Manifold Learning and Unwrapping Using Density Ridges

A Dissertation Presented
by

Matineh Shaker

to

The Department of Electrical and Computer Engineering

in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

in

Electrical Engineering

Northeastern University
Boston, Massachusetts

September 2016
Dedicated to my mother, Maliheh, my father, Ali, my brother, Amir Hossein, my grandmothers and late grandfathers.
## Contents

List of Figures ................................................................. vi

Acknowledgments ............................................................ xi

Abstract ................................................................................. xii

1 Introduction ........................................................................ 1

1.1 Motivation: Using Density Ridges to Unwrap Manifolds ............. 3

1.2 Principal Curves and Density Ridges ........................................ 5

1.2.1 Orthogonal Local Principal Curves ..................................... 6

1.3 Relevant Topics from Differential Geometry .............................. 7

1.3.1 Properties of Density Ridges ............................................. 10

1.4 Outline of the Dissertation .................................................. 12

1.4.1 Contribution Statement .................................................. 13

2 Curvilinear Coordinates ....................................................... 14

2.1 Principal Curve Projection .................................................. 15

2.2 Curvilinear Coordinates ...................................................... 18

2.3 Diffeomorphic Projection Model ........................................... 19

2.3.1 Learning the Diffeomorphic Projection Model .................... 19

2.3.2 Projecting Out-of-Sample Test Data .................................. 21

2.4 Experiments and Results .................................................... 22

2.5 Discussion and Conclusions ............................................... 25

3 Natural Images Manifolds ..................................................... 28

3.1 MNIST handwritten digits .................................................. 28
### 3.1.1 Feature Space Visualization

3.2 Frey Faces

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Notation and Background</td>
<td>46</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Critical Sets</td>
<td>46</td>
</tr>
<tr>
<td>4.2</td>
<td>Curvilinear Coordinates</td>
<td>49</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Local Curvilinear Coordinates for Exponential PDFs</td>
<td>51</td>
</tr>
<tr>
<td>4.3</td>
<td>Manifold Unwrapping</td>
<td>52</td>
</tr>
<tr>
<td>4.4</td>
<td>Experiments</td>
<td>53</td>
</tr>
<tr>
<td>4.5</td>
<td>Discussion</td>
<td>55</td>
</tr>
</tbody>
</table>

### 5 Efficient Density Derivative Estimation

5.1 Introduction

5.1.1 Related work

5.2 Kernel Functions

5.2.1 Univariate Kernels

5.2.2 Separable Multivariate Kernels with derivatives

5.2.3 Scaled Multivariate Kernels

5.2.4 Kernel Densities

5.3 Efficient Implementation

5.3.1 Tree Construction Algorithm

5.3.2 Computational complexity

5.4 Experimental Results

5.4.1 Speedup on Synthetic Data

5.4.2 Speedup on Ridge Score

5.4.3 Speedup on Principal Curves Projection

5.4.4 Speedup as a Function of Number of Evaluation Points

5.5 Conclusion

5.6 Proof 1

5.7 Proof 2
6 Conclusion

6.0.1 Future Work

A Graph-Constrained Inverse Covariance Estimation: Application in Traumatic Brain Injury Detection

A.1 Gaussian Graphical Models

A.2 Application to Injury Detection in TBI

A.2.1 Subjects and Data Acquisition

A.2.2 a Priori Graph

A.2.3 Identifying an Abnormal Brain

A.2.4 Identifying Individual Abnormal Regions

A.3 Experiments

A.3.1 Node-only Versus Neighborhood Versus Fully-connected Graphs

A.3.2 Neighborhood Versus Random Prior Graphs

A.3.3 Selecting the Optimal Penalty Parameter $\lambda$

A.3.4 Investigating Other DTI Measures

A.3.5 Correlations with Behavioral Measures

A.3.6 Individual Abnormal Regions Identification

A.4 Discussion

Bibliography
List of Figures

1.1 Noisy one-dimensional manifold (light green) embedded in $\mathbb{R}^2$ and the density ridge estimate (dark blue). The small window shows a selected portion of the density ridge. We can see the corresponding local maxima of the gradient flow in the figure [1]. ........................................ 4

1.2 A manifold with a one-dimensional ridge/principal curve. The local basins of attraction can be decomposed using orthogonal principal curves. $M$ is the intrinsic manifold, $T_pM$ denotes the tangent space of $M$ at $p$. The green area denoted in the lower right sketch is the area of $T_pM$ constrained by the exponential map [1]. ........................................ 7

1.3 Illustration of the eigenvectors (red arrows) of the kernel density Hessian matrix at a point on the manifold $M$ (thick black line). A figurative illustration of the contours of the kernel centered at the same point is also shown to illustrate the bandwidth of the kernel [1]. ................. 10

1.4 (a) Nonlinear and nonconvex data set with density ridges shown in red(top) and blue(second); (b) A single data point projected to the two one-dimensional density ridges, trajectories shown in red and then along the gradient to the local mode(green). The density ridges as shown in Fig 1.4(a) are marked with small black dots [1]. ........................................ 12

2.1 Projection trajectories following Hessian eigenvector flow for two different choices of the strict monotonically increasing function on Gaussian pdf. ........................................ 16

2.2 Convergence of a given point on the Rosenbrock density function based on Equation (2.1); green: gradient, light blue: small eigen-vectors of Hessian, red: large eigen-vector of Hessian. The gradient and large eigenvector are being alligned over the iterations until stopping criteria is satisfied. ........................................ 17
2.3 Principal curve projection on swiss roll. (a), (b), (c): blue data points with projected red points. Columns show three orientations of same data. Gaussian kernel width is 5 and the bias in density estimate causes the mis-orientation. (d), (e), (f): same as above but the kernel width is too small for the data and the projections are noisy. (g), (h), (i): Color coded data points and projected black points using mean-shift projection method in [2], and with kernel width of 1,3 and 5. None of them show an accurate projection to the ridge of the density. Please note that this is not an example of 1-d unwrapping and it only presents projection to one of the ridges.

2.4 Density estimation and principal curves in crescent and two-moon data sets. In (c) the two moons are closer and aligned forming a circle-like shape.

2.5 Projection of ’crescent’ data set without reducing the dimension using benchmark manifold learning techniques, and the proposed principal curves (PC). Only PC unwrapping results in meaningful embeddings. The benchmark algorithms fail at providing a meaningful unwrapping of the data.

2.6 Projection of ’two-moon’ data set using benchmark manifold learning techniques, and the proposed principal curves (PC). Proposed method not only unwraps the manifold embedded in the data, but separates the two moons into clusters while unwrapping. Benchmark methods fail in unwrapping.

2.7 Projection of two moon data set while the moons are closer, still results in perfect unwrapping/clustering.

2.8 Out-of-sample unwrapping. Training points marked as dots, test points marked as crosses. Color coding is included to illustrate that the sign change along the diffeomorphism.

3.1 Left: Isotropic Gaussian kernel learned on PCA projection of number 1 in digits data (digits-1), reducing dimension of digits data set to 3. Maximum likelihood cross validation is used for chosing the kernel width hyperparameter.

3.2 The projection trajectories to the first and second surfaces according to Equation 2.1.

3.3 Three dimensional data of 6 of mnist digits in grey color and the corresponding first and second principal surface in blue and red. We clearly see that the data follows a nonlinear submanifold.

3.4 mnist-1 and mnist-7 unwrapped
3.5 (a) mnist-0 images on top of the 2d projection of first principal surface. The orientation constitute the main source of variation in the lower variance dimension. (b) mnist-0 on top of 2d projection of second principal surface. The main mode of variation here is the thickness across high variance dimension.  

3.6 Unwrapped mnist-0 digits with a selection of images of the actual digits on top. Orientation and thickness of digits smoothly vary across horizontal and vertical directions, respectively.  

3.7 Unfolding mnist-0 digits using two benchmark manifold learning methods. Unlike the proposed method, no smooth variation of structure is preserved using these methods.  

3.8 Unfolding mnist-2 digits using the proposed method and t-SNE. Similar to Figure 3.7(b) for mnist-0, no smooth variation of structure is preserved using t-SNE, while proposed method shows smooth change of thickness and shape in horizontal and vertical coordinates, respectively.  

3.9 Unfolding mnist-9 digits using the proposed method, which shows smooth change of thickness and orientation in horizontal and vertical coordinates, respectively.  

3.10 Unfolding mnist-9 digits using benchmark manifold learning methods. Similar to previous results, no smooth structure is captures.  

3.11 2D projection of one of principal surfaces of mnist digits 0,1,2. We can see separate clusters while orientation and thickness within clusters varies smoothly.  

3.12 Unwrapping of mnist digits 0,1,2. vertical axis shows the thickness variations, while horizontal axis value changes based based on clusters of digits (0 on the right, 1 on the left and 2 in the middle).  

3.13 Convolutional neural network used for classifying mnist digits in order to visualize feature space (layer before Softmax) to track the training process. Here is the order of layers: 5*5 convolutions to 32 channels followed by ReLU and 2*2 max-pooling, 5*5 convolutions to 64 channels followed by ReLU and 2*2 max-pooling, Two fully connected layers of 1024 and 10, and Softmax. Dropout is also used during training, to improve generalization.  

3.14 Visualization of features from test set of first 6 digits of mnist, evaluated at the last fully-convolutional layer before training.  

3.15 Features evaluated after 1 epoch of training.  

3.16 Features evaluated after 3 epoch of training.  

3.17 Features evaluated after 6 epoch of training. As training progresses to more epochs, the same digits are clustered together more strongly.
3.18 Frey faces on top of 2D projection using first and second PC projections. A random selection of actual face images shown on top of the unwrapped coordinates for illustration of structure.

4.1 The pdf $p(x) \propto e^{-\frac{1}{2}(x_1^2+(x_1-x_2)^2)}$ and its critical sets.

4.2 Three hump camel (THC) distribution with its critical sets and clusters.

4.3 Local and global unwrapping of Three hump camel. Top: blue, red, and green clusters as in 4.2, and the global unwrapping with respect to the main mode (red cluster) shown in black. Second row shows the local unwrapping with respect to each of the three clusters.

4.4 (a): Data drawn from the pdf in 4.1 and kernel density estimate using the unconstrained bandwidth matrix plug-in estimator [3]. (b) estimated pdf in the form of exponential of polynomial using polynomial regression of log of kernel density estimate. The zero level sets of delta is also shown.

4.5 Factorization of delta from log-polynomial estimation of pdf in distorted Gaussian.

4.6 An estimation result on delta factorization.

4.7 Kernel density estimate and log-polynomial estimation of data drawn from three hump camel pdf presented in Figure 4.2(a). The procedures are as described in Figure 4.4.

4.8 The unwrapping using factors of delta by both analytical and estimated pdfs. The actual pdf and estimated pdf show close results which points out the goodness of log-polynomial estimate of pdf on this data drawn from highly skewed multimodal pdf.

5.1 Full graph (left): Nodes violating the constraint are marked with bold red. Pruned graph (right): Nodes which meet the maximum derivative constraint are marked with bold blue.

5.2 Node-edge relationship in the tree.

5.3 Comparison of number of multiplications in computing first and second order derivatives of kernel density estimator of one single point in $\mathcal{N}(0, I_{d \times d})$ distribution, using the conventional direct method and the proposed efficient method with dimensions from 1 to 100. The curves show the experimental results, while the markers represent the corresponding theoretical results according to the Facts (3) and (4).

5.4 Comparing running times of the proposed method and the direct method on sample sizes of 100, 150, and 200. (a) dimension range of 1 to 100. (b) zoomed-in version of (a) showing the details of lower dimensions. The markers point to dimensions after which the proposed method is faster than the direct method. It can be seen that as the number of samples increases, the implementation of the proposed method acts superior to the direct method.
5.5 Calculation time for generating first and second derivatives of kernel from randomly drawn
$\mathcal{N}(0, I_{200 \times 200})$ and increasing number of data points, using both direct (green) and proposed
(blue) implementation indicating significant reduction in complexity and the rate of increase
for time versus number of points. .......................................................... 73

A.1 Illustration of the prior graph through 10 brain regions. The vector assigned to region 1 in
the adjacency matrix is [1110110000], emphasizing connections of regions 2, 3, 5, 6 and
disconnections of regions 4, 7, 8, 9, 10 to region 1. Bold lines in Figure A.1(b) show the
subgraph associated with region 1. .......................................................... 84

A.2 Abnormal Region detection based on the $\chi^2$ distribution and our greedy forward sorting of
regions. The dashed lines correspond to the 95th percentile of the CDF of the $\chi^2$ distribution.
Each of the colored curves represent, for each subject, the accumulated Mahalanobis distance
given by the subsets of sorted regions. Regions added after the point at which the subjects
Mahalanobis distance curve exceeds the 95% threshold on the CDF of $\chi^2$ distribution are
flagged as abnormal. .......................................................... 86

A.3 (a) 90% CIs of the AUC, as a function of the penalty parameter $\lambda$ for the prior graph, fully-
connected graph, and node-only graph; (b) 90% CIs of the BIC as a function of the penalty
parameter $\lambda$ for each model; (c) Number of model parameters for a given penalty weight for
prior and fully-connected graphs; (d) AUC as a function of the number of model parameters
for the prior and fully-connected graphs. Both models show similar AUC performance, but
the prior graph has better model complexity and BIC. ........................................ 88

A.4 Comparison of the neighborhood graph model performance (blue dashed line) with 1000
random graphs with the same number of edges. ........................................... 89

A.5 Selection of the optimum penalty parameter for each model based on minimizing leave-one-
out sum of squared Mahalanobis distances to the model mean. ......................... 90

A.6 Abnormality maps show which regions are affected in each subject for different graphical
models. Left: TBI subjects, Right: Normal Controls (NC). Top to Bottom: node-only graph,
neighborhood graph, full graph. .......................................................... 93

A.7 Abnormality maps based on z-scores for two different thresholds. Top: threshold is $|z| > 2$,
Bottom: $|z| > 3.58$ (Bonferroni correction for the number of regions) .............. 94
Acknowledgments

I would like to thank my advisor, Prof. Deniz Erdogmus for guiding and supporting me over the years. You have set an example of excellence as a researcher, mentor, and original thinker.

I would like to thank my thesis committee members Prof. Dana Brooks, Prof. Jeniffer Dy, and Prof. Sylvain Bouix, who were also my mentors, for all of their guidance through this process; your discussion, ideas, and feedback have been absolutely invaluable.

I would like to thank my collaborators Jonas N. Myhre and Devrim Kaba who contributed greatly to this research. I am very grateful to both of you.

I would like to thank my fellow graduate students at BSPIRAL, Marzieh Haghighi, Jamshid Sourati, Mohammad Moghaddamfalahi, Hooman Nezamfar, Asieh Ahani, Athena Nouhi, Nastaran Ghadar, Seyhmust Guler, Yeganeh Marghi, Jaume Coll-Font, Setareh Ariaifar, Sadegh Salehi, Esra Cansizoglu, and Paula Gonzalez for their constant enthusiasm, encouragement, and supportive feedbacks.

I would like to thank my amazing family for the love, support, and constant encouragement I have gotten over the years. In particular, I would like to thank my mother who was my best teacher as I grew up, my father for his warm support, and my dear brother for the motivation he brings to my life. I undoubtedly could not have done this without you.

I would like to dedicate this dissertation to my parents, my brother, two grandmothers and my late grandfather who left us not long ago.

I would specially like to thank my greater Boston family, Zanyar Golabi, Mohammad Araghchini and Nafiseh Masoumi. Your love, encouragement and laughter have kept me smiling and inspired. You are and always will be my family.

Finally, I would like to thank my fellow band members at Sayeh Ensemble, Hamed Tabkhi, Maryam Vaziri, and Pouya Shabanpour for all the amazing time and long hours we spent together to create, practice, and perform music. This was one of the most special experiences in my life and I am very grateful to all of you.
Abstract

Manifold Learning and Unwrapping Using Density Ridges

by

Matineh Shaker

Doctor of Philosophy in Electrical Engineering
Northeastern University, September 2016
Dr. Deniz Erdogmus, Adviser

Manifold learning is used for determining a coordinate system for high dimensional data on its intrinsic low-dimensional manifold, in order to (approximately) unwrap the manifold to an isomorphic Euclidean space, or to reduce data dimensionality. Defining a global or local manifold on data also provides a base for data classification, clustering, and visualization. Research on manifold learning within a density ridge estimation framework has shown great potential in recent work for both estimation and denoising of manifolds, building on the intuitive and well-defined notion of principal curves and surfaces. However, the problem of unwrapping or unfolding manifolds has received relatively little attention within the density ridge approach, despite being an integral part of manifold learning in general. We seek natural curvilinear coordinate systems along the manifold, coordinate frames that are geometrically meaningful based on the data distribution. In this dissertation, two approaches are presented towards this end.

In the first approach, we define the ridges of the (at least twice continuously differentiable) probability density function explicitly based on its gradient and Hessian, and in practice, demonstrate results using a kernel density estimate approximation based on samples drawn from this density. Specifically, a given point is projected to the ridges of the probability density by solving differential equations that follow ascent trajectories whose tangent vectors are gradient vector projected to subspaces spanned by all eigenvectors of the Hessian except one (the one that will become the tangent vector of the ridge to which the point is projected). Once the original point is projected to each ridge by designating each eigenvector as the “tangent” once, the
ridge projections are used as initial conditions to gradient-ascent differential equations. The curvlength of these latter trajectories, combined with information regarding direction of arrival to the mode, are used to construct a curvilinear coordinate vector for the original point with respect to the mode it climbs to in this process. A natural consequence of this process is that each point gets a curvilinear coordinate relative to the mode that contains the original point in its attraction basin (according to gradient-ascent), hence mode-based clusters are naturally endowed with local charts with curvilinear coordinate systems. We provide experimental results with both synthetic and real data sets, such as MNIST handwritten digits and Frey Faces. Manifold traversing for images of digits gives an approximation on the underlying smooth manifold representing geometric variations like orientation and thickness of digits in the two leading unwrapped dimensions. Results on synthetic datasets, such as crescent and two-moon, showed that the proposed curvilinear coordinates capture the underlying manifold of a mode more successfully compared to benchmark data-oriented methods available in the literature, illustrating our assertion that the ridges of the distribution and the local maximum point at the intersection of these curves, form a natural geometric skeleton for the curvilinear structure of the data distribution in the vicinity of this mode.

The second approach aims to avoid solving differential equations that involve eigendecomposition of Hessians along every point in the associated trajectories. Instead, we recognize that an alternative characterization of critical surfaces and curves (which include ridge-, cliffside-, and valley-paths) is possible using the rank of a matrix sequence that has i nits columns powers of the Hessian multiplying the gradient at a point. The so-called Omega matrix sequence, then as a set of zero level-sets for their respective determinants points to d-dimensional critical sets. Approximating the determinant with a polynomial expansion yields a polynomial curvilinear coordinate system based on the factorization of the matrix determinant. For an exponential density model with polynomial bases in the exponent, the Omega matrix entries and its determinant become polynomials, in which case, the manifold unwrapping problem is reduced to one of polynomial factorization. We provide unwrapping results based on known and estimated probability density functions in the exponential family with polynomial bases. Experiments on synthetic and real data sets show that this procedure generates similar results to the ridge tracing method, but with polynomial warping of coordinates, as expected.
Chapter 1

Introduction

High dimensional data are present in many areas of machine learning, signal processing, computer vision, bioinformatics, language processing, etc. Any process in nature which progresses as a result of many different variables, measurable or not, will potentially generate high dimensional data. Images containing millions of pixels, videos with millions of frames, documents with thousands of features, clinical data of patients with hundreds or thousands of indicators, and DNA data containing thousands of genes, are all examples of high dimensional data we are confronted with.

The high dimensionality of data increases the computational time and memory requirements of storing data, as well as covering the useful information in thousands of dimensions of noise. In many real world applications, high dimensional data distributions often exhibit clear low-dimensional underlying structures, sometimes referred to as the manifold. In addition, the nonlinear intrinsic structure prevents the use of fast linear machine learning techniques. In general such underlying low-dimensional surfaces may have complicated shapes and may have to be defined locally at best.

In practice manifold learning is often posed as either learning coordinates that describe the intrinsic manifold or simply as unwrapping, stretching or unfolding the manifold such that it can, to some extent, be treated as a linear subspace in \( \mathbb{R}^D \).

Most algorithms in manifold learning rely on the so-called manifold assumption [4]. The assumption is that the data cloud in a vector space lies on or close to a hypersurface of lower intrinsic dimension. Doing inference on or along this hypersurface would enable the use of simple linear methods which are fast and theoretically well-defined. A much used example to illustrate these concepts is the so-called swiss roll [5]. It consists of a 2-D plane embedded in 3-D and folded into a swiss roll shape. Calculating distances along this shape using standard Euclidean geometry would fail, and so will simple methods like for example linear regression if we wanted to estimate the data relationships. See [5], for illustrations and details.

There exists many algorithms and methods that globally or locally try to unwrap or unfold nonlinear manifolds, but none of which use the density ridges for this purpose. Principal curves and density ridge
algorithms, [6, 2, 7], find curves or surfaces that are smooth estimates of the underlying manifold, but the possible intrinsic nonlinear structure will still be present such that linear operations in the ambient space will fail to represent the manifold.

Manifold learning and unwrapping has been intensively studied due to the realization that many high-dimensional data distributions observed in practical applications have low entropy; samples tend to tightly concentrate around surfaces with much smaller local dimensionality than the data vector space. Based on this observation, much effort has gone into finding global nonlinear dimensionality reduction algorithms that attempt to preserve approximations of data-pair distances constrained to the manifold [8]. There is also both traditional and more recent work on local dimensionality reduction, as well as the realization that a global manifold unwrapping may not be the most meaningful, or efficient, approach in terms of reducing dimensionality as much as one could. Existing approaches, thus, can be categorized into three: (1) global methods such as Kernel PCA, ISOMAP, and Maximum Variance Unfolding [9, 5, 10], that tend to preserve global properties in the low-dimensional representation, (2) local methods such as Local Linear embedding or Laplacian eigenmaps [11, 12], that aim to preserve the local geometry in the embedded space [13, 14], and (3) techniques based on global alignment of multiple linear models [15].

Another method based on curvilinear component analysis (CCA) [16] uses similar ideas to this work in term of finding mappings between data space and unfolded spaces. This task is done in two stages of learning the unfolding transformation by a neural network, and then nonlinear projection of the input space through the mapping. This kind of representation aims to visualize the data and helps in understanding the structure of data.

A vast majority of the effort goes to data-oriented approaches and little has been done in characterizing underlying local manifold structures for a given probability density function (pdf). The critical surfaces (ridge, valley, and saddle surfaces) defined recently using the gradient and Hessian matrix eigendecomposition of the probability density function (pdf) [2] can allow us to define curvilinear coordinate systems for charts anchored at the modes (local maxima) of the underlying pdf. Such a characterization, would then allow us to develop manifold/chart learning and unwrapping techniques and algorithms that rely only on sample estimates of local gradient and Hessian, avoiding global pdf estimation in practice.

In general, manifold learning algorithms most often consists of two stages. One to remove unwanted/irrelevant dimension, and one to deal with non-trivial data structures. This will often result in algorithms that are a compromise between geometric intuition and reduction of dimension. In this work the goal is to unwrap manifolds estimated by density ridges in order to acheive geometrically meaningful unwrapping. This yields two fields of literature to reference, the topics related to principal curves/density ridges and topics related to manifold learning. We start with principal curves and surfaces.

Research related to the manifold-estimation part of our work is usually described by the interrelated terms principal curves, principal surfaces and principal manifolds. Principal curves and related subjects have
been studied in a wide array of settings, often under different names and configurations. The most common case is principal curves, which are smooth one-dimensional curves embedded in $\mathbb{R}^d$. They were originally posed as smooth self-consistent curves passing through the 'middle' of the data by Hastie and Stuetzle [6]. Kegl et al. proposed to constrain the length of the principal curves, enabling a more thorough theoretical analysis and simpler implementation [17]. Einbeck et al. suggested an algorithm for finding local principal curves based on local linear projections [18]. Our work is motivated by the work of Ozertem and Erdogmus, where local principal curves are defined as one-dimensional ridges of the data probability density [2]. This framework also extends naturally to cover higher dimensional principal surfaces by using higher dimensional density ridges. See also the work of Bas, [19], for further details and applications.

In the last few years several very interesting works related to the density ridge interpretation have been introduced. Genovese et al. have provided a theoretical analysis of non-parametric density ridge estimation [20] and Chen provides asymptotic theory and formulates density ridges as so-called solution manifolds [21].

Another closely related setting to the principal curves and derivatives of the probability density is filament estimation, which is in practice very similar to principal curves [22, 23]. For estimation purposes, Pulkkinen et al., have proposed several practical improvements for the density ridge estimation framework of Ozertem and Erdogmus [2], both in the generative model and as a method of optimization [24, 25]. Finally we mention the recent work of Gerber and Whitaker, [7], where the original framework of Hastie and Stuetzle [6] is reformulated to avoid regularization both for principal curves and surfaces.

Beyond metastudies of principal curves and density ridges themselves, several applications can be found. Examples include template based classification [26], floating ice identification in satellite images [27], galaxy identification [21], character skeletonization [28] and clustering [29] to name a few.

In the field of manifold learning within the machine learning literature, several works exists that are closely related to ours in terms of manifold unwrapping. The closest in principles and ideas are the local tangent space alignment algorithm, [30], the LOGMAP algorithm for calculating normal coordinates of a manifold, [31], and the manifold chart stitching of Pitelis et al. in [32]. The manifold parzen algorithm and contractive autoencoders should also be mentioned, [33, 34], as they are, similar to our work, algorithms that tries to learn representations along non-linear manifolds via estimates of the probability density.

1.1 Motivation: Using Density Ridges to Unwrap Manifolds

A kernel density ridge can be used to completely describe a biased version, called a surrogate, of a manifold embedded in $\mathbb{R}^D$ given sufficient samples and bounded noise [20]. The main motivation for using density ridges to locally unwrap nonlinear manifolds comes from the following.

- As the ridge is estimated via a probability density estimate, all ridges of dimensions higher than zero
Figure 1.1: Noisy one-dimensional manifold (light green) embedded in $\mathbb{R}^2$ and the density ridge estimate (dark blue). The small window shows a selected portion of the density ridge. We can see the corresponding local maxima of the gradient flow in the figure [1].

will by construction induce a gradient flow contained along the ridge/manifold surrogate. This is a trivial consequence of the definition of critical set [2].

- In the one-dimensional case the gradient flow completely describes local sections (attraction basins) of the manifold estimate, and can thus be unwrapped directly by representing the manifold in approximate arc-length coordinates. This is analogous to pulling a curly string taut.

- In the $d$-dimensional case the gradient flows that converge to the same point can be approximated and unwrapped by a local linear projection. This is analogous to flattening a curled up sheet.

To further emphasize the first point: Unless the underlying manifold is a straight line or flat in all directions, the kernel size of the density estimators used – presented in equation (1.1) on page 8 – has to be bounded. This will thus inherently lead to local maxima along the density ridges, which further leads to a gradient flow contained along the ridge. For clarity, we split the next part of our motivation into the one-dimensional case and the general $d$-dimensional case, starting with the one-dimensional case.

In the one dimensional case all points along the ridge can be described by the piece-wise arc-length of the gradient flow integral curve from the point to the local mode where the gradient vanishes. This will create a complete description of the coordinate distances from all points on the ridge to their corresponding local maxima collected in local sections of the curve. In Figure (1.1), we see a 2-D version of the synthetic ‘swiss roll’ dataset with added Gaussian noise, [11]. It is clearly a one-dimensional manifold (curve) embedded in $\mathbb{R}^2$. The density ridge is shown in dark blue dots, while the noisy data is shown in light
green dots. We see that the ridge is a good approximate for the manifold. This is key to our proposition; by following the gradient flow of the density along the ridge we have a way of tracing the underlying manifold and thus calculating distances along the manifold.

In the $d$-dimensional case, the gradient flow will generate unique curves from each point on the density towards the local modes. The manifold can thus be intuitively separated into parts based on where the gradient flow converges to the same critical points. Each part can then be mapped by a linear projection approximating the logarithmic map [31]).

Continuing from local to global unwrapping, we note that as we are using a kernel density estimate the gradient flow of the probability density is inherently local. Thus piecewise distances – or any other further operations – will also only be local, and to obtain a global unwrapping we need some strategy to combine local unwrappings. the local attraction basins are flattened intuitively, but globally the representation is not meaningful.

To obtain a global representation we take inspiration from the concept of parallel transport from differential geometry [35] and introduce a translation along the tangent vectors of the one-dimensional ridges (translation along geodesics in the $d$-dimensional case) to obtain a complete unwrapped, or simply flat, representation of the ridge. This is analogous to parallel transporting vectors along a geodesic, see for example [36], except that the piece-wise local distance along the geodesic is also added to the transported vector.

To conclude this section we add two further comments that should be noted: (1) If the intrinsic manifold turns out to be an embedded surface ($S \subset \mathbb{R}^2$) or hypersurface ($S \subset \mathbb{R}^d$, $d \geq 3$), we have to consider the case of non-zero curvature, where an isometric (distance-preserving) mapping or unfolding cannot be guaranteed. For simplicity, in the rest of this thesis we assume that the manifolds we are working with are isometric to $\mathbb{R}^d$. (2) In some cases, the definition of the density ridges allows for multiple orthogonal one-dimensional ridges, up to $d$ if $f(x) : \mathbb{R}^d \to \mathbb{R}$. In such cases local orthogonal coordinate systems can be estimated by first estimating the orthogonal ridges and then follow the gradient flow along each separate curve. This is useful in situations where the underlying one-dimensional manifold is corrupted by noise that changes along the manifold.

1.2 Principal Curves and Density Ridges

Principal curves were originally introduced by Hastie and Stuetzle [6]. Several extensions were made, [17, 18, 37], until Ozertem and Erdogmus, [2], redefined principal curves and surfaces as being the ridges of a probability density estimate, [2]. Given a probability density $f(x)$, its gradient $g(x) = \nabla^T f(x)$ and Hessian matrix $H(x) = \nabla \nabla^T f(x)$, the ridge can be defined in terms of the eigendecomposition of the Hessian matrix.
CHAPTER 1. INTRODUCTION

Definition 1. A point $x$ is on the $d$-dimensional ridge, $R$ of its probability density function, when the gradient $g(x)$ is orthogonal to at least $D - d$ eigenvectors of $H(x)$ and the corresponding $D - d$ eigenvalues are all negative.

We express the spectral decomposition of $H$ as $H(x) = Q(x)\Lambda(x)Q(x)^T$, where $Q(x)$ is the matrix of eigenvectors sorted according to the size of the eigenvalue and $\Lambda_{ii}(x) = \lambda_i$, $\lambda_1(x) > \lambda_2(x) > \ldots$, is a diagonal matrix of sorted eigenvalues. Furthermore $Q(x)$ can be decomposed into $[Q_\perp(x) \; Q_\parallel(x)]$, where $Q_\perp$ is the $d$ first eigenvectors of $Q(x)$, and $Q_\parallel$ are the $D - d$ smallest. The latter is referred to as the orthogonal subspace due to the fact that when at a ridge point, all eigenvectors in $Q_\perp$ will be orthogonal to $g(x)$.

Of great value is the recent paper by Genovese et al., [20], which showed that the Hausdorff distance between a $d$-dimensional manifold embedded in $D$ dimensions and the $d$-dimensional ridge of the density is bounded under certain restrictions wrt. noise and the closeness of the density estimate to the true density. They also show that the kernel density ridges are consistent estimators of the true underlying ridges and refer to the ridge as a surrogate of the underlying manifold. Thus, $\hat{R}$ is a point set representing the underlying manifold with theoretically established bounds under Hausdorff loss.

1.2.1 Orthogonal Local Principal Curves

In the special case of a projection onto a principal curve, $d = 1$, we note that the construction of the orthogonal subspace allows for choosing $d$ different orthogonal subspaces to project the gradient to. The most intuitive setting, which is most commonly used [2, 20, 38] is to use the top eigenvector(s) corresponding to eigenvalue(s) as the orthogonal subspace. We recall that the eigenvalues and eigenvectors of the Hessian matrix of a function points to the directions of largest second order change. In the case of a density ridge, the change in density along the ridge is assumed to be much lower than the change in density when moving away from the ridges, hence the choice of largest eigenvector pointing away from the ridge. This local ordering, though intuitive from the Hessian point of view, has not been well studied, and could in some cases actually give non-intuitive results. We will not go further into this in this thesis.

The importance of this local decomposition is that considering a $d$-dimensional ridge, the ridge tracing algorithm, presented in Chapter 2, can be applied on each of the local orthogonal subspaces. In the case of e.g. different local scales of noise this could yield a better description of the data. Figure 1.2 shows a conceptual illustration of the idea, where the overall dominating nonlinearity can be described by a principal curve/one-dimensional density ridge, but since the orthogonal structure varies along the ridge local decomposition provides further detail.
CHAPTER 1. INTRODUCTION

1.3 Relevant Topics from Differential Geometry

*Manifold learning* is a framework which is inspired by concepts from differential geometry [31, 5, 32, 11, 7, 30]. The main idea is that data sets or data structures seldom fill the vector space they are represented in. Even in low dimensions, e.g. $\mathbb{R}^3$, data often concentrate around clearly bounded subregions or so-called *manifolds*.

In some works the problem of learning representations on or along a manifold has been termed manifold unwrapping or unfolding [39, 40]. The idea is that learning the structure of a manifold keeps the local structures intact more strongly than global structures, so that the (global) nonlinearities can be unfolded, unrolled or unwrapped. This is perhaps the term most related to our work.

Unfolding or unwrapping a manifold can intuitively be performed in two different ways. Either we use some function that stretches or flattens the manifold such that we can use linear methods to calculate distances along the manifold, or we can somehow estimate the structure of the manifold such that distance measures can be defined directly along the manifold again allowing linear operations in the resulting coordinate system.

The ideas and concepts of manifold learning stems from differential geometry, the field of mathematics which studies smooth geometric objects and, closely related, smooth functions. Its main object of interest is the manifold, which can be roughly regarded as a well behaving smooth (topological) space.

A clear definition of a manifold can be found in either books of Lee or Tu, [35, 41]:

Figure 1.2: A manifold with a one-dimensional ridge/principal curve. The local basins of attraction can be decomposed using orthogonal principal curves. $M$ is the intrinsic manifold, $T_pM$ denotes the tangent space of $M$ at $p$. The green area denoted in the lower right sketch is the area of $T_pM$ constrained by the exponential map [1].
CHAPTER 1. INTRODUCTION

Definition 2. A (topological) manifold is a second countable, locally Euclidean, Hausdorff space.

Local Euclidean structure is analogous to how humans perceive the surface of the earth. At smaller scales traversing a path along the surface will seem like a straight line, but on larger scales paths along the surface of the earth are clearly curved. A Hausdorff space is a space where two separate points have disjoint neighborhoods \([42]\). E.g. a surface embedded in \(\mathbb{R}^3\) that intersects with itself will have points that shares neighborhoods and is thus not Hausdorff.

In simple terms; the intuition behind differential geometry and manifolds is to describe an object locally at a certain point or a certain homogeneous region using derivatives. Although often not explicitly stated the kernel density estimate and its derivatives – see e.g. \([3]\) – can be interpreted in a differential geometry setting. We recall that Genovese et al., \([20]\), has shown that kernel density derivatives can be used to estimate a smoothed version of an underlying manifold sampled from data with noise.

Before we go into further discussion of the connection between the kernel density estimate of data sampled from a manifold and differential geometry, we introduce a few basic concepts.

First: throughout this discussion we are talking about submanifolds of \(\mathbb{R}^D\) \([35]\). Given a manifold \(M\) diffeomorphic to \(\mathbb{R}^d\), at each point \(p \in M\) the tangent space, \(T_pM\), is the Euclidean space of dimension \(d\) which is tangent to \(M\) at \(p\) \([35]\). See Figure 1.2 for a sketch of the related concepts. The term tangent to, can intuitively be interpreted as either the space of tangent vectors of all possible curves passing through \(p\) or the space spanned by the partial derivatives of the parametrization of \(M\) at \(p\) \([35]\). A disjoint union of all tangent spaces of \(M\) is called the tangent bundle. We denote the coordinate transformation from the tangent space \(T_pM\) to the manifold \(M\) as the exponential map, and the inverse transformation from the \(M\) to the \(T_pM\) as the log map \([31]\). Finally we note that vectors in \(T_pM\) can be expressed by a local basis of differentials \(E_i = \frac{\partial p}{\partial x^i}\). These are called the local coordinates at \(p\), \([35]\). These local coordinates represents a Euclidean subspace of same dimension \(d\) as \(M\).

As for the statistical model, we assume the same model as in \([20]\) where the data points, \(X = \{x_i\}_{i=1}^n \in \mathbb{R}^D\), are sampled with noise from a distribution supported on \(M\). If we let \(P_M\) be the distribution of points on and along \(M\) and \(\Phi_\sigma\) be a Gaussian distribution with zero mean and \(\sigma\)I covariance – also in \(\mathbb{R}^D\) – that represents noisy samples that does not lie directly on the manifold, we get the following model\(^1\):

\[
P = P_M \ast \Phi_\sigma.
\]

The kernel density estimator is defined as follows:

\[
\hat{p}(x) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{\sigma^2} K \left( \frac{||x - x_i||}{\sigma^2} \right),
\]

\(^1\ast\) denotes convolution.
where $K(\cdot)$ is any symmetric and positive semi-definite kernel function. Note that we skip the normalizing constant, as it is simply a matter of scale and we are only interested in the direction of the gradient and the Hessian eigenvectors. We also note that in this work we restrict ourselves to use the Gaussian kernel $K(x_j, x_i) = \exp(-\|x_i - x_j\|^2/2\sigma^2)$. From the kernel density estimate we can calculate the gradient as follows (notation adapted from [20]):

$$g(x) = -\frac{1}{n} \sum_{i=1}^{n} \frac{x - x_i}{\sigma^2} K \left( \frac{\|x - x_i\|}{\sigma^2} \right). \quad (1.2)$$

And finally the Hessian matrix:

$$H(x) = \frac{1}{n} \sum_{i=1}^{n} \left( u_i u_i^T - \frac{1}{\sigma^2} I \right) K \left( \frac{\|x - x_i\|}{\sigma^2} \right), \quad (1.3)$$

where $u_i = (x - x_i)$. From these equations we see that the gradient is simply the average of all vectors pointing out from $x$ weighted by the kernel function value of the norm of the vectors. We also note that depending on the kernel size $\sigma^2$ the gradient will in practice be the average of the vectors pointing from $x$ to all points within the neighborhood covered by the particular kernel size. If a point and its neighbors within the kernel size distance lies on or close to the manifold $M$, this clearly is a good candidate for a local tangent space estimate.

If we look at the expression for the Hessian matrix, we notice that it turns out to be proportional to the sample covariance matrix at $x$ with the variance normalized wrt the kernel size $\sigma^2$. Like the kernel density gradient, the weight of the points taken into account in calculating the covariance is governed by the kernel size, leading to a local sample covariance estimate at $x$. We note that if a very large kernel size is used, the Hessian matrix will resemble the standard sample covariance matrix and eigendecomposition would yield standard PCA. Considering this we can interpret the Hessian matrix as fixing and aligning a local Gaussian distribution to a local subset of the data. For a one-dimensional curve with multidimensional additive noise the local Gaussian will clearly have a strong correlation along the curve which will be reflected in the eigenvectors of the local Hessian estimate. A simple illustration of this is shown in Figure 1.3. Considering the case of a principal surface we can continue our principal component analogy. Assuming the points have been sampled from some surface with some level of noise, the kernel density Hessian estimate at $x$ will in practice calculate a covariance matrix estimate based on the points within the range covered by the kernel centered at $x$. Unless the noisy points that does not lie on $M$ completely dominate, we expect that most of the variance is concentrated along the manifold, and thus the largest eigenvectors according to eigenvalue will span the local tangent space $T_x M$.

Since by definition the kernel density gradient, $g(x)$, lies in the span of $d$ Hessian eigenvectors if $x$ is on the $d$-dimensional ridge, the corresponding $d$ eigenvectors forms a natural basis for local coordinates
CHAPTER 1. INTRODUCTION

Figure 1.3: Illustration of the eigenvectors (red arrows) of the kernel density Hessian matrix at a point on the manifold $M$ (thick black line). A figurative illustration of the contours of the kernel centered at the same point is also shown to illustrate the bandwidth of the kernel [1].

Consequently the parallel Hessian eigenvectors, $Q||$, can be calculated for each $x$ that lies on the density ridge, also for out of sample points, and will thus form an approximate tangent bundle on the ridge estimate of the manifold, $\hat{R}$.

1.3.1 Properties of Density Ridges

Throughout this section we consider a single connected submanifold $M$ of Euclidean space without boundary that satisfies the properties given in [20]. Recall that a principal curve can be approximated by a one-dimensional density ridge and a principal surface by a 2-dimensional density ridge. Some properties are general, while some apply only to principal curves, or vice versa. We start by introducing the gradient flow, which we use in both cases later in this dissertation.

An integral curve of the positive gradient flow is a curve $\lambda(t)$ such that $\lambda'(t) = \nabla f(\lambda(t))$. For every point $x_0$ in the domain of $f$ there exists an integral curve that starts at $x_0$. Given a critical point$^2$ $m$, $\nabla(m) = 0$, all points that have integral curves that converges to the critical point are said to lie in the basin of attraction of $m$ [43]. This property is the foundation of mode based clustering algorithms, e.g mean shift clustering [44, 45].

The idea of a basin of attraction can be extended to hold for density ridges as well. Given a density ridge of dimension $d$ a local mode will still satisfy the criteria for being a point on the ridge. The differential equation – discussed in next chapter – for projecting points towards the density ridge always follows the gradient, so the density ridge(s) can be divided into basins of attraction based on the original points before projection. This naturally divides a manifold estimated by a density ridge into non-overlapping subsets. Closely related: a chart of a smooth manifold is a diffeomorphism from a neighborhood $U$ on the manifold to $V \subset \mathbb{R}^d$, $\phi : U \to V$. The local density ridge unwrapping algorithm, presented in Chapter 2 can be

---

$^2$A local mode in the case where the function $f$ is a probability density function.
CHAPTER 1. INTRODUCTION

considered an algorithm for learning the map \( \phi \), and we thus have a chart for each separate basin of attraction of the manifold.

We end this section with a short summary of important properties and observations for principal curves and surfaces separately.

**Principal curves:**

- The gradient flow integral curves \( \gamma \) are geodesics.
- The gradient flow separates the manifold into non-overlapping regions \( C_i \) - attraction basins, such that \( M = \bigcup_{i=1}^{\#\text{charts}} C_i \).
- Curves do not have intrinsic curvature, and thus principal curves can always be unfolded isometrically.
- The gradient flow curve starting on points on the manifold is completely contained along the one-dimensional manifold, \( \gamma \in M \).

**Principal surfaces:**

- The gradient flow separates the manifold into non-overlapping regions - attraction basins. Again \( M = \bigcup_{i=1}^{\#\text{charts}} C_i \).
- The gradient flow curves \( \gamma \) are not necessarily geodesics. If the initial point is on the manifold the gradient flow integral curves will be restricted to the manifold. Thus, if \( \gamma(0) \in M, \gamma(t) \in M \ \forall t \).
- Curvature must be introduced, isometric unfolding can no longer be guaranteed for arbitrary surfaces.
- The direction of the gradient determines the gradient flow along the manifold.

We conclude this section with an illustration in Figure 1.4(a). We see a two dimensional sample from a Gaussian distribution superimposed with an underlying one-dimensional nonlinear structure. The two orthogonal density ridges found using the RungeKuttaFehlberg solver (RKF45) are shown in red and blue dotted lines.

In Figure 1.4(b) we see the stages for a single data point from Figure 1.4(a). First the point is projected to each of the two orthogonal density ridges using the RKF45 algorithm, shown in red. Second, the point is projected along the ridge by following the gradient towards the local mode shown in green. We see that by calculating the piece-wise Euclidean distances between the gradient ascent steps – the green points – we get approximate coordinate distance along the intrinsic nonlinear geometry.


CHAPTER 1. INTRODUCTION

Figure 1.4: (a) Nonlinear and noncovex data set with density ridges shown in red(top) and blue(second); (b) A single data point projected to the two one-dimensional density ridges, trajectories shown in red and then along the gradient to the local mode(green). The density ridges as shown in Fig 1.4(a) are marked with small black dots [1].

1.4 Outline of the Dissertation

In the first method presented in Chapter 2, we propose that the orthogonal curvilinear coordinate systems revealed locally by the eigenvector flow derived from the Hessian of data density can be used as a mean to obtain local charts around modes. These charts can be stitched to form an atlas to define a global map of the data space, providing a base for data classification or clustering. A given point is projected to the ridges of the probability density by solving a differential equation which forces the gradient to be in the direction of the eigenvector corresponding to the largest eigenvalue of the Hessian. A curvilinear coordinate is then determined as the curve length along each ridge from the projection point to the mode.

In Chapter 3, we provide experiments on real datasets such as MNIST handwritten digits and Frey Faces using the method of Chapter 2, by following gradient on the projected points to density ridges. Manifold traversing in digits gives an approximation on the underlying smooth manifold representing geometric variations like orientation and thickness, and in faces dataset give the facial expression and rotation angle of the head.

The approach presented in Chapter 4, is based on the assumption that \( d \)-dimensional critical surfaces of a probability density function (pdf) provides a natural skeleton for the data, and therefore unwraps it. The main idea here is to define a measure which could easily result in critical sets. This measure is later defined as the determinant of a matrix, whose zeros give the critical sets including ridges, valleys, and saddle points. We provide unwrapping results based on analytical density distributions and estimated distributions from the
CHAPTER 1. INTRODUCTION

data drawn from analytical density functions.

In Chapter 5, an algorithmically efficient method is proposed to estimate kernel density derivatives in high dimensions, resulting in significantly reducing the total computation times for calculating kernel matrix, gradient, Hessian and principal curves projection.

An application of extracting information and classifying high dimensional data using sparsity constraints is presented in Appendix A on diffusion tensor images of the normal brains and the brains of patients with traumatic brain injuries (TBI).

1.4.1 Contribution Statement

This dissertation is the result of collaboration with Jonas Myhre (University of Tromso), Devrim Kaba (General Electric Research), and Sylvain Bouix (Brigham and Women’s Hospital at Harvard Medical School). The implementation of curvilinear coordinates and unwrapping based on the method in Chapter 2 was done by both Jonas and myself. This work resulted in the conference paper titled Invertible Nonlinear Cluster Unwrapping [13]; a journal paper is under preparation with the tentative title Manifold Unwrapping Using Density Ridges [1]. Parts of the Introduction Chapter is derived from this journal paper in progress, for which Jonas is the first author.

Similarly, the topic of computationally efficient kernel density estimates and derivatives was initiated by me and was completed in collaboration with Jonas, both in terms of algorithm design and implementation. This work resulted in a journal paper titled Computationally Efficient Exact Calculation of Kernel Density Derivatives [46].

The experiments on the unwrapping method in Chapter 4 based on critical sets was based on an original idea from Devrim in constructing the so-called Omega matrix to characterize critical sets. The implementations and experiments including log-polynomial density estimation and polynomial factorization algorithms was done by me. This work was resulted in a conference paper titled Manifold Unwrapping Using Critical Surfaces [47]. All the experiments in Chapter 3 on the manifolds of natural images was done by me.

Finally, the work on traumatic brain injury (TBI) detection using sparsity constraints on Gaussian graphical models was the result of my internship at Brigham and Women’s Hospital at Harvard medical school. In this work, I designed the algorithm and implemented it along with all the experiments. Sylvain provided the data and the specific graph used for the modeling, as well as validating the results and giving ideas on experiment designs. This work resulted in a conference paper titled Sparse Model Learning for High Dimensional Diffusion MRI Data in Traumatic Brain Injury [48], and a journal paper titled Subject-Specific Abnormal Region Detection in Traumatic Brain Injury Using Sparse Model Selection on High Dimensional Diffusion Data Medical Image Analysis, which, at the time of submitting this document, has been accepted for publication in the journal of Medical Image Analysis.
Chapter 2

Curvilinear Coordinates

The method presented in chapter is the result of collaboration with Jonas Myhre from University of Tromso. This work resulted in a conference papers as "Invertible nonlinear cluster unwrapping" [13], and a journal paper under preparation as "Manifold unwrapping using density ridges" [1].

In this Chapter, the first method for unwrapping manifolds is presented. We start by recalling that the probability density function along a local principal curve/one-dimensional density ridge will contain local maxima. For a point \( x \in \hat{R} \) the gradient, \( g(x) \), is by definition orthogonal to all except one of the eigenvectors of the local Hessian \( H(x) \). Thus any gradient on the ridge which is not exactly zero will point along the ridge towards a local mode. This restricts the gradient flow, [43], along the ridge to stay on the ridge, and thus we can formulate a differential equation that sends the points on the principal curve, along the curve/ridge towards the local mode.

We note that Arias-Castro et al. showed that a gradient ascent scheme for estimating the gradient flow lines converges uniformly to the true integral curve lines [43]. Then, by using either a gradient ascent scheme – like in mean shift clustering – or as in this work a differential equation solver, we can estimate the gradient flow lines along a ridge, and by calculating the piece-wise Euclidean distances between the steps of the gradient flow we can estimate the distance from a point to a local mode along the ridge.

Since this is the only possible path between the point on the ridge and the local mode it converges to, it is by construction a geodesic. These geodesic distances along the principal curve we will take as local coordinates in the basin of attraction of the local mode, \( m_i \), with the smallest eigenvector of \( H(m_i) \) as basis. The intuition is that the curvature as described by the Hessian eigenvalues should be lower along the ridge, than orthogonally off the ridge. Thus the choice of lowest eigenvector as basis.

The curse of dimensionality is a problem whether a density estimate is explicitly specified or is implicitly assumed. For instance, graph-based or affinity-based methods still suffer from issues such as neighborhood size or kernel width selection. We assert that knowing exactly what density estimator is used is a strength because theoretical asymptotic behavior analysis of the density estimate exists, hence the
The main difference of the proposed framework from existing manifold learning methods based on graphs and local tangent space approximations is that we specify an unwrapping method for a probability density, not a data set. Interpretability, invertibility and clarity are the strengths of the proposition.

The proposed framework considers the data space as a curved manifold. Thus, the manifold has the same dimensionality of the original data. As a special case, consider linear Principal Component Analysis (PCA) which simply rotates from data coordinates to linear principal component coordinates. This method generalizes linear PCA in a way that data coordinates (Cartesian) are transferred to a unwrapped instance of the cluster using principal curves of the density model. The unwrapping happens by projection of data points to the principal curves of the density, and calculating the curve length from the projection point to the mode of the density along the principal curve. These curved distances are termed as curvilinear coordinates.

In summary this chapter includes: (1) the introduction of an atlas-based manifold charting concept using principal curves and a methodology for generating Cartesian-curvilinear coordinate pairs for the probability density function (pdf) that underlies data, (2) an algorithm to learn, within a chart defined on the attraction basin of a mode of the pdf, from a training set of Cartesian-curvilinear coordinate pairs, a diffeomorphism that allows transformations between Cartesian and curvilinear coordinates (relative to the mode, which acts as the local origin within the associated chart). The first contribution is based on the concept of locally defined principal curves [2], and the second contribution is based on extending the concept of diffeomorphic deformable landmark based image registration [49].

2.1 Principal Curve Projection

Principal curves are defined as the ridges of the data pdf; every point on a principal curve is a local maximum of the pdf in the subspace orthogonal to the curve at that point [2]. Given pdf $p(x)$ for a random vector $X \in \mathbb{R}^d$, the principal curves are represented based on the (column) gradient $g(x) \doteq \nabla^T p(x)$ and Hessian matrix $H(x) \doteq \nabla^T \nabla p(x)$ of the pdf.

On a ridge of a pdf, the gradient becomes an eigenvector of the Hessian and is tangent to the ridge. All remaining eigenvectors form the so-called orthogonal subspace, and have negative eigenvalues (assuming second order Taylor approximation is sufficient to establish local maximality, which would generally be the case, for instance in Gaussian mixture models). At an isolated mode (strict local maximum) of the pdf, all eigenvalues of the Hessian are negative. When the Hessian eigenvalues at the mode are distinct and negative, the principal curves intersect orthogonally at the mode of the pdf, and form a natural curved coordinate system for a chart.\footnote{Note that this definition replaces the Hastie’s concept of being the expected value in the orthogonal subspace with the concept of being the local maximum in the orthogonal subspace [6].}

\footnote{The reader should be aware that pathological cases may exist and we are still investigating these to report results in a future}
CHAPTER 2. CURVILINEAR COORDINATES

Figure 2.1: Projection trajectories following Hessian eigenvector flow for two different choices of the strict monotonically increasing function on Gaussian pdf.

Each data point \( x \in \mathbb{R}^d \) is projected to a selected ridge (identified by the designated tangent eigenvector at the data point, which is the initial condition for the differential equation) by following the Hessian eigenvector flow according to

\[
\dot{y}(t) = Q_{\perp}(y(t))Q_{\perp}^T(y(t))g(y(t)),
\]

where \( y(t) \in \mathbb{R}^d \) indicates each point on the trajectory climbing to the ridge, while the initial point is \( y(0) = x \). Here \( \dot{y}(t) = dy(t)/dt \) for some arbitrary curve parameterization \( t \). The vector \( g(y) \) is the gradient at point \( y \). The orthogonal subspace \( Q_{\perp}(y) \) is the \( d \times (d - 1) \) matrix of eigenvectors of the Hessian (with orthonormal columns). The excluded eigenvector, \( Q_{\parallel}(y) \) is the eigenvector designated at \( y(0) \) to be the tangent direction of the target ridge. The iterations stop on the ridge at \( y(T) = y_T \). Hessian at point \( x \) is decomposed as

\[
H(.) = Q_{\perp}(.)\Lambda_{\perp}(.)Q_{\perp}^T(.) + Q_{\parallel}(.)\Lambda_{\parallel}(.)Q_{\parallel}^T(.)
\]

In Figure 2.2, the process of convergence of Equation (2.1) is shown for a data point on the non-convex Rosenbrock density function [50]. Here, the gradient is shown with green vector, small and large eigen-vectors of Hessian with light blue and red vectors, respectively. Over the iterations the gradient and large eigenvector are being converged until the stopping criteria is satisfied.

A strict monotonically increasing function of the pdf has the same ridges as the pdf itself [2].
Figure 2.2: Convergence of a given point on the Rosenbrock density function based on Equation (2.1); green: gradient, light blue: small eigen-vectors of Hessian, red: large eigen-vector of Hessian. The gradient and large eigenvector are being alligned over the iterations until stopping criteria is satisfied.

However, since the Hessian matrix will be different for the two cases, the projection trajectories will be different and the curvilinear coordinates obtained using two different strict monotonically increasing functions would be related through a diffeomorphisms that has a diagonal Jacobian (i.e. each individual curvilinear coordinate will be deformed by an invertible nonlinear mapping). Figure 2.1(a) illustrates the trajectory of curvilinear projections to the ridges for five data points, each with a distinct color, using the Hessian and gradient of the pdf itself. For the Gaussian pdf, taking the natural logarithm of the pdf makes the Hessian constant and equal to the negative-inverse-covariance of the Gaussian. This leads to linear projections as in standard linear PCA and as shown in Figure 2.1(b).
CHAPTER 2. CURVILINEAR COORDINATES

Algorithm 1 Local density ridge unwrapping.

Require: Input points, $x_i$, projected to a one-dimensional density ridge $\hat{R}$ by solving Equation (2.1).
1: Compute the trajectory, $Z_i$, from each point on the density ridge to the associated mode by solving Equation (2.3).
2: For each trajectory $Z_i = [z_1^i \ z_2^i \ \cdots \ z_N^i]$ calculate:

$$c_i = \sum_{t=1}^{N-1} \|z_t^i - z_{t+1}^i\|_2$$

(2.4)

Ensure: Local manifold chart coordinate $c_i$.

2.2 Curvilinear Coordinates

Once a point $x$ is projected to the ridge at location $y_T$, solving the (gradient ascent) differential equation

$$\dot{z}(t) = g(z(t))$$

(2.3)

using initial condition $z(0) = y_T$ until it reaches the mode (e.g. when a suitable stopping criterion such as the norm of the gradient becoming sufficiently close to zero is achieved) generates a trajectory that follows the ridge. Here $z(t)$ is the tracing path. The entry of the curvilinear coordinate vector $c(x)$ corresponding to selected ridge is defined as the curve length from $y_1$ to the mode. The sign of each curvilinear coordinate is determined by arbitrarily selecting signs for eigenvectors of the Hessian at the mode, and considering the direction from which trajectory is approaching to the mode relative to the selected mode-eigenvector directions. This process could be repeated for all $d$ ridges (by designating different Hessian eigenvectors at $x$ as tangent directions, leading to the projection of this point to different ridges using (2.1)). When this process is completed for all $d$ ridges for all $x$ in a chart, one obtains the proposed local unwrapping around the mode under investigation.

Solving this equations gives coordinate lengths along the ridge and thus yields a complete local coordinate description for every point in the local basin of attraction of $m_i$. In practice we solve this using a Runge Kutta fourth and fifth order pair. This is an adaptive stepwise solver, which yields benefits both in terms of speed and accuracy. For density ridge estimations we use the kernel density estimate from Equation (1.1), its gradient, Equation (1.2), and the Hessian matrix from Equation (1.3) as presented in the previous section. The ridge projection for each data point considered is independent from other projections, so processing each point in parallel is possible. In this work we have used the parallel processing toolbox in MATLAB to achieve a local speedup proportional to the number of cores on the computer. We have summarized the steps necessary to perform local principal curve unwrapping in Algorithm 1.

The process described above implicitly defines a vector-valued diffeomorphic mapping denoted by $c(x)$, indicating that one obtains $d$ curvilinear coordinates from $d$ Cartesian coordinates. A mode at Cartesian
coordinate \( m \) in the data space has curvilinear coordinates \( c(m) = 0 \) in its own chart. As an example of the projection to principal curves and unwrapping the coordinates, the "two-moon" dataset is projected and unwrapped in Figures 2.4(a) and 2.5(b).

### 2.3 Diffeomorphic Projection Model

Given a training set of Cartesian and curvilinear coordinate pairs \( \{(x_1, c(x_1)), \ldots, (x_n, c(x_n))\} \) obtained using the methodology described above, we can now consider the problem of finding a diffeomorphic model that can be used to approximately determine the curvilinear coordinates of out-of-sample data (test data not seen in the training process) by interpolating the chart. Ideally, given the density model and the corresponding \( f(x) \) and its derivatives, a test data point could be projected using the same numerical integration methodology (e.g., RK4), however, that requires solving \( d \) differential equations numerically, where at each step of each trajectory approximation, Hessian eigendecomposition is necessary - and for high dimensional data, this may become a computational hindrance. In this section, we generalize a diffeomorphic landmark matching methodology commonly used in image registration, which provides a diffeomorphism (both Cartesian to curvilinear and curvilinear to Cartesian transformations exist and these inverse mappings are continuously differentiable, at least once). The current proposition does not guarantee a conformable map to emerge from this modeling process, although we know that the underlying true Cartesian to orthogonal curvilinear transformation is conformal diffeomorphic.

#### 2.3.1 Learning the Diffeomorphic Projection Model

The purpose of this learning process is to identify a diffeomorphic model which approximates the transformations between Cartesian and curvilinear coordinates within a mode, for out-of-sample projections. Given such a model, dimension reduction of test samples would involve simply passing the Cartesian test data through the learned diffeomorphic model to get its curvilinear coordinates, and discard unwanted coordinates from this vector (i.e., replace small curvilinear coordinates with zeros). Similarly, a curvilinear coordinate vector with reduced dimension can be pulled back to the full dimensional space by appending zeros for discarded dimensions and using the inverse of the diffeomorphic mapping identified as described here. Consequently, the model learned here offers a fast and convenient approximation for both nonlinear dimension reduction and pull-back operations under the proposed framework.

In the process of building this diffeomorphic transformation model between Cartesian and curvilinear coordinates, we adopt ideas from the deformable image registration literature. These techniques develop a diffeomorphic large deformation model by cascading diffeomorphic models for small deformations, essentially by numerically approximating the solution of a differential equation with boundary and smoothness conditions imposed on the approximate solution [51]. Small deformations parametrize a displacement field
Chapter 2. Curvilinear Coordinates

\( \mathbf{u} \), which is added to the initial point to find the transformation as \( \phi(\mathbf{x}) = \mathbf{x} + \mathbf{u}(\mathbf{x}) \). Clearly, starting from \( \mathbf{x} \) and adding a small perturbation has the advantage that the Jacobian of the overall mapping is identity plus a perturbation (hence nonsingular for all \( \mathbf{x} \)), leading to a diffeomorphism.

These approximations fail in the case of large non-linear deformations, and is thus not suitable for our setting. The large deformation framework employed here, on the other hand, cascades such small perturbations optimally in order to guarantee a one-to-one, smooth, and continuous mapping, with nonsingular Jacobian guarantees implying a diffeomorphism [52]. A key element of a diffeomorphic mapping is that it preserves the topology (but not angles since it is not conformal in general), and is consistent under composition of transformations. These properties are useful when transforming the coordinates in manifold unwrapping and dimensionality reduction, implying transitive inverse consistent mappings.

The Euler-Lagrange equation for solving the large deformation diffeomorphic mapping is studied in [53], [54], and [49] for variational formulation of image matching. This setting parametrizes the transformation by means of velocity vectors \( \mathbf{v} \) tangent to each displacement vector \( \mathbf{u} \). In our version of this model, input training data \( \{ \mathbf{x}_1, \ldots, \mathbf{x}_n \} \) and corresponding transformed curvilinear coordinates \( \{ \mathbf{c}_1, \ldots, \mathbf{c}_n \} \), both coordinates in \( \mathbb{R}^d \), are connected via the diffeomorphic change of coordinate \( \phi : \mathbb{R}^d \to \mathbb{R}^d \). \( \phi \) is the solution of the ordinary differential equation (ODE)

\[
\frac{d\phi(\mathbf{x}, t)}{dt} = \mathbf{v}(\phi(\mathbf{x}, t), t),
\]

where \( t \in [0, 1] \) is the curve parametrization for the differential equation trajectory, and the initial point \( \phi(\mathbf{x}, 0) = \mathbf{x} \) corresponds to the identity transform. Here, \( \phi(\mathbf{x}, t) \) is the Lagrangian trajectory defined as the position at time \( t \), which was at \( \mathbf{x} \) at time 0. The final transformation (solution of the differential equation) \( \phi(\cdot, 1) \) is therefore controlled through the velocity field \( \mathbf{v}(\cdot, t) \), and is given as:

\[
\phi(\mathbf{x}, 1) = \mathbf{x} + \int_0^1 \mathbf{v}(\phi(\mathbf{x}, \tau), \tau) d\tau.
\]

Since such \( \phi(\mathbf{x}, t) \) is not unique, the optimal diffeomorphic match is constructed by minimizing deformation energy \( \| L\mathbf{v} \|^2 \), where \( L \) is a linear differential operator on the velocity field. This is analogous to thin-plate splines [55]. In addition, we want to minimize the distance between the curvilinear coordinates \( \mathbf{c} \) and the endpoints of the transformation \( \phi(\mathbf{x}, 1) \), resulting in the following optimization problem for the velocity field of (2.6):

\[
\hat{\mathbf{v}}(\mathbf{x}, t) = \arg\min_{\mathbf{v}(\mathbf{x}, t)} \int_{\mathbb{R}^d \times [0, 1]} \| L\mathbf{v}(\mathbf{x}, t) \|^2 \, d\mathbf{x} \, dt \\
+ \sum_{i=1}^n [\mathbf{c}_i - \phi(\mathbf{x}_i, 1)]^T [\mathbf{c}_i - \phi(\mathbf{x}_i, 1)].
\]
CHAPTER 2. CURVILINEAR COORDINATES

In the above equation the first and second terms are the regularization and the data terms respectively. This optimization problem poses two problems: the infinite dimensional parameter space of the velocity field and the continuous nature of the integral. The first issue is elegantly alleviated by noticing that the minimizer of (2.7) must take the following form (for reasons similar to kernel regression emerging from the representation theorem for Reproducing Kernel Hilbert Space (RKHS)):

\[
\hat{\mathbf{v}}(\mathbf{x}, t) = \sum_{i=1}^{n} k(\phi(\mathbf{x}_i, t), \mathbf{x}) \sum_{j=1}^{n} (\mathbf{K}^{-1}(t))_{ij} \dot{\phi}(\mathbf{x}_j, t),
\]

(2.8)

where \( \mathbf{K}_{ij}(t) = k(\phi(\mathbf{x}_i, t), \phi(\mathbf{x}_j, t)) \), where \( k \) is the Green’s function for \( L^T L \); that is, \( L^T L k(\mathbf{x} - \mathbf{y}) = \delta(\mathbf{x} - \mathbf{y}) \). Hence, this equivalent optimization problem is obtained [49]:

\[
\dot{\mathbf{\Phi}}(t) = \arg \min_{\mathbf{\Phi}(t)} \int_{0}^{1} \mathbf{\Phi}(t) \mathbf{K}^{-1}(t) \mathbf{\Phi}(t) dt \\
+ \sum_{i=1}^{n} [\mathbf{c}_i - \phi(\mathbf{x}_i, 1)]^T [\mathbf{c}_i - \phi(\mathbf{x}_i, 1)]
\]

subject to \( \mathbf{\Phi}(0) = [\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_n^T]^T \)

(2.9)

This reformulation has reduced the problem from finding the velocity field on the entire space \( \mathbb{R}^d \) to finding \( n \) trajectories.

Finally we can approximate these \( n \) velocity fields trajectories on \( t \in [0, 1] \) to be piece-wise constant on \( M \) sub-intervals (as in Euler/RK2 numerical integration),

\[
\{t_m = 1/M\}_{m=0}^{M},
\]

resulting in:

\[
\int_{0}^{1} \dot{\mathbf{\Phi}}(t) \mathbf{K}^{-1}(t) \mathbf{\Phi}(t) dt \approx M \sum_{m=0}^{M-1} \left( (\mathbf{\Phi}(t_{m+1}) - \mathbf{\Phi}(t_m))^T \mathbf{K}^{-1}(t_m) (\mathbf{\Phi}(t_{m+1}) - \mathbf{\Phi}(t_m)) \right).
\]

(2.10)

2.3.2 Projecting Out-of-Sample Test Data

We continue to use the landmark matching framework [49], which essentially results in a kernel regression type transformation for projecting points that are not in the original landmark/training set. The out-of-sample test data point projection rule consists of a weighted sum of estimated velocities at each training sample, times the inverse pairwise kernel matrix seen earlier during training. The estimated velocity
CHAPTER 2. CURVILINEAR COORDINATES

at time $t \in [t_{i-1}, t_i]$ for a training point $x_i$ is given as

$$\dot{\phi}(x_n, t) = \frac{1}{(1/M)} \phi(x_n, t_i) - \phi(x_n, t_{i-1}),$$

(2.11)

and we get the velocity field for an out-of-sample data point $\tilde{x}$ as

$$\dot{v}(\tilde{x}, t) = \sum_{i=1}^{n} k(\phi(x_n, t), \tilde{x}) \sum_{j=1}^{n} (K^{-1}(t))_{ij} \dot{\phi}(x_j, t).$$

(2.12)

2.4 Experiments and Results

For a given training set $\{x_1, \ldots, x_n\}$, after obtaining a suitable pdf estimate (using mixture models, kernel density estimates, exponential families, etc.), based on equations (2.1) and (2.3), one can accurately determine corresponding curvilinear coordinates $\{c_1, \ldots, c_n\}$. Clearly, during the projection process, any numerical integration method can be employed; here 4th order Runge Kutta integrator (RK4) or its variable step-length versions are recommended due to favorable computational complexity versus error accumulation characteristics compared to more basic methods like Euler integration.

Figure (2.3) shows the curvilinear projection method on swiss roll. The top and middle row are the results when using kernel density estimation with anisotropic Gaussian kernels with distinct bandwidth matrix for each data point, derived based on local covariance matrix in the neighborhood of the data point. While this defines the orientation of anisotropic kernels, a scalar kernel width is globally optimized for all the data points utilizing maximum likelihood objective and leave-one-out cross validation. The kernel width for results in top and middle row of Figure (2.3) are $\sigma = 5$ and $\sigma = 1$ respectively, which three columns show different views. As we could see, $\sigma = 5$ is a better kernel width choice for this data. The iterative projection is terminated when the angle between $g(y)$ and $g(y)H(y)$ is sufficiently close to zero, which is set to $\varepsilon = 0.001$. The bottom row shows the ridge projection results when using subspace constrained mean shift [2] with same bandwidth selection, and kernel widths of $\sigma = 1$, $\sigma = 3$, $\sigma = 5$ from left to right respectively. This illustrates the advantage of the proposed curvilinear projections, as the mean shift projection results are not desirable.

In 2.4 the density estimation and principal curves in crescent and two-moon data sets are shown. In 2.4(c) the two moons are closer and aligned forming a circle-like shape. To illustrate commonalities and differences between the proposed coordinate unfolding method and several benchmark methods, we compare the results of algorithms on a `crescent'-shaped cluster, commonly used to evaluate algorithms that handle nonlinearities in Figure 2.5. Specifically, ISOMAP [5], Local Linear Embedding (LLE) [11], Laplacian Eigenmaps [12], and t-SNE [56] are used.

The results of applying algorithms on crescent and two-moon datasets are shown in Figures 2.5 and
Figure 2.3: Principal curve projection on swiss roll. (a), (b), (c): blue data points with projected red points. Columns show three orientations of same data. Gaussian kernel width is 5 and the bias in density estimate causes the mis-orientation. (d), (e), (f): same as above but the kernel width is too small for the data and the projections are noisy. (g), (h), (i): Color coded data points and projected black points using mean-shift projection method in [2], and with kernel width of 1, 3 and 5. None of them show an accurate projection to the ridge of the density. Please note that this is not an example of 1-d unwrapping and it only presents projection to one of the ridges.

2.6. Only PC unwrapping results in meaningful embeddings. For the two-moon dataset, proposed principal curve (PC) unwrapping method not only unwraps the manifold embedded in the data, but separates the two moons into clusters while unwrapping. For this experiment, we only used the differential equations to find the projections as we did not use any out-of-sample extensions for the other algorithms. Visual comparison of the proposed curvilinear coordinates with alternatives shows that the proposed charting method reveals an
CHAPTER 2. CURVILINEAR COORDINATES

Figure 2.4: Density estimation and principal curves in crescent and two-moon data sets. In (c) the two moons are closer and aligned forming a circle-like shape.

Figure 2.5: Projection of ‘crescent’ data set without reducing the dimension using benchmark manifold learning techniques, and the proposed principal curves (PC). Only PC unwrapping results in meaningful embeddings. The benchmark algorithms fail at providing a meaningful unwrapping of the data. Appropriate and intuitive unwrapping of the data.

All benchmark methods have either twisted or warped the data set in a way that leaves little room for interpretation and seems quite counter-intuitive. These methods will all give reasonable results with the dimension reduced to one, but when we try to look at the geometry behind the transformation, here in two-dimensions, the curvilinear representation gives, by far, the most intuitive representation.

In Figure 2.7, the unwrapping of two moon data set is shown while the moons are closer, and it still results in perfect unwrapping/clustering. The benchmark results are not shown since they failed similar to the
Figure 2.6: Projection of 'two-moon' data set using benchmark manifold learning techniques, and the proposed principal curves (PC). Proposed method not only unwraps the manifold embedded in the data, but separates the two moons into clusters while unwrapping. Benchmark methods fail in unwrapping.

To illustrate the quality of diffeomorphic model based out-of-sample coordinate transformations, we split an $n = 200$ crescent data into 35 training samples and 165 test samples and run the learning and kernel-model-based transformation/projection. The results, shown in Figure 2.8, demonstrate that the out-of-sample projections are reasonably accurate, except in some cases when test points fall outside the convex hull of the training set.

### 2.5 Discussion and Conclusions

In this chapter, we showed that ideal principal curve projections using differential equation solutions involving eigenvectors of the density Hessian yield a natural orthogonal curvilinear coordinate system for each chart imposed around each mode, with the local maximum at the origin of this chart. Furthermore, we illustrated the effect of using different monotonically increasing manipulations to the data density, namely the effect of the function $h$ on the resulting projection trajectories and curvilinear coordinates for the charts. Finally, we developed a methodology to learn diffeomorphic Cartesian-to-curvilinear coordinate transformations which yield simple out-of-sample forward projection and pull back expressions in the form of a kernel machine model. Results showed that the proposed curvilinear coordinates capture the
CHAPTER 2. CURVILINEAR COORDINATES

Figure 2.7: Projection of two moon data set while the moons are closer, still results in perfect unwrapping/clustering.

Figure 2.8: Out-of-sample unwrapping. Training points marked as dots, test points marked as crosses. Color coding is included to illustrate that the sign change along the diffeomorphism.
underlying manifold of a mode more successfully compared to benchmark data-oriented methods available in the literature, proving our assertion that the ridges of the distribution and the local maximum point at the intersection of these curves, form a natural geometric skeleton for the curvilinear structure of the data distribution at this mode.
Chapter 3

Natural Images Manifolds

Manifold traversing in natural images gives an approximation on the underlying manifold near which images of the same class or with similar features exist. In this chapter the experiments are performed on MNIST digits [57], based on the method presented in Chapter 3. The goal is to detect a smooth manifold on geometric variations like rotations, translations and also more nonlinear deformations in the shape of the objects in the dataset.

3.1 MNIST handwritten digits

In this experiment we use MNIST handwritten images of a certain digit. MNIST images of the digit, for instance 1, is projected onto its first three (Euclidean) principal components. In order to find the principal curves on this data set, an isotropic Gaussian kernel is learned on the PCA projections. Maximum likelihood cross validation is used for choosing the kernel width hyperparameter.

The use of PCA as a linear projection should not influence the geometric properties of the data. Ideally PCA would remove all ambient space except $\mathbb{R}^{d+1}$ such that the codimension\footnote{If $M \subset \mathbb{R}^d$ is embedded in $\mathbb{R}^D$ the codimension is $D - d$.} is one.

In Figures 3.1(a) and 3.1(c) the log likelihood function versus kernel width as well as the optimum bandwidth measures are shown for all MNIST digits and only digit one (mnist-1), respectively. In order to test the representation power of the density estimate after dimensionality reduction, samples are drawn from the generative model given by density estimate and then an inverse PCA transform is applied. Drawn Samples in the lower half of Figures 3.1(b) and 3.1(d) show a reasonable reconstruction of digits from the underlying model.

In Figure 3.2 the projection trajectories to the first and second surfaces according to Equation 2.1 are shown. Figure 3.3 shows the principal surfaces for 6 of the digits (the rest of them show similar results), and 3.4 shows the unwrapping based on Algorithm 1 in 3 dimensions. Specifically, Figure 3.4(a) shows the...
CHAPTER 3. NATURAL IMAGES MANIFOLDS

Figure 3.1: Left: Isotropic Gaussian kernel learned on PCA projection of number 1 in digits data (digits-1), reducing dimension of digits data set to 3. Maximum likelihood cross validation is used for choosing the kernel width hyperparameter.

three dimensional data of mnist-1 in grey color and the corresponding first and second principal surface in blue and red. We clearly see that the data follows a nonlinear submanifold and that the variation of the images varies along a low dimensional surface.

For visualization, we project these into the two leading Euclidean principal components. We see that the coordinates represents a clear structure in the data – a straight up version of the digit around the origin and different, but symmetric variations along the two axes. In Figure 3.5 mnist-0 images are shown on top of the 2d projection of first and second principal surfaces. In 3.5(a), the orientation constitute the main source of variation while in 3.5(b), the main mode of variation is the thickness across high variance dimension.
CHAPTER 3. NATURAL IMAGES MANIFOLDS

Figure 3.2: The projection trajectories to the first and second surfaces according to Equation 2.1

Unwrapped mnist-0 digits with a selection of images of the actual digits is shown in Figure 3.6. Orientation and thickness of digits smoothly vary across horizontal and vertical directions, respectively. To compare the results with benchmark manifold learning methods, we do the same experiment on mnist-0 using Laplacian Eigenmap [58] and t-SNE [56], shown in Figure 3.7. We can see that, unlike the proposed method, no smooth variation of structure is preserved using these methods. Similar experiment is performed on mnist-2 and mnist-9 in Figures 3.8 to 3.10. See figures captions for more details.

In Figure 3.12, the unwrapping of mnist digits 0,1,2 combined is presented We can see separate clusters while orientation and thickness within clusters varies smoothly.

3.1.1 Feature Space Visualization

In this section, we present experiments on classifying MNIST digits using convolutional neural networks (CNN), in order to visualize the feature space to tracking the training process across epochs (iterations). As shown in Figure 3.13, input mnist images are fed into to 5-by-5 convolutional layers, followed by ReLU and 2-by-2 max-pooling layers, followed by two fully connected layers of 1024 and 10 nodes, followed by a Softmax layer. The features used for visualization are the evidences after 10-node layer before computing probabilities (Softmax). 2000 images from first 6 digits are used here.
Figure 3.3: Three dimensional data of 6 of mnist digits in grey color and the corresponding first and second principal surface in blue and red. We clearly see that the data follows a nonlinear submanifold.

The features are derived from “test set” evaluation at the last fully-convolutional 10-dimensional vector, before training, after 1, 3, and 6 epochs of training, respectively. The results are shown in Figures
3.14 to 3.17. We can see that as training progresses the same digits are clustered together. This experiment is to test the capability of the proposed dimensionality reduction algorithm in order to be used for multi-class "unsupervised" visualization.

### 3.2 Frey Faces

In this section we test the density ridge unwrapping on the Frey faces data set. Again, for visualization purposes we reduce the dimension of the data set down to three principal components.

In Figure 3.18 we see the results of the density ridge algorithm. A selection of original input images are displayed on top of the selected coordinates. We see that the main horizontal direction, exhibiting the largest variance, captures the sad/happy expression of the faces. The second direction with lower variance is interpreted as the orientation of the face, where it is looking straight ahead in the middle.
Figure 3.5: (a) mnist-0 images on top of the 2d projection of first principal surface. The orientation constitute the main source of variation in the lower variance dimension. (b) mnist-0 on top of 2d projection of second principal surface. The main mode of variation here is the thickness across high variance dimension.
Figure 3.6: Unwrapped mnist-0 digits with a selection of images of the actual digits on top. Orientation and thickness of digits smoothly vary across horizontal and vertical directions, respectively.
Figure 3.7: Unfolding mnist-0 digits using two benchmark manifold learning methods. Unlike the proposed method, no smooth variation of structure is preserved using these methods.
Figure 3.8: Unfolding mnist-2 digits using the proposed method and t-SNE. Similar to Figure 3.7(b) for mnist-0, no smooth variation of structure is preserved using t-SNE, while proposed method shows smooth change of thickness and shape in horizontal and vertical coordinates, respectively.
Figure 3.9: Unfolding mnist-9 digits using the proposed method, which shows smooth change of thickness and orientation in horizontal and vertical coordinates, respectively.
Figure 3.10: Unfolding mnist-9 digits using benchmark manifold learning methods. Similar to previous results, no smooth structure is captured.
Figure 3.11: 2D projection of one of principal surfaces of mnist digits 0,1,2. We can see separate clusters while orientation and thickness within clusters varies smoothly
Figure 3.12: Unwrapping of mnist digits 0,1,2. vertical axis shows the thickness variations, while horizontal axis value changes based on clusters of digits (0 on the right, 1 on the left and 2 in the middle).

Figure 3.13: Convolutional neural network used for classifying mnist digits in order to visualize feature space (layer before Softmax) to track the training process. Here is the order of layers: 5*5 convolutions to 32 channels followed by ReLU and 2*2 max-pooling, 5*5 convolutions to 64 channels followed by ReLU and 2*2 max-pooling, Two fully connected layers of 1024 and 10, and Softmax. Dropout is also used during training, to improve generalization.
Figure 3.14: Visualization of features from test set of first 6 digits of mnist, evaluated at the last fully-convolutional layer before training.
CHAPTER 3. NATURAL IMAGES MANIFOLDS

Figure 3.15: Features evaluated after 1 epoch of training.

Figure 3.16: Features evaluated after 3 epoch of training.
Figure 3.17: Features evaluated after 6 epoch of training. As training progresses to more epochs, the same digits are clustered together more strongly.
Figure 3.18: Frey faces on top of 2D projection using first and second PC projections. A random selection of actual face images shown on top of the unwrapped coordinates for illustration of structure.
Chapter 4

Manifold Unwrapping Using Critical Surfaces

The unwrapping method in this chapter based on critical sets, was based on an original idea from Devrim Kaba in designing the so-called Omega matrix and the way this matrix relates to the critical sets. The implementations and experiments including log-polynomial density estimation algorithm was done by me. This work was resulted in a conference paper named "Manifold unwrapping using critical surfaces" [47]. All the experiments in Chapter 3 on the manifolds in natural images was done by me.

In this chapter, the specific aims are as follows: (1) We provide a new characterization of the critical surfaces of a pdf using gradient and Hessian, avoiding the requirement for any eigendecomposition of the Hessian. (2) We demonstrate that different dimensional critical sets of a pdf are the zero-level sets of a sequence of matrix determinants; this characterization makes identifying the d-dimensional underlying local manifold trivial and could allow manifold learning by developing direct sample estimators of the zero-level set of this determinant form. (3) For the exponential pdf family, we show that this determinant will take a polynomial form, therefore the critical sets will be zero-level sets of polynomial functions. (4) For the exponential of polynomials, we provide a polynomial factorization based methodology to define a curvilinear (polynomial) coordinate system for charts anchored at the modes of the pdf. These charts could be used for local manifold unwrapping or, if one mode is taken as a reference, its chart could be extended to assign global curvilinear coordinates to every point in the space, therefore offering a global manifold unwrapping solution (which should be used sparingly and with care in practice). We suggest that the critical surfaces of a pdf provide a natural skeleton for any given pdf, therefore are good candidates to anchor any manifold/chart curvilinear coordinate system to. We also suggest that mode-based charts are good candidates towards building atlas-based manifold unwrapping solutions for complex data distributions in high dimensional spaces; that is, after mode-based clustering each cluster gets its own chart with the mode at its origin.

Here, we do not attempt to make contributions in the following areas: (1) a data-oriented algorithm...
that seeks to achieve local or global manifold unwrapping and dimension reduction, (2) an unwrapping that attempts to optimize a particular statistic of the coordinates or projections, (3) an application of the proposed framework in real data sets. Our goal here is to provide a theoretical framework on which effective solutions can be built.

4.1 Notation and Background

Throughout this chapter \( x \in \mathbb{R}^n \) denotes a random vector. The function \( p(x) \) denotes its probability density function (pdf) and is assumed to be at least twice continuously differentiable. Let \( f(x) = a(p(x)) \) be a function of the pdf, where \( a : \mathbb{R} \to \mathbb{R} \) is at least twice continuously differentiable and is strictly monotonically increasing.\(^1\) We will denote the transpose of the gradient of \( f(x) \) by \( g(x) \) and its Hessian will be denoted by \( H(x) \). We use standard mathematical notation. We will be referring the invariance of a certain subspace under a linear transformation. Recall that a subspace \( \mathcal{V} \subseteq \mathbb{R}^n \) is called \( M \)-invariant, for a linear transformation \( M : \mathbb{R}^n \to \mathbb{R}^n \), if \( M \mathcal{V} \subseteq \mathcal{V} \).

4.1.1 Critical Sets

The notion of critical points from calculus (which are, of course, zero dimensional) have higher dimensional analogues. Such sets are called critical sets and have the following formal definition [59].

**Definition 1.** For \( d = 0, 1, \ldots, n - 1 \), the \( d \)-dimensional critical set, denoted \( C^d \), is the set of all points \( x \in \mathbb{R}^n \) such that \( g(x) \) is orthogonal to at least \( n - d \) eigenvectors of \( H(x) \).\(^2\)

The definition above has a natural consequence, \( C^d \) is a growing set with increasing \( d \). That is,

\[
C^d \subseteq C^{d+1}
\]

(4.1)

Hence, points which belong to \( C^d \) but not to a critical set of lower dimension are of particular interest, and they deserve a special name.

**Definition 2.** A point \( x \in C^d \setminus C^{d-1} \) is called a regular point of \( C^d \).

This definition of the critical sets is standard in the literature. However, it does not become as handy if one wants to locate and plot the critical sets. Hence, an alternative characterization of the critical sets is favorable. The following approach is inspired by the techniques of geometric control theory and the eventual characterization will resemble the famous controllability matrix of Kalman. We assume that all

---

\(^1\)The critical surfaces of \( p(x) \) and \( f(x) \) are identical [59], therefore the choice of \( a \) will only nonlinearly warp coordinates along each curvilinear axis we define in a diffeomorphic fashion. Specifically for a pdf in the form of exponential of polynomials, such as a Gaussian, \( f(x) = \ln p(x) \) becomes a natural choice. In fact, with this choice, for a Gaussian pdf, the unwrapping we define will coincide with linear principal component analysis.

\(^2\)For the purposes of this definition, we consider \( g(x) = 0 \) to be orthogonal to all \( n \) eigenvectors of the Hessian.
eigenvalues of $H(x)$ are distinct\(^3\) for all $x \in \mathbb{R}^n$. Hence, at each $x$, $H(x)$ has $n$ distinct eigenvectors, and since $H(x)$ is a real symmetric matrix, the eigenvectors of $H(x)$ form an orthogonal basis. If $x$ is a regular point of $C^d$, that is $g(x)$ is orthogonal to exactly $n - d$ eigenvectors of $H(x)$ (and not more), then since the eigenvectors form an orthogonal basis, $g(x)$ must be in the span of the remaining $d$ eigenvectors. Without loss of generality, suppose that $g(x)$ is in $W_d := \text{span}\{q_1(x), \ldots, q_d(x)\}$. We know from linear algebra that $W_d$ is $H(x)$-invariant. Hence, $W_d$ is an $H(x)$-invariant subspace containing $g(x)$. We further claim that $W_d$ is the smallest $H(x)$-invariant subspace containing $g(x)$. In order to see this, suppose on the contrary that there exists a smaller subspace $W \subsetneq W_d$, which contains $g(x)$ and is $H(x)$-invariant. It is an easy exercise to show that $W$ is spanned by at most $d - 1$ eigenvectors of $H(x)$. So that $g(x)$ is orthogonal to at least $n - d + 1$ eigenvectors $H(x)$, which is a contradiction to the fact that $x$ is a regular point of $C^d$. On the other hand, the smallest $H(x)$-invariant subspace containing $g(x)$ has a characterization in terms of $H(x)$ and the subspace $G(x)$, which is the span of $g(x)$:

$$W_d = H(x)^{n-1}G(x) + \cdots + H(x)G(x) + G(x). \tag{4.2}$$

We are not going to provide a proof for this, we refer to [60] for details. Therefore, $x$ is a regular point of $C^d$ if and only if the smallest $H(x)$-invariant subspace containing $g(x)$ is of dimension $d$. It is possible to represent the dimension of the smallest invariant subspace as the rank of a certain matrix. For $d = 0, \ldots, n - 1$, we define the matrices

$$\Omega_d(x) := \begin{bmatrix} H(x)^d g(x) & \cdots & H(x)g(x) & g(x) \end{bmatrix}. \tag{4.3}$$

Then we have $\dim(W_d) = \text{rank} \Omega_{n-1}(x)$. Hence, we proved the following.

**Proposition 1.** For $d = 0, 1, \ldots, n - 1$, the point $x$ is a regular point of $C^d$ if and only if $\text{rank} \Omega_{n-1}(x) = d$.

This proposition provides a very simple test to find the number $d$, for which $x$ is a regular point of $C^d$. We are not merely interested in classification of points $x$. The identification of the critical sets $C^d$ as zero sets of certain functions is also desirable. The approach we introduced above becomes handy in this vein too. First, note that as an immediate consequence of the proposition:

$$C^d = \{ x \in \mathbb{R}^n \mid \text{rank} \Omega_{n-1}(x) \leq d \}. \tag{4.4}$$

Due to the special structure of the matrix $\Omega_{n-1}(x)$, the condition $\text{rank} \Omega_{n-1}(x) \leq d$ can, in fact, be characterized in terms of the determinants involving the matrix $\Omega_d(x)$. We define

$$\Delta_d(x) := \det(\Omega_d(x)^T \Omega_d(x)). \tag{4.5}$$

\(^3\)This is done to avoid pathological situations in the initial disposition. Clearly it is incorrect for almost all (probability density) functions.
CHAPTER 4. MANIFOLD UNWRAPPING USING CRITICAL SURFACES

Then, we have the following proposition.

**Proposition 2.** For \(d = 0, 1, \ldots, n - 1\), we have

\[
C^d = \{x \in \mathbb{R}^n | \Delta_d(x) = 0\}.
\]

**Proof.** Clearly, \(\Delta_d(x) = 0\) if and only if the columns of \(\Omega_d(x)\) are linearly dependent, i.e. \(\text{rank } \Omega_d(x) \leq d\). So, if we can prove that \(\text{rank } \Omega_{n-1}(x) \leq d\) if and only if \(\text{rank } \Omega_d(x) \leq d\), then the result follows from (4.4). If \(\text{rank } \Omega_{n-1}(x) \leq d\), then the necessity of \(\text{rank } \Omega_d(x) \leq d\) is obvious. For the sufficiency, note that since \(\text{rank } \Omega_d(x) \leq d\), the column \(H(x)^dg(x)\) can be written as a linear combination of the remaining columns of \(\Omega_d(x)\). Hence, so does any column of the form \(H(x)^ig(x)\) where \(d \leq i \leq n - 1\). Therefore, \(\text{rank } \Omega_{n-1}(x) \leq d\). 

The inclusion (4.1) together with Proposition 2 suggest a relationship between \(\Delta_d(x)\) and \(\Delta_{d-1}(x)\). The matrices \(\Omega_d(x)\) and \(\Omega_{d-1}(x)\) differ only by a single column

\[
\Omega_d(x) = \begin{bmatrix} H(x)^dg(x) & \Omega_{d-1}(x) \end{bmatrix}.
\]

Introducing, for compactness, \(v_d(x) = H(x)^dg(x)\) and \(O_{d-1} = (I - \Omega_{d-1}(x)(\Omega_{d-1}(x)^T\Omega_{d-1}(x))^{-1}\Omega_{d-1}(x)^T)\):

\[
\Delta_d(x) = \det \begin{bmatrix} v_d(x)^Tv_d(x) & v_d(x)^T\Omega_{d-1}(x) \\ \Omega_{d-1}(x)^Tv_d(x) & \Omega_{d-1}(x)^T\Omega_{d-1}(x) \end{bmatrix}.
\]

Taking Schur complement, we get

\[
\Delta_d(x) = \Delta_{d-1}(x)(v_d(x)^TO_{d-1}v_d(x)).
\]

If we set \(\delta_0(x) := g(x)\) and for \(d\) with \(1 \leq d \leq n - 1\) define

\[
\delta_d(x) = g(x)^TH(x)^dO_{d-1}H(x)^dg(x),
\]

then we get

\[
\Delta_d(x) = \Delta_{d-1}(x)\delta_d(x) \quad (4.6)
\]

The equation (4.6) not only shows the relationship between \(\Delta_d(x)\) and \(\Delta_{d-1}(x)\), but also provides a factorization:

\[
\Delta_d(x) = \prod_{i=0}^d \delta_i(x) \quad (4.7)
\]
We would like to note that the matrix $\Omega_{n-1}(x)$ is a square matrix. Hence, $\Delta_{n-1}(x) = (\det(\Omega_{n-1}(x)))^2$. Therefore, we will simply use $\det(\Omega_{n-1}(x))$ to identify the points on the $(n - 1)$-dimensional critical set $C^{n-1}$.

### 4.2 Curvilinear Coordinates

We are going to use the machinery we developed in Section 4.1.1 to provide a local curvilinear coordinate system to perform manifold unwrapping. The reader should note that providing a system of curvilinear coordinates in $\mathbb{R}^n$ amounts to finding $n$ distinct functions $\{f_1(x), \ldots, f_n(x)\}$ such that the determinant of the Jacobian matrix

$$
\det \left( \frac{\partial (f_1, \ldots, f_n)}{\partial (x_1, \ldots, x_n)} \right) = \begin{vmatrix}
\frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\
\vdots & \ddots & \vdots \\
\frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n}
\end{vmatrix}
$$

is non-zero for all $x \in \mathbb{R}^n$. A trivial example is the Euclidean coordinate system which is given by the functions $f_i(x) = x_i$ for $i = 1, \ldots, n$. Here it is important to note that in $\mathbb{R}^n$, the equation $x_i = 0$ defines a hyperplane, not a line in general. Obviously, there are infinitely many ways of assigning curvilinear coordinates in $\mathbb{R}^n$. However, we are particularly interested in those which arise from critical sets of the probability density function. The set of all $x$ satisfying $f_i(x) = 0$ for some $i$ is, in general, an $(n - 1)$-dimensional hypersurface. Hence, this already gives a clue that a possible factorization of $\det(\Omega_{n-1}(x))$ might provide such a set of functions.

Moreover, if one wants to obtain the $i^{th}$ curvilinear coordinate axis, then one needs to solve the system of equations

$$
f_1(x) = \cdots = f_{i-1}(x) = f_{i+1}(x) = \cdots = f_n(x) = 0. \quad (4.8)
$$

Even though picking certain factors of $\det(\Omega_{n-1}(x))$ to assign curvilinear coordinates seems to be a good choice, it comes with an unexpected drawback, which we would like to discuss in the next example.

**Example 1.** Consider the pdf given by

$$
p(x) \propto e^{-\frac{1}{2}(x_1^2 + (x_1 - x_2)^2)}
$$

where we omitted the scaling factor which would guarantee that the double integral over the whole $\mathbb{R}^2$ gives
CHAPTER 4. MANIFOLD UNWRAPPING USING CRITICAL SURFACES

Figure 4.1: The pdf \( p(x) \propto e^{-\frac{1}{2}(x_1^2 + (x_1 - x_2)^2)} \) and its critical sets.

1. Then the scaled gradient and the Hessian of \( \ln(p(x)) \) are given by

\[
g(x) = \begin{bmatrix} x_2^2 - 2x_1 \\ -2x_2^3 + 2x_1x_2^2 \end{bmatrix} \quad H(x) = \begin{bmatrix} -2 & 2x_2 \\ 2x_2 & 2x_1 - 6x_2^2 \end{bmatrix},
\]

so that

\[
det(\Omega_1(x)) = 2x_2(4x_1^3 - 14x_1^2x_2^2 + 12x_1x_2^4 - 2x_1x_2^2 - 2x_2^6 + x_2^4). \tag{4.9}
\]

Hence, our candidate functions as curvilinear coordinates are

\[
f_1(x) = x_2 \\
f_2(x) = 4x_1^3 - 14x_1^2x_2^2 + 12x_1x_2^4 - 2x_1x_2^2 - 2x_2^6 + x_2^4 \tag{4.10}
\]

The leftmost plot in Figure 4.1 shows the contour lines of \( p(x) \) and the 1-dimensional critical sets. An obvious component of the 1-dimensional critical set is the \( x_2 = 0 \) line, i.e. the second curvilinear coordinate axis due to (4.8). The remaining 3 components are given by \( f_2(x) = 0 \). Hence, (4.10) is certainly not appropriate for assigning coordinates! Moreover, (4.10) is, in fact, an irreducible polynomial, that is, one cannot find two bi-variate polynomials \( q_1(x) \) and \( q_2(x) \) with real coefficients such that \( q_1(x)q_2(x) \) will give (4.10). This can be checked using the algebraic geometry software SINGULAR. Therefore, the remaining three components cannot be represented as the zero sets of distinct bivariate polynomials, the irreducible polynomial (4.10) gives all three at once.

The problem we described in the previous example is, in fact, easy to tackle. The polynomial (4.10) gives us three components, although we need just one. In this particular example, there is a simple way of eliminating the unwanted components. The idea is, to treat (4.10) as a polynomial in \( x_1 \) only, and
solve for \( x_1 \) in terms of \( x_2 \). As the highest power of \( x_1 \) appearing in (4.10) is 3, this can be done efficiently. In fact, MATLAB’s `solve` command can find all three roots. Then, we pick the equation corresponding to the component we would like to use as a curvilinear coordinate axis. In our case, we pick the component shown in the middle in Figure 4.1 to obtain the curvilinear coordinate axes. This approach has an obvious limitation. Because, the celebrated Abel-Ruffini Theorem states that, in general, there is no solution in radicals to polynomial equations of degree five or higher\(^4\) [61]. If \( p(x) = e^{-P(x)} \) for some polynomial \( P(x) \), then \( \text{det}(\Omega_{n-1}(x)) \) is a polynomial too. In the rest of the chapter we will focus on the exponential pdf family.

### 4.2.1 Local Curvilinear Coordinates for Exponential PDFs

**Algorithm 2** Local Curvilinear Coordinates

**In:** \( g(x), H(x) \) for \( f(x - \bar{x}) \) (assuming the mode of interest \( \bar{x} \) is brought to the origin)

**Out:** \( f_1(x) = (f_{11}(x), \ldots, f_{1n}(x)) \), local curvilinear coordinate system around the origin.

1. Calculate \( \text{det}(\Omega_{n-1}(x)) \)
2. Check, if \( \text{det}(\Omega_{n-1}(x)) \) has a trivial factor of the form \( x_i^j \) for some \( i, j \geq 1 \). If so, check if the standard basis vector \( e_i \) is an eigenvector of \( H(0) \) for such an \( i \). If the answer is affirmative, assign \( x_i \) as a coordinate function.
3. Factor out the trivial factors of \( \text{det}(\Omega_{n-1}(x)) \) to obtain a new polynomial \( \hat{P}(x) \).
4. Calculate the components of \( \hat{P}(x) = 0 \) using MATLAB’s `solve` command, or approximate them using SINGULAR’s `hnexpansion` command or other available algorithms. If the total number of components obtained are more than \( n \), then pick those whose normal at the origin (the mode) is an eigenvector of \( H(0) \).

As we discussed above, even if we can factorize \( \text{det}(\Omega_{n-1}(x)) \) into \( n \) distinct factors, they might still not provide the curvilinear coordinate system we are looking for. Moreover, in case the power of some \( x_i \) appearing in one of the factors is very high, then we might not be able to solve for \( x_i \), in terms of other variables to overcome a possible complication similar to the one discussed in Example 1. However, there is an intuitive approach to solve both of these problems at once. One can always approximate the relevant components of \( C^{n-1} \) around a mode. Without loss of generality, suppose that the mode is located at the origin. Then, approximating a component of \( C^{n-1} \) around the origin amounts to calculating the Newton-Puiseux expansion (or variants) of the polynomial \( \text{det}(\Omega_{n-1}(x)) \) at the origin. As the name suggests, this is a very old technique, which is commonly used by algebraic geometers today. The approximation algorithm is originally for \( n = 2 \), and it is implemented in algebraic geometry software like SINGULAR under the name Hamburger-Noether expansion. For arbitrary \( n \), an approximation algorithm is available [62]. The discussion of these approximation algorithms in detail is out of the scope of this work. However, we would like to present the main ideas in this section. Given a multivariate polynomial equation \( P(x) = 0 \), the aim is to find a multivariate power series expansion \( S \) of, lets say, \( x_1 \), in terms of the other variables \( x_2, \ldots, x_n \);

\[
x_1 = S(x_2, \ldots, x_n), \text{ such that } P(S(x_2, \ldots, x_n), x_2, \ldots, x_n) = 0.
\]

\(^4\)Due to this fact, in general, MATLAB’s `solve` command cannot be used to calculate these components.

51
CHAPTER 4. MANIFOLD UNWRAPPING USING CRITICAL SURFACES

Any truncation $S_j$ of the power series $S$ gives an approximation, and hence, the equation $x_1 - S_j(x_2, \ldots, x_n) = 0$ gives an approximation of a component of $C^{m-1}$. Of course, this can be done for any $x_i$. Moreover, the algorithms mentioned above provide as many power series expansions as the number of components of $C^{m-1}$ passing through the mode (origin) around which we do the approximation. If we have more than $n$ components passing through, like in Example 1, then as a rule of thumb, we choose those components whose normal at the mode is an eigenvector of $H(0)$. This guarantees that we have an orthogonal system of local curvilinear coordinates. These approximation algorithms are recursive. We will present this construction of curvilinear coordinates in an example in the next section, where we will employ it to do local manifold unwrapping around modes. Algorithm 2 describes our local curvilinear coordinates algorithm.

4.3 Manifold Unwrapping

We would like to use the local curvilinear coordinates algorithm of the previous section to do manifold unwrapping. Again, as before, we assume that either the pdf or its gradient is known or estimated from samples. Sample estimators for gradient, Hessian and other derivatives of the pdf are available [63, 64, 65, 66]. Moreover, we suppose that there are finitely many critical points, i.e. $C^0$ has finitely many elements, and that we know the location of all the modes. We will denote the number of modes by $m$ and $\bar{x}_i$ will denote the $i^{th}$ mode. Each mode will be placed at the origin of a local chart with curvilinear coordinates.

Unwrapping starts with mode-based clustering. Each data point is assigned to a mode, therefore to a chart. Second, local curvilinear coordinate systems at each mode are calculated using Algorithm 2. For $m$ modes, this amounts to calculating $mn$ functions. The $j^{th}$ coordinate function for the $i^{th}$ mode is denoted as $f_{ij}(x)$. Then

$$f_i(x - \bar{x}_i) = (f_{i1}(x - \bar{x}_i), \ldots, f_{in}(x - \bar{x}_i))$$

gives the coordinates of $x$ in the curvilinear coordinate system at the $i^{th}$ mode. We present the algorithm in an example.

Example 2. Three Hump Camel Function. The following is a slightly modified version of an optimization test function called the Three Hump Camel (THC) Function

$$P(x) = 6x_1^2 - 9x_1^4 + \frac{27}{6} x_1^6 + 3x_1x_2 + \frac{3}{2}x_2^2.$$  \hspace{1cm} (4.11)

The left most plot in Figure 4.2 shows the contours and the critical sets of the pdf $e^{-P(x)}$. The pdf has three

\footnote{The modes are easy to find by mode based clustering of samples.}
modes located at \((-1,1), (0,0)\) and \((1,-1)\). Moreover, we have
\[
\det(\Omega_1(x)) = 8748x_1^{10} + 10935x_1^9x_2 - 17496x_1^8 \\
- 23328x_1^7x_2 + 11421x_1^6 + 17982x_1^5x_2 \\
+ 1215x_1^4x_2^2 - 2268x_1^4 - 5184x_1^3x_2 \\
- 972x_1^2x_2^2 - 81x_1^2 + 243x_1x_2 + 81x_2^2.
\]

The MATLAB command `solve(det(omega1), x2)` calculates a function which can be used as a coordinate function at any of the modes. According to (4.8), the zero set of this function defines a curvilinear coordinate axis. This axis is plotted in the middle in Figure 4.2. The remaining three coordinate functions (one for each mode) has to be approximated. We approximate these functions using SINGULAR up to orders 2 and 3. The right most plot in Figure 4.2 shows all three local coordinate systems assigned by the local curvilinear coordinates algorithm.

### 4.4 Experiments

We performed a manifold unwrapping experiment using Example 2. We created a random sample from the pdf \(e^{-P(x)}\) where \(P(x)\) is the modified Three Hump Camel function of Example 2. The three clusters are determined using mode-based clustering. These clusters are shown in the right plot of Figure 4.2. Each local coordinate system was calculated in Example 2, which is shown in the left plot of Figure 4.2.

In the first part of the experiment we did manifold unwrapping of the whole data set using the curvilinear coordinate system located at the mode in the middle, which is the origin. The upper plot in Figure 4.3 shows the unwrapped data after this transformation.
In the second part, we transformed each cluster using its native curvilinear coordinate system which is centered at the mode associated with cluster. The transformed clusters are shown in the lower plots of Figure 4.3.

Now, the goal is to repeat the above experiments, but without using the analytical density function and only based on the data drawn from those distributions. In order to achieve a polynomial as the determinant of Omega matrix, we need to have a pdf in the form of exponential of a polynomial resulting in polynomial log-pdf, gradient and Hessian. Fitting a log-polynomial pdf to a given data is a very challenging task without convergence guarantee, which is simplified here by first estimating the pdf using kernel density estimation and then polynomial fitting to the log of the kernel density estimate.

In a work by Chacon and Duong [3], a data-driven kernel density and density derivative estimation method is proposed which incorporates unconstrained bandwidth matrix selectors inspired by ideas from cross-validation (CV), plug-in techniques (PI) and smoothed cross-validation (SCV). This is achieved by utilizing advances in matrix analytic theory which allow mathematically and computationally tractable representations of higher order derivatives of multivariate vector valued functions. The theoretical asymptotic
Figure 4.4: (a): Data drawn from the pdf in 4.1 and kernel density estimate using the unconstrained bandwidth matrix plug-in estimator [3]. (b) estimated pdf in the form of exponential of polynomial using polynomial regression of log of kernel density estimate. The zero level sets of delta is also shown.

properties of squared errors for density estimators using the proposed selectors are studied [67]. Based on the simulation study in [3], the plug-in selector here is chosen so that density estimate represents large deviation from normality such as multimodality and high skewness.

Figure 4.4(a) shows the KDE estimate of the distorted gaussian pdf presented in 4.1. In Figure 4.4(b) the log of kde estimate is used for polynomial regression. The exponential of estimated polynomial is shown in 4.4(b), along with the factors of the determinant of Omega matrix. The log polynomial estimate of the pdf resembles the original analytical pdf in 4.1. The order of the polynomial is chosen by goodness of fit measures in regression such as R-square and mean-square error measures. In Figure 4.5 the factors from the estimated pdf in 4.4(b) are shown separately. Figure 4.6 shows a numerical estimation on the factors of delta which qualify as ridges, with their normals as eigen-vectors of the Hessian at the mode. The estimated curved ridge is slightly misplaced compared to the exact factorization, which is due to the estimation errors.

The same procedure is repeated in Figure 4.7 for the three-hump-camel pdf. The unwrapping using factors of delta using analytical pdf and estimated pdf are shown in Figure 4.8. The actual pdf and estimated pdf show close results which points out the goodness of log-polynomial estimate of pdf on this data drawn from highly skewed multimodal pdf.

4.5 Discussion

In this chapter, we started with the premise that the critical sets of the pdf provide a good skeleton to define local and global manifold structures on. Critical sets are characterized by the gradient and the Hessian of a twice-continuously differentiable pdf, hence their determination may ultimately avoid the global
CHAPTER 4. MANIFOLD UNWRAPPING USING CRITICAL SURFACES

Figure 4.5: factorization of delta from log-polynomial estimation of pdf in distorted Gaussian.

Figure 4.6: An estimation result on delta factorization
density estimation step - as a matter of fact, there have been recent work on statistically effective algorithms for estimating the derivatives of a pdf, as mentioned.

We provided a simple characterization of all critical surfaces that does not require the eigendecomposition of the Hessian as in earlier work that introduced the concept. We further demonstrated that for the family of exponential of polynomials the critical surfaces are found as the zero-level sets of polynomials; hence polynomial factor based curvilinear coordinate assignments that respect underlying critical surfaces became possible. We demonstrated how this realization can be utilized for local chart and global manifold unwrapping in the case of the exponential family.
Figure 4.8: The unwrapping using factors of delta by both analytical and estimated pdfs. The actual pdf and estimated pdf show close results which points out the goodness of log-polynomial estimate of pdf on this data drawn from highly skewed multimodal pdf.
Chapter 5

Efficient Density Derivative Estimation

The method presented in chapter is the result of collaboration with Jonas Myhre from University of Tromso. This work resulted in a journal paper named "Computationally efficient exact calculation of kernel density derivatives"[46].

Machine learning research related to the derivatives of the kernel density estimator has received limited attention compared to the density estimator itself. This is despite of the general consensus that most of the important features of a data distribution, such as modes, curvature or even cluster structure, are characterized by its derivatives. In this chapter we present a computationally efficient algorithm to calculate kernel density estimates and their derivatives for linearly separable kernels, with significant savings especially for high dimensional data and higher order derivatives. It significantly reduces the number of operations (multiplications and derivative evaluations) to calculate the estimates, while keeping results exact (i.e. no approximations are involved). The main idea is that the calculation of multivariate separable kernels and their derivatives, such as the gradient vector and the Hessian matrix involves significant number of redundant operations that can be eliminated using the chain rule. A tree-based algorithm that calculates exact kernel density estimate and derivatives in the most efficient fashion is presented with the particular focus being on optimizing kernel evaluations for individual data pairs. In contrast, most approaches in the literature resort to approximations of functions or downsampling. Overall computational savings of the presented method could be further increased by incorporating such approximations, which aim to reduce the number of pairs of data considered. The theoretical computational complexity of the tree-based and direct methods that perform all multiplications are compared. In experimental results, calculating separable kernels and their derivatives is considered, as well as a measure that evaluates how close a point is to the principal curve of a density, which employs first and second derivatives. These results indicate considerable improvement in computational complexity, hence time over the direct approach.
CHAPTER 5. EFFICIENT DENSITY DERIVATIVE ESTIMATION

5.1 Introduction

Kernel density estimation (KDE) is an established non-parametric approach that is widely used in pattern analysis, computer vision, dimensionality reduction, and clustering [68, 69, 70, 71]. In the KDE literature, methods that tackle overall computational complexity primarily approach from the perspective of reducing pairwise kernel evaluations; examples include Nystrom approximation [72], Fast Gauss Transform [73, 74] and sparse dictionary learning methods [75].

However, in applications such as seeking modes or principal surfaces of the density [2, 44], kernel density derivative estimation (KDDE) contains important information regarding multivariate datasets. An outline of statistical problems which can be solved using estimators of density derivatives can be found in [76]. In addition, when dealing with high dimensional data, features of the density such as local extrema, valleys and ridges, are in many cases more important than the density itself [77].

Despite their usefulness, higher order KDDE have received less attention due to the high computational cost, and the mathematical intractability of bandwidth selection. The approaches have mostly focused on bandwidth selection [3, 78] and asymptotic analysis of error [67], not on computational efficiency. Stemming from fast multipole methods in multibody physics, series approximations has been investigated and shown to reduce computational complexity, but implementation only exists for univariate Gaussian KDDE [79]. To the best of our knowledge, there has not been a study in the literature which reduces the computational complexity for general multivariate kernel density derivative estimation. In Table 5.1 our proposed method is placed in context with recent and/or benchmark KDE and KDDE research papers. We have chosen to include papers about bandwidth issues, even though it is not directly addressed in this chapter. This is simply to illustrate the position of our method in the most common frameworks of KDE and KDDE literature. In contrast, graphical representation of functions and derivatives similar to the presentation made here, and forward and backward flow algorithms have been used in other fields such as in numerical optimization and neural networks [80].

In general, multivariate kernels and their derivatives are implemented by direct computation of the kernel function (matrix) and its derivatives with respect to each dimension. However, we show in this chapter that in the special case of the widely used separable multivariate kernel [81], there exists an algorithmic improvement on the computational complexity. The proposed algorithm presents a fast tree-based implementation of multivariate KDDE up to any derivative order, without any form of downsampling, approximation or incremental error. As it is shown in the rest of the chapter, direct computation has significant redundancies, and is essentially a brute force algorithm.

We start with a quick motivating example with some simplification of notation that will be clarified in Section 5.2. Assume a separable four dimensional kernel is evaluated at the point \( \mathbf{x} = [x_1, x_2, x_3, x_4]^T \). A gradient element would be \( \nabla_4 K(\mathbf{x}) = k(x_1)k(x_2)k(x_3)k'(x_4) \), and similarly two Hessian elements...
\[ \nabla^2_{14} K(x) = k(x_1)k(x_2)k(x_3)k''(x_4) \quad \text{and} \quad \nabla^2_{34} K(x) = k(x_1)k(x_2)k'(x_3)k'(x_4). \]

One can observe that carrying out calculation of these quantities would yield the computation of \( k(x_1)k(x_2)k(x_3) \) and \( k(x_1)k(x_2) \) redundant with three and two repetitions, respectively. This redundancy grows as the number of dimensions and the derivative order increases, which leads to significant room for optimizing the algorithm.

Linearly separable kernels (e.g. Gaussian, Epanechnikov) are widely used in the kernel machines field and the main contribution of this work stems from this repetitive structure that emerges in differentiation of linearly separable kernels due to the chain rule. This consistent product structure among linearly separable multivariate kernels and derivatives allows us to design an algorithm that eliminates multiplication redundancies, as well as repeated univariate kernel and derivative calculations that appear in a direct implementation. The scope of this work does not include bandwidth selection for KDE/KDDE, and the proposed algorithm applies to any kernel scaling.

The rest of the chapter is as follows. In 5.1.1 a survey on similar works on computational efficiency of kernel density and derivative estimation is presented. In Section 5.2, we present the basics of the separable kernel function. Section 5.3 presents the tree-based efficient algorithm and its computational analysis. Section 5.4 presents experiments showing the computational efficiency of the algorithm on ridge score calculation as well as curvilinear projections of samples to the local ridges of a KDE [82], which uses the kernel, its gradient, and its Hessian. The last section contains experiments illustrating the computational savings of the new method compared to the direct implementation.

### 5.1.1 Related work

As mentioned above, we have found no similar work; that reduces computational complexity without any effect on accuracy or without restrictions on dimension or degree of derivatives. We present a short overview of the most relevant, and well-proven methods for speeding up the kernel density estimator. In Section 5.2.3 we show how any bandwidth matrix can be used by our method, and thus in principle any matrix can be used. Because of this we will not focus on methods regarding bandwidth selection in this section.

The most commonly used efficient method for univariate KDE is the binning approximation \([83, 84, 85]\), in which the interval is divided to \( G(< n) \) grid points, where \( n \) is the number of sample points. The complexity of KDE based on the grid counts is reduced from \( O(nm) \) to \( O(Gm) \) where \( m \) is the number of evaluation points. Further savings can also be obtained by considering the repetitive kernel evaluations, and using fast Fourier transform (FFT) for discrete convolution \([83]\).

In higher dimensions, one of the most used in kernel machine setting, is perhaps the Nyström method, \([72]\). It is a well established method that performs a reduced-rank approximation on the Gram (kernel) matrix by randomly choosing a subset of row and columns of the matrix and eigendecomposition of the smaller system. The method which essentially is an out-of-sample extension considerably speeds up
multivariate kernel machines while controlling the margin of error. It has proven to work well for a wide range of kernel based algorithms, including kernel density estimation.

When using Gaussian kernels for KDE, the summation of $n$ Gaussians evaluated at $m$ targets causes quadratic computational complexity of the summation as $O(mn)$, which prevents scalability of the algorithm in practice. The fast Gauss transform [73] tackles this problem by using Hermite polynomials to approximate the exponential term of the Gaussian kernel, and reducing the complexity to linear $O(m + n)$. This can provide significant speedup, but the cost of extension to higher dimensional problems grows exponentially with dimension.

The improved fast Gauss transform [74] deals with this issue by a new efficient estimate of the sum of Gaussians in higher dimensions. In this method $n$ sample points are divided into $k$ clusters, where the maximum distance of a point to a cluster center is minimized. This is equal to solving the $k$-center problem which is NP-hard and has a greedy solution [86]. This partitioning and the use of a new multivariate Taylor expansion dramatically reduces the cost of fast Gauss transform, but at the cost of reduced accuracy.

In the same setting of using Gaussian, or other RKHS kernels, kernel herding [87] can be used to draw random samples from the set. The kernel herding algorithm uses a negative autocorrelation sampling technique that moves away from oversampled regions, yielding it better than iid samples. It is also shown to converge faster than iid samples in kernel feature space.

There also exists an online sparsification process used in kernel recursive least squares, [75], that reduces computational complexity. The evaluation of a kernel density at a point can be considered as a sum of weights and a set of feature vectors. If this is relaxed and only an approximate sum is used, some of the feature vectors of the reproducing Hilbert space can be considered linearly independent, and thus evaluations can be saved.

For fast computation of univariate kernel derivative estimators, there exist a method [79] which is inspired by the fast multipole methods used in fast Gauss transform. Unlike binning methods which lack a precise control over accuracy, this approximation algorithm reduces computational complexity from $O(nm)$ to $O(n + m)$, while the desired accuracy can be specified by user. The algorithm is based on Taylor series expansion of Gaussian kernels and truncation of later terms while setting the desired error.

5.2 Kernel Functions

In this section, we present the framework for computing multivariate kernels and their derivatives, that are separable after an affine transformation of the input argument. First, the base kernel functions are covered, followed by kernel density, gradient and Hessian estimators. Then, the structure of the new efficient implementation is described, and its computational cost in a $d$-dimensional space for derivatives of order $p$ is presented and compared with direct implementation. We also note that the kernel framework presented
Table 5.1: The proposed method in contrast with recent or benchmark KDE and KDDE research papers.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Focus</th>
<th>Dimensionality</th>
<th>Article</th>
</tr>
</thead>
<tbody>
<tr>
<td>KDE</td>
<td>bandwidth</td>
<td>univariate</td>
<td>[88], [89], [90]</td>
</tr>
<tr>
<td>KDE</td>
<td>bandwidth</td>
<td>multivariate</td>
<td>[91], [92]</td>
</tr>
<tr>
<td>KDE</td>
<td>efficiency</td>
<td>univariate</td>
<td>[83], [84], [85]</td>
</tr>
<tr>
<td>KDE</td>
<td>efficiency</td>
<td>multivariate</td>
<td>[72], [73], [74], [75]</td>
</tr>
<tr>
<td>KDDE</td>
<td>bandwidth</td>
<td>univariate</td>
<td>[78]</td>
</tr>
<tr>
<td>KDDE</td>
<td>bandwidth</td>
<td>multivariate</td>
<td>[3], [67]</td>
</tr>
<tr>
<td>KDDE</td>
<td>efficiency</td>
<td>univariate</td>
<td>[79]</td>
</tr>
<tr>
<td>KDDE</td>
<td>efficiency</td>
<td>multivariate</td>
<td>proposed method</td>
</tr>
</tbody>
</table>

below is designed for use in KDE and KDDE, as the truncation in Section 5.2.1 will yield a kernel that is not necessarily positive semi-definite.

5.2.1 Univariate Kernels

Let \( k(x) : \mathbb{R} \rightarrow \mathbb{R} \) be an arbitrary zero-mean, unit-width (with some appropriate definition of width) one-dimensional kernel, with \( k^{(j)}(x) \) as its \( j \)-th derivative. A univariate bounding box is introduced to truncate the kernels for finite support, which helps reduce computational complexity in applications with many samples. The truncated adaptation is given as

\[
k^{(j)}_{t}(x; [x_l, x_h]^T) = k^{(j)}(x)b(x; [x_l, x_h]^T)\alpha([x_l, x_h]^T),
\]

where \( b(x; [x_l, x_h]^T) \) is the bounding box expanding from the lower bound \( x_l \) to the upper bound \( x_h \). Notice that \( k^{(0)}(x) \) represents the kernel itself, \( k(x) \). If infinite support kernels are desired, the bounding box is eliminated by setting its extremes to \( \pm \infty \). Here \( \alpha([x_l, x_h]^T) \) is the normalization factor which assures that \( k_t(x; [x_l, x_h]^T) \) integrates to one, and

\[
b(x; [x_l, x_h]^T) = \begin{cases} 
1, & \text{if } x_l \leq x \leq x_h \\
0, & \text{otherwise}
\end{cases}
\]

The normalization factor, \( \alpha([x_l, x_h]^T) \), is found as in (5.3) to satisfy (5.4). This essentially corrects the effect of truncation by computing the probability of the kernel tails, and accounting for it.

\[
\alpha([x_l, x_h]^T) = [k^{(-1)}(x_h) - k^{(-1)}(x_l)]^{-1},
\]

\[
\int_{x_l}^{x_h} k_t(x; [x_l, x_h]^T)\alpha([x_l, x_h]^T)dx = 1,
\]
where \( k^{(-1)}(x) \) returns the value of the first order integral or the cumulative distribution function of the kernel at \( x \).

### 5.2.2 Separable Multivariate Kernels with derivatives

The multivariate separable and isotropic kernel \( K_{I}(x) : \mathbb{R}^{d} \to \mathbb{R} \) can be written as

\[
K_{I}(x) = \prod_{l=1}^{d} k(x_{l}), \quad (5.5)
\]

where \( x_{l} \in \mathbb{R} \) represents the \( l \)-th component of \( x = (x_{1}, ..., x_{d})^{T} \). Here \( K_{I}(x) \) is a scalar value obtained from the product of all univariate kernels \( k(x_{l}) \). Notice that truncation subscript and bounding box parameter vector can hereafter be eliminated, as the kernels can have finite or infinite support. Also, the equations in what follows are only valid for \( x \in \text{Support} \{ K_{I}(.) \} \); and values are zero outside the support. The first derivatives are obtained using the operator \( \nabla_{c} = \partial / \partial x_{c} \) with \( c = 1, ..., d \), and are

\[
\nabla_{c} K_{I}(x) = k^{(1)}(x_{c}) \prod_{\substack{l=1 \\ \forall l \neq c}}^{d} k(x_{l}), \quad (5.6)
\]

where \( x_{c} \in \mathbb{R} \). Here \( \nabla K_{I}(x) \) represents the row vector of first derivatives or gradient. The second order derivatives are obtained with \( \nabla^{2}_{rc} = \partial^{2} / \partial x_{r} \partial x_{c} \) where \( r, c = 1, ..., d \), and are

\[
\nabla^{2}_{rc} K_{I}(x) = \delta_{rc} k^{(2)}(x_{c}) \prod_{\substack{l=1 \\ \forall l \neq c}}^{d} k(x_{l}) \\
+ (1 - \delta_{rc}) k^{(1)}(x_{r}) k^{(1)}(x_{c}) \prod_{\substack{l=1 \\ \forall l \neq r \\ l \neq c}}^{d} k(x_{l}), \quad (5.7)
\]

where \( x_{r} \in \mathbb{R} \), and \( \delta_{rc} \) is the Kronecker delta function. The first term of (5.7), is the derivative of (5.6) with respect to \( x_{r} \) when \( r = c \), and the second term is the derivative when \( r \neq c \). The \( d \times d \) Hessian matrix \( \nabla^{2} K_{I}(x) \) contains all of these second order partial derivatives.

### 5.2.3 Scaled Multivariate Kernels

Thus far, the scale (inverse-width) has been an identity matrix for the kernels and their derivatives. It is a well known fact that the critical factor in performance of kernel density estimators and kernel machines is the appropriate selection of width and orientation [89]. The most general anisotropic kernel employs a data dependent affine scale matrix \( S \succ 0 \), see e.g. [81]. A particular restriction is to use a diagonal matrix
S. In the class of isotropic kernels, $S$ consists of a positive scalar multiple of identity matrix. The scaled multivariate kernel $K_S(x)$ is given by

$$K_S(x) = \beta(S) K(Sx). \quad (5.8)$$

Starting with the last equation, in the rest of the chapter, we will omit the identity scale matrix subscript from the base multivariate kernel in the notation. The normalization factor $\beta(S)$ is derived based on the requirement that $K_S(x)$ must integrate to one.

$$\int_{\mathbb{R}^d} \beta(S) K(Sx) dx = 1. \quad (5.9)$$

Defining $\tilde{x} = Sx$ results in $d\tilde{x} = |S| \, dx$. Since $\int_{\mathbb{R}^d} K(x) dx = 1$, we have $\beta(S) = |S|$, and

$$K_S(x) = |S| K(Sx). \quad (5.10)$$

The gradient and Hessian of $K_S(x)$ are given as

$$\nabla K_S(x) = |S| \nabla K(Sx) S. \quad (5.11)$$

$$\nabla^2 K_S(x) = |S| S^T \nabla^2 K(Sx) S. \quad (5.12)$$

### 5.2.4 Kernel Densities

Given a set of $n$ samples $(X_1, ..., X_n)$, the kernel estimator of the underlying density is defined as

$$f(x) = \sum_{i=1}^{n} w_i K_{S_i}(x - X_i), \quad (5.13)$$

where $w_i$ is the weight of each sample point in the aggregation. Typically (if the samples are assumed/known to be independent identically distributed), $w_i = 1/n, \forall i$. In the most general case, each data point $X_i$ has a distinct scaling, and that is denoted by the use of $K_{S_i}(x - X_i)$. Consequently, the gradient and the Hessian of the density is given as

$$g(x) = \sum_{i=1}^{n} w_i \nabla K_{S_i}(x - X_i). \quad (5.14)$$

$$H(x) = \sum_{i=1}^{n} w_i \nabla^2 K_{S_i}(x - X_i). \quad (5.15)$$

Higher order derivatives are structured as supersymmetric higher order tensors due to repeated outer products of $\nabla$ (a rank-1 higher-order differential operator). Linearity ensures the same form is preserved, i.e.
summation over samples and differentiation are interchangeable in order of operation.

5.3 Efficient Implementation

In this section we present the algorithm for exact and efficient implementation of the multivariate affine separable kernel and derivative calculations presented in Section 5.2. As mentioned in the introduction, implementing (5.5) to (5.7) in a brute force fashion results in calculating the same set of kernel, derivative, and multiplication operations repeatedly, which is clearly not computationally efficient.

The primary factor in designing the algorithm without any redundant computations is the appropriate selection of the data structure. To this end, we propose a computation tree, $T$, which calculates separable kernels and their derivatives up to order $p$ simultaneously, and without redundancy. $T$ is a rooted layered tree with layers $l = 1, \ldots, d$, which make its depth equal to the number of dimensions. Each node of the tree keeps track of the derivative order, while each edge contains the weight of $k^{(j)}(x_l)$ or simply $k^{(j)}_l$ for $j \in \{0, \ldots, p\}$. Traversing the tree layer by layer on a path, similar to a breadth first search [93], is equivalent to calculating and storing the cumulative multiplications one dimension at a time. Depending on the node values, a full traversal would correspond to either of the products in (5.5) to (5.7). Each node represents a single computation stage of multiplying the weight of its arriving edge by the value recorded at its predecessor. Consequently, each node contains the information of the corresponding path starting from the root.

An example of the tree growth when $d = 3$ and $p = 2$ is shown in Figure 5.1 on the right. Excluding the root, there are three layers in the tree, and nodes store values of 0, 1, 2 representing kernel, first derivative, and second derivative, respectively. Since the maximum derivative order is $p$, the number of children of each node is constrained by the sum of the derivative orders of the path leading to that node. In other words, the summation of the node values in each path could not exceed $p$. In Figure 5.1 on the left, the branches which violate this constraint are shown as red. Finally, the nodes of the unbalanced tree, Figure 5.1 on the right, contains all the required terms of equations (5.5) to (5.7).

5.3.1 Tree Construction Algorithm

In order to outline the tree construction algorithm, the required function definitions and notations representing its nodes and edges are introduced in this section.

**Definition 3.** Let $T(E, V)$ be a tree with $n_v$ nodes and $n_v - 1$ edges. Each node in $V$ takes the index $l$ as the level indicator, and $m$ as the counter of nodes of the same level, while $m$ itself is a function of $l$. The value of a node $v_{lm}$ is equal to the derivative order $j$, and is shown by $|v_{lm}|$. Except for the root, corresponding to each node $v_{lm}$, there exists an edge with the weight of $v_{lm}^W = k^{(j)}(x_l)$. Finally, $v_{lm}^P$ indicates the parent of $v_{lm}$, as shown in Figure 5.2.
Figure 5.1: Full graph (left): Nodes violating the constraint are marked with bold red. Pruned graph (right): Nodes which meet the maximum derivative constraint are marked with bold blue.

**Definition 4.** Function $a : V \rightarrow \mathbb{R}$ defined on each node, returns the cumulative multiplications recursively:

$$a(v_{lm}) = a(v_{lm}^P) v_{lm}^W. \quad (5.16)$$

**Definition 5.** Function $b : V \rightarrow \mathbb{R}^+$ defined on each node, outputs the cumulative summation of derivative orders recursively:

$$b(v_{lm}) = b(v_{lm}^P) + |v_{lm}|. \quad (5.17)$$

**Fact 1.** Subject to the constraint $\sum_l |v_{lm}| \leq p$ on the growth of $T$, the number of nodes in the $l$-th layer is equal to $\binom{l+p}{p}$, making the set of nodes as $V = \{v_{lm} \mid l = 1, \ldots, d; m = 1, \ldots, \binom{l+p}{p}\}$.

**Fact 2.** The number of tree leaves and consequently the number of output terms is $\binom{d+p}{p}$. 
Fact 1, which is proved in the Appendix 5.6, immediately results in Fact 2 by setting \( l = d \). Due to the derivative order constraint, when \( \sum_l |v_{lm}| = p \), the node \( v_{lm} \) can only have one child followed by a branch of single-children-nodes all with \( |v_{lm}| = 0 \). Expanding the nodes by valid numbers of children significantly minimizes memory allocation, and allows for using this algorithm in high dimensions and high derivative orders. In Figure 5.1, on the right, the blue nodes show the level where the maximum derivative order is reached. From a memory management point of view, an advantage of the algorithm is that all the information of the path leading to each node is encoded in that node. Thus, for every new layer, \( l + 1 \), only the data on \( l \)-th layer is needed, and the rest of the memory used for the tree can be deallocated. In Algorithm 3 the graph construction algorithm is presented in full.

![Figure 5.2: Node-edge relationship in the tree](image)

**Algorithm 3 Kernel-Derivatives-Efficient\((d, p)\)**

1: Initialize \( a(\text{Root}) = 1 \), \( b(\text{Root}) = 0 \)
2: for \( l \): 1 to \( d \) do
3:     for \( m \): 1 to \( \binom{l+p}{p} \) do
4:         \( b(v_{lm}) = b(v^P_{lm}) + |v_{lm}| \)
5:     if \( b(v_{lm}) = 0 \) then
6:         Expand \( v_{lm} \) by \( p + 1 \) nodes.
7:         Set \( v_{lm} \) as parent of all the new nodes.
8:     else
9:         Calculate the kernel multiplication \( a(v_{lm}) = a(v^P_{lm}) v^W_{lm} \).
10:     end if
11:     if \( b(v_{lm}) = 1 \) then
12:         Expand \( v_{lm} \) by \( p \) nodes with values 0 to \( p - b(v_{lm}) \) and repeat steps 7 and 9.
13:     else if \( b(v_{lm}) = 2 \) then
14:         Expand \( v_{lm} \) by \( p - 1 \) nodes with values 0 to \( p - b(v_{lm}) \) and repeat steps 7 and 9.
15:     else if \( b(v_{lm}) = p \) then
16:         Expand \( v_{lm} \) by 1 new node with \( v_{lm} \) as parent and value 0.
17:     end if
18: end if
19: end for
20: end for
5.3.2 Computational complexity

Given one dimensional kernels and derivatives \( k^{(j)}(x_l) \) as inputs, according to equations (5.5) to (5.7), the only operations for calculating the multivariate kernels and their order \( p \) derivatives are multiplications. The rest of this section focuses on counting the number of multiplications in the proposed efficient algorithm and comparing it with the direct method of simply calculating all products in the separable kernels.

**Fact 3.** The number of multiplications in the efficient algorithm is \( \sum_{l=1}^{d} \binom{l+p}{p} \).

**Proof.** Based on the Algorithm 1, all computational stages occur at the nodes. Therefore, the number of operations is equal to the total number of the tree nodes, which is \( \sum_{l=1}^{d} \binom{l+p}{p} \) according to Fact 1.

**Fact 4.** The number of multiplications in the direct algorithm is \( (d-1) \binom{d+p}{p} \).

**Proof.** Direct implementation can be thought of as our graph based approach, except for the fact that the number of nodes in each layer is constant and equal to \( \binom{d+p}{p} \), according to Fact 2. Since there are \( d \) layers, \( d - 1 \) edges connect the nodes in a sequence, independent from top and bottom nodes, similar to a two-dimensional array. Therefore, the total number of multiplications will be \( (d-1) \binom{d+p}{p} \).

Here, \( (d-1) \binom{d+p}{p} \) can approximately be written as \( \sum_{l=1}^{d} \binom{l+p}{p} \). Comparing this form with \( \sum_{l=1}^{d} \binom{l+p}{p} \) highlights that that later is a significantly smaller number, hence proves the algorithmic improvement.

5.4 Experimental Results

We demonstrate the speedup achieved of the proposed algorithm on synthetically generated data from the Normal density with a range of dimensions and number of data points. In addition we show that the algorithm works in a practical setting, by tracing principal curves, [2] of a collection of data sets from the UCI machine learning repository [94]. In the experiments all samples of data points are evaluated, and therefore \( n \) represents both number of data points and evaluation points. The multivariate normal data is scaled using both isotropic and full anisotropic scale matrices \( S \) with the parameterization shown in Appendix 5.7. The experiments are designed to emphasize the advantage of the proposed method over the brute force direct method in terms of number of calculations (multiplications in separable kernels) and running time. The proposed fast algorithm was prototyped and programmed in MATLAB.

5.4.1 Speedup on Synthetic Data

For the first set of experiments we fix the number of sample points \( n = 1 \) to only investigate the effect of increasing dimension \( d \) on the two methods. Figure (5.3(a)) and Figure (5.3(b)) compare the number of multiplications for gradient and Hessian, respectively. As can be seen from the plots, the proposed
method outperforms the direct method significantly as the derivative order increases. The markers show the corresponding results of the theoretical formula derived in the Facts (3) and (4), which exactly match the implementation results.

The second set of experiments on synthetic data is shown in Figure (5.4), which compares running times of the proposed and direct methods on sample sizes of $n = 100$, $n = 150$, and $n = 200$. The plot on the right in which dimensions range from $d = 1$ to $d = 25$ shows the zoomed-in version of the curves on the left with more details. One can notice that in lower dimensions the proposed method is slightly slower than the direct method. The reason is due to the fact that function calls and branching operations needed for running the proposed method creates cpu overhead that slows down the execution on small data sets. However, as the dimension increase, redundant calculations in the direct method makes it grow faster than the efficient implementation. Another point to mention is that by increasing number of sample point, $n$, the threshold where the proposed method outperforms the direct method decreases. This results in better performance of the proposed algorithm over the direct method for every dimensions $d$ as $n$ grows asymptotically.

### 5.4.2 Speedup on Ridge Score

One motivation for calculating gradient and Hessian of the density estimate is mentioned in [95]. The paper presents a method that uses locally defined principal curves to trace the data. In a $d$-dimensional space, a point $x$ is on the $d'$-dimensional principal manifold iff. the gradient $g(x)$ is orthogonal to at least
Figure 5.4: Comparing running times of the proposed method and the direct method on sample sizes of 100, 150, and 200. (a) dimension range of 1 to 100. (b) zoomed-in version of (a) showing the details of lower dimensions. The markers point to dimensions after which the proposed method is faster than the direct method. It can be seen that as the number of samples increases, the implementation of the proposed method acts superior to the direct method.

\[ \mathbf{H}(x) \] eigenvectors of the Hessian \( \mathbf{H}(x) \). Let \( \mathbf{q}_i(x) \) and \( \lambda_i(x) \) be the \( i \)-th eigenvector and eigenvalue pair of the Hessian of \( f(x) \), sorted in a way that \( \lambda_1 \geq \cdots \geq \lambda_{d'} \geq \lambda_{d'+1} \geq \cdots \geq \lambda_d \). The \( d' \) leading eigenvectors form a tangential space \( \mathbf{H}_\parallel \), while the smallest \( d-d' \) smallest eigenvectors span the orthogonal subspace \( \mathbf{H}_\perp(x) = \text{span}\{\mathbf{q}_i(x)\} \), where \( \mathbf{g}(x)^T \mathbf{q}_i(x) = 0 \) for \( i = (d'+1), \ldots, d \). Therefore, if \( x \) is on the principal curve, then \( \mathbf{g}(x) \) is collinear with \( \mathbf{H}_\perp \), such that the following ridge score is equal to zero.

\[
\zeta(x) = \frac{\mathbf{g}(x)^T \mathbf{H}_\perp \mathbf{g}(x)}{||\mathbf{g}(x)||^2 ||\mathbf{H}(x)\mathbf{g}(x)||}.
\] (5.18)

Table 5.2 presents the computational time of calculating \( f(x), \mathbf{g}(x), \mathbf{H}(x) \) for a \( d+1 \) sized, \( \mathcal{N}(\mathbf{0}, \mathbf{I}_{d\times d}) \), random sample using anisotropic kernels. The same functions as well as ridge score calculations are presented in Table 5.3 using isotropic kernels for one single data point. As we can observe from both tables, eliminating the operational redundancies of the direct method has a significant effect in reducing the running time, especially for higher dimensions. In Table 5.2 the running time of calculating full scale matrices are also added to the computation time.

### 5.4.3 Speedup on Principal Curves Projection

In this section we test the algorithm from Section 5.4.2 on a selection of real datasets from the UCI machine learning repository, [94], using 4th order Runge Kutta solvers [2]. The dimension of the datasets
Table 5.2: Total computation times for calculating kernel matrix, gradient, and Hessian with varying dimension. In this experiment Gaussian anisotropic kernels are used.

<table>
<thead>
<tr>
<th>Dimension</th>
<th># of data points</th>
<th>Direct (sec)</th>
<th>Proposed (sec)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>0.005</td>
<td>0.005</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>0.01</td>
<td>0.013</td>
<td>0.77</td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>0.25</td>
<td>0.136</td>
<td>1.8</td>
</tr>
<tr>
<td>50</td>
<td>51</td>
<td>3.22</td>
<td>1.117</td>
<td>2.9</td>
</tr>
<tr>
<td>100</td>
<td>101</td>
<td>53.63</td>
<td>15.432</td>
<td>3.5</td>
</tr>
<tr>
<td>250</td>
<td>251</td>
<td>2700.16</td>
<td>750.306</td>
<td>3.6</td>
</tr>
</tbody>
</table>

Table 5.3: Ridge score and kernel matrix computation times with varying dimension and isotropic kernel for one single data point.

<table>
<thead>
<tr>
<th>Dimension / # of data</th>
<th>Ridge score time (sec)</th>
<th>Kernel matrix time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Direct (sec)</td>
<td>Proposed (sec)</td>
</tr>
<tr>
<td>5 / 1</td>
<td>0.0007</td>
<td>0.0009</td>
</tr>
<tr>
<td>25 / 1</td>
<td>0.0035</td>
<td>0.0045</td>
</tr>
<tr>
<td>50 / 1</td>
<td>0.0119</td>
<td>0.0122</td>
</tr>
<tr>
<td>100 / 1</td>
<td>0.0499</td>
<td>0.0408</td>
</tr>
<tr>
<td>250 / 1</td>
<td>0.3159</td>
<td>0.2391</td>
</tr>
<tr>
<td>500 / 1</td>
<td>1.5123</td>
<td>1.0063</td>
</tr>
<tr>
<td>800 / 1</td>
<td>5.0368</td>
<td>2.4922</td>
</tr>
<tr>
<td>1000 / 1</td>
<td>9.5334</td>
<td>4.2797</td>
</tr>
</tbody>
</table>

varies from 9 to 500, while the sample sizes are generally small. Results are shown in Table 5.4. We note that the speedup ratio is consistent with respect to the earlier results and thus showing that the implementation is efficient when used in a practical setting as well.

If we look at the actual computation times, we see that the number of data points are a much bigger influence than the dimension. The reason for this is simply that the prototype code is implemented in MATLAB and cannot be vectorized sample-wise, leading to a slight mismatch between algorithm and platform. For this illustration it is of less importance as we see that the speedup ratios are consistent. A C++ version is currently being developed to resolve this issue.

Table 5.4: Principal curve projections for a selection of UCI data sets.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Dimension</th>
<th># of data points</th>
<th>Direct time (sec)</th>
<th>Proposed time (sec)</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancer</td>
<td>9</td>
<td>699</td>
<td>16211</td>
<td>25344</td>
<td>0.64</td>
</tr>
<tr>
<td>Wine</td>
<td>13</td>
<td>178</td>
<td>595</td>
<td>908</td>
<td>0.66</td>
</tr>
<tr>
<td>Cholesterol</td>
<td>21</td>
<td>264</td>
<td>208</td>
<td>271</td>
<td>0.77</td>
</tr>
<tr>
<td>Thyroid</td>
<td>21</td>
<td>300</td>
<td>334</td>
<td>454</td>
<td>0.74</td>
</tr>
<tr>
<td>Ovarian</td>
<td>100</td>
<td>216</td>
<td>2330</td>
<td>1812</td>
<td>1.29</td>
</tr>
<tr>
<td>USPS</td>
<td>256</td>
<td>50</td>
<td>882</td>
<td>571</td>
<td>1.54</td>
</tr>
<tr>
<td>Madelon</td>
<td>500</td>
<td>30</td>
<td>2342</td>
<td>1336</td>
<td>1.75</td>
</tr>
</tbody>
</table>
5.4.4 Speedup as a Function of Number of Evaluation Points

In theory each pairwise kernel computation is completely independent of other pairs, so one should expect linear change in computation time when adding more data with fixed dimension. In Figure 5.5 we see that this is in fact not the case. The computational time of the efficient algorithm is growing slower as a function of number of data points than the direct method. However, after $d = 200$ the time taken is slightly higher than the linear time expected from the theory. This latter effect is due to overhead time spent on memory allocation, which the direct method naturally suffers more from as more memory is needed to store the full tree. Overall, the efficient method significantly outperforms the direct method in increasing proportions as the number of samples increase.

![Graph](image)

Figure 5.5: Calculation time for generating first and second derivatives of kernel from randomly drawn $\mathcal{N}(0, I_{200 \times 200})$ and increasing number of data points, using both direct (green) and proposed (blue) implementation indicating significant reduction in complexity and the rate of increase for time versus number of points.

5.5 Conclusion

We presented a parametrization of linearly separable kernels using an affine input space transformation. This parametrization allowed us to develop a computationally efficient graph-based method to evaluate high dimensional kernels and their derivatives up to any order, with zero error. The possibility of using infinite or finite support kernels exists in this formulation, and although we presented results using Gaussian kernels, other kernels such as generalized Gaussians can be used. Another practical kernel type is
CHAPTER 5. EFFICIENT DENSITY DERIVATIVE ESTIMATION

Epanechnikov, which has finite support by design and desirable geometrical properties for Kernel Density Estimation [81].

5.6 Proof 1

Proof of Fact 1. Subject to the constraint \( \sum |v_{lm}| \leq p \) on the growth of \( T \), the number of nodes in the \( l \)-th layer is equal to \( \binom{l+1}{p} \). The proof of this is based on mathematical induction for all \( l = 1, \ldots, d \). Let the number of nodes in each layer be \( n_l \).

Base case. When \( l = 1 \), the first dimension contains all the derivative orders \( j = 0, \ldots, p \) which makes \( n_1 = p + 1 \). The nodes with index \( j \) at the first layer are expanded by \( p + 1 - j \) child nodes in the second layer, and therefore \( n_2 = (p + 1) + p + \cdots + 2 + 1 = \binom{2+p}{2} \) which supports the statement for \( l = 2 \).

Induction step. Let \( l' \) be given as the layer number, and suppose the hypothesis is true for \( l = l' \). Using Pascal’s rule about binomial coefficients, \( n_{l'+1} \) can be written as

\[
\binom{l' + 1 + p}{p} = \binom{l' + p}{p} + \binom{l' + p}{p - 1}
\]

where the left term is the number of nodes at the layer \( l' + 1 \). Looking back at the tree structure and Figure 5.1 as an example, it can be observed that the tree has a self-repeating pattern. At the new layer \( l' + 1 \), the nodes are divided into two parts. The upper part is originally expanded from \( v_{11} \), and it exactly repeats the nodes of previous layer. The reason is that the starting node of this branch is the node containing \( j = 0 \) at the first layer, which does not change the maximum derivative order constraint. As a result, the branch can be shifted one layer back to the root, and therefore this recursive nature propagates to the whole tree. In (5.19), this branch is represented by the term \( \binom{l' + p}{p} \) which is equal to \( n_{l'} \), according to the induction hypothesis.

The lower part of the tree is expanded from all other nodes of the first layer expect \( v_{11} \). One can observe that these branches are identical to those of a tree with the same number of layers but with the maximum derivative order of \( p - 1 \). Again the reasoning is based on the maximum derivative constraint. Since at the first layer the node value is nonzero, the nodes have one less derivative order for the valid expansion. Based on the induction hypothesis, the number of nodes in the layer \( l' + 1 \) subject to the constraint \( \sum |v_{lm}| \leq (p - 1) \), will be \( \binom{l' + 1 + (p-1)}{p-1} \) or \( \binom{l' + p}{p-1} \), and equal to the second term of the the right side of (5.19). Thus the induction hypothesis is proved for \( l = l' + 1 \), and the statement is generally true.

5.7 Proof 2

In the case of multivariate separable Gaussian kernels, the exponent has the Mahalanobis distance form. Therefore, decomposing the inverse of covariance matrix (precision matrix) gives the matrix that scales and
orients the input data relative to the desired kernel bandwidth.

\[ \Sigma_i^{-1} = S_i^T S_i. \]  \hspace{1cm} (5.20)

In order to extract \( S_i \), the eigendecomposition of \( C_i \) is utilized

\[ \Sigma_i = Q_i \Lambda_i Q_i^T, \]  \hspace{1cm} (5.21)

where the eigenvectors and eigenvalues of \( C_i \) are the columns and diagonal elements of orthogonal \( Q_i \) and diagonal \( \Lambda_i \), respectively. From (5.20) and (5.21), the scale matrix \( S_i \) is written as follows

\[ S_i = \Lambda_i^{-1/2} Q_i^T. \]  \hspace{1cm} (5.22)

Using this formation, \( S_i \) is the bandwidth matrix affiliated with \( X_i \). Similarly, a Cholesky factorization of \( C_i = R_i^T R_i \) would yield \( S_i = R_i \) as an option.
Chapter 6

Conclusion

In this dissertation, original manifold unwrapping and nonlinear dimensionality reduction techniques are presented with the goal of unfolding by determining a curvilinear coordinate system that is coherent with the underlying ridge (principal curve) structure. These unwrapping methods enable us to quantify latent variables describing high dimensional data that is concentrated around low dimensional manifolds, to visualize high dimensional data for exploratory analysis, and to track training and modeling algorithms by feature space visualization.

In the first method, we define the ridges of the density explicitly based on the gradient and Hessian eigenvectors (illustrated on kernel density estimates), and propose a ridge-based curvilinear coordinate assignment procedure that is consistent with Hessian eigenvector tensor fields locally. We provide experimental results with both synthetic and real data, including MNIST handwritten digits and Frey Faces. These results demonstrate that, relative to competing methods that do not explicitly aim to maintain local geometric structure in charts anchored to modes of the probability density function, the proposed method preserves local manifold structure better (in a qualitative fashion, via a smoother mapping from Cartesian to curvilinear coordinates).

In the second method, we provide a new characterization of the critical surfaces of a pdf using a matrix sequence that is constructed using the powers of the Hessian matrix and the gradient vector. This alternative characterization eliminates the need to perform eigendecomposition on the Hessian altogether, and one-dimensional critical sets (which include ridges) are identified as the zero-level-set of the determinant of a particular element of this matrix sequence. Specifically for the exponential family with polynomial bases, this characterization allows us to convert the problem of manifold unwrapping to a problem of polynomial factorization, for which there exists a rich numerical analysis literature that future machine learning researchers can draw from. This method offers a curvilinear coordinate system in the form of polynomials, therefore, relative to the ridge-based method, the resulting coordinates are warped in a polynomial fashion along each ridge; a distortion that can be fixed if desired, by using the inverse of each polynomial factor (which will
necessarily be invertible to be a legitimate curvilinear coordinate system). We demonstrated that, with the polynomial factorization based chart unfolding around each mode, manifolds can be locally unwrapped around each mode, but also using chart-neighboring relationships, relative to a reference mode anywhere in the data space. The latter procedure forms a global unwrapping atlas that consists of stitched local charts for each mode-based cluster. Unwrapping results based on known and estimated probability density functions in the exponential family demonstrate that coordinates obtained from estimated density ridges approximate coordinates obtained from true ridges accurately.

6.0.1 Future Work

One interesting direction for future work is a continuation of the idea presented in Chapter 4, by implementing a method based on exponential family density estimates. Currently, this density estimation is performed based on regressing a polynomial on log of the kernel density estimate of the data (an approximate minimum KL divergence exponential family model that approximates a KDE model). Specific exponential family model fitting and polynomial-series approximation procedures designed for high dimensional data will lead to better results in situations where data is limited and has extremely high dimensionality. For instance, one could use a maximum-entropy approach to fitting smooth exponential modals to high dimensional data with desired bases used as constraints.

Consider $N$ independent and identically distributed observations $X = \{X_n \in \mathbb{R}^d \mid i = 1, \ldots, N\}$ such that for all $k \in K$

$$E_p[\phi_k(x)] = \alpha_k \quad \text{(6.1)}$$

Note that $\alpha_k$’s can be empirically estimated moments of $x$. The expectations $E_p[\phi_k(x)]$ under distribution $p$ are matched to expectations under empirical distributions. This problem is generally under-determined; there are many distributions consistent with these observations. Therefore, in max-ent principle, one selects the model that has maximum entropy defined as

$$H(p) = E[-\log p(x)] = -\int p(x) \log(p(x)) dx \quad \text{(6.2)}$$

to obtain model $p^*$, subject to these equality constraints (6.1) [96]. More formally:

$$p^* = \arg \max_{p \in P} H(p) \quad \text{subject to} \quad E_p[\phi_k(x)] = \alpha_k \quad \text{for all } k \in K \quad \text{(6.3)}$$

By calculus of variations it can be shown that the solution $p^*$ takes the form

$$p^\theta(x) \propto \exp \sum_{k \in K} \theta_k \phi_k(x), \quad \text{(6.4)}$$
where $\theta \in \mathbb{R}^d$ represents a parametrization of distribution in exponential family format. Based on the maximum entropy interpretation, $\theta$’s are the Lagrange multipliers associated with constrained optimization problem in 6.3. The intuition behind using maximum entropy distribution is to chose the density with maximum uncertainty (as flat as possible) while remaining faithful to data by means of specified empirical moments $\alpha_k$. This estimation method, along with reliable numerical polynomial approximation and factorization methods, is expected to further improve the results and applicability of the proposed manifold unwrapping method in Chapter 4.
Appendix A

Graph-Constrained Inverse Covariance Estimation: Application in Traumatic Brain Injury Detection

This work on Traumatic Brain Injury (TBI) detection problem using sparsity constraints on Gaussian graphical models was the result of my internship at Brigham and Women’s Hospital at Harvard medical school. In this work, I designed the algorithm and implemented it along with all the experiments. Sylvain provided the data and the specific graph used for the modeling, as well as validating the results and giving ideas on experiment designs. This work result in a conference paper, ”Sparse model learning for high dimensional diffusion MRI data in traumatic brain injury” [48], and a journal paper ”Subject-Specific Abnormal Region Detection in Traumatic Brain Injury Using Sparse Model Selection on High Dimensional Diffusion Data Medical Image Analysis”, which is accepted at journal of Medical Image Analysis.

Abnormalities of Diffusion Tensor Imaging (DTI) data in neuroimaging studies are traditionally detected at the population level by directly comparing regions of interest across patients and healthy controls, and verifying why different in these regions. The assumption behind these types of analyses is that conditions in patients have homogeneous spatial patterns of abnormalities. However, in diseases such as traumatic brain injury (TBI) or multiple sclerosis, a common spatial pattern of injury is unlikely to occur, violating the main hypothesis of standard population studies.

With an estimated 10 million people world-wide affected annually by a TBI, the burden that this condition imposes on society makes it a considerable public health problem [97, 98, 99]. Importantly, a significant percentage (10-15%) of individuals diagnosed with mild TBI experience persistent post-concussive symptoms (PPCS), which may lead to long-term disabilities [100]. Symptoms range from physical, such as headache; cognitive, such as difficulty concentrating; and emotional/behavioral, such as irritability
APPENDIX A. GRAPH-CONSTRAINED INVERSE COVARION MATRIX ESTIMATION: APPLICATION IN TRAUMATIC BRAIN INJURY

and impulsivity. In the majority of these chronic cases, there is no radiological evidence of injury from conventional magnetic resonance imaging (MRI) or computed tomography (CT), and little is known about the pathophysiology underlying the injury. Thus establishing radiological evidence of brain injury is a critical first step towards the proper diagnosis and monitoring of TBI, and may lead to establishing neuroimaging biomarkers to help predict recovery versus PPCS and to assess better the impact of therapies on the injured brain.

Recent methods for injury detection in mild TBI patients have been developed by estimating a model of "healthy" DTI features and testing whether brain regions have outside-of-normal-range values for a particular subject’s brain (see [101] for a nice overview). Typically, each region is modeled by the mean and standard deviation of the DTI feature of interest over all healthy individuals, and individual TBI subject’s data are z-transformed using these healthy population parameters. Finally regions with a z-score above a given threshold (typically 2 standard deviations) are flagged as abnormal, and statistics such as the number of abnormal regions or the average z-score over the brain are compared between TBI and controls. Methods mostly differ from each other based on how the mean and standard deviation are estimated, and how bias is avoided when testing normal controls that have been used to estimate the “healthy” model parameters [102, 103, 104]. Most methods study one DTI feature at a time (except for [105], which uses four DTI features in a multivariate setting), but none of the current techniques model the inter-dependence of DTI features between neighboring brain regions. Another interesting result from our previous work, suggest that DTI changes are observable in gray matter regions in these patients (potentially related to glial scaring), and thus one should study the full brain as opposed to only white matter in this population [104].

In this chapter, we extend the multiple univariate setting of [104] to a high dimensional Gaussian multivariate model which accounts for inter-region interactions. One of the main challenge we need to overcome is a relatively small number of healthy subjects (in the order of 50) compared to the number of parameters to estimate (in the order of 10,000). Our method thus relies on the estimation of a sparse representation of the region co-dependencies as modeled by a precision matrix.

Although not as thoroughly studied in diffusion MRI, sparse representation of inter-region interactions is the subject of much research in fMRI. Extracted networks capture higher order dependencies among variables, and therefore are effective in exploring local interactions of brain regions [106]. Unfortunately, the estimation of these functional connectivities from subject to subject can be difficult to do robustly and recent research has focused on imposing a prior to the sparse representation. One such example is the work of [107], which uses structurally-weighted least absolute shrinkage and selection operator (LASSO) regression, and models the directional functional interactions of resting state fMRI data based on structural connectivity constraints encoded by 358 cortical landmarks derived from DTI data [108].

Our work is similar in spirit, with some key differences. Here, we use DTI to evaluate subtle tissue changes in TBI patients by detection of outliers compared to a model of normal brain derived from 145 brain
regions of 34 healthy subjects. A feature vector containing fractional anisotropy (FA) measures over 145 brain regions represents each subject. We model the distribution of these features in the healthy subjects as a multi-dimensional Gaussian distribution as represented by a precision matrix. Our method relies on the theorem that conditional independence of two variables given others is equivalent to setting the corresponding precision matrix entity to zero \[109\]. We leverage this theorem by imposing a brain neighborhood prior graph on the structure of the precision matrix, reducing the number of parameters to estimate by favoring interactions of proximal regions and ignoring the interactions of regions which are far away from each other.\footnote{Note that a sparse precision matrix does not imply a sparse covariance matrix; therefore distant brain regions are not assumed to be independent with this constraint – only conditionally independent as shown in Eq. (A.1).}

The multi-dimensional Gaussian model is further regularized by an \(L_1\) sparsity constraint and estimated using the graphical LASSO \[110\].

### A.1 Gaussian Graphical Models

Let \(x = [X_1, X_2, \ldots, X_d]\) be a \(d\)-dimensional random vector so that it has a multivariate Gaussian distribution \(x \sim \mathcal{N}(\mu, \Sigma)\), with \(d\)-dimensional mean vector \(\mu\), and a \(d \times d\) covariance matrix \(\Sigma\). In a Gaussian graphical model, an unweighted undirected graph with adjacency matrix \(G\), can be used to represent the conditional dependence structure between the individual variables \(X_i\). More specifically, the edge structure of \(G\) can be imposed onto the inverse covariance matrix, also known as the precision matrix, \(\Sigma^{-1} \equiv \Theta = \{\theta_{ij}\}\), and conditional independence between \(X_i\) and \(X_j\) can be expressed as a zero in the corresponding location in \(\Theta\):

\[X_i \perp \!\!\!\perp X_j \Leftrightarrow G_{ij} = 0 \Leftrightarrow \theta_{ij} = 0\]  \hspace{1cm} (A.1)

The proof can be found in \[109\].

One key benefit of this representation is that one can use a priori information to impose a conditional independence structure to the model. This is particularly useful in scenarios where a high dimensional \(\Theta\) needs to be estimated with only a few samples, and expert knowledge about the data set can help guide sparse model learning. By using a graph \(G\) which sets many of the precision matrix elements to zero before the estimation process, we can greatly reduce the number of parameters of the model, and thereby increase the robustness of the optimization.

In addition, we assume global sparsity of the model, and thus add an \(L_1\) penalty term to further regularize the model. Following \[111\], let \(X\) be the \(n \times d\) data matrix representing \(n\) observations, \(S\) be the \(d \times d\) sample covariance matrix, and \(G\) the a priori graph, the maximum a-posteriori (MAP) estimate of \(\Theta\) given \(X\) and \(G\) is:
APPENDIX A. GRAPH-CONSTRAINED INVERSE COVARIANCE ESTIMATION: APPLICATION IN TRAUMATIC BRAIN INJURY

\begin{align}
\hat{\Theta}_{MAP} &= \arg \max_{\Theta} \log p(\Theta|X,G) \\
&= \log \det \Theta - tr(S\Theta) - \rho \|\Theta\|_{L_1} \\
\text{with } \theta_{ij} &= 0 \text{ when } G_{ij} = 0. \tag{A.2}
\end{align}

where $\rho$ is a scalar controlling the $L_1$ norm penalty weight.

The optimization method uses the graphical LASSO algorithm [110], which can elegantly incorporate $G$ into the optimization process.

In the following section, we describe how this graphical model with the addition of an a priori graph can be applied to the problem of estimating a multivariate Gaussian distribution of DTI features in healthy subjects and use this model to detect brain injuries in subjects with mild TBI.

A.2 Application to Injury Detection in TBI

Our driving hypothesis for using graphical models is that brain regions next to each other have similar, or at least highly related, DTI signal in healthy subjects. We thus model these interactions by only considering edges connecting proximal regions in the graph imposed on the precision matrix. If a TBI subject has a region with abnormal signal, having modeled the healthy region-to-region interaction will help us increase our sensitivity to classifying a TBI brain as abnormal, compared to looking at each region independently.

A.2.1 Subjects and Data Acquisition

In this work, we used the data described in [104]. There are $n = 34$ healthy subjects, $p = 11$ TBI patients who reported symptoms (see Table A.2.1 for details), such as headaches, emotional dysregulation and memory impairments at the time of data collection, as well as $m = 11$ normal controls demographically matched to TBIs. The normal controls are separated from healthy subjects for validation purposes. Subjects underwent MRI scanning, including a high resolution diffusion tensor imaging scan and a high resolution structural T1 weighted scan. Each T1 image was segmented using the FreeSurfer software [112], resulting in 176 gray matter (GM), white matter (WM), and cerebrospinal fluid (CSF) sections. CSF sections and sections smaller than 300 $mm^3$ were excluded from the analysis, as these smaller regions led to unstable estimation of mean/std of the DTI measures and many failed to pass normality tests. The remaining 145 sections (83 in GM and 62 in WM) were registered onto the diffusion space using a non-linear diffeomorphic registration algorithm [113]. The average FA was computed in each region for each subject. The outcome of the image processing procedure is a feature vector of the average FA in $d = 145$ brain structures in each
Table A.1: Description of individual mTBI subjects adapted from [104]

<table>
<thead>
<tr>
<th>ID</th>
<th>Age</th>
<th>Gender</th>
<th>Source of Injury</th>
<th>Duration since Injury</th>
<th>Symptoms</th>
</tr>
</thead>
<tbody>
<tr>
<td>TB01</td>
<td>45</td>
<td>F</td>
<td>MVA*</td>
<td>17.0</td>
<td>Cognitive impairment, emotional dysregulation, depression</td>
</tr>
<tr>
<td>TB02</td>
<td>38</td>
<td>M</td>
<td>MVA</td>
<td>106.6</td>
<td>Mild memory impairment, mild executive function impairment, emotional dysregulation</td>
</tr>
<tr>
<td>TB03</td>
<td>44</td>
<td>F</td>
<td>MVA</td>
<td>121.3</td>
<td>Dizziness, exhaustion, periodic limb movements, hypersomnia, depression and anxiety</td>
</tr>
<tr>
<td>TB04</td>
<td>30</td>
<td>M</td>
<td>Sports Injury</td>
<td>2.6</td>
<td>Diplopia, fatigues easily, executive function impairment</td>
</tr>
<tr>
<td>TB05</td>
<td>42</td>
<td>M</td>
<td>MVA</td>
<td>138.0</td>
<td>Cognitive impairment, memory executive function impairment</td>
</tr>
<tr>
<td>TB06</td>
<td>28</td>
<td>M</td>
<td>Assault</td>
<td>27.0</td>
<td>Anxiety, depression, insomnia, ADHD, intrusive thoughts, memory deficits, overeating</td>
</tr>
<tr>
<td>TB07</td>
<td>24</td>
<td>M</td>
<td>Blast Exposure</td>
<td>70.3</td>
<td>Anxiety, panic attacks, hypervigilance, overeating, difficulty concentrating</td>
</tr>
<tr>
<td>TB08</td>
<td>25</td>
<td>M</td>
<td>Blast Exposure</td>
<td>83.3</td>
<td>Depression, memory impairment difficulty w/rapidly presented information</td>
</tr>
<tr>
<td>TB09</td>
<td>29</td>
<td>M</td>
<td>Blast Exposure</td>
<td>51.4</td>
<td>Irritability, nightmares, depression, panic attacks, cognitive and memory impairment</td>
</tr>
<tr>
<td>TB10</td>
<td>24</td>
<td>M</td>
<td>Blast Exposure</td>
<td>55.9</td>
<td>Headaches, memory impairment, problems concentrating, irritability, anxiety, nightmares</td>
</tr>
<tr>
<td>TB11</td>
<td>39</td>
<td>M</td>
<td>Sports Injury</td>
<td>9.5</td>
<td>Facial pain, memory/executive function impaired, emotional dysregulation</td>
</tr>
</tbody>
</table>

*MVA= Motor Vehicle Accident

subject. More details about data acquisition and processing can be found in [104]. In addition, the same procedure was applied for the other standard DTI measures: mean diffusivity (MD), radial diffusivity (RD), and axial diffusivity (AD).

A.2.2 a Priori Graph

Given this data set, we will have to estimate a $145 \times 145$ precision matrix based on 34 observations. In order to reduce the number of parameters to estimate, we chose to design a simple graph that will only consider the relationship between neighboring regions in the brain. Two regions were considered to be neighbors if they were connected in a template FreeSurfer segmentation using 26-connectivity. Our motivation for choosing this graph for TBI stems from the knowledge that nearby regions in healthy subjects will tend to have similar tissue properties and thus similar DTI signal (note that we are not considering tensor orientation). This neighborhood network $G$ is illustrated in Figure A.1. Each brain structure is represented as a node in the graph, and conditional dependence is only considered between regions connected by an edge, whereas all other relationships are ignored. Bold lines in Figure A.1(b) show the subgraph associated with region 1. The adjacency matrix corresponding to the complete neighborhood graph is shown in Figure A.1(c). One can observe a large number of parameters that will be set to 0 in $\Theta$. Note that the conditional independence of
two non-neighboring regions imposed by this graph does not enforce unconditional independence; pairs of regions that are not immediate neighbors are allowed to have correlations.

![Regions IDs in a brain slice](image)

![Illustration of a neighborhood graph corresponding to the brain regions](image)

![Neighborhood graph adjacency matrix for the full brain](image)

Figure A.1: Illustration of the prior graph through 10 brain regions. The vector assigned to region 1 in the adjacency matrix is $[1110110000]$, emphasizing connections of regions 2, 3, 5, 6 and disconnections of regions 4, 7, 8, 9, 10 to region 1. Bold lines in Figure A.1(b) show the subgraph associated with region 1.

**A.2.3 Identifying an Abnormal Brain**

Let $X$ be the $n \times d$ matrix representing the set of $d$ features in $n$ healthy subjects, $Y$ the $m \times d$ matrix capturing the observations in $m$ normal controls, and $Z$ the $p \times d$ matrix representing the set of $p$ TBI
patients. Normal controls are healthy subjects matched to patients demographically, and are separated from the healthy training set \( X \) for validation purposes.

The overall design is to generate a model \((\mu_X, \Theta_X)\) based on the healthy subject data \( X \) and test whether a TBI subject \( i \) is abnormal by measuring the Mahanobis distance of its feature vector \( z_i \) to the model:

\[
d_M(z_i) = \sqrt{(z_i - \mu_X)^T \Theta_X (z_i - \mu_X)}
\] (A.3)

As the Mahalanobis distance follows a \( \chi^2 \) distribution, a threshold for an abnormal brain based on this distance can be theoretically derived (e.g., above the 95th percentile of the expected Mahalanobis distances). However, in our work, we test the discriminatory power of our model by computing the Mahalanobis distances of TBI subjects \((Z)\) and matched controls \((Y)\) and evaluate its classification performance using Receiver Operating Characteristic (ROC) curve analysis.

### A.2.4 Identifying Individual Abnormal Regions

The method we have presented thus far has the ability to identify whether a subject’s imaging profile is overall abnormal. The natural next step is to identify which regions are most affected in this subject and thus provide some information that could potentially be linked to the pathophysiology of the brain injury, or help targeting therapies to particular brain areas. Given \( k \) regions, we propose a greedy forward sorting approach to identify these abnormal regions as follows. Let \( R_u \) be the ordered set of sorted regions from most normal to most abnormal, \( R_u = \{1, \ldots, k\} \) be the set of all regions, and \( d_R \) be the Mahalanobis distance computed by only taking into account the regions in subset \( R \subseteq R_u \). We build \( R_s \) by incrementally adding the region \( r_i \in R_u \setminus R_s \), which minimizes \( d_{R_i} \), where \( R_i = R_s \cup \{r_i\} \). This process is repeated until all regions have been sorted from most normal to most abnormal. The procedure is detailed in Alg. 4

**Algorithm 4** Sorting regions from most normal to most abnormal

1: \( R_u = \{1, \ldots, k\} \)
2: \( R_s = () \)
3: for \( i : 1 \) to \( k \) do
4: \( r_i = \arg \min_{j \in R_u \setminus R_s} \left( d_{R_s \cup \{j\}} \right) \)
5: \( R_s = R_s \cup \{r_i\} \)
6: end for
7: return \( R_s \)

The output of this algorithm is an ordering of regions along with \( k \) Mahalanobis distances, \( d_{R_i} \) of the corresponding subsets of sorted regions. The last step consists of comparing the subject’s sorted \( D_i \)s with the theoretical distribution of the Mahalanobis distance (the \( \chi^2 \) distribution with \( i \) degrees of freedom) and finding the first region after which the subject’s sorted distances exceed the 95th percentile of the \( \chi^2 \) distribution. Let \( F_{\chi^2}(D, l) \) be the cumulative distribution function of the \( \chi^2 \) distribution with \( l \) degrees of freedom.
freedom and $\hat{k} = \arg \max_k (F_{\chi^2}(D_k, k) < 0.95)$. Thanks to our sorting process, the regions that are not in the subset of size $\hat{k}$ will generate increasingly unlikely Mahalanobis distances and can be flagged as abnormal. This thresholding procedure is illustrated in Figure A.2.4.

Figure A.2: Abnormal Region detection based on the $\chi^2$ distribution and our greedy forward sorting of regions. The dashed lines correspond to the 95th percentile of the CDF of the $\chi^2$ distribution. Each of the colored curves represent, for each subject, the accumulated Mahalanobis distance given by the subsets of sorted regions. Regions added after the point at which the subjects Mahalanobis distance curve exceeds the 95% threshold on the CDF of $\chi^2$ distribution are flagged as abnormal.

A.3 Experiments

In order to evaluate the performance of the prior neighborhood graph approach, we tested three different graph structures as follows:

1. The neighborhood prior graph as described in Section A.2.2, with an $L_1$ sparsity constraint.

2. A node-only graph with all off-diagonal elements set to zero in the precision matrix.

3. A fully connected graph evaluating all off-diagonal terms with an $L_1$ sparsity constraint.

We tested the robustness of each model by performing a cross validation procedure as follows. Using a leave-one-out strategy, we generated $n - 1$ models $(\mu_i, \tilde{\Theta}_i)$ from $X$ without the $i$-th element. For each model $(\mu_i, \tilde{\Theta}_i)$, we then calculate $d_{M,X_i}$ for all TBI subjects in $Z$ and for all control subjects in $Y$. In addition, we repeated this procedure for a range of regularization parameter $\rho$ (from $10^{-2}$ to 10) to evaluate the impact of this parameter on performance. Thus, for each $\rho$ we had $n$
sets of “TBI vs. Controls” Mahalanobis distances and were able to compute confidence intervals of various classification performance measures (in our case the area under the receiver operating characteristic curve – AUC).

As described earlier, the maximization of the posterior distribution in (A.2), iteratively minimizes certain edges of the graph in two ways: 1) Data driven, where natural interaction of variables among all samples estimate the edges in the graph or precision matrix elements; 2) Prior model driven, where a predefined graph is imposed to the model which sets certain edges to zero, without iterative learning.

In the following experiments, the performance of the node only graph (diagonal precision matrix) is evaluated to illustrate the importance of multivariate vs. univariate analyses. Graphical LASSO is clearly not needed in this diagonal precision matrix design.

### A.3.1 Node-only Versus Neighborhood Versus Fully-connected Graphs

In Figure A.3, all three graph types are examined. In addition, the evaluation is performed for different $\rho$ values to observe the impact of this regularization parameter on the classifier performance.

Figure A.3(a) compares the 90% confidence intervals (CI) of the AUC($\rho$) functions of 34 cross-validation instances across graph types. The confidence intervals are computed using the functional box plot method [114], and the envelope of the 90% central region is shown in Figure A.3. One can observe that both the neighborhood and the full graph clearly outperform the node-only model. In addition, while these two graphs have comparable average performance over all cross-validation, the neighborhood graph has a tighter 90% confidence interval.

The advantage of the prior graph over a fully-connected graph is even clearer when considering the Bayesian information criterion (BIC), as given by

$$BIC = -2 \ln p(X|\Theta) + j \ln n,$$

where $p(X|\Theta)$ is the maximized value of the likelihood function, $j$ is the number of parameters estimated, and $n$ is the number of training samples. In our case, $j$ represents the number of non-zero values in the estimated precision matrix $\Theta$. BIC is a criterion for model selection among a finite set of models, and balances the goodness of fit ($p(X|\Theta)$) with a penalty term for the number of model parameters. This criterion penalizes models which increase their likelihood by overfitting the data. Using BIC as a model selection criteria, the prior graph model is preferred due to its lower BIC. Figure A.3(c) compares the number of parameters estimated (model order) for the two multivariate models. In Figure A.3(d), one can observe that the neighborhood graph always has a higher AUC than the full graph for the same model complexity.
Figure A.3: (a) 90% CIs of the AUC, as a function of the penalty parameter $\rho$ for the prior graph, fully-connected graph, and node-only graph; (b) 90% CIs of the BIC as a function of the penalty parameter $\rho$ for each model; (c) Number of model parameters for a given penalty weight for prior and fully-connected graphs; (d) AUC as a function of the number of model parameters for the prior and fully-connected graphs. Both models show similar AUC performance, but the prior graph has better model complexity and BIC.

A.3.2 Neighborhood Versus Random Prior Graphs

To check the importance of expert knowledge in model selection, 1000 random graphs were generated so that they have the same number of edges as the prior graph but at uniformly random locations. Figure A.4 compares the AUC and BIC of the neighborhood graph to the 75%, 85% and 95% central regions of the randomly generated graphs. The neighborhood graph has an AUC that is higher than the 95% central region of random graphs. Similarly the BIC is almost always lower for the neighborhood graph than it is for random graphs. This result illustrates that the better performance of the neighborhood prior is not due to overfitting, but because of the selection of an appropriate graphical model. The percentile of the prior graph
performance at various $\rho$ compared to the random graphs distribution is shown in Table (A.2).

![Figure A.4: Comparison of the neighborhood graph model performance (blue dashed line) with 1000 random graphs with the same number of edges.](image)

**Table A.2: Percentile of the prior graph performance on the random graph distribution**

<table>
<thead>
<tr>
<th>Percentile</th>
<th>0.01</th>
<th>0.02</th>
<th>0.1</th>
<th>0.2</th>
<th>0.4</th>
<th>0.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>90% central region</td>
<td>98%</td>
<td>99%</td>
<td>100%</td>
<td>98%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>80% central region</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>70% central region</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**A.3.3 Selecting the Optimal Penalty Parameter $\rho$**

While the above analyses provide valuable information on the quality of the different models under different regularization by the parameter $\rho$, one does need to select a single optimal $\hat{\rho}$ value to estimate the final model. In order to find this optimum, the Mahalanobis distance of each training point to the model mean estimated with the remaining training points is calculated. The optimum $\rho$ minimizes the leave-one-out sum of squared distances, which is $\hat{\rho} = 0.3$ for the prior graph and $\hat{\rho} = 0.38$ for the full graph, as shown in Figure A.5. Table A.3 compares the performance of the three models at optimum values of $\rho$. Once again, the neighborhood prior graph model outperforms both the node-only and full graph priors. Note that the performance of the node-only graph does not depend on the value of $\rho$.

In order to put our results in context with traditional "z-score" approaches [115, 116, 104, 101], we also performed the computation of the AUC of the mean absolute z-score over all regions as a potential measure to distinguish patients from controls.

Z-scores were computed with respect to the mean and standard deviation of FA in each region over $X$, the training set of healthy subjects. As expected, this method does not perform as well as the multivariate models.
APPENDIX A. GRAPH-CONSTRAINED INVERSE COVARIANCE ESTIMATION: APPLICATION IN TRAUMATIC BRAIN INJURY DETECTION

Figure A.5: Selection of the optimum penalty parameter for each model based on minimizing leave-one-out sum of squared Mahalanobis distances to the model mean.

Table A.3: AUC analysis for optimal multivariate models and independent z-scores

<table>
<thead>
<tr>
<th>Measure</th>
<th>Full Graph</th>
<th>Prior Graph</th>
<th>Node-only Graph</th>
<th>mean absolute z-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUC</td>
<td>0.83</td>
<td>0.86</td>
<td>0.69</td>
<td>0.65</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>0.64</td>
<td>0.73</td>
<td>0.73</td>
<td>0.64</td>
</tr>
<tr>
<td>Specificity</td>
<td>0.91</td>
<td>1</td>
<td>0.64</td>
<td>0.64</td>
</tr>
</tbody>
</table>

A.3.4 Investigating Other DTI Measures

The z-score analysis of [104] only found statistically significant differences for FA. Nevertheless, we further tested, using our multivariate method, the other most common DTI measures: Mean Diffusivity (MD), Axial Diffusivity (AD), and Radial Diffusivity (RD). For all experiments, we used the prior graph and the same regularization parameter \( \hat{\rho} = 0.3 \). As in the previous work, only FA reached significance, although we hypothesize that AD could reach significance given a larger sample size (see Table A.4). Consequently, all subsequent analyses focused solely on FA.

Table A.4: p-value of Wilcoxon ranksum tests and AUC for the most common DTI metrics.

<table>
<thead>
<tr>
<th>Measure</th>
<th>p</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>FA</td>
<td>0.016</td>
<td>0.86</td>
</tr>
<tr>
<td>MD</td>
<td>0.168</td>
<td>0.68</td>
</tr>
<tr>
<td>AD</td>
<td>0.088</td>
<td>0.72</td>
</tr>
<tr>
<td>RD</td>
<td>0.265</td>
<td>0.64</td>
</tr>
</tbody>
</table>

A.3.5 Correlations with Behavioral Measures

Similarly to [104], we performed Spearman correlations between the Mahalanobis distance and behavioral measures in BI subjects. The results presented in Table A.5 are very similar to our previous work, with "Digit Symbol", a measure of processing speed, the only behavioral test significantly correlated with
imaging (ρ = -0.62, p = 0.04). Note however that our sample of 11 TBI subjects is quite small, and we expect better correlations with a larger number of subjects.

Table A.5: Spearman correlations between Behavioral measures and Mahalanobis distances based on FA in TBI subjects.

<table>
<thead>
<tr>
<th>Test</th>
<th>Subtest</th>
<th>ρ</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>California Verbal Learning Test II</td>
<td>Trials 1-5</td>
<td>0.27</td>
<td>0.42</td>
</tr>
<tr>
<td></td>
<td>Short Delay Free Recall</td>
<td>-0.57</td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td>Short Delay Cued Recall</td>
<td>-0.37</td>
<td>0.26</td>
</tr>
<tr>
<td></td>
<td>Long Delay Free Recall</td>
<td>-0.35</td>
<td>0.29</td>
</tr>
<tr>
<td></td>
<td>Long Delay Cued Recall</td>
<td>-0.23</td>
<td>0.49</td>
</tr>
<tr>
<td>Processing Speed</td>
<td>Digit Symbol Search</td>
<td>-0.62</td>
<td>0.04*</td>
</tr>
<tr>
<td></td>
<td>Symbol Search</td>
<td>-0.45</td>
<td>0.16</td>
</tr>
<tr>
<td>Digit Span</td>
<td>Digit Span Search</td>
<td>-0.30</td>
<td>0.37</td>
</tr>
<tr>
<td>Trail Making</td>
<td>Trail Making A</td>
<td>-0.02</td>
<td>0.96</td>
</tr>
<tr>
<td></td>
<td>Trail Making B</td>
<td>0.37</td>
<td>0.26</td>
</tr>
<tr>
<td>Controlled Oral Word Association</td>
<td></td>
<td>-0.18</td>
<td>0.60</td>
</tr>
<tr>
<td>STROOP</td>
<td></td>
<td>0.17</td>
<td>0.61</td>
</tr>
</tbody>
</table>

A.3.6 Individual Abnormal Regions Identification

In this section, we present the results of the detection of individual abnormal regions as described in section A.2.4. Each subfigure in Figure A.6 shows a $k \times l$ matrix. The $k$ rows represent the regions and the $l$ columns the individual subjects. The intensity associated with each region in each figure corresponds to its respective amount of "abnormality". We define this abnormality $a_i$ as the following differential

$$a_i = \frac{(D_i - D_{i-1})}{(D_i - D_{i-1})},$$

where $D_i$ is the Mahalanobis distance of the sorted subset of size $i$ and $\tilde{D}_i$ is 95% threshold of the $\chi^2$ distribution, i.e., $F_{\chi^2}(\tilde{D}_i, i) = 0.95$.

We also present the equivalent figures for standard z-score analyses in Figure A.7. In this figure, we present regions with an absolute z-score greater than 2 as well as those greater than 3.58, the threshold corresponding to a Bonferroni correction for the number of regions.

One can observe that both the neighborhood and full graph display similar patterns of detections, whereas the node-only graph displays many false positives. The z-score method show similar results to the node only graph at $|z| > 2$ and a subset of the multivariate techniques at $|z| > 3.58$.

A.4 Discussion

Graphical models are a powerful and flexible technique to impose a structure on a multivariate Gaussian model, which has allowed us to constrain the estimation of a model of DTI signal based on a small
data set of healthy subjects. We applied this method to detect whether subjects who experienced a TBI had an abnormal DTI scan, by measuring the Mahalanobis distance of their data to the model.

We tested three different graph structures, a node-only graph, a fully connected graph, and a neighborhood graph, which only connects regions that are next to each other in the brain. The ability of each method to accurately detect an abnormal brain was tested by classifying TBI vs NC subjects using their Mahalanobis distance to the model under study and computing the corresponding AUC.

Our results demonstrate that multivariate approaches (full and neighborhood graph) clearly outperform the univariate approaches, including standard z-score analyses. While both full and neighborhood graph show similar AUCs, the neighborhood graph leads to a better model when taking into account model complexity, i.e., the number of non-zero elements in the precision matrix. Furthermore, our cross-validation experiments show that although the sample size is small, the results are quite robust as the 90% central region width of the AUC is less than 0.05 for the neighborhood graph. Moreover, the neighborhood model always outperforms randomly generated graph with the same number of edges, indicating that the “expert” knowledge embedded in the graph is indeed a valuable prior to constrain the estimation of the model.

Importantly, the flexibility of graphical models can allow us to test a number of prior graphs, including network-based graph generated from diffusion MRI and/or functional MRI network analyses. This is certainly a topic we plan to further investigate in future work. Another possible extension is the study of DTI (or more generally diffusion MRI) measures in combination, by using a nested precision matrix design, although larger sample sizes would be needed for such complex models.

We have also shown that our multivariate analysis can detect individual regions with abnormal data. In fact, our results show fewer false positives in NCs and more regions detected in mTBIs compared to classical independent z-score analyses. Nevertheless, this aspect of our work was exploratory and further development inspired by factor analysis techniques should be investigated.

Finally, we tested the connection between imaging data and symptomatology, but unfortunately were not able to find strong relationships between behavioral measures and DTI beyond a single measure (Digit Symbol, a measure of processing speed). We believe the main reason is the small sample size, but also the fact that we have only looked at the overall Mahalanobis distance (a global imaging measure). With more data, one could investigate connections between symptoms and subsets of regions corresponding to known networks associated with a particular brain function, which we think will lead to stronger relationships between imaging and behavioral measures.
Figure A.6: Abnormality maps show which regions are affected in each subject for different graphical models. Left: TBI subjects, Right: Normal Controls (NC). Top to Bottom: node-only graph, neighborhood graph, full graph.
Figure A.7: Abnormality maps based on z-scores for two different thresholds. Top: threshold is $|z| > 2$, Bottom: $|z| > 3.58$ (Bonferroni correction for the number of regions)
Bibliography


BIBLIOGRAPHY


[38] E. Bas, “Extracting structural information on manifolds from high dimensional data and connectivity analysis of curvilinear structures in 3d biomedical images,” *Northeastern University, Boston, MA*, 2011.


BIBLIOGRAPHY


BIBLIOGRAPHY


BIBLIOGRAPHY


