ orthonormal eigenvectors corresponding to the $D-d$ largest eigenvalues of $\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{y})$.

(b) The differentiability condition on the profile k ensures that $\hat{f}$ is twice continuously differentiable so that all quantities used in the SCMS updates are well defined no matter how the algorithm is initialized. The condition that the kernel K is integrable and the conditions on k imposed in the proposition imply that k is bounded, its derivative k' is nondecreasing and negative on $[0, \infty)$, and both $k(x)$ and $k'(x)$ converge to zero as $x \to \infty$. The profile $k(x) = e^{-x}$ of the widely used Gaussian kernel satisfies these conditions.

(c) At the price of complicating the notation, the proof can straightforwardly be extended to more general kernel density estimates of the form

$$\hat{f}(\boldsymbol{x}) = \frac{c}{nh^D} \sum_{i=1}^{n} k\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|_{\boldsymbol{K}_i}^2 \right),$$

where $\|\boldsymbol{y}\|_{\boldsymbol{K}_i}^2 = \boldsymbol{y}^T \boldsymbol{K}_i \boldsymbol{y}$, with $\boldsymbol{K}_i$, $i = 1, \ldots, n$ being symmetric and positive definite $D \times D$ matrices. The potential usefulness of considering such more general estimates, which may account better for anisotropy and local scale information in the data sample, has been argued in [84].

4.7 Modified SCMS Algorithm

An inspection of the proof of Proposition 4.2 shows that all three statements remain valid if $\boldsymbol{V}_j$, $j = 1, 2, \ldots$, is an arbitrary sequence of $D \times (D - d)$ matrices having orthonormal columns. Thus for the convergence results to hold, $\boldsymbol{V}_j$ does not have to be the matrix whose columns are the $D - d$ orthonormal eigenvectors corresponding to the largest eigenvalues of $\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{y}_j)$. Of course, for the outputs of the algorithm to be meaningful the columns of $\boldsymbol{V}_j$ should be (nearly) orthogonal to the gradient of $\hat{f}$ at points on the d-dimensional principal surface of $\hat{f}$. The choice of $\hat{\boldsymbol{\Sigma}}^{-1}$ was motivated in [96] by Definition 3.1 and the connection to principal components when the underlying pdf is Gaussian. In this case the local inverse covariance matrix (of the Gaussian pdf, not estimated from data) is just the inverse covariance matrix of the Gaussian pdf up to a constant at any point with eigendirections the principal component directions. In practice, the density estimate $\hat{f}$ is never Gaussian so the use of $\hat{\boldsymbol{\Sigma}}^{-1}$ seems less well motivated for the SCMS algorithm than simply using the estimated Hessian $\hat{\boldsymbol{H}}$, which is a more natural choice in the context of Definition 3.1, as well as requiring slightly fewer operations to compute. At points $\boldsymbol{x}$ on the d-dimensional principal surface of $\hat{f}$, the gradient $\nabla \hat{f}(\boldsymbol{x})$ is orthogonal to exactly $D - d$ eigenvectors of $\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x})$ and to exactly $D - d$ eigenvectors of $\hat{\boldsymbol{H}}(\boldsymbol{x})$, and these two sets of eigenvectors are the same (see [96]). The eigenvalues of $\hat{\boldsymbol{H}}(\boldsymbol{x})$ associated with these eigenvectors are $-\hat{f}(\boldsymbol{x})$ times the corresponding eigenvalues of $\hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x})$ and so we form $\boldsymbol{V}_j$ from the $D - d$ eigenvectors of $\hat{\boldsymbol{H}}(\boldsymbol{y}_j)$ corresponding to the $D - d$ *smallest* eigenvalues of $\hat{\boldsymbol{H}}(\boldsymbol{y}_j)$.

In this section, we compare the use in the SCMS algorithm of $\hat{\boldsymbol{\Sigma}}^{-1}$, $\hat{\boldsymbol{H}}$, and two local estimates (local to $\boldsymbol{y}_j$) of the covariance matrix of $\hat{f}$ due to Wang and Carreira-Perpiñán [124]. In the resulting three variations of the original SCMS algorithm, the mean shift vectors and output updates are computed using (3.6) and Step 5 of the SCMS algorithm, respectively, but instead of the local inverse covariance matrix in (3.7), three different matrices are used. Let $\{\boldsymbol{y}_j^1, \ldots, \boldsymbol{y}_j^n\}$ denote the set of outputs after the jth iteration, where $\boldsymbol{y}_j^{(i)}$ is the output of the algorithm when it is initialized to the ith data point $\boldsymbol{x}_i$, $i = 1, \ldots, n$. In the jth iteration, the proposed matrices at a point $\boldsymbol{x}$ (set to one of the points $\boldsymbol{y}_j^{(i)}$) are [49]

(i) The Hessian of $\hat{f}$,

$$\hat{\boldsymbol{H}}(\boldsymbol{x}) = \frac{c}{nh^{2+D}} \sum_{i=1}^{n} \left(-\boldsymbol{I} + \frac{2(\boldsymbol{x} - \boldsymbol{x}_i)(\boldsymbol{x} - \boldsymbol{x}_i)^T}{h^2} \right) \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{2h^2} \right),$$

where c is the kernel profile normalization factor and $\boldsymbol{I}$ is the $D \times D$ identity matrix;

(ii) The estimated local covariance matrix using the κ nearest *data points*,

$$\hat{\boldsymbol{\Sigma}}_{\kappa}(\boldsymbol{x}) = \frac{1}{\kappa - 1} \sum_{\boldsymbol{x}_i \in N_{\kappa}(\boldsymbol{x})} (\boldsymbol{x}_i - \boldsymbol{m}_{\kappa}(\boldsymbol{x}))(\boldsymbol{x}_i - \boldsymbol{m}_{\kappa}(\boldsymbol{x}))^T,$$

where $N_{\kappa}(\boldsymbol{x})$ is the set of the κ nearest neighbors of $\boldsymbol{x}$ in the observed data set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, and $\boldsymbol{m}_{\kappa}(\boldsymbol{x})$ is the average over members of $N_{\kappa}(\boldsymbol{x})$;

(iii) The estimated local covariance matrix using the κ nearest *outputs*,

$$\hat{\boldsymbol{\Sigma}}_{\kappa,j}(\boldsymbol{x}) = \frac{1}{\kappa - 1} \sum_{\boldsymbol{y}_j^{(i)} \in N_{\kappa,j}(\boldsymbol{x})} (\boldsymbol{y}_j^{(i)} - \boldsymbol{m}_{\kappa,j}(\boldsymbol{x}))(\boldsymbol{y}_j^{(i)} - \boldsymbol{m}_{\kappa,j}(\boldsymbol{x}))^T,$$

where $N_{\kappa,j}(\boldsymbol{x})$ is the set of the κ nearest neighbors of $\boldsymbol{x}$ among the outputs $\{\boldsymbol{y}_j^1, \ldots, \boldsymbol{y}_j^n\}$ at the jth iteration and $\boldsymbol{m}_{\kappa,j}(\boldsymbol{x})$ is the average over members of $N_{\kappa,j}(\boldsymbol{x})$. In this case we update all the outputs in each iteration.

For each matrix above the matrix $\boldsymbol{V}_j$ at Step 4 of the SCMS algorithm is given by

$$\boldsymbol{V}_j = [\boldsymbol{v}_{d+1}, \dots, \boldsymbol{v}_D],$$

where $\boldsymbol{v}_i, i = d+1, \dots, D$ are the $D-d$ eigenvectors corresponding to the $D-d$ *smallest* eigenvalues. The projection step and termination criterion are the same as in Steps 5 and 6, respectively, in the SCMS algorithm. Proposition 4.2 guarantees that each of the resulting three SCMS algorithm variations stops after a finite number of iterations.

The projection of the MS vectors onto the subspace spanned by the eigenvectors of the Hessian matrix corresponding to the $D-d$ smallest eigenvalues complies with Definition 3.1, since a point $\boldsymbol{x}$ is located on the d-dimensional principal surface if the gradient at $\boldsymbol{x}$ is orthogonal to the $D-d$ smallest eigenvectors of the Hessian at $\boldsymbol{x}$ and the corresponding eigenvalues are negative [37]. The matrices in (ii) and (iii) follow Wang and Carreira-Perpiñán [124]. There the authors computed the blurred MS vectors using the blurring version of the MS algorithm [13] and then a corrector projective step is computed to constrain the motion to be orthogonal to the underlying manifold.

Although using only the κ nearest neighbors instead of the whole data set to estimate the projection matrix does not change the theoretical complexity in each iteration, in practice with a finite data set the running time significantly reduces. A good value of κ will in general depend on the structure of the underlying manifold. In our simulations we chose κ to be between 4 and 6 percent of the number of observations, but setting κ in general is beyond the scope of this paper. We note that the authors in [124] suggest that κ typically should grow sublinearly with the sample size n.

In the rest of this section, we present a simulation example using the original SCMS algorithm and our three variations on the two and three-dimensional spiral and two-dimensional circle. The input data are generated as

$$x_i = u_i + e_i, \quad i = 1, \ldots, n,$$

where the u_i's are independently and uniformly selected on the two or three-dimensional spiral or circle, called the generative curve, and the e_i's are independent, zero mean spherical Gaussian random vectors of appropriate dimension, independent of the u_i's and with component variance σ^2. We used $\epsilon = 0.01$ in the stopping criterion in Step 6 of the SCMS algorithm in all runs. For the two-dimensional spiral we used $n = 1000$ data samples, $\sigma^2 = 1$ for the noise variance, $h = 2$ for the bandwidth of the kernel density estimator, and $\kappa = 50$ nearest neighbors for computing the two variations of the local covariance matrix. For the three-dimensional spiral we used $n = 600$, $\sigma^2 = 0.6$, $h = 3$, and $\kappa = 40$. For the two-dimensional circle we chose $n = 500$, $\sigma^2 = 0.4$, $h = 0.35$, and $\kappa = 40$. For performance evaluation we computed the average squared Euclidean distance between the output points and the closest points on the generative curve, and the average running time, in seconds. All simulations were run using Matlab on a desktop computer with an Intel Core i7-870 processor.

Table 4.1 shows the results for the two and three-dimensional spirals and two-dimensional circle using the original SCMS algorithm and the three variations using the Hessian, the local covariance matrix using the original data points (Cov. 1), and the local covariance matrix using the output points in each iteration (Cov. 2) in place of the inverse covariance matrix. Performance in terms of closeness to the generative curve is similar for all 4 variations though, interestingly, use of the local covariance matrices gives no worse performance. In terms of runtime, the local covariance matrices perform significantly better, as expected. Adaptive optimization of the local neighborhood size κ should yield improved

Table 4.1: Performance results for the two, three-dimensional spirals, and circle.

2-d Spiral	SCMS	Hessian	Cov. 1	Cov. 2
Running time (sec.)	11.34	11.34	3.91	3.85
Av. Squared Euclidean Distance	0.074	0.075	0.077	0.077
3-d Spiral	SCMS	Hessian	Cov. 1	Cov. 2
Running time (sec.)	109.89	111.56	19.53	17.89
Av. Squared Euclidean Distance	0.273	0.299	0.152	0.152
2-d Circle	SCMS	Hessian	Cov. 1	Cov. 2
Running time (sec.)	22.900	23.011	7.490	6.950
Av. Squared Euclidean Distance	0.789	0.788	0.739	0.719

performance. Figures 4.1, 4.2, and 4.3 show the generative curve, the simulated data points, and the output points from the four versions of the algorithm, for the two-dimensional and the three-dimensional spirals and two-dimensional circle, respectively. All four versions of the algorithm show similar performance visually.

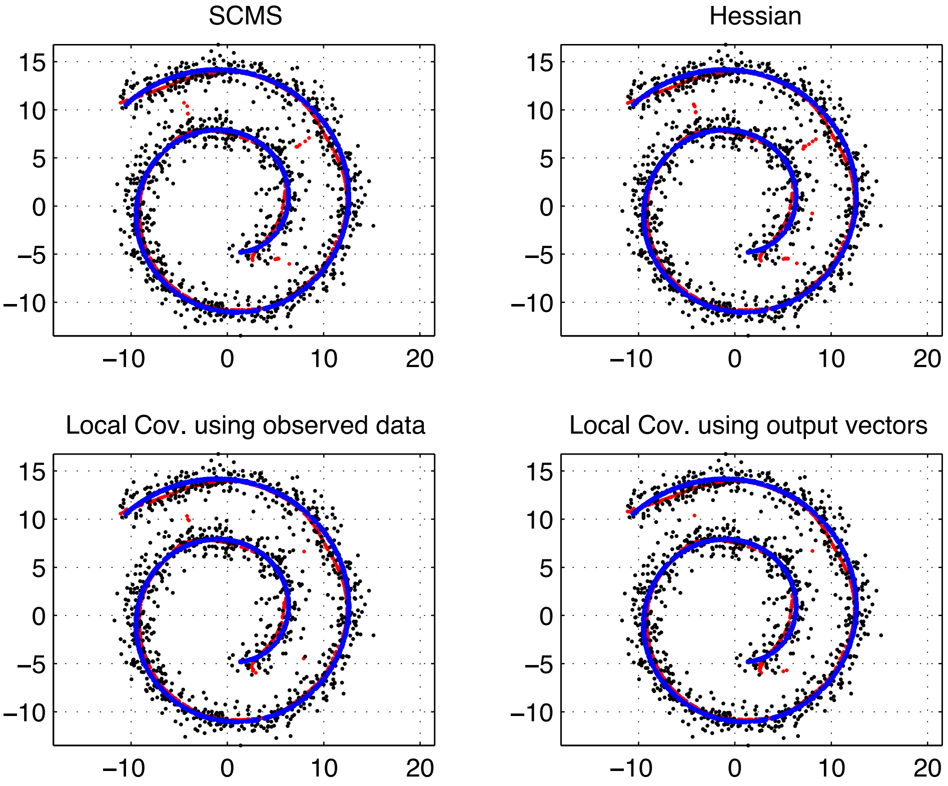

Figure 4.1: The blue points are $n = 1000$ samples uniformly selected on the two dimensional spiral generative curve, the red points are the outputs of each algorithm, and the black points are the observed data points generated by adding independent, zero mean Gaussian noise to the points on the generative curve.

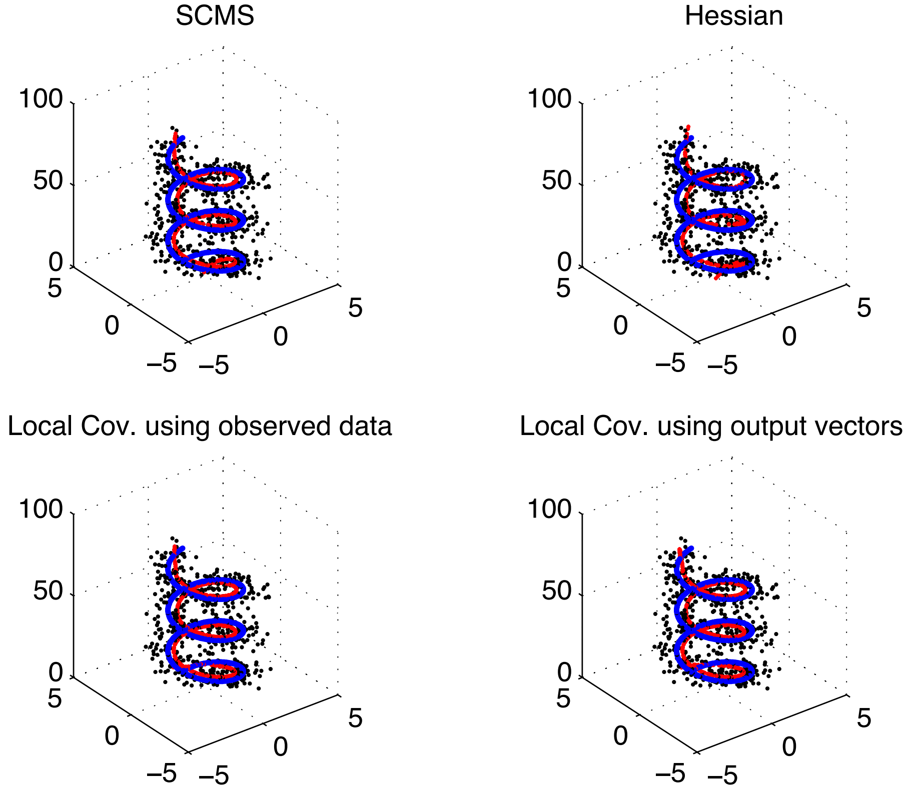

Figure 4.2: The blue points are $n = 600$ samples uniformly selected on the three dimensional spiral generative curve, the red points are the outputs of each algorithm, and the black points are the observed data points generated by adding independent, zero mean Gaussian noise to the points on the generative curve.

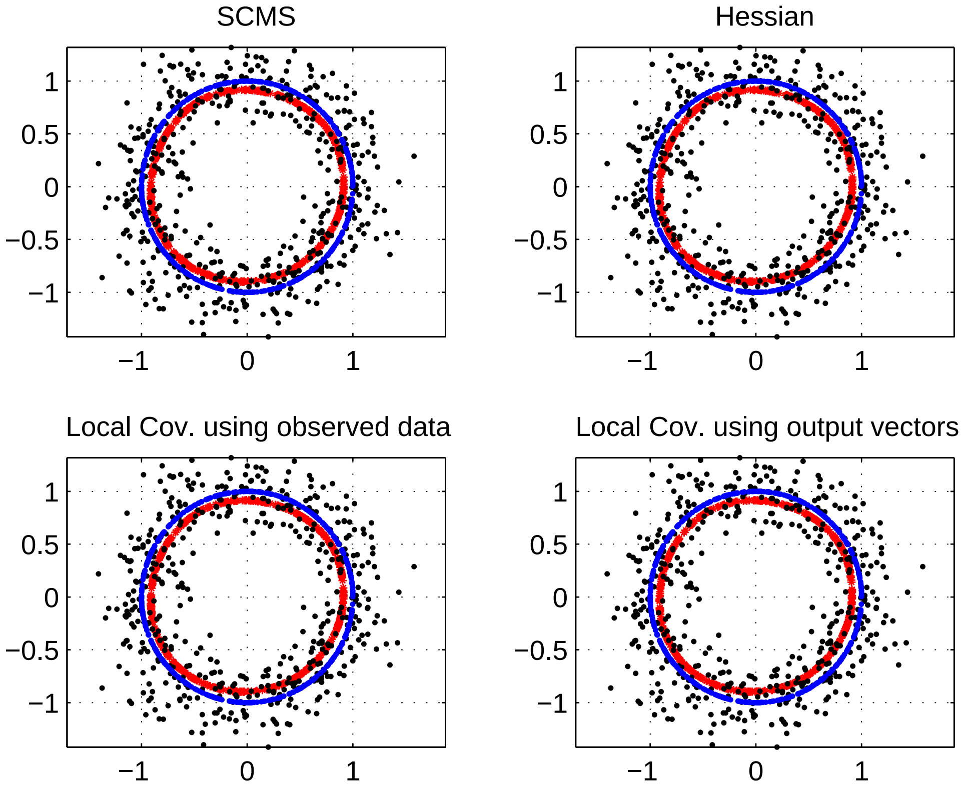

Figure 4.3: The blue points are $n = 500$ samples uniformly selected on the three dimensional spiral generative curve, the red points are the outputs of each algorithm, and the black points are the observed data points generated by adding independent, zero mean Gaussian noise to the points on the generative curve.

4.8 Sequential Data and Effect of the New Samples

In the standard SCMS algorithm, it is assumed that the entire data set is given in advance and new observations cannot be added to the data set during the process. However, when the SCMS algorithm is used over data sets in real world applications, we may confront difficult situations where a complete set of the observations is not available in advance. For example, in applications such mobile robotics, data are presented as a stream and all of the data are not available beforehand. Consider also a situation where the SCMS algorithm has converged, but new observations become available. Running the algorithm with augmented data set will change the location of the output points. Thus, the effect of the new observations on the output needs to be studied. When the new observations are added to the data set, if the SCMS algorithm is run on the updated data set there is no way to update the previous output points by just looking at the incoming observations. Running the SCMS algorithm on the entire data set is time consuming and increases the complexity, which prevents the algorithm from quickly responding to the new incoming data. In this section, we propose an adaptive version of the SCMS algorithm that can update the output points on the principal curve/surface by observing the new data sequentially.

4.8.1 Adaptive SCMS algorithm

When the new observations are available, it is clear that they have insignificant effect on the data points that are far from them. Therefore, we can consider the effect of the new samples just on their neighbors in a certain neighborhood and assume that the rest of the output points do not change. In other words, instead of running the SCMS algorithm on the entire data set, we just run the algorithm on the nearest neighbors. The output points associated with the nearest points will be modified, but the rest of the output points remain unchanged. The adaptive SCMS (ASCMS) algorithm is given as follows

(a) Let $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ denote the data set. Run the SCMS algorithm on $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ and save the outputs.

(b) Let $\boldsymbol{x}_{new}$ denote the new incoming data. Find the k nearest neighbors of $\boldsymbol{x}_{new}$ in $\mathcal{X}$. Let $\boldsymbol{x}_{n_1}, \ldots, \boldsymbol{x}_{n_k}$ denote the k nearest neighbors of $\boldsymbol{x}_{new}$.

(c) Run the SCMS algorithm on the new data set $\{\boldsymbol{x}_{new}, \boldsymbol{x}_{n_1}, \ldots, \boldsymbol{x}_{n_k}\}$.

(d) Repeat (b) and (c) as long as new observations are available.

We test the effectiveness of the ASCMS algorithm for adaptive estimation of a principal curve on a noisy circle and noisy straight line. For the circle, the size of the initial data set is five, and new observations are made from a noisy circle sequentially. The observations have an additive form $\boldsymbol{x} + \epsilon$, where $\boldsymbol{x}$ is uniformly selected point on a unit circle and ϵ is an additive Gaussian noise with independent components having zero mean and variance 0.1. The stopping threshold is set to 0.01. The outputs of the algorithm at certain times are shown in Fig. 4.4. The blue points are the current input data, the red points represent the outputs of the algorithm, and the green point is the new observed data. It can be observed that as the number of the observed data increases, the output points move to on or near the generative circle.

A similar experiment is repeated for a noisy straight line. Clean points on a straight line with length 10 are uniformly selected and corrupted by adding a Gaussian noise with variance $\sigma^2 = 1.2$. The samples are given to the adaptive SCMS algorithm one by one.

Fig. 4.5 shows the performance of the proposed adaptive SCMS algorithm at certain times. The current input data are shown by blue points, the red points are the outputs of proposed algorithm, and the new observed data is showed by a green point. It can be observed from Fig. 4.5 that the output points gradually converge to a straight line as the number of observed points increases.

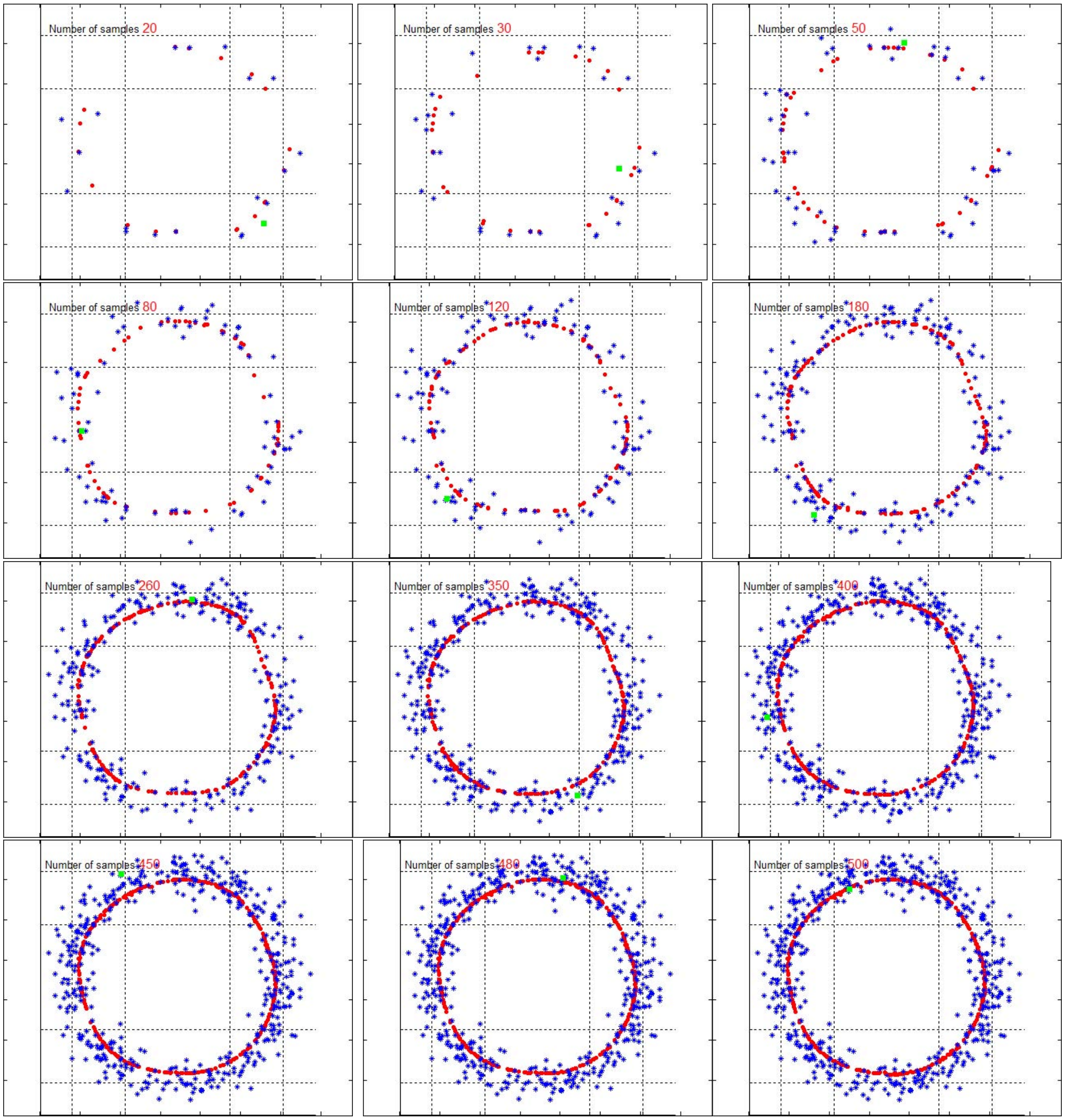

Figure 4.4: Output of the adaptive SCMS algorithm in certain times. The blue stars are the current input data, the red circles are output of the algorithm and the green square represents the new observed data.

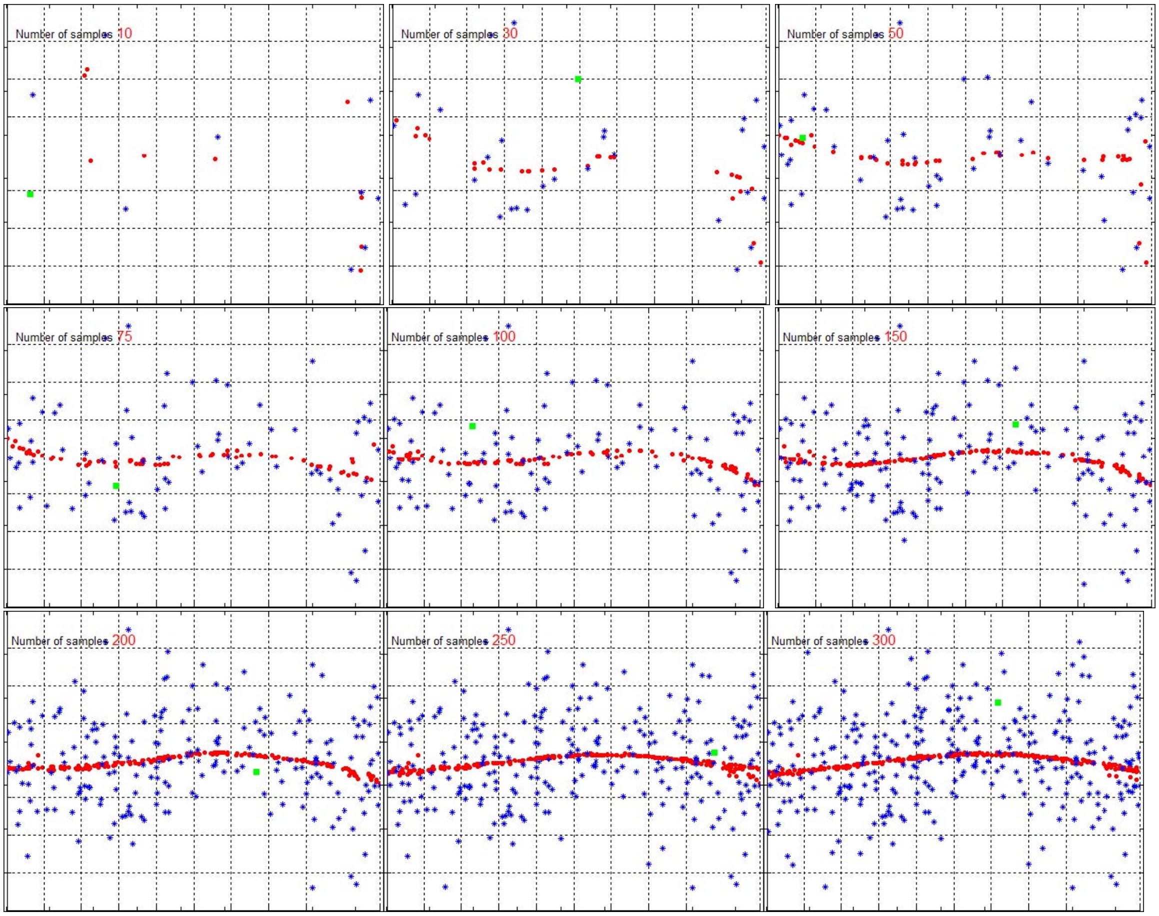

Figure 4.5: Output of the adaptive SCMS algorithm in certain times. The blue stars are the current input data, the red circles are output of the algorithm and the green square represents the new observed data.

Chapter 5

Nonlinear Dimensionality Reduction for Noisy Observations

5.1 Introduction

In certain situations, the observed high-dimensional data usually lie on or near a low-dimensional manifold, embedded in the high-dimensional space, as a result of which the observed data will have an intrinsically low-dimensional structure. For example, consider a system that records gray scale images of an individual under different poses and lighting conditions. Although the input dimensionality may be quite high, e.g., 4096 for 64 pixel by 64 pixel images, the structure of interest of these images lies on a three-dimensional manifold that can be parameterized by two pose variables and a lighting angle [116].

The goal of dimensionality reduction is to find a low-dimensional representation of high dimensional data while preserving the original information as much as possible. Many different algorithms have been introduced to accomplish this goal [66]. These can be classified as linear dimensionality reduction techniques and nonlinear dimensionality reduction techniques. The most popular technique for linear dimensionality reduction is principal component analysis (PCA) [69]. This technique assumes that the data can be well represented in a low-dimensional linear subspace of the high-dimensional space of the data. For nonlinear underlying manifolds, different techniques have been proposed, including locally linear embedding (LLE) [104], ISOMAP [116], kernel PCA [110], and maximum variance unfolding (MVU) [125], among others.

In this chapter, we first briefly review a selection of the more popular dimensionality reduction techniques. Then we show how the SMCS algorithm can be used to improve the performance of the nonlinear dimensionality reduction techniques in the presence of noise.

5.2 Dimensionality Reduction Techniques

Principal component analysis, also known as the Karhunen-Loeve transform [123], is a popular linear dimensionality reduction technique. PCA is a procedure that linearly transforms correlated variables into uncorrelated variables called principal components such that the first principal component has maximum variance, the second principal component has maximum variance under the constraint that it be uncorrelated with the first one, and so on. Assume that $\boldsymbol{X} \in \mathbb{R}^D$ is a vector consisting of D correlated zero mean random variables. PCA projects $\boldsymbol{X}$ to a d dimensional ($d \ll D$) linear subspace such that the projection captures most of the variability in $\boldsymbol{X}$. To find the first principal component, we wish to find a unit vector $\boldsymbol{b}_1 \in \mathbb{R}^D$ such that the variance of $\boldsymbol{b}_1^t\boldsymbol{X}$ is maximized. If $y = \boldsymbol{b}_1^t\boldsymbol{X}$, then the variance is given by $E(y^2) = \boldsymbol{b}_1^t \boldsymbol{\Sigma} \boldsymbol{b}_1$, where $\boldsymbol{\Sigma}$ is the covariance matrix of $\boldsymbol{X}$. The standard approach to find the unknown vector $\boldsymbol{b}_1$ is to use Lagrange multipliers [69]:

$$\arg\max_{\boldsymbol{b},\lambda}\{\boldsymbol{b}^t\boldsymbol{\Sigma}\boldsymbol{b} - \lambda(\boldsymbol{b}^t\boldsymbol{b} - 1)\},$$

where λ is the Lagrange multiplier. Taking the derivative of the above cost function with respect to $\boldsymbol{b}$ gives $\boldsymbol{\Sigma}\boldsymbol{b} = \lambda\boldsymbol{b}$. Thus λ is the eigenvalue and $\boldsymbol{b}_1$ is the eigenvector of the covariance matrix $\boldsymbol{\Sigma}$. It is straightforward to show that λ must be the largest eigenvalue of the covariance matrix and therefore $\boldsymbol{b}_1$ is the eigenvector corresponding to the largest eigenvalue. In general, for any linear transformation $\boldsymbol{y} = \boldsymbol{b}^t\boldsymbol{X}$, where $\boldsymbol{b}$ is a $D \times d$ $(1 \leq d \leq D)$ transformation matrix, the trace of the covariance matrix of $\boldsymbol{y}$, $tr(\boldsymbol{\Sigma}_{\boldsymbol{y}})$ will be maximized if $\boldsymbol{b}$ consists of the d eigenvectors of $\boldsymbol{\Sigma}$ corresponding to the d largest eigenvalues. In a similar way, it can be shown that $tr(\boldsymbol{\Sigma}_y)$ is minimized if the transformation matrix $\boldsymbol{b}$ consists of the d eigenvectors corresponding to the smallest eigenvalues of $\boldsymbol{\Sigma}$ [69]. To apply PCA to input data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, the covariance matrix $\boldsymbol{\Sigma}$ is replaced by the sample covariance matrix $\hat{\boldsymbol{\Sigma}}$.

The PCA algorithm can be summarized as follows

(a) The sample mean vector [1] $\hat{\boldsymbol{\mu}}$ and the sample covariance matrix [2] $\hat{\boldsymbol{\Sigma}}$ of the input data are computed and data are mean centered.

(b) The d eigenvectors corresponding to the d largest eigenvalues of $\hat{\boldsymbol{\Sigma}}$ are selected to construct the $D \times d$ transformation matrix.

(c) The data samples are mapped to the d dimensional space using the transformation matrix computed in (b).

[1] The sample mean $\hat{\boldsymbol{\mu}}$ is defined by $\hat{\boldsymbol{\mu}} = \sum_{i=1}^n \boldsymbol{x}_i/n$.

[2] The sample covariance matrix is defined by $\hat{\boldsymbol{\Sigma}} = \sum_{i=1}^n(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}})^T/(n-1)$

Another important property of PCA is that the projection onto the linear subspace minimizes the squared reconstruction error $\sum_{i=1}^{n} \|\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i\|^2$, where $\hat{\boldsymbol{x}}_i, i = 1, \ldots, n$ is an estimate of $\boldsymbol{x}_i$. In other words, the principal components of a set of data in $\mathbb{R}^D$ provide a sequence of best linear approximations to the data in d-dimensional subspace ($d < D$).

5.2.1 Locally linear embedding

Locally linear embedding (LLE) is a popular nonlinear dimensionality reduction technique that is used for mapping high-dimensional data to a low-dimensional space [128]. The LLE algorithm is based on a simple geometric intuition. The high-dimensional data, which is assumed to lie near a smooth nonlinear manifold, is mapped into a lower dimensional space such that the local structure in the data is preserved during the mapping [104]. In other words, nearby points in the high-dimensional space remain near each other in the low-dimensional space. The LLE algorithm first finds neighbors of each data point and computes coefficients that best reconstruct that data sample using its neighbors. It is assumed that each data point and its neighbors lie on or are close to a locally linear patch of the underlying manifold. There are two popular ways to find the neighbors of each data point, both of which are based on the Euclidean distance. The first is to select the k nearest samples for each data point $\boldsymbol{x}_i$, where k is chosen by the user. The second is to choose the neighbors of the data point $\boldsymbol{x}_i$ to be the points inside a ball with fixed radius centered at $\boldsymbol{x}_i$.

The reconstruction of the i-th data point $\boldsymbol{x}_i$ is computed as $\sum_{j \in N_i} W_{ij} \boldsymbol{x}_j$, where N_i is the set of indices of the neighbors of $\boldsymbol{x}_i$ and the coefficients W_{ij} satisfy $\sum_{j \in N_i} W_{ij} = 1$ for each $i = 1, \ldots, n$, where n is the number of input data points (note that $i \notin N_i$). The reconstruction error is defined as

$$e(\boldsymbol{W}) = \sum_{i=1}^{n} \left\| \boldsymbol{x}_i - \sum_{j \in N_i} W_{ij} \boldsymbol{x}_j \right\|^2,$$

where $\boldsymbol{W}$ is the matrix whose (i, j)-th component is W_{ij}. To find the optimal W_{ij}, the reconstruction error is minimized. The optimal coefficients W_{ij} can be found by solving a least squares problem [108].

In the final step, each high-dimensional $\boldsymbol{x}$ is mapped into a low-dimensional $\boldsymbol{y}$ such that the local geometry in the high-dimensional space is preserved in the low-dimensional space. This goal is achieved by minimizing the following cost function [104]

$$\Phi(\boldsymbol{Y}) = \sum_{i=1}^{n} \left|\left| \boldsymbol{y}_i - \sum_{j \in N_i} W_{ij} \boldsymbol{y}_j \right|\right|^2,$$

where $\boldsymbol{y}_i \in \mathbb{R}^d, i = 1, \ldots, n$ is the representation of $\boldsymbol{x}_i$ in the lower dimensional space and $\boldsymbol{Y}$ is the $d \times n$ matrix whose ith column is $\boldsymbol{y}_i$, $i = 1, \ldots, n$. In order to make the problem well posed, it is assumed that the low-dimensional coordinates $\boldsymbol{y}_i$ are centered around the origin $\sum_{i=1}^{n} \boldsymbol{y}_i = 0$. In addition, to avoid degenerate solutions, the embedding vectors are enforced to have unit covariance matrix $\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{y}_i \boldsymbol{y}_i^t = \boldsymbol{I}$ (where $\boldsymbol{I}$ denotes $d \times d$ identity matrix). The objective function can be reformulated as

$$\Phi(\boldsymbol{y}) = \sum_{i=1}^{n} \sum_{j=1}^{n} M_{ij} \boldsymbol{y}_i^t \boldsymbol{y}_j,$$

where $M_{ij} = \delta_{ij} - W_{ij} - W_{ji} + \sum_{k=1}^{n} W_{ik} W_{jk}$ and $W_{ij} = 0$ if $j \notin N_i$, $i, j = 1, \ldots, n$. Let $\boldsymbol{M}$ and $\boldsymbol{W}$ denote matrices whose (i, j)th elements are M_{ij} and W_{ij}, respectively. Then, it is not difficult to show that $\boldsymbol{M} = (\boldsymbol{I} - \boldsymbol{W})^t (\boldsymbol{I} - \boldsymbol{W})$. It is proved that the cost function is minimized under the given constraints if the columns of $\boldsymbol{Y}^T$ are the eigenvectors associated with the lowest eigenvalues of $\boldsymbol{M}$ [109]. The LLE algorithm is summarized in three steps as follows

(a) Compute the k nearest neighbors of each data point.

(b) Compute the weight matrix $\boldsymbol{W}$ that best reconstruct each data point using its k nearest neighbors.

(c) Find the $d+1$ bottom eigenvectors [1] of $(\boldsymbol{I}-\boldsymbol{W})^T(\boldsymbol{I}-\boldsymbol{W})$. Discard the eigenvector $[1, 1, 1, 1 \ldots]$ corresponding to the eigenvalue zero.

(d) Set the qth row of $\boldsymbol{Y}$ to be the q smallest eigenvector.

5.2.2 ISOMAP

ISOMAP is a nonlinear dimensionality reduction technique that tries to preserve the intrinsic geometry of the data. In other words, the ISOMAP technique is looking for a mapping from the high-dimensional observation space into a low-dimensional feature space that preserves the intrinsic metric structure of the observations as much as possible [115]. To achieve this goal, it tries to preserve the geodesic distances between data points. For neighboring points, the Euclidean distance provides a good approximation to the geodesic distance. For points far from one another, the geodesic distance can be approximated by adding up the lengths of the paths between neighboring points. The neighborhood relations are represented by a graph G such that data point i is connected to its neighbor j with an edge of weight $d_X(i,j)$, which is the Euclidean distance between i and j [116]. The

[1] The eigenvectors corresponding to the $d+1$ smallest eigenvalues

ISOMAP technique estimates the geodesic distance $d_M(i,j)$ between any point i and j by computing the shortest path using distances $d_G(i,j)$ in the graph G. Simple algorithms, such as Floyd-Warshall algorithm [41], can be employed to find the shortest path in the graph G. The Floyd-Warshall algorithm finds the shortest path in the graph between data points i and j, $i,j = 1,\ldots,n$, $i \neq j$ as an estimation of the geodesic distance. Let function $P(i,j,k)$ return the shortest possible path from i to j using vertices only from the set $\{1,2,\ldots,k\}$ as intermediate points along the way. Now, having $P(i,j,k)$, we can use a recursive algorithm to find the shortest path from each i to each j using only vertices 1 to $k+1$. The function P is initialized by $P(i,j,0) = d_X(i,j)$, where $d_X(i,j)$ is the weight of the edge between vertices i and j. We can find $P(i,j,k+1)$ using the following recursion [21]

$$P(i,j,k+1) = \min\{P(i,j,k), P(i,k+1,k) + P(k+1,j,k)\}. \tag{5.1}$$

The algorithm first computes $P(i,j,1)$ for all (i,j) pairs, then increases k by one, and using (5.1), finds $P(i,j,k+1)$ for all (i,j) pairs. This process continues until $k = n$ and we have found the shortest path for all (i,j) pairs using intermediate vertices. The shortest path can be considered as the estimated geodesic distance between all pairs in the graph. The estimated geodesic distances construct a matrix of graph distances $\boldsymbol{D}_G$ whose (i,j)th element is $d_G(i,j)$. The final step of the ISOMAP technique is applying the classical multidimensional scaling (MDS) [23] to the distance matrix $\boldsymbol{D}_G$ to generate an embedding of the data in a d ($d < D$) dimensional space. The cost function is defined by $\sum_{i=1}^{n}\sum_{j=1}^{n}(d_G(i,j) - d_Y(i,j))^2$, where $d_Y(i,j)$ denotes the Euclidean distance between the transformed points in the lower dimensional space. Using the MDS algorithm [23], we can find a representation of the data in the lower dimensional space such that the above cost function is minimized. The ISOMAP technique can be summarized as follows

(a) Construct the graph G by connecting points i and j if they are closer than some predefined ϵ or if i is one of the k nearest neighbors of j. Set the edge lengths equal to $d_X(i, j)$, which is the Euclidean distance between i and j.

(b) Initialize $d_G(i, j) = d_X(i, j)$ if i, j are connected by an edge, otherwise set $d_G(i, j) = \infty$.

(c) Use the Floyd-Warshall algorithm to find the shortest path distances between all pairs of points in G. The matrix of final values D_G will contain the estimated geodesic distances between all pairs of points in G.

(d) Apply the MDS technique to the distance matrix $\boldsymbol{D}_G$ to generate an embedding of the data in a low-dimensional space.

5.2.3 Kernel PCA

Kernel PCA (KPCA) is an extension of the standard PCA, which has been used for feature selection in a high-dimensional feature space. In KPCA, the principal components are computed in a higher dimensional feature space that is related to the input space through some nonlinear mapping. In other words, KPCA tries to find the low-dimensional latent structure of the input data by nonlinearly mapping it to a higher dimensional space and finding principal components in that space [103].

Let $\boldsymbol{x}_i, i = 1, \ldots, n$ be a set of mean centered observations in $\mathbb{R}^D$. Let $\Phi : \mathbb{R}^D \rightarrow \mathbb{R}^N$ be a mapping that transforms data samples into an N ($N > D$ and $N = \infty$ is not excluded) dimensional space, called feature space F. It is assumed that the transformed data in the feature space are mean centered $\sum_{i=1}^{n} \Phi(\boldsymbol{x}_i) = \boldsymbol{0}$. Let $\boldsymbol{v}$ and λ denote an arbitrary eigenvector and the corresponding eigenvalue of the covariance matrix in the feature space F, represented by $\mathbf{\Sigma}_F$. Then, we have [110]

$$\lambda \sum_{i=1}^{n} \alpha_i \Phi(\boldsymbol{x}_k)^t \Phi(\boldsymbol{x}_i) = \frac{1}{n} \sum_{i=1}^{n} \alpha_i \big(\Phi(\boldsymbol{x}_k)^t \sum_{j=1}^{n} \Phi(\boldsymbol{x}_j)\big) \big(\Phi(\boldsymbol{x}_j)^t \Phi(\boldsymbol{x}_i)\big)$$

$$k = 1, \dots, n,$$

where $\alpha_i, i = 1, \dots, n$ are computed from $\mathbf{v} = \sum_{i=1}^{n} \alpha_i \Phi(\boldsymbol{x}_i)$. Let $\mathbf{K}$ be the inner product matrix, also called the Gramian matrix, whose (i, j)th element is $\Phi(\boldsymbol{x}_i)^t \Phi(\boldsymbol{x}_j)$. Then, the last equality can be simplified as

$$n\lambda \boldsymbol{K} \alpha = \boldsymbol{K}^2 \alpha,$$

where $\alpha = [\alpha_1, \dots, \alpha_n]^t$. The above equality is equivalent to $\boldsymbol{K}\alpha = n\lambda\alpha$, which gives us α [110]. The principal components are found by the projection of the transformed data on the eigenvectors of the covariance matrix in the feature space. If $\boldsymbol{v}^k$ denotes the kth eigenvector of $\mathbf{\Sigma}_F$, then the nonlinear principal components in $\boldsymbol{v}^k$'s direction are computed by

$$\Phi(\boldsymbol{x})^t \boldsymbol{v}^k = \sum_{i=1}^{n} \alpha_i \Phi(\boldsymbol{x}_i)^t \Phi(\boldsymbol{x}).$$

The kernel trick makes it possible to compute the dot product in the feature space without actually mapping the data into the feature space F. Since the feature space F is nonlinearly related to the input space via Φ, the contour lines of the projection onto the principal eigenvectors in F become nonlinear in the input space.

The KPCA can be summarized as follows

(a) Compute the inner product matrix $\boldsymbol{K}$ (Gramian matrix).

(b) Find the nonzero eigenvalues and corresponding eigenvectors of $\boldsymbol{K}$ ($n\lambda\alpha = \boldsymbol{K}\alpha$).

(c) Find the nonlinear principal components by projecting the transformed data onto the eigenvectors of the covariance matrix in the feature space.

The kernel PCA does not inherit all the advantages of the original PCA. For example, the reconstruction of the data samples is not a trivial task in the KPCA. Data can be reconstructed in the feature space F. However, finding the corresponding $\boldsymbol{x}$ in the initial space is difficult and sometimes even impossible [66].

5.2.4 Maximum variance unfolding

Similar to the previous techniques, the maximum variance unfolding (MVU) algorithm is based on simple geometric intuition. This algorithm tries to preserve distances and angles between k nearest neighbors and pull apart the rest of the points by maximizing their total variance [125]. Let $\boldsymbol{x}_i$ and $\boldsymbol{y}_i, i = 1, \dots n$ denote the input and output of the algorithm, respectively. If $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are themselves neighbors or common neighbors of another point in the data set, then the local isometry is preserved if

$$\|\boldsymbol{x}_i - \boldsymbol{x}_j\| = \|\boldsymbol{y}_i - \boldsymbol{y}_j\|,$$

where $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ are corresponding embedded points in the lower dimensional space. The MVU algorithm constructs a graph with n nodes such that each node is connected to its k nearest neighbors (k is a parameter of the algorithm). The local geometry constraint tries to preserve both lengths and angles between connected edges to the same node. To achieve this goal, all neighbors of each node are connected and by preserving the distances along the edges in the new graph, both the angles and lengths in the original graph will be preserved. To remove a translation degree of freedom, the outputs $\boldsymbol{y}_i, i = 1, \ldots, n$ are forced to be centered on the origin, $\sum_{i=1}^{n} \boldsymbol{y}_i = \mathbf{0}$. The MVU algorithm tries to keep neighbors together and pull non-neighbor points as far apart as possible. Let D_{ij} denote the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Let $\boldsymbol{K}$ be the inner product matrix such that its (i, j)-th element is given by $K_{ij} = \boldsymbol{y}_i^t \boldsymbol{y}_j$ and $\eta_{ij} \in \{0, 1\}$ indicates whether there is a edge between nodes i and j.

The MVU technique yields to the following optimization problem [125]

$$\begin{aligned}
&\text{Maximize } tr(\boldsymbol{K}),\\
&\text{subject to}\\
&\sum_{i=1}^{n}\sum_{j=1}^{n} K_{ij} = 0,\\
&K_{ii} - 2K_{ij} + K_{jj} = D_{ij} \text{ for all } (i, j) \text{ with } \eta_{ij} = 1,\\
&\boldsymbol{K} \text{ is a symmetric, positive semi-definite matrix.}
\end{aligned}$$

The above optimization problem is an example of semi-definite programming. There is a large literature on efficiently solving these problems. As well, there are a number of tool boxes. Outputs $\boldsymbol{y}_i, i = 1, \ldots, n$ can be recovered from the inner product matrix. If $v_{\alpha,i}$ denotes the ith element of the eigenvector corresponding to the eigenvalue λ_α, then the (i, j)th element of the inner product matrix can be written as $K_{ij} = \sum_{\alpha=1}^{n} \lambda_\alpha v_{\alpha,i} v_{\alpha,j}$. In this case the ith element of $\boldsymbol{y}_\alpha$ corresponding to the $\boldsymbol{x}_\alpha$ is $\boldsymbol{y}_{\alpha,i} = \sqrt{\lambda_\alpha} v_{\alpha,i}$. The MVU algorithm can be summarized as follows

(a) Compute k nearest neighbors for each data point and connect each input data to its neighbors as well as each neighbor to other neighbors of the same input.

(b) Compute the inner product matrix that is centered on the origin and preserves the distances of all edges in the graph.

(c) Compute the low-dimensional embedding from the top eigenvectors of the inner product matrix learned by the semi-definite programming.

5.3 Nonlinear Dimensionality Reduction in the Presence of Noise

Often it is reasonable to assume that the observed data set has an intrinsically low-dimensional structure but is corrupted by some noise. In this case, applying common nonlinear dimensionality reduction techniques, such as locally-linear embedding (LLE) [104], ISOMAP [116], or kernel PCA [110] algorithms, on the noisy observations may not lead to a meaningful low-dimensional representation of the data. When the observed noisy data are not located on an underlying manifold of interest, we need first to estimate the points on the underlying manifold before applying a dimensionality reduction technique. Principal curves and surfaces provide a reasonable low-dimensional representation of data and can be used as a preprocessing step before applying common nonlinear dimensionality reduction techniques. As mentioned before, the SCMS algorithm has the capability to estimate principal curves and surfaces. After this estimation step, one can apply dimensionality reduction techniques to obtain a representation of the data in a low-dimensional space. To illustrate how this works we used the SMCS algorithm as a preprocessing step before the LLE and the ISOMAP.

5.3.1 The SMCS algorithm before the LLE

We selected 500 samples uniformly from a three-dimensional spiral. Then independent, three-dimensional zero mean Gaussian noise with per component variance 0.7 is added to those samples. Fig. 5.1 shows a scatterplot of the noisy observations and the output of the SCMS algorithm. To assess the performance of the SCMS algorithm, we selected 12 clean data points from the spiral (the markers in Fig. 5.1 represent the selected points) and applied the LLE algorithm [104] to obtain their one-dimensional representation. The three dimensional spiral has an intrinsic dimensionality of one, thus we reduced the dimension of the selected points to one. The second row in Fig. 5.2 shows the representation of the selected clean points in one-dimensional space. Then we applied the LLE algorithm to the estimates of these points computed as the output of the SCMS algorithm that was run on the noisy data. The first row in Fig. 5.2 shows the representation of the estimated points (output of the SCMS algorithm) after applying the LLE algorithm. The third row in Fig. 5.2 is a one-dimensional representation of the noisy points after directly applying the LLE algorithm. It can be observed from Fig. 5.1 and Fig. 5.2 that although the observed data was corrupted by Gaussian noise, applying the LLE algorithm to the output of the SCMS algorithm gives a one-dimensional representation very similar to that of the clean data. On the other hand, applying the LLE algorithm directly to the noisy version of the observed data changes the pairwise distances and the one-dimensional order of the points, which is not desirable.

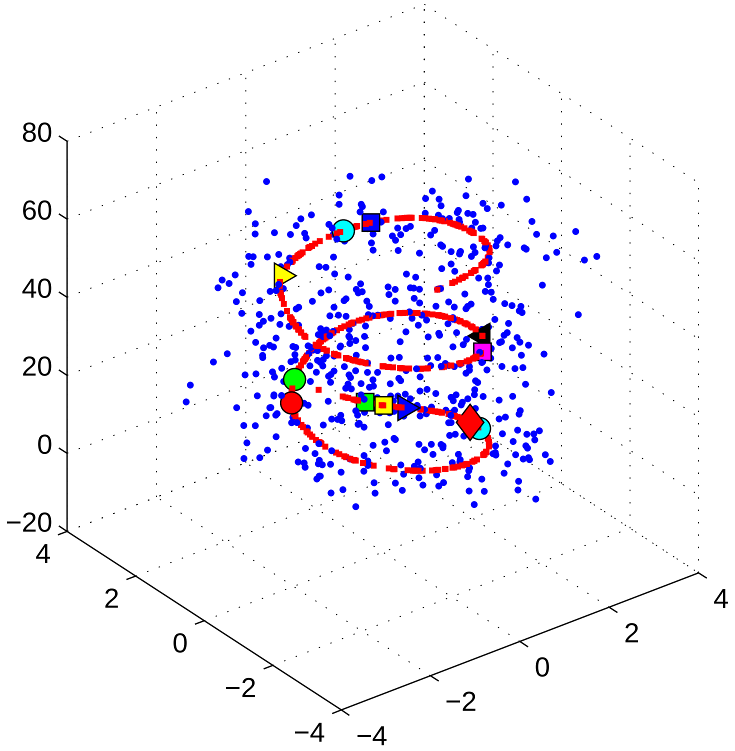

Figure 5.1: Applying the SCMS algorithm on the noisy data in order to estimate the clean data.The red points represent the output of the SCMS algorithm and the blue points are the noisy data.

5.3.2 The SMCS algorithm before the ISOMAP

To demonstrate the effectiveness of the SCMS algorithm as a preprocessing step for nonlinear dimensionality reduction using ISOMAP, we uniformly selected 750 samples from a two-dimensional spiral. The sample is corrupted by adding independent, two-dimensional zero mean Gaussian noise with per component variance 0.7. Fig. 5.3 shows a scatterplot of the noisy observations and the output of the SCMS algorithm. To show the performance of the SCMS algorithm, we selected 10 clean data points from the spiral (the markers in

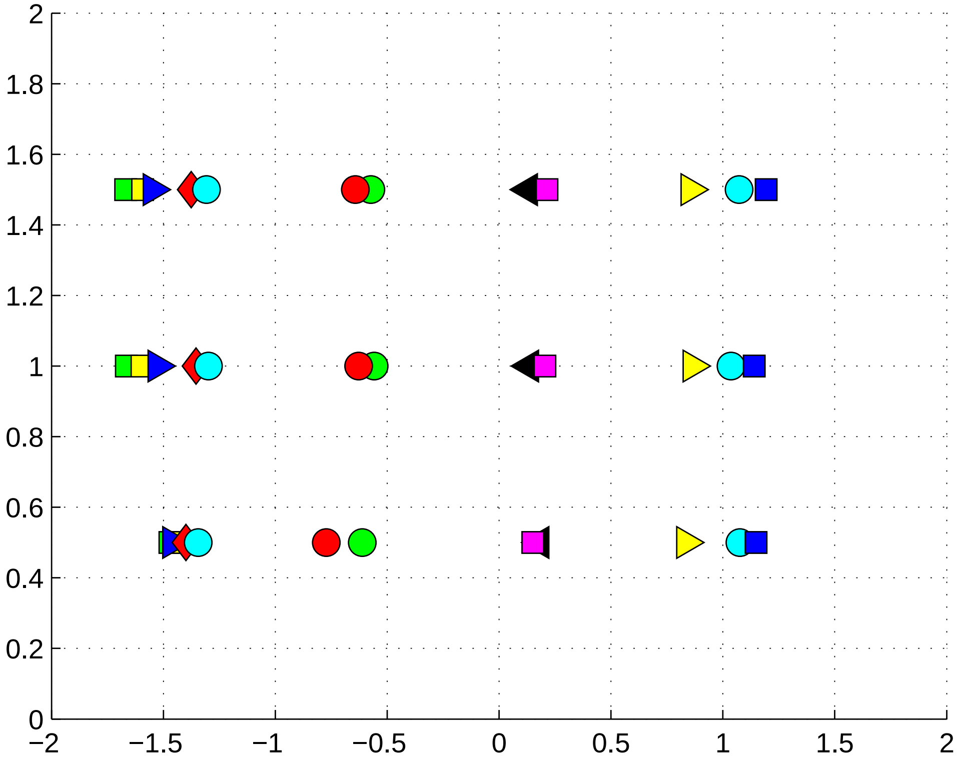

Figure 5.2: The first row shows the output of the LLE algorithm when applied to the SCMS estimates of the selected points.The second row shows the output of the LLE algorithm applied to the clean data points. The third row shows the output of the LLE algorithm applied directly to the noisy data points.

Fig. 5.3 represent the selected points) and applied the ISOMAP algorithm [116] to them to obtain a one-dimensional representation. The intrinsic dimensionality of two-dimensional spiral is one, thus we reduced the dimension of the selected points to one. The second row in Fig. 5.4 shows the representation of the selected clean points in one-dimensional feature space generated by ISOMAP. Then we applied the ISOMAP algorithm to the estimates of the selected points computed as the output of the SCMS algorithm that was run on the noisy data. The first row in Fig. 5.4 shows the representation of the estimated points (output of the SCMS algorithm) after applying the ISOMAP algorithm. We also directly applied the ISOMAP algorithm on the noisy data points to find the one-dimensional representation of them. The third row in Fig. 5.4 represents the one-dimensional representation of the noisy points in the ISOMAP feature space. It can be observed from Fig. 5.3 and Fig. 5.4 that although the observed data was corrupted by Gaussian noise, applying the ISOMAP algorithm to the output of the SCMS algorithm gives a one-dimensional representation very similar to that of the clean data. The order of the selected points and distance between them are preserved. On the other hand, applying the ISOMAP algorithm directly to the noisy version of the observed data changes pairwise distances and the one-dimensional order of the selected points.

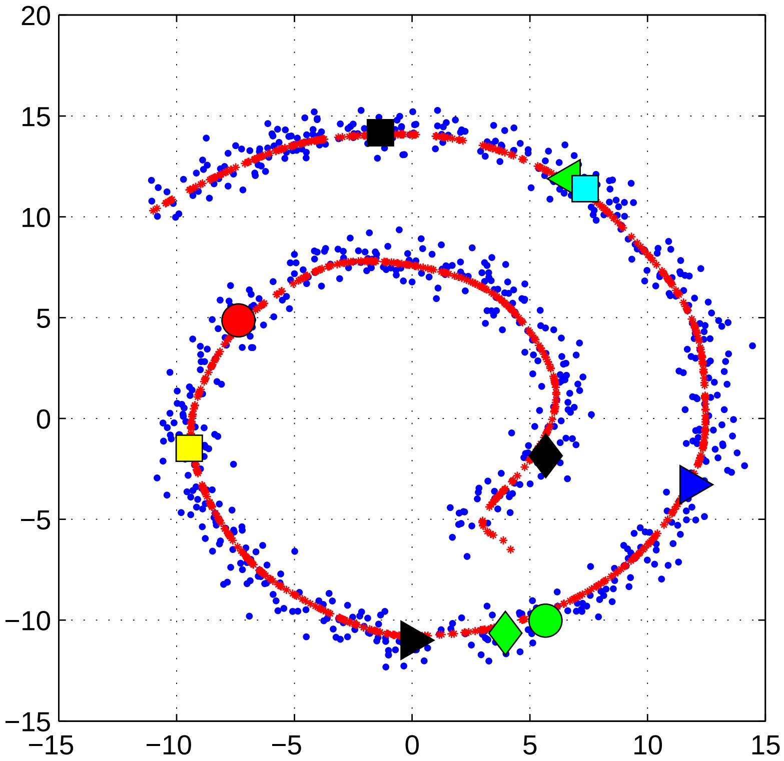

Figure 5.3: Applying the SCMS algorithm on the noisy data in order to estimate the clean data.The red points represent the output of the SCMS algorithm and the blue points are the noisy data. The markers show the selected clean data points from the spiral

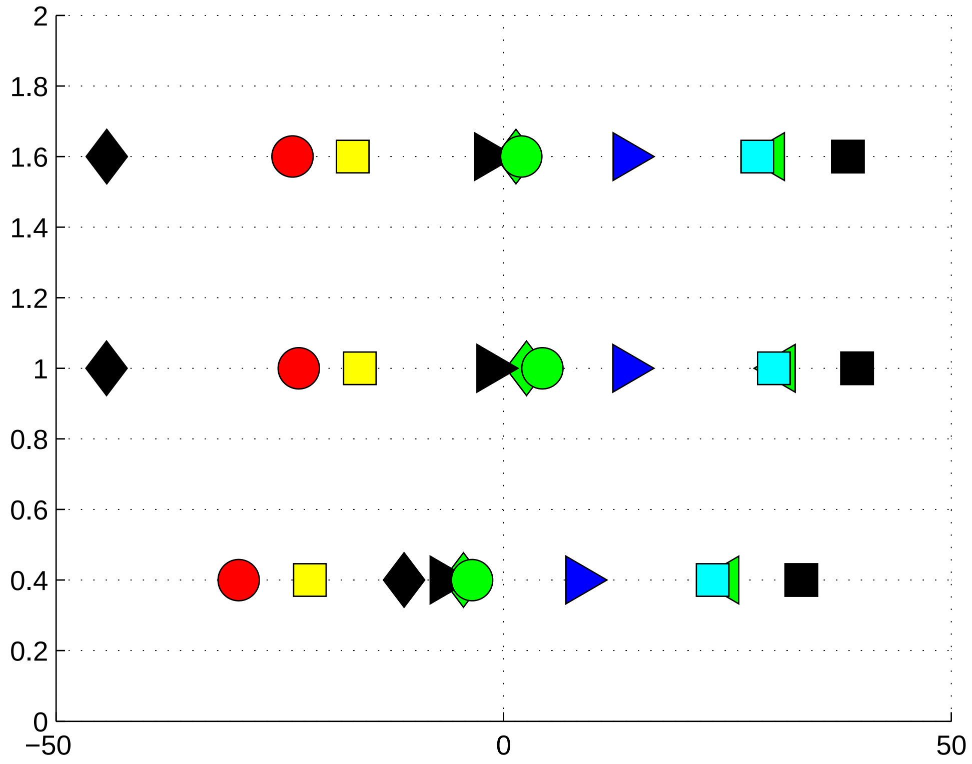

Figure 5.4: The first row shows the output of the ISOMAP algorithm when applied to the SCMS estimates of the selected points.The second row shows the output of the ISOMAP algorithm applied to the clean data points. The third row shows the output of the ISOMAP algorithm applied directly to the noisy data points.

Chapter 6

New Applications of the MS and SCMS

6.1 Hough Transform

6.1.1 Introduction

The problem of detecting straight lines in digital images is of great importance in image processing and machine vision. The detection of lines can be used in a variety of applications, such as object detection/recognition [93, 78, 26], data base navigation [75], target tracking [59, 22], and camera calibration [118]. The classical Hough transform, introduced by Hough [62], has been widely used in the image processing community as a technique for detecting straight lines. The Hough transform basically implements a voting procedure for all potential lines in an image, such that at the termination step the algorithm keeps the lines with high voting scores. The original motivation for the algorithm was the detection of straight lines in photographs obtained in cloud chambers [61]. Later the Hough transform was extended to detect the position of shapes with a specified parametric form, most commonly circles or ellipses [35][76]. In its extended form, the algorithm creates meaningful groups of features that satisfy some parametric constraint. The transform was popularized in the computer vision community in the 90s when Ballard revealed the potential application of the Hough transform to detect arbitrary shapes [5]. In the rest of this section we focus solely on the Hough transform for detecting straight lines; however the given results can be generalized for finding parametric curves.

The main idea behind the algorithm is to consider sets of colinear points in an image [64]. A set of image pixels that lie on a straight line can be described by $y = mx + b$, where (x, y) denote the pixel's location in the image. By considering the characteristics of a straight line in terms of the slope m and the intercept b, for each pixel p there are infinitely many straight lines passing through it. In other words, an arbitrary pixel p with coordinates (x, y) on the image defines a bundle of straight lines in $m - b$ space and each straight line on the image maps to a single point in $m - b$ space. Thus a single line connecting any two arbitrary pixels p and q lies on the intersection of two of straight lines representing p and q in $m - b$ space. The main practical difficulty of the previous representation arises for vertical lines, since the slope m becomes infinite. Duda and Hart [35] introduced an alternative line parametrization, called normal parametrization, given by

$$\rho = x\sin(\theta) + y\cos(\theta), \tag{6.1}$$

where ρ is the algebraic line distance from the origin and θ is the angle of the normal line. The new $\rho - \theta$ space is called the Hough space [35]. Figure 6.1 illustrates ρ and θ for an arbitrary line on the plane. The idea is the same as before: an arbitrary pixel p on the image is mapped to all straight lines that pass through it. This yields a sine-like curve in the Hough space. Then we have the following observations:

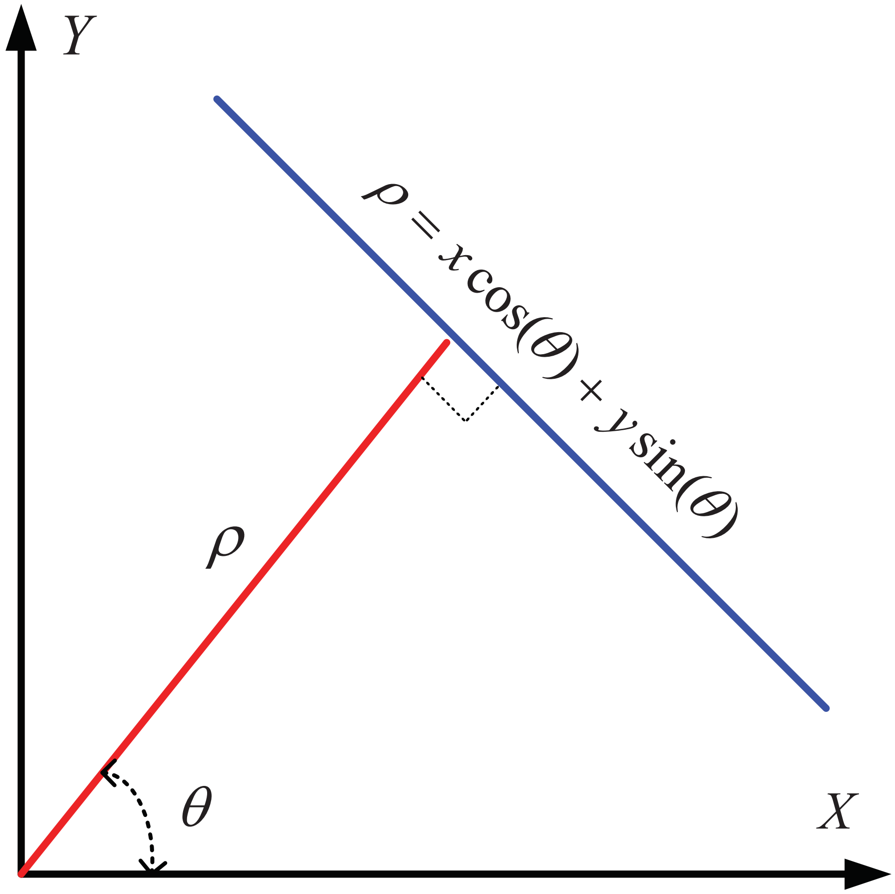

Figure 6.1: The algebraic distance of the straight line from the origin is denoted by ρ and θ is the angle of the normal line

- A pixel in the image corresponds to a sinusoidal curve in the Hough space.
- A point in the Hough space corresponds to a straight line in the image.
- Points lying on the same straight line in the image correspond to sinusoidal curves passing through a common point in the Hough space.

- Points lying on the same sinusoidal curve in the Hough space correspond to straight lines passing through the same point in the image.

The above observations have been used to detect colinear points in an image. The computational burden can be reduced considerably by specifying an acceptable error in ρ and θ. To this end, the Hough space is quantized into finite intervals or accumulator cells. As the algorithm runs, each pixel of the image is transformed into a discretized curve and the accumulator cells that lie along this curve are incremented. The resulting peaks in the accumulator array are the candidates to represent straight lines in the image. The Hough transform algorithm for line detection can be summarized as follows

1. Find all edges in the image, e.g. using Canny edge detector [10].

2. Initialize all accumulator cells in Hough space to zeros.

3. Map edge points to the Hough space and increment corresponding accumulator cells.

4. Find the local maxima in the accumulator space. The local maxima represent the straight lines in the image.

As mentioned before, quantizing the Hough space specifies an error in the values of ρ and θ and in fact the major source of error comes from the finite size of the accumulator array. In other words, the resolution of the accumulator determines the accuracy of the detected lines. The cell size must not be too small, otherwise some votes will fall in the neighboring bins, which will reduce the visibility of the main bin. The cell size must also not be too big, since this would result in the generation of inaccurate lines. Furthermore, there exists no criterion to guess the total number of the straight lines in an image. We usually define a threshold and if the total number of votes for a cell is higher than the specified threshold, the cell represents a line in the image. The threshold is usually chosen heuristically and for each specific problem it needs to be determined separately. In the rest of this section we show how using the mean shift algorithm can address the above mentioned problems.

6.1.2 Proposed algorithm

The first step is similar to the original Hough transform technique. All the edges in the image are found using an edge detector and the output is given as a binary (zero, one) image. Each two edge pixels represent a possible straight line with a specific ρ and θ. Therefore, if D denotes the total number of the edge pixels then each pixel generates $D-1$ points in the Hough space and the total number of points in the Hough space will be $D \times (D-1)/2$. The Hough space is not required to be quantized, hence the computed pair of parameters (ρ, θ) is more precise than the Hough algorithm. The two-dimensional points in the Hough space are given as input data to the mean shift algorithm. The algorithm starts from the points in the Hough space and iteratively tries to estimate modes of the underlying pdf. The output of the algorithm is a small set containing the modes of the estimated pdf. Each mode represents a cluster corresponding to the set of all data points converging to that mode. The modes with the highest number of converging data points will be our candidates to represent the straight lines in the image. Instead of computing the Hough parameters for all pairs of the edge pixels, which increases the computational cost, we can find (ρ, θ) pair just for k nearest neighbors of each pixel. Intuitively, if two pixels are far from each other, the probability that they fall in a line is not significant. The proposed algorithm can be summarized as follows

1. Find all edges in the image, e.g. using Canny edge detector [10].

2. For each edge pixel, find its k nearest neighbors.

3. Each edge pixel and its k nearest neighbors define k straight lines. Compute the pairs of (ρ, θ) for these lines. If D denotes the total number of the edge pixels, then we will have $D \times k$ points in the Hough space.

4. Apply the mean shift algorithm on the points in the Hough space and round the outputs to the nearest integer.

5. The rounded outputs of the mean shift algorithm that attracted the highest number of the data points are the most likely lines in the image space.

6.1.3 Simulation results

For the Hough transform in the following simulations the parameter θ, measured in degrees, is quantized to 180 cells, $-90 \leq \theta \leq 90$, and the parameter ρ is quantized into $\sqrt{N^2 + M^2}$ cells for a $M \times N$ image. The bandwidth h for the MS algorithm and the number of the nearest neighbors k are set for each simulation separately. We first test the performance of the proposed algorithm on simple binary images and then we perform simulations on gray scale images.

- The input image is a 400×400 binary image. We uniformly select 40 points on each of the following lines $y = x + 25$, $y = -0.1x + 50$, and 20 points are uniformly selected on each of $y = 3.8x - 379$, $y = -0.2x + 224$. We also randomly select 180 points from the image in order to have a total of 300 edge pixels. The bandwidth h used in the MS algorithm is equal to 5 and for each edge pixel we computed its

10 nearest neighbors ($k = 10$). Figure 6.2 shows the edge points and the detected straight lines using the proposed method and the Hough transform. The detected lines using the two algorithms may seem similar in Figure 6.2, but comparing the detected values of (ρ, θ) with the exact values reveals that the proposed algorithm generated more accurate results. Table 6.1 compares the computed values of the pair (ρ, θ) using the two algorithms with their exact values. It can be observed that computed (ρ, θ) using the proposed method is closer to the actual values. The bins with the highest number of votes in the Hough space contain 41, 19, 12, 10, 7, 6, 6, and 6 votes, which makes it difficult to guess the right number of straight lines. For the proposed method, the first 8 dominant modes attract 134, 127, 92, 70, 30, 20, 20, and 19 points. By observing the number of the attracted points for each mode, we can guess that there are two large length lines and two lines of smaller length.

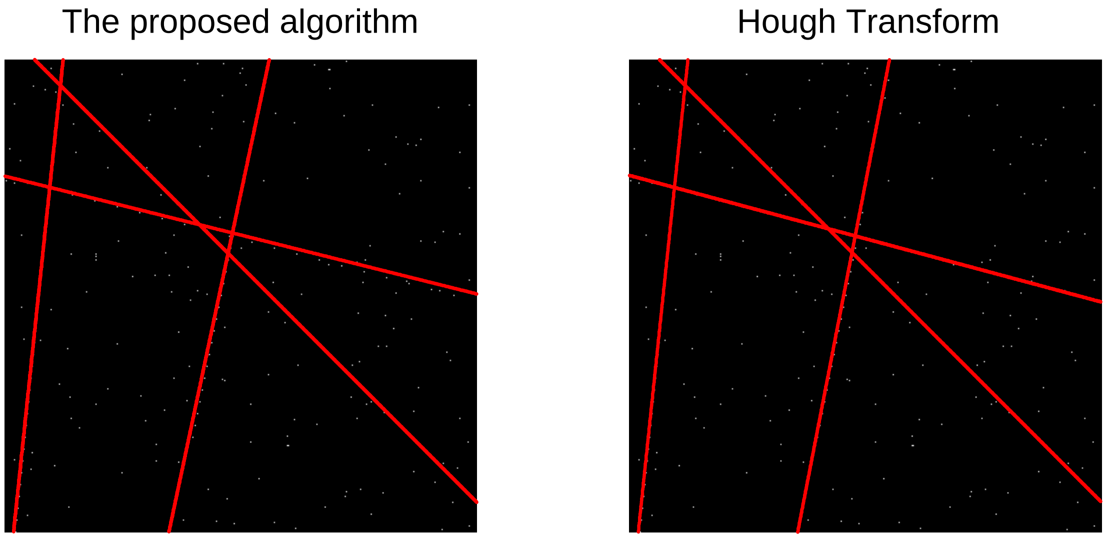

Figure 6.2: The white pixels represent the selected points and the red lines are the detected lines using the proposed algorithm and the Hough transform. The bandwidth h for the proposed algorithm is 5 and for each pixel we computed its 10 nearest pixels.

Table 6.1: Computed values of ρ and θ for each line using the proposed method and the Hough transform.

	Line 1	Line 2	Line 3	Line 4
Exact value of θ	84.29	-45.00	-14.74	78.69
θ using the MS	84	-45	-15	78
θ using the HT	85	-45	-11	75
Exact value of ρ	49.75	-17.67	-96.45	219.65
ρ using the MS	50	-18	-96	220
ρ using the HT	50	-19	-95	217

Table 6.2: The total number of votes for the bins with the highest votes and the total number of the attracted points by the most attractive modes (vase).

Mode/Bin	1	2	3	4	5	6	7	8	9	10	11	12
The proposed algorithm	1758	1632	1571	1477	684	521	517	492	258	257	245	223
The Hough Transform	98	88	87	76	38	33	28	28	27	26	26	25

- The input is a binary image. For the proposed algorithm we choose the bandwidth $h = 5$ and the number of the nearest neighbors for each pixel is set to be 20. Table 6.2 shows the cells with the highest number of votes and the mode estimates that attract the highest number of points. From Table 6.2 it can be observed for the proposed algorithm that the number of the attracted points decreases after the eighth mode so we can predict that the number of the straight lines should be eight. Note that this observation cannot be made using the output of the Hough transform. Figure 6.3 shows the first eight detected lines using the proposed technique and using the Hough transform. It is clear from Figure 6.3 that the proposed algorithm has successfully detected eight straight lines but the Hough transform has missed one of them.

- The input is a gray scale image containing seven straight lines separating black and white areas. Figure 6.4 shows the original image and the extracted edges. We select

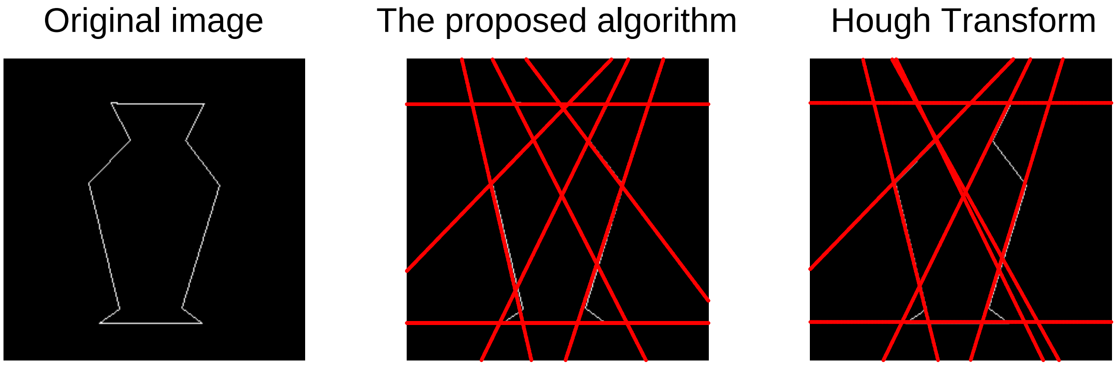

Figure 6.3: The image on the left is the original binary image, the middle image shows the detected lines using the proposed technique and the right image shows the detected lines using the Hough transform

Table 6.3: The total number of votes for the bins with the highest votes and the total number of the attracted points by the most attractive modes (eight lines).

Mode/Bin	1	2	3	4	5	6	7	8	9	10	11	12
The proposed algorithm	4959	4881	4471	3570	3150	2520	1055	700	360	306	86	72
The Hough Transform	181	150	131	113	111	99	45	41	29	20	20	19

the bandwidth h to be 5, and the number of the nearest neighbors is set to be 20. Table 6.3 shows the local peaks in the Hough space and the modes with the highest number of attracting points. The number of attracted points decreases after the eights mode, which indicates that the number of the straight lines in the image should be seven. Figure 6.5 shows the detected lines using the proposed algorithm and using the Hough transform.

- The input image is a gray scale box. We first apply an edge detector to extract the edge pixels. Figure 6.6 shows the box and the extracted edges. The bandwidth h for the mean shift algorithm is 5, and for each pixel we compute its 20 nearest neighbors. The local peaks in the Hough space and the modes with the highest number of the

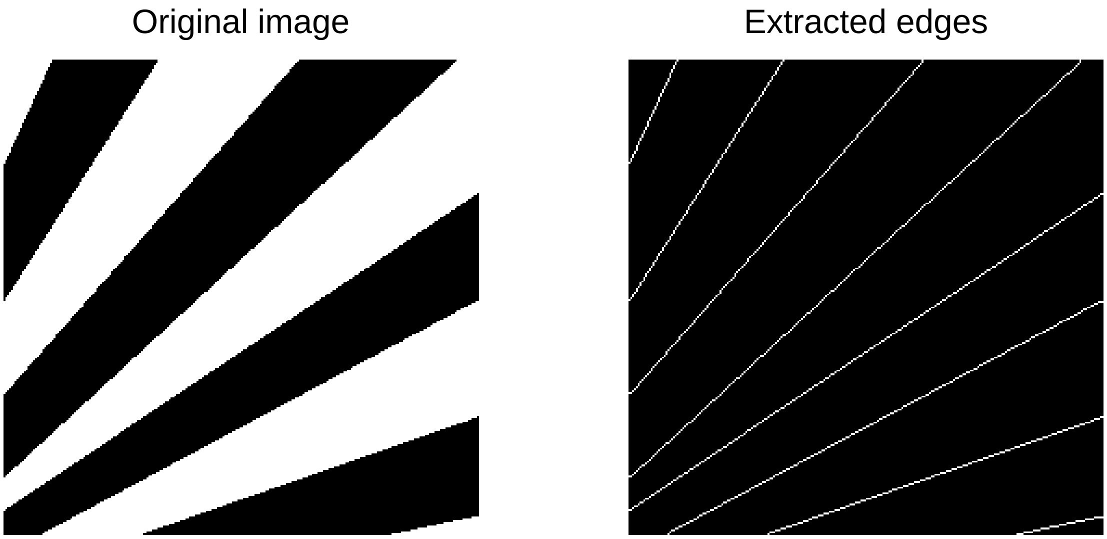

Figure 6.4: The left image is the original image and the right image shows the extracted edges.

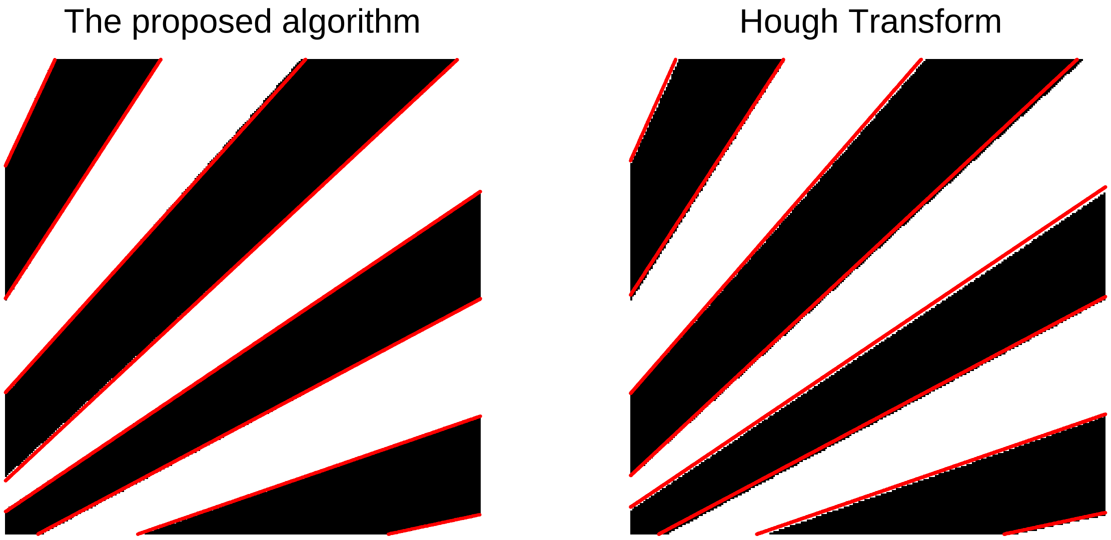

Figure 6.5: The left image shows the detected line using the proposed algorithm and the right image shows the detected lines using the Hough transform.

Table 6.4: The total number of votes for the bins with the highest votes and the total number of the attracted points by the most attractive modes.

Mode/Bin	1	2	3	4	5	6	7	8	9	10	11	12
The proposed algorithm	2479	1831	1689	1462	1310	684	597	540	210	150	137	132
The Hough Transform	84	82	74	61	55	47	38	34	30	28	27	26

attracted points are shown in Table 6.4. From Table 6.4, we observe that the number of straight lines in the image should be eight. Figure 6.7 compares the first 9 detected lines using the proposed algorithm and the Hough transform. The proposed algorithm has found all the straight lines successfully, but the Hough transform has missed one of the lines.

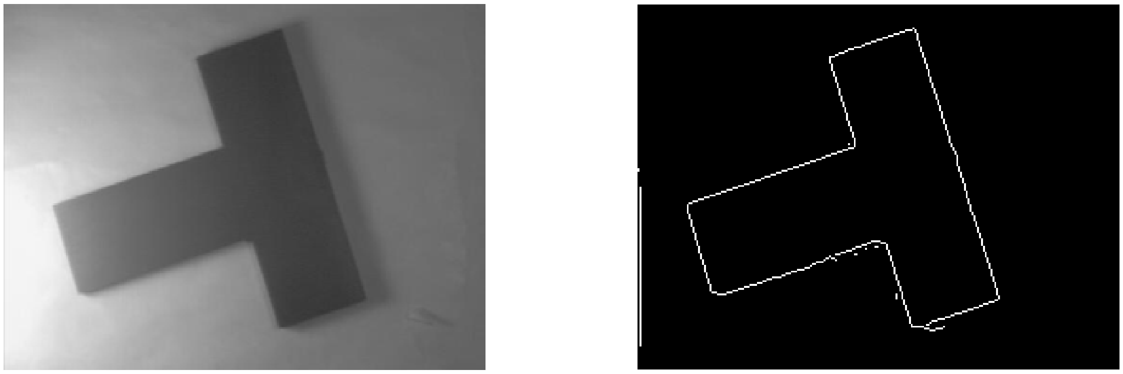

Figure 6.6: The image on the left shows the original gray scale image and the image on the right side shows the extracted edges.

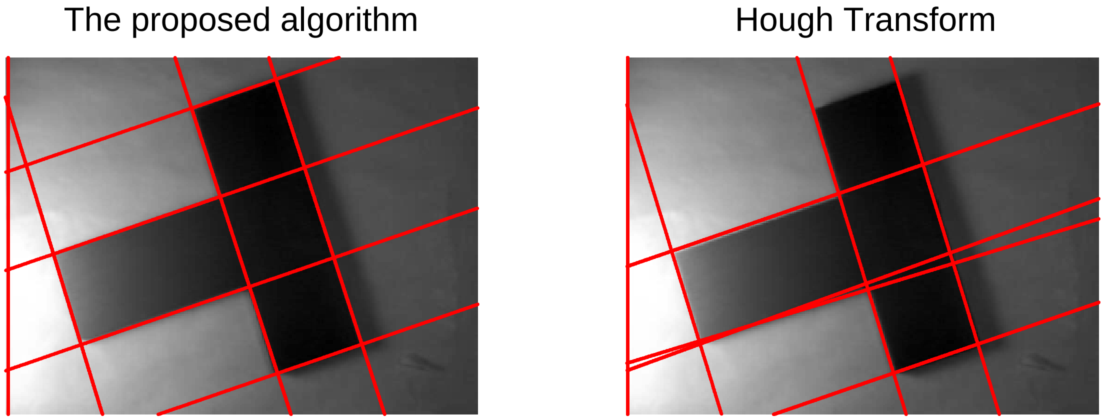

Figure 6.7: The left figure shows the detected straight lines using the proposed technique and the right figure shows the detected lines using the Hough transform.

6.2 Noisy Source Vector Quantization via SCMS Algorithm

6.2.1 Introduction

Vector quantization is an important building block used in lossy data compression. A vector quantizer encodes (maps) vectors from a multidimensional input space into a finite subset of the space, called the codebook. The design of quantizers has been extensively studied. A classical result shows that an optimal quantizer of a given codebook size has to satisfy the Lloyd-Max conditions [88][90]. This gives rise to the Lloyd-Max algorithm, an iterative method for scalar quantizer design that alternates between optimizing the codebook and the partition induced by the codebook. The generalized version of the Lloyd-Max algorithm, known as the LBG algorithm, is used to design (locally) optimal vector quantizers [85][53]. The classical problem of optimal vector quantization assumes that the source is available noise free to the quantizer. However, in some situations the source output may be corrupted by noise due to, e.g., measurement errors [102]. In this case, only a noisy version of the source is available for the quantization, and the quantizer's goal is then to minimize the expected distortion between the clean (unobserved) source and the output of the quantizer. Some practical examples where this model may apply are pilot's speech in the presence of aircraft noise, digital signal processing at transmitter or receiver that introduce quantization and round-off errors, satellite images affected by measurement error, or speech signal for a mobile phone in a noisy environment.

The theory of noisy source coding was first investigated by Dobrushin and Tsybakov [32] who analyzed the optimal rate-distortion performance. The structure of the optimal noisy source quantizer under the mean square distortion was studied by Fine [40], Sakrison [106], and Wolf and Ziv [126]. It has been shown that for the mean square distortion an optimal noisy source quantization system can be decomposed into an optimum estimator followed by an optimum source coder operating on the estimator output [126]. Some properties of an optimum noisy source quantizer, and its relations with the optimal estimator for the general problem, are derived by Ayanoglu [2]. By appropriately modifying the given distortion measure, Ephraim and Gray [38] showed the noisy source quantization problem becomes a standard quantization problem for the noisy source using the modified distortion measure. The problem of empirical vector quantizer design for noisy sources has been investigated by Linder, Lugosi, and Zeger[87]. The classical results imply that in order to minimize the mean square distortion with respect to the clean data, one needs to quantize the conditional expectation of the clean data given the noisy data. Thus, we need to find a good approximation, in the minimum mean square error (MMSE) sense, of the clean data from the observed noisy data. In practical situations where the statistics of the data and noise are unknown, the clean data can be estimated by applying nonparametric techniques, such as kernel regression [92], based on training data. However, in practice training data from the clean source may not be available and the designer of the quantizer only has access to the noisy observations.

6.2.2 Vector quantization

A fixed rate N-level vector quantizer $Q : \mathbb{R}^D \to \mathcal{C}$ is a mapping from the D-dimensional Euclidean space $\mathbb{R}^D$ into a finite set $\mathcal{C} = \{\boldsymbol{c}_1, \ldots, \boldsymbol{c}_N\}$ of N distinct points in $\mathbb{R}^D$. The set $\mathcal{C}$ is called the codebook and the elements of $\mathcal{C}$ are called the codevectors. Every N point vector quantizer Q partitions $\mathbb{R}^D$ into N regions or cells, $R_i, i = 1, \ldots, N$ [46]. The ith quantizer cell is given by $R_i = \{\boldsymbol{x} : Q(\boldsymbol{x}) = \boldsymbol{c}_i\}, i = 1, \ldots, N$. From the definition of the quantizer cells, it follows that

$$\bigcup_{i=1}^{N} R_i = \mathbb{R}^D \text{ and } R_i \cap R_j = \emptyset \text{ if } i \neq j.$$

The performance of a fixed rate quantizer in approximating the input vector is measured using a non-negative function $d : \mathbb{R}^k \times \mathbb{R}^k \to [0, \infty)$ called the distortion measure. The quantity $d(\boldsymbol{x}, Q(\boldsymbol{x}))$ measures the reconstruction error in representing $\boldsymbol{x}$ by $Q(\boldsymbol{x})$. There are different criteria to measure the distortion, including the squared error distance, the rth power distortion, and the weighted squared error. For a D-dimensional random vector $\boldsymbol{X}$ the overall distortion D of a quantizer Q is the expected value of the reconstruction error and is given by

$$\begin{aligned} D &= Ed(\boldsymbol{X}, Q(\boldsymbol{X})) \\ &= \int_{\mathbb{R}^D} d(\boldsymbol{x}, Q(\boldsymbol{x}))f(\boldsymbol{x})d\boldsymbol{x} \text{ if } \boldsymbol{X} \text{ has pdf} f. \end{aligned}$$

The most common and tractable distortion measure is the mean square error (MSE) distortion,i.e., $D = E\|\boldsymbol{X} - Q(\boldsymbol{X})\|^2$. An N-level quantizer Q is called the nearest neighbor vector quantizer if for all $\boldsymbol{x} \in \mathbb{R}^D$, $Q(\boldsymbol{x}) = \arg\min_{\boldsymbol{c}_i \in \mathcal{C}} d(\boldsymbol{x}, \boldsymbol{c}_i)$. Among all N-points vector quantizers, the optimal vector quantizer Q^* is defined as follows

Definition 6.1. *Let $\mathcal{Q}_N$ denote the family of all D-dimensional N-level quantizers. $Q^* \in \mathcal{Q}_N$ is an optimal quantizer for source $\boldsymbol{X}$ if*

$$Ed(\boldsymbol{X}, Q^*(\boldsymbol{X})) = \min_{Q \in \mathcal{Q}_N} Ed(\boldsymbol{X}, Q(\boldsymbol{X})).$$

The optimal quantizer Q^* depends on the distribution of $\boldsymbol{X}$ and the distortion measure d and is not necessarily unique. We have the following necessary conditions for optimality [86]

(a) Nearest neighbor condition (NNC): Any nearest neighbor quantizer has minimum distortion among all N-level vector quantizers with the same codebook.

(b) Centroid condition (CC): Consider all N-level vector quantizers with given cells $R_i, i = 1, \ldots, R_N$. Among these, the quantizer Q with output points

$$\boldsymbol{c}_i = \arg\min_{\boldsymbol{c} \in \mathbb{R}^D} E\Big[d(\boldsymbol{X}, \boldsymbol{c}) | \boldsymbol{X} \in R_i)\Big], i = 1, \ldots, N.$$

has minimum distortion.

The CC and the NNC are necessary but not sufficient conditions for the optimality of an N-level vector quantizer.

6.2.3 Nosiy source vector quantization

Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be jointly distributed k-dimensional random vectors with $\boldsymbol{X}$ representing the clean source and $\boldsymbol{Y}$ representing the noisy version of $\boldsymbol{X}$. The problem of noisy source vector quantization is to approximate the clean data $\boldsymbol{X}$ with the lowest distortion based on quantizing its noisy version $\boldsymbol{Y}$ at a given fixed rate. Formally, our encoder is a member of the set of all N-level quantizers $\mathcal{Q}_N$ on $\mathbb{R}^k$. Assuming that $E\|\boldsymbol{X}\|^2$ is finite, the noisy source quantization problem is to find $Q^* \in \mathcal{Q}_N$ with minimum distortion

$$E\|\boldsymbol{X} - Q^*(\boldsymbol{Y})\|^2 = \min_{Q \in \mathcal{Q}_N} E\|\boldsymbol{X} - Q(\boldsymbol{Y})\|^2. \tag{6.2}$$

It has been shown that the structure of an optimal N-level quantizer Q^* can be obtained via a useful decomposition [106][126]. The following summarizes these results.

Proposition 6.1. *Let $m(\boldsymbol{y}) = E[\boldsymbol{X}|\boldsymbol{Y} = \boldsymbol{y}]$. Then an optimal quantizer Q^* is given by $Q^*(\boldsymbol{y}) = \hat{Q}(m(\boldsymbol{y}))$, where $\hat{Q} \in \mathcal{Q}_N$ is an MSE optimum N-level quantizer for $m(\boldsymbol{Y})$, i.e., $\hat{Q} = \arg\min_{Q \in \mathcal{Q}_N} E\|m(\boldsymbol{Y}) - Q(m(\boldsymbol{Y}))\|^2$. Furthermore, $\min_{Q \in \mathcal{Q}_N} E\|\boldsymbol{X} - Q(\boldsymbol{Y})\|^2 = E\|\boldsymbol{X} - m(\boldsymbol{Y})\|^2 + \min_{Q \in \mathcal{Q}_N} E\|m(\boldsymbol{Y}) - Q(m(\boldsymbol{Y}))\|^2$.*

Proof. Let $Q \in \mathcal{Q}_N$ be an arbitrary N-level quantizer. Then

$$\begin{aligned} E[\|\boldsymbol{X} - Q(\boldsymbol{Y})\|^2|\boldsymbol{Y}] &= E[\|\boldsymbol{X} - m(\boldsymbol{Y})|\boldsymbol{Y}\|^2] \\ &\quad + \|m(\boldsymbol{Y}) - Q(\boldsymbol{Y})\|^2 \\ &\quad + 2E[(\boldsymbol{X} - m(\boldsymbol{Y}))^T|\boldsymbol{Y}](m(\boldsymbol{Y}) - Q(\boldsymbol{Y})) \\ &= E[\|\boldsymbol{X} - m(\boldsymbol{Y})\|^2|\boldsymbol{Y}] \\ &\qquad + \|m(\boldsymbol{Y}) - Q(\boldsymbol{Y})\|^2, \end{aligned}$$

where the inner product term disappears after taking iterated expectations, first conditioned on $\boldsymbol{Y}$. Since the first term of the last expression does not depend on Q, in order to minimize $E[\|\boldsymbol{X} - Q(\boldsymbol{Y})\|^2] = E(E[\|\boldsymbol{X} - Q(\boldsymbol{Y})\|^2|\boldsymbol{Y}])$, we have to find $Q \in \mathcal{Q}_N$ that minimizes $E[\|m(\boldsymbol{Y}) - Q(\boldsymbol{Y})\|^2]$. Now suppose $\{c_1, \ldots, c_N\}$ are the codewords of Q and let $Q(\boldsymbol{y}) = c_j$. Then,

$$\begin{aligned} \|m(\boldsymbol{y}) - Q(\boldsymbol{y})\|^2 &= \|m(\boldsymbol{y}) - c_j\|^2 \\ &\geq \min_{1 \leq i \leq N} \|m(\boldsymbol{y}) - c_i\|^2, \end{aligned}$$

which shows that

$$E[\|m(\boldsymbol{Y}) - Q(\boldsymbol{Y})\|^2] \geq E[\min_{1 \leq i \leq N} \|m(\boldsymbol{Y}) - c_i\|^2].$$

This means that given $\boldsymbol{Y} = \boldsymbol{y}$, the optimum encoding rule is to form a nearest neighbor quantizer using the codepoints of Q and encode $m(\boldsymbol{y})$ optimally with this quantizer. It then follows that if $\hat{Q}$ denotes the optimum N-level (nearest neighbor) quantizer minimizing

$E\|m(\boldsymbol{Y}) - \hat{Q}(m(\boldsymbol{Y}))\|^2$, then

$$\min_{Q \in \mathcal{Q}_N} E[\|m(\boldsymbol{Y}) - Q(\boldsymbol{Y})\|^2] = E[\|m(\boldsymbol{Y}) - \hat{Q}(m(\boldsymbol{Y}))\|^2]$$

and Q^* defined by $Q^*(\boldsymbol{Y}) = \hat{Q}(m(\boldsymbol{Y}))$ is the optimum noisy source quantizer such that

$$\begin{aligned}\min_{Q \in \mathcal{Q}_N} E[\|\boldsymbol{X} - Q(\boldsymbol{Y})\|^2] &= E[\|\boldsymbol{X} - Q^*(\boldsymbol{Y})\|^2] \\ &= E[\|\boldsymbol{X} - m(\boldsymbol{Y})\|^2] \\ &+ E[\|m(\boldsymbol{Y}) - \hat{Q}(m(\boldsymbol{Y}))\|^2],\end{aligned}$$

as claimed. □

Thus, in order to minimize the distortion, one needs to find a good approximation of the clean data $\boldsymbol{X}$ based on the observed noisy data $\boldsymbol{Y}$. In practical situations where the statistics of the data and noise are unknown, the clean data can be estimated using nonparametric techniques, such as kernel regression, based on training data. If a set of training data $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1\ldots,n}$ is available in advance, the conditional expectation $m(\boldsymbol{y}) = E[\boldsymbol{X}|\boldsymbol{Y} = \boldsymbol{y}]$ of $\boldsymbol{X}$ given $\boldsymbol{Y}$ can be estimated using the kernel regression method as

$$\hat{m}(\boldsymbol{y}) = \frac{\sum_{i=1}^{n} \boldsymbol{x}_i K_h(\boldsymbol{y} - \boldsymbol{y}_i)}{\sum_{i=1}^{n} K_h(\boldsymbol{y} - \boldsymbol{y}_i)}, \tag{6.3}$$

where $K_h : \mathbb{R}^k \to [0, \infty)$ is an integrable kernel function with bandwidth h. In this paper we assume that the designer of the quantizer only has access to the noisy observations and training data from the clean source are not available.

6.2.4 Applying the SCMS algorithm

Since very little is known theoretically about the performance of the SCMS algorithm, we will use numerical examples to assess how well the SCMS algorithm approximates the clean data for the purposes of quantization. We compare the performance of the obtained system with that of a system using the kernel regression method trained on a data set consisting of pairs of clean and noisy data samples. In particular, in two different scenarios we compare the mean square distortion that results from quantizing the output of the SCMS algorithm with the near-optimal distortion resulting from quantizing the estimated clean data using the kernel regression method. We note that the kernel regression method is asymptotically optimal in the limit of large training set sizes.

6.2.5 Quantization of a noisy line

We examine the performance of the SCMS algorithm as a preprocessing step for noisy vector quantization on a straight line in $\mathbb{R}^2$. In the design stage, we uniformly select 500 samples from the straight line of length 4 and perturb them by additive, independent zero-mean bivariate Gaussian noise with per component variance 0.4. These points are fed the SCMS algorithm and the resulting 500 output points are then used as a training set to design a vector quantizer using the LBG algorithm. For testing, we select another 500 samples from the straight line and perturb them by noise, the noisy data is then fed to the SCMS algorithm, and the output of the algorithm is quantized using the designed vector quantizer. Fig. 6.8 shows the performance of the SCMS algorithm to estimate the clean data. In Fig. 6.8, the black points are the noisy observations and the blue points are outputs of the SCMS algorithm. It can be observed from Fig. 6.8 that the outputs of the SCMS algorithm lie close to the straight line. The red points represent the codewords computed by the LBG vector quantization algorithm. As expected, the codewords are distributed uniformly on the straight line. We vary the number of codewords and run the simulations for quantizers of codebook size 1, 2, 4, 8, 16, and 32. Fig. 6.9 shows the output points of the SCMS algorithm and the computed codewords for each choice of the codebook size. The blue points in Fig. 6.9 represent the output points of the SCMS algorithm and the red points represent the codewords computed by the LBG algorithm. To compare the performance of the SCMS approach with the theoretical optimum, we generate 500 pairs of clean and noisy data points to train a kernel regression function in order to estimate the conditional expectation of the clean data given the noisy version. Another 500 noisy data points are then generated and fed to the kernel regression method and the output is used to train a vector quantizer using the LBG method. In the testing phase, another 500 noisy data points are generated and fed to the kernel regression estimator, and the output is quantized using the vector quantizer obtained in the training phase. Table 6.5 compares the mean square distortions resulting from the quantization of the estimated clean data using the kernel regression method and the output of the SCMS algorithm, respectively, as a function of the number of codevectors (ranging from 2 to 128). Although the SCMS algorithm does not have access to clean data, the simulation results indicate that the resulting mean square distortion is close to that achieved by the near-optimal scheme where the clean data estimates are obtained using the kernel regression method.

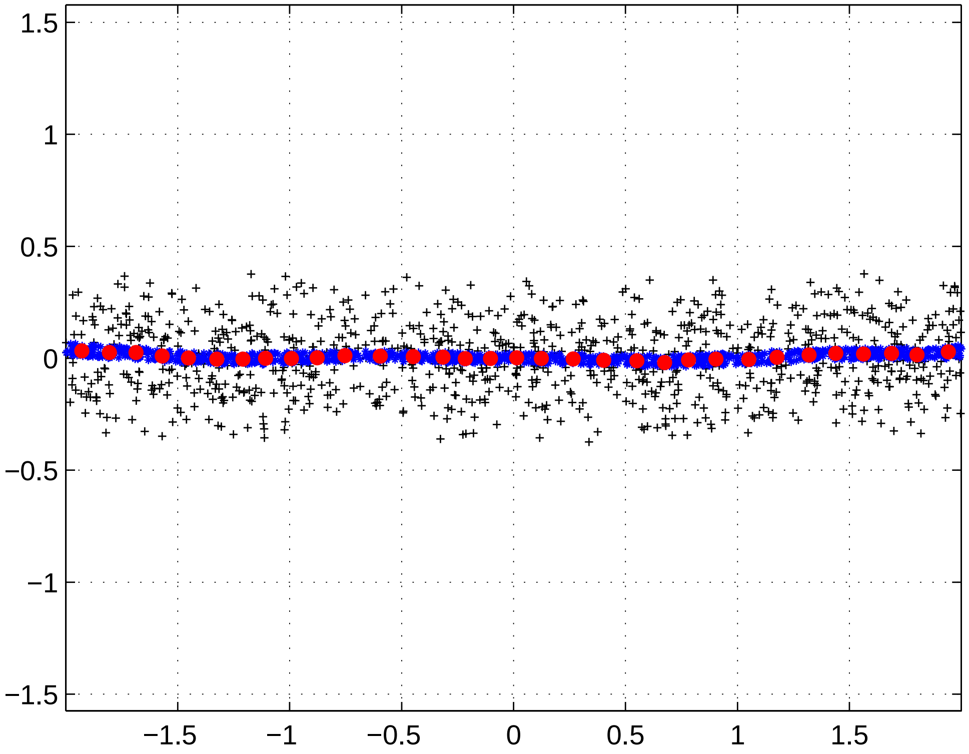

Figure 6.8: *Quantization of a noisy line.* The black points represent the noisy observations. The noisy data is fed to the SCMS algorithm, and the SCMS algorithm generates the blue points as an estimate of the clean data. The blue points are used for vector quantization using the LBG algorithm. The red points are the output of the LBG vector quantization algorithm.

6.2.6 Quantization of a noisy circle

The simulation setup is similar to that of a noisy line, but now we consider the uniform distribution on the unit circle as the clean source and the additive bivariate zero-mean Gaussian noise has per sample variance 0.3. For training and testing, two sets of 1024 noisy data

[1] Strictly speaking, this distortion is only near the theoretical optimum since the kernel estimate converges to the desired conditional expectation only in the limit of large training set sizes. Also the LBG algorithm is not guaranteed to produce globally optimal quantizers.

Table 6.5: *Quantization of a noisy line.* The mean square distortion resulting from the quantization of the output of the SCMS algorithm and the (near) optimal mean square distortion for different number of codebook sizes ranging from 2 to 128 for the noisy line.

Number of the codevectors	2	4	8	16	32	64	128
Optimal distortion [1]	0.4986	0.2507	0.1287	0.0639	0.0330	0.0172	0.0081
SCMS algorithm	0.5001	0.2683	0.1477	0.0731	0.0415	0.0271	0.0135

Table 6.6: *Quantization of a noisy circle.* The mean square distortion resulting from the quantization of the output of the SCMS algorithm and the (near) optimal mean square distortion for different number of codebook sizes ranging from 2 to 128 for the noisy circle.

Number of the codevectors	2	4	8	16	32	64	128
Optimal distortion [1]	0.7271	0.3858	0.2016	0.1064	0.0595	0.0367	0.0294
SCMS algorithm	0.7274	0.3945	0.2120	0.1220	0.071	0.0498	0.0379

points are generated for the SCMS approach and 1024 pairs of clean and noisy data points are generated for the kernel regression approach. Fig. 6.10 shows the performance of the SCMS algorithm to estimate the clean data on the circle. The black points represent the clean data, the blue points are the outputs of the SCMS algorithm, and the red points represent the codewords computed by the LBG algorithm. It can be observed in Fig. 6.10 that the codewords are nearly uniformly distributed. Fig. 6.11 shows the output of the SCMS algorithm and the computed codewords for simulations with quantizer codebook sizes 1, 2, 4, 8, 16, and 32. Table 6.6 compares the mean square distortions for the quantization of the SCMS estimates and that of the kernel regression method, respectively, as the number of the codevectors ranges from 2 to 128. The measurements indicate that the mean square distortion achieved by the quantization of the output of the SCMS algorithm is close to the near-optimum mean square distortion obtained by the quantization of the estimates using the kernel regression method.

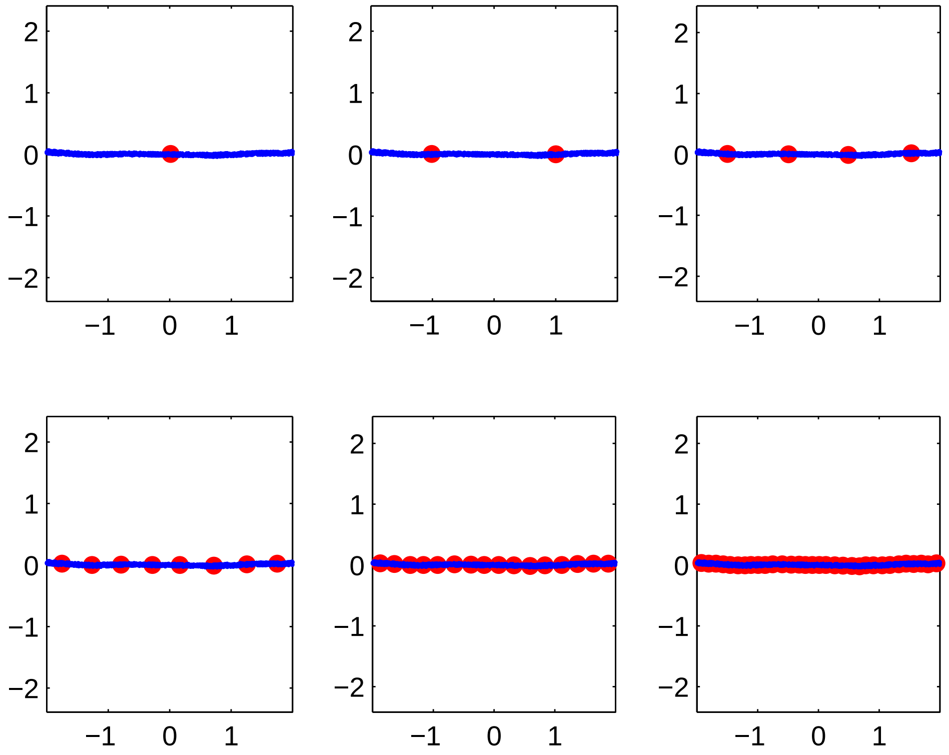

Figure 6.9: *Quantization of a noisy line.* The blue points represent the output of the SCMS algorithm applied to the points from the noisy line, and the red points are the codewords generated by the LBG vector quantization algorithm.

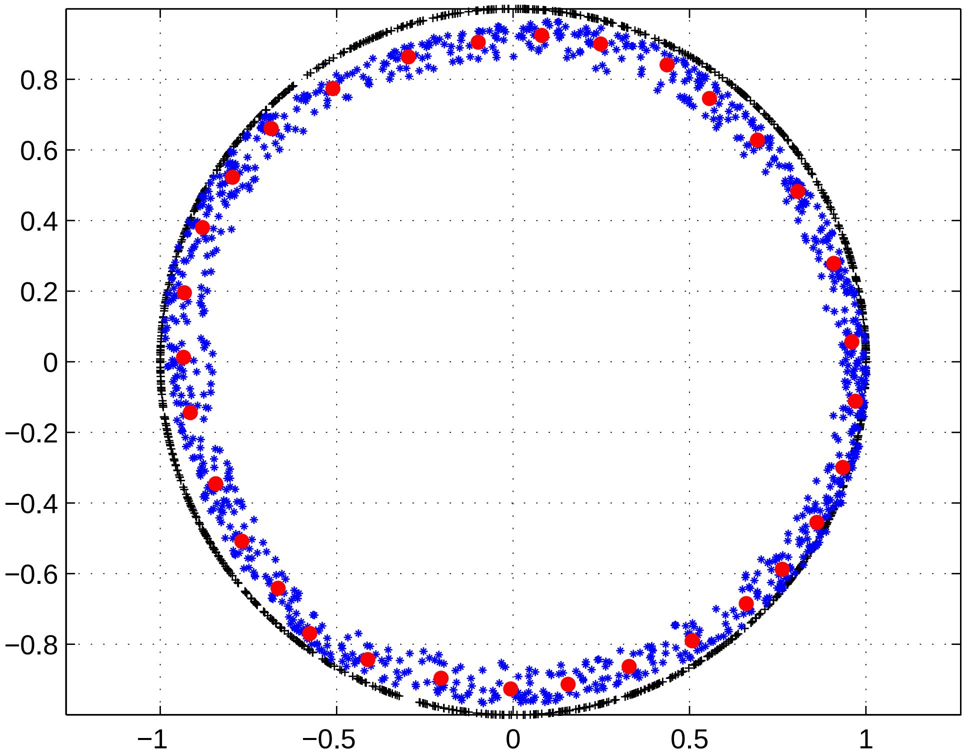

Figure 6.10: *Quantization of a noisy circle.* Applying the SCMS algorithm on the noisy data in order to estimate the clean data. The estimated clean data is used for the vector quantization. The black points represent the clean data, the blue points represent the output of the SCMS algorithm, and the red points are the codewords generated by the LBG vector quantization algorithm.

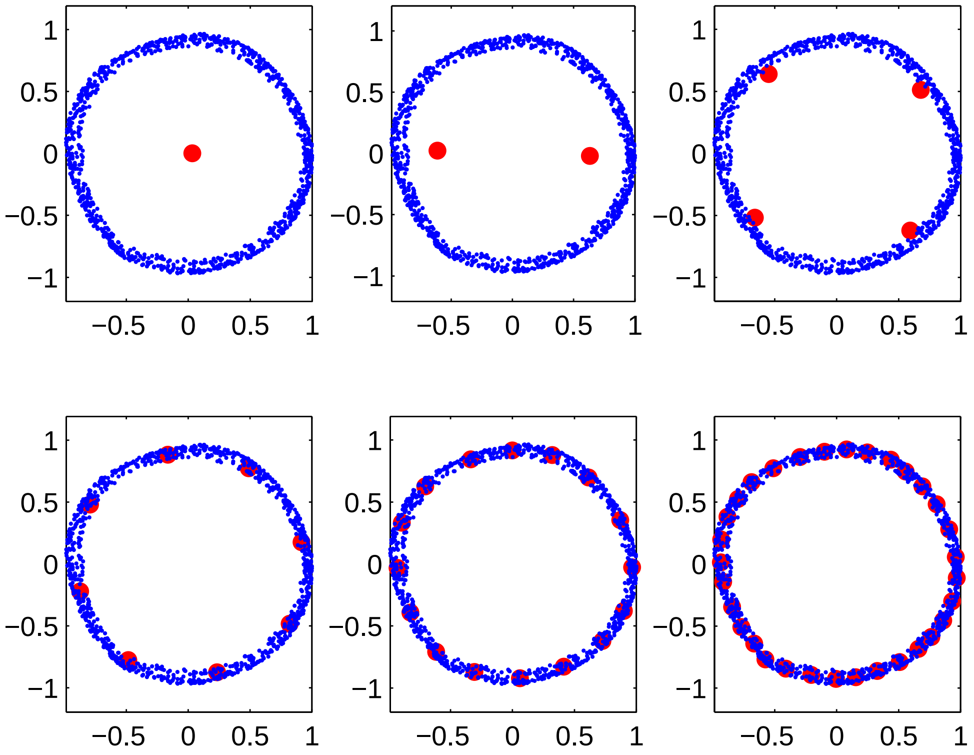

Figure 6.11: *Quantization of a noisy circle.* The blue points represent the output of the SCMS algorithm applied to the points from the noisy circle, and the red points are the codewords generated by the LBG vector quantization algorithm.

6.3 Character Skeletonization

6.3.1 Introduction

Skeletonization, also called thinning or medial axis transformation [4], is the process of extracting a region based shape to represent the general form of an object in two or three-dimensional space. In other words, the skeletonization process tries to find a medial representation of a digital object that is equidistant to its boundaries. The skeletonized object, called the skeleton, is a collection of thin arcs and curves that requires less pixels (voxels in three-dimensional space) to be represented, while the underlying shape can still be recognized with the human's perception. Skeletonization is an important preprocessing technique that has been used in numerous applications in machine learning [45], image segmentation [129], statistical pattern recognition [3], and data compression [51].

The concept of skeletonization was introduced by Blum [8], who used an intuitive model of fire propagation on a grass field, where the field has the form of the given shape. Intuitively, if one sets fire at all points on the shape's contour, assuming the fire is propagating within the shape at a uniform speed, then a point is on the medial axis if two or more firefronts meet at that point. Based on this observation, Blum introduced medial axis transform (MAT) to find the skeleton of a shape. The MAT computes the closest boundary points for each point in an object. An inner point is on the skeleton if it has at least two closest boundary points. Fig. 6.12 shows skeletonization of a rectangle based on the definition given by Blum [97]. Since Blum's work, different definitions and techniques for the skeletonization of an object have been proposed (for example, see [112][105][72] among others).

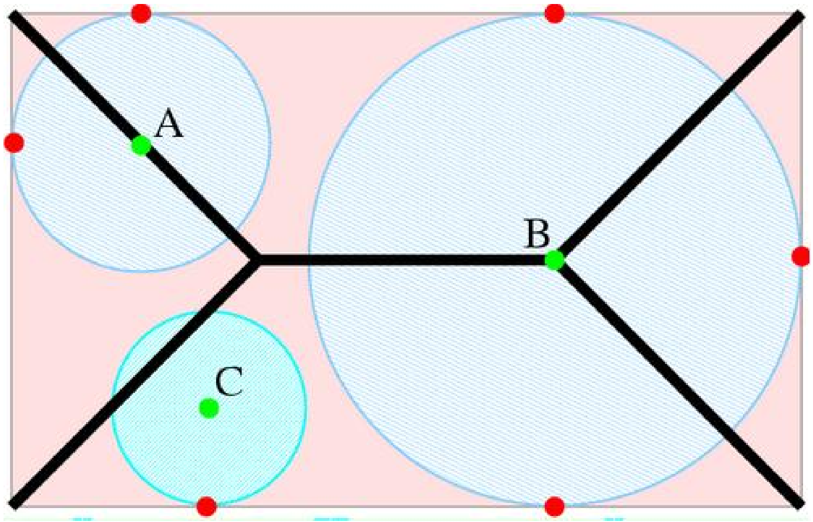

Figure 6.12: The green points are three randomly selected inner points, and the red points represent the closest boundary points. The skeleton is marked by thick black line segments. From the definition given by Blum, both points A and B are skeletal since they have two closest points on the boundary. But point C does not belong to the skeleton [97].

Skeletonization was initially used to find medial representation of two-dimensional digital objects. In such instances, the pixels on the border of the object are transformed to background pixels until a medial representation of the object is obtained. Later, skeletonization was extended to three-dimensional objects. Similar to the approach used with two-dimensional objects, the object voxels on the border are considered as background voxels until a skeletonized representation of the object is obtained [4]. Skeletons represent the underlying structure of digital objects and provide knowledge about how components are connected together to form the whole object [65]. In this section, we are dealing with 2-dimensional skeletonization and propose a weighted version of the SCMS algorithm to find the medial axis for objects in digital images. We test the proposed algorithm on the skeletonizaion of hand-written numbers, which is an important problem in image processing.

6.3.2 Weighted SCMS algorithm

A practical application of skeletonizaion is the reduction of computational complexity in character recognition. It is found that the recognition of the medial representation of a character requires less processing time compared to processing the raw image [72][30]. For example ZIP codes on envelopes from U.S. postal mail have five digits. Each ZIP code is segmented into five digits. The segmented small images are 16×16 gray scale images such that each pixel takes an integer value from 0 to 255. The task is to recognize each digit from a 16×16 matrix of pixel intensities in order to find the address automatically [58]. A neural network, as a nonlinear classification technique, can receive each digital image and predict the digit [77][99]. Feeding a neural network with a 16^2-dimensional vector greatly increases the computational cost. On the other hand, finding a medial representation of each digit and using it to train the neural network significantly reduces the processing time.

In this section we slightly modify the SCMS algorithm in order to use it effectively for skeletonization. Specifically, we define a weight factor for each term in the summation of kernel functions at each iteration. The weighted SCMS algorithm is based on the following three observations in gray scale images

- Pixels at the background make no contribution to finding the medial axis.
- Pixels at the edge of an object with low intensity make limited contribution to finding the medial axis.
- Pixels at the center of an object with high intensity play the main role of finding the medial axis.

Based on the above three observations, we assign a weight $0 \leq w \leq 1$ to each pixel. The weight w can be simply computed by dividing each pixel's intensity by the highest available intensity in the image. Let $i(p)$ represent the intensity of an arbitrary pixel p; then the weighted SCMS algorithm can be summarized as follows

1. Consider an $m \times n$ gray scale image I as a two-dimensional surface and assign a two-dimensional vector $\boldsymbol{x}_k, k = 1, 2, \ldots, mn$ to each pixel $p_{ij}, i = 1, \ldots, m,\ j = 1, \ldots, n$. The two-dimensional vector $\boldsymbol{x}_k$ consists of the x and y coordinates of each pixel.

2. Set $\epsilon > 0$, $j = 1$, and initialize the SCMS algorithm to an arbitrary point $\boldsymbol{y}_1$.

3. Compute i_{max} as the highest intensity among all the pixels, i.e., $i_{max} = \max_{i \in I} i(p)$.

4. Assign a weight w to each pixel p by $w = i(p)/i_{max}$.

5. Evaluate the weighted mean shift vector as follows

$$\boldsymbol{m}_{h,g}(\boldsymbol{y}_j) = \frac{\sum_{k=1}^{mn} w_k \boldsymbol{x}_k g\big(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_k}{h}\|^2\big)}{\sum_{k=1}^{nm} g\big(\|\frac{\boldsymbol{y}_j - \boldsymbol{x}_k}{h}\|^2\big)} \tag{6.4}$$

6. Evaluate the gradient, the Hessian matrix, and the local inverse covariance matrix $\hat{\boldsymbol{\Sigma}}^{-1}$ given in (3.7) at $\boldsymbol{y}_j$. Perform the eigendecomposition of $\hat{\boldsymbol{\Sigma}}_j^{-1} = \hat{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{y}_j)$ and find its eigenvalues and eigenvectors.

7. Let $\boldsymbol{V}_j = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{D-d}]$ be the $D \times (D - d)$ matrix whose columns are the $D - d$ orthonormal eigenvectors corresponding to the $D - d$ largest eigenvalues of $\hat{\boldsymbol{\Sigma}}_j^{-1}$.

8. Compute $\boldsymbol{y}_{j+1} = \boldsymbol{V}_j \boldsymbol{V}_j^T \boldsymbol{m}(\boldsymbol{y}_j) + \boldsymbol{y}_j$.

9. Stop if $\|\boldsymbol{y}_{j+1} - \boldsymbol{y}_j\| < \epsilon$; otherwise increment j by 1 and go to step 5.

Note that instead of the local inverse covariance matrix in step (6), we can use the three new matrices introduced in 4.7.

There are several other approaches for skeletonization (see [7][24][112][79][72] among others). The main advantage of the proposed method is its easy and straightforward implementation for real world applications. In contrast to the most other techniques, it does not require any pre or post processing in order to improve the output. The only parameter that needs to be set in advance is the bandwidth h. Furthermore, the smoothness of the generated curves is inherited from the smoothness of the underlying pdf or its estimate.

6.3.3 Simulation results

For the simulations in this section, we used the MNIST handwritten digits database [81][1]. The MNIST database contains real world data and has been extensively used to test the performance of different techniques and algorithms. Each digit is centered in a 28×28 gray scale image. Fig. 6.13 shows a sample image for each digit from the MNIST data base. We arbitrarily chose 5 samples for each handwritten digit and applied the weighted SCMS algorithm to find the medial axis. Fig. 6.14 shows the selected digits and the output of the weighted SCMS algorithm for digits 0, 1, 2, 3, and 4. Fig. 6.15 shows the performance of the proposed algorithm to find the medial axis for digits 5 to 9. In all the simulations the kernel function is Gaussian with the bandwidth $h = 1$ and the stopping criteria is $\epsilon = 0.05$.

[1] The MNIST database can be accessed for free using http://yann.lecun.com/exdb/mnist

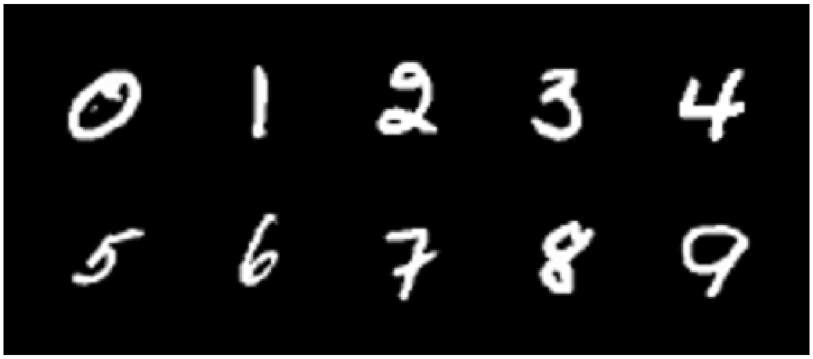

Figure 6.13: Sample images for different digits from the MNIST database.

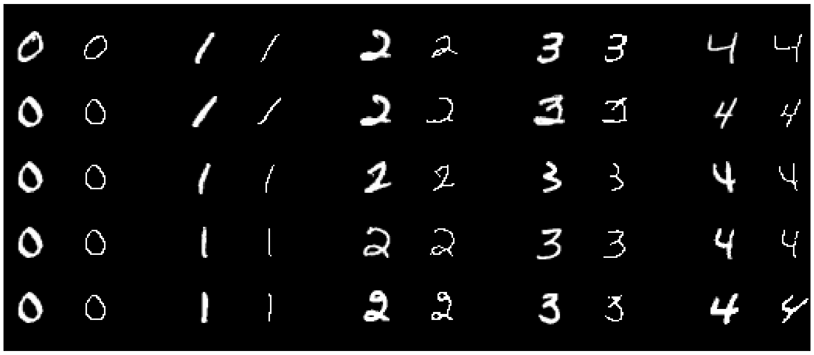

Figure 6.14: Skeletonization of handwritten digits using the weighted SCMS algorithm- numbers 0, 1, 2, 3, and 4.

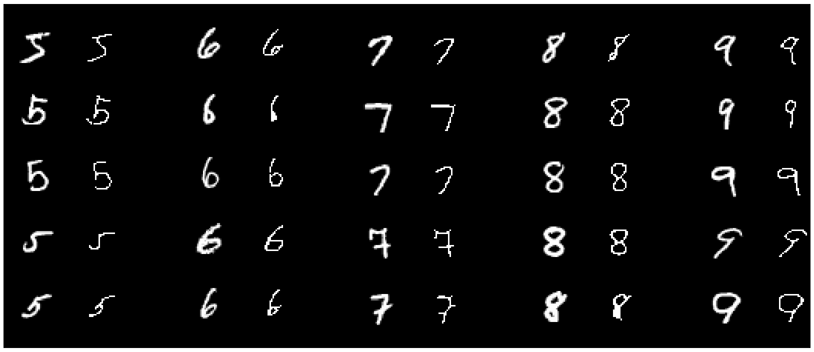

Figure 6.15: Skeletonization of handwritten digits using the weighted SCMS algorithm- numbers 5, 6, 7, 8, and 9.

6.4 Noisy kernel regression

Let $\boldsymbol{X} \in \mathbb{R}^D$ denote a real valued random variable and $Y \in \mathbb{R}$ denote a real valued random output variable, with joint density $f(\boldsymbol{X}, Y)$. We are interested in finding a relation between $\boldsymbol{X}$ and Y to predict Y given the values of $\boldsymbol{X}$. In this case, the random variable $\boldsymbol{X}$ is called the explanatory variable and Y is called the response. Suppose the relationship between $\boldsymbol{X}$ and Y can be modeled by $Y = m(\boldsymbol{X}) + \epsilon$, where ϵ is a zero mean random variable with variance σ^2. The expected prediction error is defined by [58]

$$\begin{aligned} C = E(Y - m(\boldsymbol{X})^2) = \int (y - m(\boldsymbol{x}))^2 f(\boldsymbol{x}, y) d\boldsymbol{x} dy \\ E(E[(Y - m(\boldsymbol{X}))^2 | \boldsymbol{X}]). \end{aligned} \tag{6.5}$$

It is well-known that the choice $m(\boldsymbol{x}) = E(Y|\boldsymbol{X} = \boldsymbol{x})$ minimizes C. Therefore, the best prediction of Y at any point $\boldsymbol{X} = \boldsymbol{x}$ is given by the conditional expectation of Y given $\boldsymbol{X}$, which is called the regression function [58].

Now suppose that pairs of input data $(\boldsymbol{x}_i, y_i), i = 1, \ldots, n$ are given, where y_i is the response for the ith observation $\boldsymbol{x}_i$. Since the density functions are not available, the conditional expectation $m(\boldsymbol{x})$ can be estimated by replacing the density functions by the kernel density estimates as follows [119][55]

$$\begin{aligned} m(\boldsymbol{x}) \approx \hat{m}(\boldsymbol{x}) &= \int y \frac{\sum_{i=1}^n \frac{1}{nh^{D+1}} K_1(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2) K_2(\|\frac{y-y_i}{h}\|^2)}{\sum_{i=1}^n \frac{1}{nh^D} K_1(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2)} dy \\ &= \frac{\sum_{i=1}^n K_1(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2) \int y \frac{1}{h} K_2(\|\frac{y-y_i}{h}\|^2) dy}{\sum_{i=1}^n K_1(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2)} \\ &= \frac{\sum_{i=1}^n y_i K_1(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2)}{\sum_{i=1}^n K_1(\|\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\|^2)}, \end{aligned} \tag{6.6}$$

where h is the bandwidth and K_1 and K_2 are the kernel functions that are used for density estimates. The above estimate is called Nadaraya-Watson kernel regression [92].

In practice, there are situations where only a few elements of the explanatory variable $\boldsymbol{X}$ are strongly related to Y and the rest of the elements do not have a significant effect on the response. In other words, for an explanatory variable $\boldsymbol{x} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_D]$, some $\boldsymbol{x}_i$s have no actual structural effect on the response and simply serve as noise which masks or weakens those elements that do have real explanatory value. Correctly removing the extraneus elements reduces the dimensionality of the explanatory variables and the computational cost. Therefore, it is often desirable to determine a small number of the elements that exhibit the strongest relationship to the responses. The process of selecting a subset of relevant features to use in model construction is called feature selection or variable selection. When the unwanted predictors are allowed to remain in the model, the predictive accuracy of the regression model suffers because we are in part fitting a relationship between the response and noise, which masks the predictive power of the predictors that actually have predictive power. On the other hand, it may be that the mean response has a functional relationship to all the predictors, but the predictor vectors are all on or near an underlying smooth manifold. For example, suppose there are ten predictors but the ten-dimensional predictor vectors all fall on some circle. Then we can replace the ten-dimensional predictor vectors with a two-dimensional representation, or just two “features”.

Our goal in applying a constrained mean shift algorithm to the high dimensional predictor vectors is to find a principal surface of suitably low dimension d that is close to the original predictors in some metric, for example in an L2 sense. The output of the constrained mean shift algorithm will be the projections of the original predictor vectors onto this principal surface. However, the output points are still vectors of the same dimension as the original input vectors. We thus will next need to find a d-dimensional representation of the projected points. These would be the d candidate features selected in our approach, and we replace the original regression model that had D predictors with a regression model that had the d selected features as the predictors. In our simulations, for simplicity we assume that the intrinsic dimensionality of the observed data is known in advance. The intrinsic dimensionality estimation is still an active research area and several different techniques are proposed to estimate it [82][100][71].

Furthermore, during the measurement or transmitting the data, the explanatory variables may corrupted by noise. In this case using the kernel regression formula in (6.6) with noisy explanatory variable may not generate an accurate estimate of the response. Applying the SCMS algorithm as a pre-processing step can be considered as a denoising step.

Let d denote the intrinsic dimensionality of the D-dimensional noisy observed data. The proposed technique for the kernel regression with noisy explanatory variables is as follows

- Apply the SCMS algorithm on the noisy explanatory variables (the projection step is done using $D - d$ appropriate eigenvectors).
- Apply one of the nonlinear dimensionality reduction techniques to find a d-dimensional representation of the output of the SCMS algorithm.
- Use the kernel regression technique in (6.6) to find the relation between the response and the explanatory variables.

6.4.1 Simulation results

We assume that 1000 samples of four-dimensional data, $\boldsymbol{x} = [x_1, x_2, x_3, x_4]$, as explanatory variables are available. The response y is related to $\boldsymbol{x}$ by $y = x_1^2 + x_2^3$. The first two elements of $\boldsymbol{x}$ are selected uniformly from a two-dimensional spiral and then corrupted by adding independent Gaussian noise with independent components having zero mean and variance one. The last two elements of each explanatory variable are just independent zero mean Gaussian with variance four. The bandwidth h for the SCMS algorithm is 2 and the stopping threshold is set to be 0.05. The output of the SCMS algorithm is given to the LLE algorithm in order to find the one-dimensional representation of the observed data. The number of the nearest neighbors in LLE algorithm is set to be 24. The kernel regression function in (6.6) is trained using the one-dimensional data and the response y with the bandwidth equal to 0.4. To show the effectiveness of the SCMS algorithm, we repeat the same procedure for the clean and noisy explanatory variable without applying the SCMS algorithm. Fig. 6.16 shows the kernel regression function trained using three above scenarios. It is clear from fig. 6.16 that applying the SCMS algorithm before LLE algorithm and kernel regression helps to find an accurate estimate of the response.

We repeat the simulation with a previous setup and this time assume that $y = x_1^2 + x_2$. The bandwidth h and the stopping threshold are chosen as before. Fig. 6.17 compares the kernel regression function computed under three scenarios. It can be observed from Fig. 6.17 that we obtain a more accurate kernel regression function by applying the SCMS algorithm to the noisy explanatory variables.

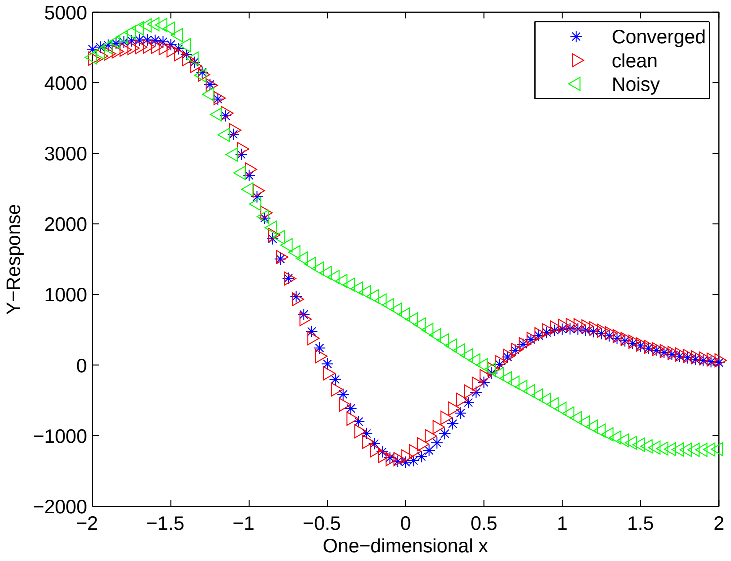

Figure 6.16: The blue stars represent the kernel regression function when it is trained using the output of the SCMS algorithm. The red triangles represent the kernel regression function when it is trained using the clean data and green triangles show the kernel regression function when it is trained using the noisy data. The response y is given by $y = x_1^2 + x_2^3$.

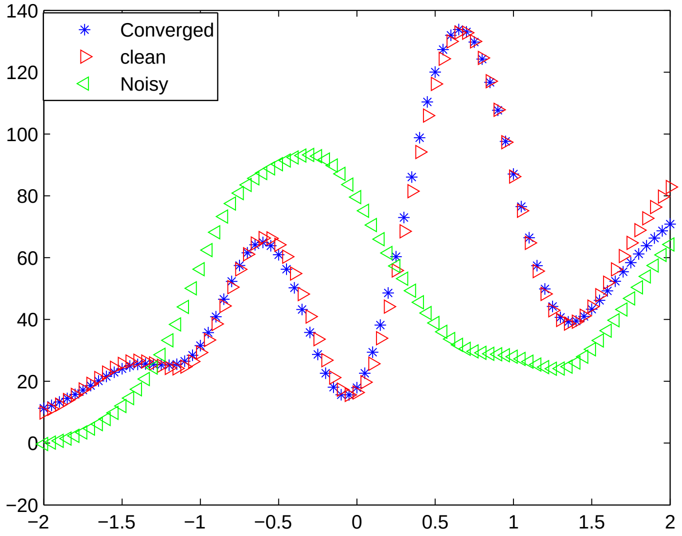

Figure 6.17: The blue stars represent the kernel regression function when it is trained using the output of the SCMS algorithm. The red triangles represent the kernel regression function when it is trained using the clean data and green triangles show the kernel regression function when it is trained using the noisy data. The response y is given by $y = x_1^2 + x_2$.

Chapter 7

Summary and future work

7.1 Summary

In this thesis, we studied the theoretical properties of some mean shift type algorithms. We also proposed some new applications for the MS and SCMS algorithms. The contributions of this thesis are summarized as follows.

- We proved that the MS algorithm with isolated stationary points generates a convergent sequence. We also provided a sufficient condition for the MS algorithm with the Gaussian kernel to have isolated stationary points. We also studied special one-dimensional case and showed that in this case the MS algorithm generates a monotone and convergent sequence with both analytic and non-analytic kernels. Furthermore, we proposed a slightly modified version of the MS algorithm in order to guarantee the convergence of the generated mode estimate sequence.
- We studied the SCMS algorithm in order to find principal curves and proved some

convergence results indicating that it inherits some of the important convergence properties of the MS algorithm. Specifically, we proved the monotonicity and convergence of the density estimate along the sequence generated by the SCMS algorithm. Then, we showed that the distance between consecutive points of the output sequence converges to zero, as does the projection of the gradient vector onto the subspace spanned by the $D - d$ largest eigenvectors of the local covariance matrix. The last two properties provide theoretical guarantees for the stopping criteria.

- We proposed three variations of the SCMS algorithm by modifying the projection step. Through the simulation we showed that with a finite data set, two of the proposed variations reduce the running time significantly.

- We proposed an adaptive version of the SCMS algorithm for situations in which the whole data set is not available in advance. In this case, the proposed SCMS algorithm observes the input data one by one and modifies the output sequence based on the new incoming data. In other words, the proposed algorithm considers the effect of new samples and makes necessary changes on output without running the algorithm on the whole data set.

- We used the SCMS algorithm as a pre-processing step before two well-known nonlinear dimensionality reduction techniques, ISOMAP and LLE, in order to improve the performance of these techniques for finding the low-dimensional representation of data in the presence of noise.

- We showed that the MS algorithm can be used to accurately find straight lines in digital images. We compared the performance of the proposed technique for finding the straight lines in a digital image with the Hough transform.

- We investigated the application of the SCMS algorithm to the problem of the noisy source vector quantization, where the clean source need to be estimated from its noisy observations before quantizing with an optimal vector quantizer. We proposed to use the output of SCMS algorithm as an estimate for the conditional expectation of the clean data given the noisy data.

- We showed how the SCMS algorithm can be used to improve the performance of the kernel regression technique when the explanatory variables are corrupted by noise. We also showed that the SCMS algorithm can be applied to find the most effective parameters to predict the output function using the kernel regression technique.

- We proposed a weighted version of the SCMS algorithm and used it to find a medial representation of a digital object.

7.2 Future work

We showed that the MS algorithm with isolated stationary points generates a convergent sequence. The proposed sufficient condition to have isolated stationary points may not be useful in real world applications. Unfortunately, a general and useful condition that leads to a set of isolated stationary points of the estimated pdf for commonly used kernels (such as Gaussian kernel) still seems to be missing in the literature. Finding the number of modes of a pdf estimate using the Gaussian kernel is still an open problem and needs to be investigated.

Extensive simulation results on artificial data demonstrated the ability of the SCMS algorithm to approximate the underlying principal curve/surface. However, the convergence of the sequence generated by the SCMS algorithm has not been proved yet, let alone its convergence to a point on the principal curve/surface. Therefore, as the first step the convergence of the procedure needs to be shown. Then if the convergence of SCMS algorithm is proved, it must to be shown that output points lie on or near to the underlying manifold.

The study of the optimality of the SCMS algorithm (i.e., its convergence to a principal-curve/surface) seems to necessitate a more careful examination of the definition of locally defined principal curves and surfaces. In particular, it is likely that existence issues should be resolved and differential geometric properties studied before optimality issues can be addressed.

Bibliography

[1] S. Avidan. Ensemble tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(2):261–271, 2007.

[2] E. Ayanoglu. On optimal quantization of noisy sources. *IEEE Trans. Inform. Theory*, 36(6):1450–1452, Nov. 1990.

[3] X. Bai, X. Yang, D. Yu, and L. J. Latecki. Skeleton-based shape classification using path similarity. *International Journal of Pattern Recognition*, 22(2):733–746, 2008.

[4] G. S. Baja. Skeletonization of digital objects. In *11th Iberoamerican Congress in Pattern Recognition*, volume 24, pages 1–13, Cancun, Mexico, Nov. 2006.

[5] D. H. Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13:111–122, Sep. 1987.

[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] G. Biau and A. Fischer. Parameter selection for principal curves. *IEEE Trans. on Information Theory*, 58:1924–1939, 2012.

[8] H. Blum. *Models for the Perception of Speech and Visual Form*, chapter A transformation for extracting new descriptors of shape, pages 362–380. MIT Press, 1967.

[9] R. A. Boyles. On the convergence of the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 45:47–50, 1983.

[10] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8:679–698, Nov. 1986.

[11] M. A. Carreira-Perpiñán. Reconstruction of sequential data with probabilistic models and continuity constraints. In *Advances in Neural Information Processing System (NIPS)*, volume 12, pages 414–420, Denver, USA, 2000.

[12] M. A. Carreira-Perpiñán. *Continuous Latent Variable Models for Dimensionality Reduction and Sequential Data Reconstruction*. PhD thesis, University of Sheffield, Feb. 2001.

[13] M. A. Carreira-Perpiñán. Fast non-parametric clustering with Gaussian blurring mean shift. In *Proc. 23th Conference on Machine Learning*, Jun. 2006.

[14] M. A. Carreira-Perpiñán. Gaussian mean shift is an EM algorithm. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29:767–776, 2007.

[15] K. Y. Chang and J. Ghosh. A unified model for probabilistic principal surfaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23:22–41, 2001.

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

[17] K. J. Cios, W. Pedrycz, R. W. Swiniarski, and L. A. Kurgan. *Data Mining: A Knowledge Discovery Approach*. Springer, 2007.

[18] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[19] D. Comaniciu, V. Ramesh, and P. Meer. The variable bandwidth mean shift and data-driven scale selection. In *Proc. 8th Intl. Conf. Computer Vision*, pages 438–445, Princeton, USA, 2001.

[20] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(5):564–575, May 2003.

[21] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT press, 1990.

[22] A. E. Cowart, W. E. Snyder, and W. H. Ruedger. The detection of unresolved targets using the Hough transform. *Computer Vision, Graphics, and Image Processing*, 21:222–238, 1983.

[23] T. F. Cox and A. A. Cox. *Multidimensional Scaling*. Chapman and Hall, 2000.

[24] A. Datta and S. K. Parui. Skeletons from dot patterns: A neural network approach. *Pattern Recognition Letters*, 18(4):335–342, 1997.

[25] K. B. Datta. *Matrix and Linear Algebra*. Prentice-Hall of India, 2004.

[26] D. Deb, S. Hariharan, U. M. Rao, and C. H. Ryu. Automatic detection and analysis of discontinuity geometry of rock mass from digital images. *Computers & Geosciences*, 34:115–126, 2008.

[27] P. Delicado. Principal curves and principal oriented points. Technical Report 309, Department dEconomia i Empresa, University Pompeu Fabra, 1998.

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

[29] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39:1–38, 1977.

[30] E. S. Deutsch. Preprocessing for character recognition. In *Proc. the IEE NPL Conference on Pattern Recognition*, pages 179–190, Teddington, UK, Jul. 1968.

[31] D. L. Donoho and C. Grimes. Hessian eigenmaps: New locally linear embedding techniques for high-dimensional data. Technical Report TR 2003-08, Department of Statistics, Stanford University, 2003.

[32] R. L. Dorbushin and B. S. Tsybakov. Information transmission with additional noise. *IEEE Trans. Inform. Theory*, 18:293–304, Sep. 1962.

[33] T. Duchamp and W. Stuetzle. The geometry of principal curves in the plane. Technical Report 250, Department of Statistics, University of Washington, 1993.

[34] T. Duchamp and W. Stuetzle. Geometric properties of principal curves in the plane. *Robust Statistics, Data Analysis, and Computer Intensive Methods: in honor of Peter Hubers 60th birthday (H. Rieder, ed.), Springer-Verlag,*, 109, 1995.

[35] R. O. Duda and P. E. Hart. Use of the Hough transformation to detect lines and curves in pictures. *Communication of the ACM*, 15:11–15, Jan. 1972.

[36] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2000.

[37] D. Eberly. *Ridges in Image and Data Analysis*. Kluwer, 1996.

[38] Y. Ephraim and R. M. Gray. A unified approach for encoding clean and noisy sources by means of waveform and autoregressive model vector quantization. *IEEE Trans. Inform. Theory*, 34(4):826–834, Jul. 1988.

[39] M. Fashing and C. Tomasi. Mean shift is a bound optimization. *IEEE Trans. on Patterns Analysis and Machine Intelligence*, 27(3):471–474, 2005.

[40] T. Fine. Optimum mean-square quantization of a noisy input. *IEEE Trans. Inform. Theory*, 11:293–294, Apr. 1965.

[41] R. W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(3):345, 1962.

[42] B. Flury. Principal points. *Biometrika*, 77:33–41, 1990.

[43] B. Flury. Estimation of principal points. *Journal of Applied Statistics*, 42:139–151, 1993.

[44] K. Fukunaga and L. D. Hostetler. Estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Trans. on Inform. Theory*, 21:32–40, 1975.

[45] X. Ge, I. I. Safa, M. Belkin, and Y. Wang. Data skeletonization via Reeb graphs. In *Advances in Neural Information Processing System (NIPS)*, volume 24, pages 837–845, Granada, Spain, 2011.

[46] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*. Springer, 1991.

[47] Y. A. Ghassabeh, T. Linder, and G. Takahara. On noisy source vector quantization via a subspace constrained mean shift algorithm. In *Proc. 26th Biennial Symp. on Communications*, pages 107–110, Kingston, Canada, 2012.

[48] Y. A. Ghassabeh, T. Linder, and G. Takahara. On the convergence and applications of mean shift type algorithms. In *Proc. 25th IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)*, pages 1–5, Montreal, Canada, 2012.

[49] 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.

[50] M. Golubitsky and V. Guilemin. *Stable Mapping and Their Singularities*. Springer-Verlag, 1973.

[51] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, Upper Saddle River, 2007.

[52] R. L. Gorsuch. *Factor Analysis*. Psychology Press, 1983.

[53] R. M. Gray, J. C. Kieffer, and Y. Linde. Locally optimum block quantizer design. *Information and Control*, 45:178–198, Jan. 1980.

[54] H. Guo, P. Guo, and Q. Liu. Mean shift-based edge detection for color image. In *Proc. International Conference on Neural Networks and Brain (ICNNB)*, pages 1118–1122, Beijing, China, Oct. 2005.

[55] L. Györfi, M. Kohler, M. Krzyzak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression*. Springer-Verlag, 2002.

[56] T. Hastie. *Principal Curves and Surfaces*. PhD thesis, Stanford University, Nov. 1984.

[57] T. Hastie and W. Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84:502–516, Jun. 1989.

[58] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2001.

[59] J. F. Hemdal. One-dimensional digital processing of images for straight line detection. *Pattern Recognition*, 31:1687–1690, 1998.

[60] R. A. Horn and C R. Johnson. *Matrix Analysis*. Cambridge University Press, 1990.

[61] P. V. C. Hough. Machine analysis of bubble chamber pictures. In *Proc. of Int. Conf. High Energy Accelerators and Instrumentation*, pages 554–558, Geneva, Switzerland, Sep. 1959.

[62] P. V. C. Hough. Method and means for recognizing complex patterns. US Patent 3069654, Dec. 1962.

[63] A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. New York: Wiley, 2001.

[64] J. Illingworth and J. Kittler. A survey of the Hough transform. *Computer Vision, Graphics, and Image Processing*, 44:87–116, 1988.

[65] A. B. Iraola. *Skeleton Based Visual Pattern Recognition: Applications to Tabletop Interaction*. PhD thesis, The University of the Basque Country Donostia, 2009.

[66] M. Verleysen J. A. Lee. *Nonlinear Dimensionality Reduction*. Springer-Verlag, 2007.

[67] J. E. Jackson. *VA User's Guide to Principal Components*. Wiley, 1991.

[68] C. Jian and Y. Jie. Real-time infrared object tracking based on mean shift. In *9th Iberoamerican Congress on Pattern Recognition, CIARP 2004*, volume 3287, pages 45–52, Puebla, Mexico, Oct. 2004.

[69] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 2002.

[70] B. Kégl. *Principal Curves: Learning, Design, and Applications*. PhD thesis, Concordia University, Dec. 1999.

[71] B. Kegl. Intrinsic dimension estimation using packing numbers. In *Advances in Neural Information Processing System (NIPS)*, volume 15, pages 681–688, Vancouver, British Columbia, Canada, 2002.

[72] B. Kégl and A. Krzyzak. Piecewise linear skeletonization using principal curves. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24:59–74, Jan. 2002.

[73] B. Kégl, A. Krzyzak, T. Linder, and K. Zeger. A polygonal line algorithm for constructing principal curves. In *Neural Information Processing Systems*, volume 11, pages 501–507, Denver, USA, Nov. 1998.

[74] B. Kégl, A. Krzyzak, T. Linder, and K. Zeger. Learning and design of principal curves. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:281–297, 2000.

[75] W. M. K. W. M. Khairosfaizal and A. J. Noraini. Eyes detection in facial images using circular hough transform. In *5th International Colloquium on Signal Processing and Its Applications, CSPA2009*, pages 238–242, Kuala Lumpur, Malaysia, Mar. 2009.

[76] C. Kimme, D. Ballard, and J. Sklansky. Finding circles by an array of accumulators. *Communication of the ACM*, 18:120–122, Feb. 1975.

[77] S. Knerr, L. Personnaz, and G. Dreyfus. Handwritten digit recognition by neural networks with single-layer training. *IEEE Trans. Neural Networks*, 16(3):962–968, Nov. 1992.

[78] D. Lagunovsky and S. Ablameyko. Straight-line-based primitive extraction in grey-scale object recognition. *Pattern Recognition Letters*, 20(10):1005–1014, 1999.

[79] L. Lam, S. W. Lee, and C. Y. Suen. Thinning methodologies-a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):869–885, 1992.

[80] M. LeBlanc and R. Tibshirani. Adaptive principal surfaces. *Journal of the American Statistical Association*, 89:53–64, 1994.

[81] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998.

[82] E. Levina and P. J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing System (NIPS)*, volume 17, Vancouver, British Columbia, Canada, 2004.

[83] J. Li, S. Ray, and B. G. Lindsay. A nonparametric statistical approach to clustering via mode identification. *Journal of Machine Learning Research*, 8:1687–1723, Aug. 2007.

[84] X. Li, Z. Hu, and F. Wu. A note on the convergence of the mean shift. *Pattern Recognition*, 40:1756–1762, 2007.

[85] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Trans. on Communications*, 28:702–710, Jan. 1980.

[86] T. Linder. *Principles of Nonparametric Learning*, chapter Learning-Theoretic Methods in Vector Quantization. Springer, 2002.

[87] T. Linder, G. Lugosi, and K. Zeger. Empirical quantizer design in the presence of source noise or channel noise. *IEEE Trans. Inform. Theory*, 43(2):612–623, Jul. 1997.

[88] S. P. Lloyd. Least squared quantization in PCM. *IEEE Trans. Inform. Theory*, 28(2):129–137, 1982.

[89] G. Matsaglia and G. P. H. Styan. Equalities and inequalities for ranks of matrices. *Linear and Multilinear Algebra*, 2(2):269–292, 1974.

[90] J. Max. Quantizing for minimum distortion. *IEEE Trans. Inform. Theory*, 6:7–12, Mar. 1960.

[91] G. McLachlan. *Discriminant Analysis and Statistical Pattern Recognition*. Wiley, 2004.

[92] E. A. Nadaraya. On estimating regression. *Theory of Probability and its Applications*, 9(1):141–142, Jul. 1964.

[93] T. Okuzono and H. Wakizako. Object detection using straight line matching in $\theta - \rho$ space. *IEEJ Trans. on Electronics, Information and Systems*, 128(6):600–606, 2008.

[94] A. Ostaszewski. *Advanced Mathematical Method*. Cambridge University Press, 1990.

[95] U. Ozertem. *Locally Defined Principal Curves and Surfaces*. PhD thesis, Oregon Health and Science University, Sep. 2008.

[96] U. Ozertem and D. Erdogmus. Locally defined principal curves and surfaces. *Journal of Machine Learning Research*, 12(4):1249–1286, 2011.

[97] K. Palágyi. Skeletonization. `http://www.inf.u-szeged.hu/~palagyi/skel/skel.html/`.

[98] E. Parzan. On estimation of probability density function and mode. *Annual of Mathematical Statistics*, 33:1065–1967, 1962.

[99] T. F. Pawlicki, D. S. Lee, J. J. Hull, and S. N. Srihari. Neural network models and their application to handwritten digit recognition. In *IEEE International Conference on Neural Networks*, pages 63–70, Jul. 1988.

[100] M. Raginsky and S. Lazebnik. Estimation of intrinsic dimensionality using high-rate vector quantization. In *Advances in Neural Information Processing System (NIPS)*, volume 18, Vancouver, British Columbia, Canada, 2005.

[101] S. Ray and B. G. Lindsay. The topography of multivariate normal mixtures. *The Annal of Statistics*, 33(5):2042–2065, 2005.

[102] D. Rebollo-Monedero, S. Rane, and B. Girod. Wyner-Ziv quantization and transform coding of noisy sources at high rates. In *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, pages 2084–2088, Stanford Univ., USA, Nov. 2004.

[103] R. Rosipal, M. Girolami, L. J. Trejo, and A. Cichocki. Kernel PCA for feature extraction and de-noising in nonlinear regression. *Neural Computing and Applications*, 10:231–243, 2001.

[104] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, Dec. 2000.

[105] K. Saeed, M. Tabedzki, M. Rybnik, and M. Adamski. K3m: a universal algorithm for image skeletonization and a review of thinning techniques. *Int. Journal of Applied Mathematics and Computer Science*, 20(2):317–335, Jun. 2010.

[106] D. J. Sakrison. Source encoding in the presence of random disturbance. *IEEE Trans. Inform. Theory*, 14:165–167, Jan. 1968.

[107] S. Sandilya and S. Kulkarni. Principal curves with bounded turn. *IEEE Trans. on Information Theory*, 48(417):2789–2793, 2002.

[108] L. K. Saul and S. T. Roweis. An introduction to locally linear embedding. Technical report, AT&T Labs and Gatsby Computational Neuroscience Unit, 2000.

[109] L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of nonlinear manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

[110] B. Schlkopf, A. Smola, and K. R. Muller. *Advances in Kernel Methods-Support Vector Learning*. MIT Press Cambridge, 1999.

[111] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, 1986.

[112] R. Singh, V. Cherkassky, and N. Papanikolopoulos. Self-organizing maps for the skeletonization of sparse shapes. *IEEE Transactions of Neural Networks*, 11(1):241–248, 2000.

[113] T. C. H. Tao. 254a, notes 3a: Eigenvalues and sums of hermitian matrices. `http://terrytao.wordpress.com/2010/01/12/254a-notes-3a-eigenvalues-and-sums-of-hermitian-matrices/`.

[114] T. Tarpey, L. Li, and B. Flury. Principal points and self-consistent points of elliptical distributions. *Annals of Statistics*, 23:103–112, 1995.

[115] J. B. Tenenbaum. Mapping a manifold of perceptual observations. In *Advances in Neural Information Processing Systems 10*, pages 682–688, MIT Press, 1998.

[116] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, Dec. 2000.

[117] R. Tibshirani. Principal curves revisited. *Statistics and Computation*, 2:183–190, 1992.

[118] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[119] A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer-Verlag, 2008.

[120] J. J. Verbeek, N. Vlassis, and B. Krose. A soft k-segment algorithm for principal curves. In *Proc. International Conference on Artificial Neural Networks*, pages 450–456, Vienna, Austria, 2001.

[121] M. P. Wand and M. Jones. *Kernel Smoothing*. Chapman and Hall, 1995.

[122] J. Wang, B. Thiesson, Y. Xu, and M. Cohen. Image and video segmentation by anisotropic kernel mean shift. In *Proc. European Conf. on Computer Vision*, pages 238–250, Prague, Czech Republic, 2004.

[123] R Wang. *Introduction to Orthogonal Transforms*. Cambridge University Press, 2012.

[124] W. Wang and M. A. Carreira-Perpiñán. Manifold blurring mean shift algorithms for manifold denoising. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2010)*, pages 1759–1766, San Francisco, USA, 2010.

[125] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semi-definite programming. *International Journal of Computer Vision*, 70:77–90, Dec. 2005.

[126] J. K. Wolf and J. Ziv. Transmission of noisy information to a noisy receiver with minimum distortion. *IEEE Trans. Inform. Theory*, 16:406–411, Jul. 1970.

[127] C. F. J. Wu. On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11:95–103, 1983.

[128] J. Xiao, Z. Zhou, D. Hu, J. Yin, and S. Chen. Self-organized locally linear embedding for nonlinear dimensionality reduction. *Advances in Natural Computation, Lecture Notes in Computer Science*, 3610:101–109, 2005.

[129] B. Yangel and D. Vetrov. Image segmentation with a shape prior based on simplified skeleton. In *8th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR 11)*, volume 6819, pages 247–260, Saint Petersburg, Russia, Jul. 2011.

[130] A. Yilmaz. Object tracking by asymmetric kernel mean shift with automated scale and orientation selection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 18–23, Minnesota, USA, 2007.

[131] H. Zhou, G. Schaefer, M. E. Celebi, and F. Minrui. Bayesian image segmentation with mean shift. In *Proc. 16th IEEE International Conference on Image Processing (ICIP)*, pages 2405–2408, Cairo, Egypt, Nov. 2009.

[132] Y. Zhu, R. He, N. Xiong, P. Shi, and Z. Zhang. Edge detection based on fast adaptive mean shift algorithm. In *Proc. International Con. on Computational Science and Engineering*, pages 1034–1039, Vancouver, Canada, Aug. 2009.