Seminar Report

Applying Semi-Supervised Locally Linear Embedding

Department of Statistics
Ludwig-Maximilians-Universität München

By Lisa Wimmer
Under the supervision of Jann Goschenhofer, Ph.D.
Munich, month dayth, 2021

Abstract

Storyline

- Goal: present SS-LLE as a local, graph-based manifold learning method incorporating prior knowledge
- Step 0: define basic mathematical concepts required to understand argumentation (plus notation)
- Step 1: introduce idea of **isometry** (most basic: MDS)
- Step 2: introduce idea of **graph-based** models
  - Achieve non-linearity
  - Common structure: build graph → derive matrix as quadratic form over graph function → derive embedding from eigenvalue problem
  - Most basic: ISOMAP (global, dense, convex)
- Step 3: introduce idea of **locality**
  - Relax global to local isometry
  - Find sparse rather than dense matrices
  - **Laplacian eigenmaps** as concept in which the others can be generalized
    - Define weighting scheme for neighborhood
    - Use Laplacian to derive matrix
    - Solve sparse eigenvalue problem
- Step 4: introduce **local linearity**
  - **LLE**
    - Obtain weights via linear reconstructions
    - Can be shown to approximate graph Laplacian (Belkin & Niyogi (2006))
  - **Hessian LLE**
    - Replace Laplacian by Hessian
- Step 5: introduce **prior knowledge**
  - **SS-LLE**
  - Improve results by pre-specifying some manifold coordinates

Extended Abstract

CHECK AGAINST INTRODUCTION

The goal of this report is to lay out the theoretical framework behind the manifold learning technique of *semi-supervised locally linear embedding (SS-LLE)*, as proposed by Yang et al. (2006), and to put it to implementation for data sampled from manifolds.

Manifold learning in general is concerned with dimensionality reduction. As data analysis employs increasingly high-dimensional data, it is frequently necessary to scale down the number of features to ensure models work as desired and remain interpretable. Dimensionality reduction is justified by the assumption that data observed in D dimensions often truly lie on a d-dimensional manifold (d-manifold), i.e., the d-dimensional generalization of a curved surface, embedded in $\mathbb{R}^D$ (with $d \ll D$). As an example for this phenomenon one might consider image data showing objects in different poses. While images are typically stored in high-dimensional pixel representations, intuitively, it is in fact a very small number of features causing the variation in the data.

A crucial property of d-manifolds embedded in $\mathbb{R}^D$ is their local topological equivalence to $\mathbb{R}^d$. This locally Euclidean behavior is exemplified by a sphere embedded in $\mathbb{R}^3$: although the sphere as a whole is entirely non-linear, on sufficiently small patches of its surface it behaves just like a flat plane in $\mathbb{R}^2$. It is precisely this fact that allows manifold coordinates to be mapped to $\mathbb{R}^d$ in a reduction of dimensionality. The goal is now to learn this mapping in an unsupervised manner. Mapping manifold coordinates to $\mathbb{R}^d$ is in general not equivalent to simple projection onto the d-dimensional coordinate hyperplanes. Instead, models must learn the intrinsic neighborhood structure of the manifold to establish a notion of "nearness" between points. As the sphere example demonstrates, standard distance metrics do not apply (globally) since points on general manifolds are connected by curved paths rather than straight lines.

Some manifold learning techniques try to retain global isometry by mapping pairwise distances to $\mathbb{R}^d$. For instance, *multi-dimensional scaling (MDS)* does so using Euclidean distances, thereby limiting the manifolds it can learn to linear ones, while *ISOMAP* generalizes this approach to non-linear manifolds by applying geodesic distances. Research indicates, however, that for non-convex manifolds it is more effective to preserve local structures only. Otherwise, solutions are prone to shortcuts, i.e., placing points close in $\mathbb{R}^D$ next to each other when they lie in fact on quite different parts of the manifold. In order to avoid such miscalculations, sparse techniques focus on merely local neighborhood structures, modeled through weighted graph representations. The information from these graphs is then condensed into a sparse matrix. Eventually, the principal eigenvectors of this matrix yield the desired low-dimensional coordinates.

One such local graph-based technique is *Laplacian eigenmaps*, a method in whose general framework other techniques may be interpreted. It employs the graph Laplacian and does well in preserving locality, yet is less adept at determining local linearity. This shortcoming is mitigated by *locally linear embedding (LLE)*and its variants. LLE is based on the idea that the embedded manifold may be approximated by locally linear neighborhoods in $\mathbb{R}^D$. Since weights resulting from linear reconstruction are believed to reflect the intrinsic geometry of the manifold, they are topological

properties and as such invariant to rotations, rescalings, and translations. By consequence, these same weights should also reconstruct points in d dimensions. LLE thus maps vicinity structures to the d-dimensional subspace and finds the coordinates that preserve them best. This requires solving the least-squares problem of minimizing reconstruction error and then the sparse eigenvalue problem of minimizing embedding cost. Convexity of both sub-problems ensures globality of local optima. A later proposition, *Hessian LLE (H-LLE)*, may be viewed as an algorithmic variant of LLE and a conceptual variant of Laplacian eigenmaps using the Hessian en lieu of the Laplacian.
These approaches have been shown to successfully retrieve manifold structures in different applications. However, their fully unsupervised functionality offers a drawback: they may fail to find a low-dimensional embedding that has an actual reflection in the real-life setting. Such situations might require the specification of some prelabeled instances. Also, it may simply be the case that manual analysis of a subset of the data is available at low cost.
When prior knowledge is at hand it is only natural to use it. Therefore, Yang et al. (2006) proposed SS-LLE as an extension to LLE that is able to harvest prior information. Both exact and inexact knowledge, the latter regularized with an uncertainty coefficient, are applicable. The information is incorporated in the second step of the algorithm by fixing some of the sought-for coordinates in advance. Perhaps unsurprisingly, Yang et al. (2006) find that careful selection of the prior points to be maximally scattered across the manifold surface works better than random sampling. Indeed, the presented results indicate considerable success of their technique.
It is the aim of this report to (1) reproduce these results, thereby creating an open-source implementation of SS-LLE, and (2) to apply SS-LLE to further manifold learning tasks for a more thorough assessment of its performance.

Contents

List of Symbols

$D \in \mathbb{N}$	Number of observed dimensions
$d \in \mathbb{N}$	Number of dimensions of embedded manifold
$m \in \mathbb{N}$	Number of dimensions of low-dimensional representation
$N \in \mathbb{N}$	Number of observed data points
$\mathcal{M} \subset \mathbb{R}^D$	d-manifold embedded in $\mathbb{R}^D$
$\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N) \in (\mathbb{R}^D)^N$	Observed coordinates of data sampled from $\mathcal{M}$
$\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_N) \in (\mathbb{R}^m)^N$	Learned coordinates of low-dimensional representation of data

List of Figures

List of Tables

1 Introduction

Machine learning problems increasingly employ data of high dimensionality. While a large amount of samples is beneficial to learning, high-dimensional feature spaces, such as in speech recognition or gene processing, pose serious obstacles to the performance and convergence of most algorithms (Cayton, 2005).

High dimensionality. Three aspects strike as particularly problematic: computational operations, interpretation of results, and geometric idiosyncrasies. Computational cost must be considered but is becoming less of an issue with technological evolution (Leist et al., 2009). By contrast, the demand for explainable results (for reasons of, say, safety or ethics) is rather intensified by the advance of complex technology. Alas, interpretation in more than a few dimensions is virtually inaccessible to humans (Doshi-Velez and Kim, 2017). The geometric aspect is often addressed as *curse of dimensionality*, a term subsuming various phenomena of high-dimensional spaces. It is generally not straightforward to infer properties of objects in these spaces as geometric intuition developed in lower dimensions can be misleading. Crucially, the exponential increase of spatial volume induces sparsity. Consequences of this behavior are, among others, a sharp incline in the number of points required to sample the feature space and a loss in meaningfulness of distances. Many learners, however, rely on these concepts[1] and see their functionality deteriorate (Verleysen and Francois, 2005).

Manifold learning. These challenges make the case for *dimensionality reduction*, that is, the endeavor of compressing problem dimensionality to a manageable size. Far from undue simplification, dimensionality reduction is justified by the belief that the latent data-generating process is indeed of much lower dimension than is observed. Consider, for example, image data showing objects in different poses. Such data are typically stored in high-dimensional pixel representations, yet it is reasonable to suppose that variation in these images is in fact caused by a small number of latent features. More formally, the data are assumed to lie on a d-dimensional *manifold* embedded in the D-dimensional observation space, with $d \ll D$ (Cayton, 2005). A crucial property of d-manifolds, i.e., the d-dimensional generalization of a curved surface, embedded in $\mathbb{R}^D$, is their local topological equivalence to $\mathbb{R}^d$ (Ma and Fu, 2011). It is precisely this fact that allows manifold coordinates to be mapped to $\mathbb{R}^d$ in a reduction of dimensionality[2]. The goal is now to learn this mapping in an unsupervised manner (Cayton, 2005). Mapping manifold coordinates to $\mathbb{R}^d$ is in general not equivalent to simple projection onto the d-dimensional coordinate hyperplanes. Instead, models must learn the intrinsic neighborhood structure of the manifold to establish a notion of "nearness" between points. Standard distance metrics do not apply (globally) as points on general manifolds are connected by curved paths rather than straight lines (Ma and Fu, 2011).

Adapt to structure of chapter 3

[1] For instance, consider support vector machines and k-nearest neighbors, both of which rely on distances, or tuning, which often requires extensive sampling of the hyperparameter space.

[2] The most intuitive example of this is probably the representation of the Earth, which is a two-dimensional manifold enclosed in three-dimensional space, on two-dimensional maps.

Local graph-based techniques. Various approaches have been proposed to retrieve points' manifold coordinates. A taxonomy may, for example, be found in van der Maaten et al. (2009). Many rely on spectral techniques, trying to find a matrix representation of the data whose principal eigenvectors are used to span a d-dimensional subspace. One group of spectral methods attempts to retain global isometry by mapping pairwise distances to $\mathbb{R}^d$. Among them, some are based on Euclidean distances and thus confined to learning linear embeddings (such as *principal component analysis (PCA)* or *multi-dimensional scaling (MDS)*). Since linearity is a strong assumption that will not hold for general manifolds, non-linear techniques are more widely applicable (van der Maaten et al., 2009). For example, *Isomap* achieves non-linearity by applying geodesic distances in the MDS setup (Tenenbaum et al., 2000). Research indicates, however, that for non-convex manifolds it is more effective to preserve local structures only. Otherwise, solutions are prone to shortcuts, i.e., placing points close in $\mathbb{R}^D$ next to each other when they lie in fact on quite different parts of the manifold (**?**). In order to avoid such miscalculations, sparse techniques focus on merely local neighborhood structures, modeled through weighted graph representations. The information from these graphs is then condensed into a sparse matrix. Eventually, the principal eigenvectors of this matrix yield the desired low-dimensional coordinates (van der Maaten et al., 2009).

Locally linear embedding. One such local graph-based technique is *locally linear embedding (LLE)*, the unsupervised algorithm SSLLE builds upon (Roweis and Saul, 2000). LLE is based on the idea that the embedded manifold may be approximated by locally linear neighborhoods in $\mathbb{R}^D$. Weights for the resulting graph are obtained by linear reconstruction of points from their neighbors. As these weights are believed to reflect the intrinsic geometry of the manifold, they are topological properties and should as such also reconstruct points in d dimensions. LLE thus maps vicinity structures to the d-dimensional subspace and finds the Euclidean coordinates that preserve them best by means of spectral decomposition (Roweis and Saul, 2000).
Much of the theoretical foundation for LLE has been discussed only in later work. In particular, Belkin and Niyogi (2001) proposed *Laplacian eigenmaps (LEM)*, a method which employs the graph Laplacian, and provided evidence for the fact that, under certain assumptions, LLE may be generalized to the same framework (**?**). A later proposition by Donoho and Grimes (2003), *Hessian LLE (HLLE)*, may be viewed as an algorithmic variant of LLE and a conceptual variant of LEM (using the Hessian en lieu of the Laplacian). The theoretical link between LLE and LEM, centered around the Laplace-Beltrami operator, has recently been found to hold less generally than previously assumed (Wu and Wu, 2018). It still appears beneficial to interpret all methods in this common framework also found by Bengio et al. (2003); a more thorough study of convergence guarantees is left to future research.

Semi-supervised extension. The above approaches have been shown to successfully retrieve manifold structures in different applications (Wu and Wu, 2018). However, their fully unsupervised functionality offers a drawback: they may fail to find a low-dimensional embedding that has an actual reflection in the real-life setting. Such situations might require the specification of some pre-labeled instances. Also, it may simply be the case that some observations already come with labels, or that annotation of a subset of the data is available at low cost (Yang et al., 2006). When

prior knowledge is at hand it is only natural to use it. Therefore, Yang et al. (2006) proposed *semi-supervised locally linear embedding (SSLLE)*, an extension to LLE that is able to harvest prior information.

Outline. Indeed, the presented results indicate considerable success of their technique. It is the aim of this report to (1) reproduce these results, thereby creating an open-source implementation of SSLLE, and (2) to apply SSLLE to further manifold learning tasks for a more thorough assessment of its performance. The rest of the report is organized as follows: chapter 2 provides a mathematical framework where fundamental concepts are briefly introduced; chapter 3 explains the idea of local graph-based manifold learning; chapter 4 presents SSLLE in detail; chapter 5 discusses the results of the conducted experiments; and chapter 6 draws final conclusions.

2 Basic Manifold Theory

2.1 Concepts in Manifold Learning

This chapter introduces the main geometric concepts considered necessary to provide a solid understanding of SS-LLE[3]. It must be noted that everything discussed here is presented through the lens of machine learning, deliberately forsaking the generality inherent to topology. Therefore, assuming features can be represented by coordinates in D-dimensional Euclidean space, all concepts are examined with regard to their meaning in $\mathbb{R}^D$. Dimensionality reduction techniques take the data observed in $\mathbb{R}^D$ to actually lie on a d-dimensional topological space that is not necessarily Euclidean but exhibits some specific properties.

Topological spaces. A *topological space* is constituted by a set T equipped with a *topology* $\mathcal{T}$. A topology is a general way of describing relations between elements in T. Consider a function $\mathcal{T} : T \to 2^T, t \mapsto \mathcal{T}(t)$, which assigns to $t \in T$ a set of subsets of T called *neighborhoods*. For $\mathcal{T}$ to be a topology[4] on T, the following properties must hold (Brown, 2006):

1. If $\mathcal{T}$ is a neighborhood of t, then $t \in \mathcal{T}$.
2. If $\mathcal{T}$ is a subset of T containing a neighborhood of t, then $\mathcal{T}$ is a neighborhood of t.
3. The intersection of two neighborhoods of t is again a neighborhood of t.
4. Any neighborhood $\mathcal{T}$ of t contains a neighborhood $\mathcal{T}'$ of t such that $\mathcal{T}$ is a neighborhood of each element in $\mathcal{T}'$.

Note that, in this general definition, neighborhoods are based on an abstract notion of "nearness". Learning the structure of a topological space effectively boils down to learning neighborhood relations. In Euclidean topological space these are directly based on distance: neighborhoods around t are constructed by ϵ-balls containing all elements within a Euclidean distance of ϵ from t. The resulting topology is also called the *metric topology* (McCleary, 2006).

[3] Obviously, the list of concepts discussed is by no means extensive. Theory is presented much more in detail (and mathematical rigor) in, for example, good book.

[4] Alternative definitions employ open subsets of T, see for example Waldmann (2014).

Topological spaces in general are not accessible via distances (or angles, for that matter) known from Euclidean spaces. The ultimate goal therefore is the interpretation of the data in a space that is again Euclidean, albeit of lower dimensionality, where such concepts are meaningful. The next step is then to study how (potentially highly non-linear) topological spaces relate to $\mathbb{R}^d$.

Homeomorphisms. Consider two topological spaces $(S, \mathcal{T}_S)$, $(T, \mathcal{T}_T)$ (denoted by the respective shorthands S, T from here) and a mapping function $f : S \to T$. If f is bijective and continuous and $f^{-1} : T \to S$ is also continuous, f is called a *homeomorphism* (Brown, 2006). Topological spaces for which such a mapping exists are *homeomorphic* to each other. Any properties of S that T shares when it is homeomorphic to S are referred to as topological properties. Two homeomorphic spaces are thus topologically equivalent (McCleary, 2006).
If there exists a non-negative integer d such that for every s in a topological space S a local neighborhood $U \ni s$, $U \subset S$, is homeomorphic to an open subset of $\mathbb{R}^d$ (sometimes called *parameter space*), S is *locally Euclidean*[5] (Ma and Fu, 2011). In other words, there is a homeomorphism $f : U \to \mathbb{R}^d$ for every element in S. The neighborhoods U are also referred to as *coordinate patches* and the associated maps f are called *coordinate charts* (Cayton, 2005). In local neighborhoods S then behaves like $\mathbb{R}^d$ (Ma and Fu, 2011).

Manifolds. *Manifolds* are now precisely such locally Euclidean topological spaces, with some additional properties. For a topological space $\mathcal{M}$ to be a d-dimensional manifold[6] (also: d-manifold) it must meet the following conditions (Waldmann, 2014):

1. $\mathcal{M}$ is Hausdorff.
2. $\mathcal{M}$ is second-countable.
3. $\mathcal{M}$ is locally homeomorphic to $\mathbb{R}^d$.

The Hausdorff condition is a separation property and ensures that for any two distinct points from $\mathcal{M}$ disjoint neighborhoods can be found (Brown, 2006). Second-countability restricts the manifold's size via the number of open sets it may possess (Waldmann, 2014).

[5] For locally Euclidean topological spaces it is thus meaningful to speak of elements as points.
[6] $\mathcal{M}$ is again a shorthand, omitting the explicit notation of the corresponding topology.

Embeddings. Recall that the data are observed in $\mathbb{R}^D$ but taken to lie on $\mathcal{M}$, locally homeomorphic to $\mathbb{R}^d$. This implies the assumption $\mathcal{M} \subset \mathbb{R}^D$ and $\mathcal{M}$ is said to be *embedded* in the ambient D-dimensional Euclidean space (Cayton, 2005). The associated *embedding* is but a map $f : \mathcal{M} \to \mathbb{R}^D$ whose restriction to $\mathcal{M}$ is a homeomorphism (Brown, 2006), or, more specifically, the canonical inclusion map identifying points on the manifold as particular points of $\mathbb{R}^D$ (Waldmann, 2014). It can be shown that $K = 2d + 1$ is sufficient to create an embedding (Ma and Fu, 2011). Figure 1 shows the so-called *S-curve* manifold embedded in $\mathbb{R}^3$. Clearly, the S-curve as a whole is far from linear, but locally homeomorphic to $\mathbb{R}^2$ and thus intrinsically two-dimensional.

Figure 1: 10,000 points sampled from the S-curve manifold. *Source:* own representation, inspired by implementation in `Python`'s `scikit-learn` library (Pedregosa et al., 2011).

Geodesics. One last aspect remains open and shall be briefly touched here, namely how to handle distances on manifolds where Euclidean metrics are not meaningful. Rather than measuring "shortcuts" between points across $\mathbb{R}^D$ (where, for instance, points in the red upper area of figure 1 would be considered deceptively close to points in the cyan mid area), it makes intuitive sense to constrain distances to the manifold surface. In order to enable the construction of such a metric, manifolds must fulfill two additional properties: *smoothness*[7] and *connectedness*[8] (Ma and Fu, 2011). For smooth, connected manifolds, *geodesic distance* is the length of the shortest curve (*geodesic*) on $\mathcal{M}$ between two points on $\mathcal{M}$. A curve c in $\mathcal{M}$ is a smooth mapping from an open interval $\Lambda \subset \mathbb{R}$ into $\mathcal{M}$. c is parametrized by a point $\lambda \in \Lambda$, such that $c(\lambda) = (c_1(\lambda), ..., c_d(\lambda))^T$ (all $c_j, j = 1, ..., d$, having a sufficient number of continuous derivatives) is a curve in $\mathbb{R}^d$. Component-wise differentiation with respect to λ yields the *velocity* of c in λ, $c'(\lambda) = (c_1'(\lambda), ..., c_d'(\lambda))^T$. The *speed* of c is given by $\|c'(\lambda)\|_2^2$, where $\| \cdot \|^2$ denotes the square norm. Distance along this curve is measured by the arc-length $L(c) = \int_{\boldsymbol{p}}^{\boldsymbol{q}} \|c'(\lambda)\|^2 d\lambda$. Eventually, geodesic distance can be derived as the length of the shortest such curve, out of the set of differentiable curves in $\mathcal{M}$ that connect $\boldsymbol{p}$ and $\boldsymbol{q}$, $\mathcal{C}(\boldsymbol{p}, \boldsymbol{q})$:
$d^{\mathcal{M}}(\boldsymbol{p}, \boldsymbol{q}) = \inf_{c \in \mathcal{C}(\boldsymbol{p}, \boldsymbol{q})} L(c)$ (Ma and Fu, 2011). Intuitively, geodesic distance can be identified with Euclidean distance in Euclidean spaces where shortest curves are just straight lines (Ma and Fu, 2011).

[7] The smoothness property is based on differentiability of coordinate charts and ensures that concepts of curvature, length and angle remain meaningful (Ma and Fu, 2011). A detailed derivation may be found, for example, in Mukherjee (2015).

[8] Connectedness means that no separation $\{U, V\}$ of a manifold $\mathcal{M}$ exists with open, non-empty and disjoint $U, V \subset \mathcal{M}$, $\mathcal{M} = U \cup V$. This may be loosely put as paths linking arbitrary pairs of manifold points (McCleary, 2006).

2.2 Formal Goal of Manifold Learning

Building on the above concepts, the data situation in manifold learning might be summarized as follows: data are observed in $\mathbb{R}^D$ but assumed to be really samples from a d-manifold $\mathcal{M}$ embedded in $\mathbb{R}^D$, meaning they can be analyzed in $\mathbb{R}^d$ if a faithful translation between respective coordinates in $\mathbb{R}^D$ and $\mathbb{R}^d$ is found[9]. The challenge is thus to unravel the manifold in a way that preserves its intrinsic structure to maximum extent (Saul et al., 2006). This goal shall be formalized in a way that will be referenced throughout the remainder of this report and that is inspired by the works of Cayton (2005) and Saul et al. (2006).

Given. Data $\mathcal{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N)$, with $\boldsymbol{x}_i \in \mathbb{R}^D \ \forall i \in \{1, 2, ..., N\}$ and $N, D \in \mathbb{N}$. $\mathcal{X}$ thus consists of N real-valued data vectors observed in D dimensions. The true data-generating process is taken to have dimensionality $d \in \mathbb{N}$, such that $\mathcal{X}$ is in fact a sample from a smooth, connected d-manifold with $\mathcal{X} \sim \mathcal{M} \subset \mathbb{R}^D$. $\mathcal{M}$ may be described by a single coordinate chart[10] $\psi : \mathcal{M} \to \mathbb{R}^d$. For manifold learning methods to yield satisfying results, $\mathcal{M}$ is always assumed to be sampled well by $\mathcal{X}$.

Goal. Find the d-dimensional representation of the data, i.e., compute $\mathcal{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_N)$, where $\boldsymbol{y}_i = \psi(\boldsymbol{x}_i) \in \mathbb{R}^d \ \forall i \in \{1, 2, ..., N\}$. The map ψ itself is not always explicitly retrieved.

Note that, while D is given a priori, the intrinsic dimensionality d is often unknown in real-life applications. Due to the lack of (finite-sample) convergence guarantees for many methods dimensionality estimations may not always be equal to d. $\mathcal{Y}$ as found by manifold learning techniques must therefore be expected to differ from the true coordinates and, in particular, to even have incorrect dimension (Saul et al., 2006). Notwithstanding this potential gap, solutions of the subsequently presented methods will be denoted by $\mathcal{Y} \in (\mathbb{R}^d)^N$ to avoid overly complicated notation.

3 Local Graph-Based Manifold Learning

3.1 Overview and Conceptual Framework

After the goal of manifold learning has been formalized, it shall now be laid out how the problem is approached by LLE as the conceptual parent of SSLLE (the incorporation of prior information is a rather different matter that will be addressed in chapter 4). Much of the theoretical foundation for LLE has been discussed only in later work. In order to provide a more integrated background, explanations will therefore be given in a broader context of local graph-based manifold learning, which also comprises LEM and HLLE. The particular relationship of the three methods shall be made clear along the way.
Local graph-based manifold learning arises from a variety of geometric intuitions and computational implementations. Nonetheless, methods share common structures that allow for interpretation in a more abstract framework (Bengio et al.

[9] It is actually a simplification to assume all data to lie *exactly* on $\mathcal{M}$, but the more general case of data lying *near* $\mathcal{M}$ is rarely considered explicitly.
[10] This is possible for any smooth, compact manifold (Cayton, 2005).

(2003), Bengio et al. (2004)). It should be noted that such a framework might be established from several angles; after all, the different methods attempt to solve the same problem and can thus often be translated into one another.
Figure 2 depicts a schematic overview on the models studied here, representing the specific perspective taken within this report. All of these belong to the realm of *spectral* models. The non-spectral group includes, for instance, techniques based on neural networks and is not discussed here (van der Maaten et al., 2009).

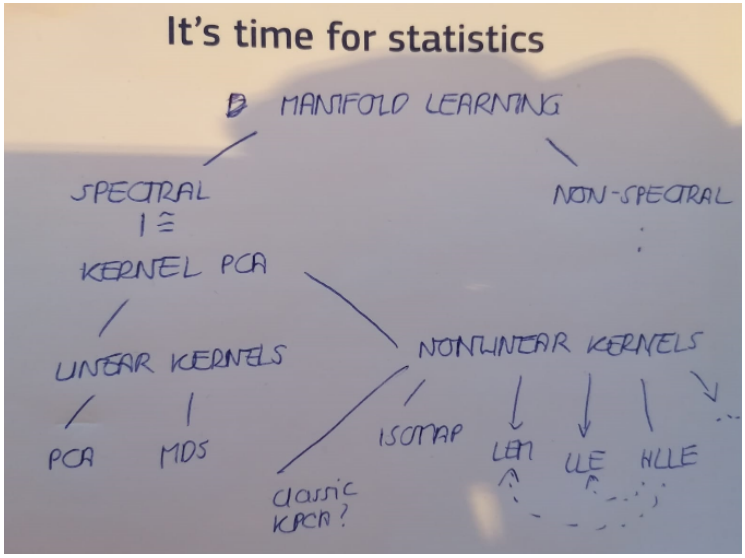

Figure 2: A schematic overview on selected methods of manifold learning (the list is by no means extensive and could arguably be ordered in several alternative ways). *Source:* own representation, inspired by a similar example given in van der Maaten et al. (2009) and re-interpreted with the findings in Bengio et al. (2004).

As visually indicated in figure 2 , this report will sketch the idea behind local graph-based manifold learning in the light of *kernel principal component analysis (kernel PCA)*. Kernel PCA was actually proposed first and later shown to link the other concepts by a unified idea (Ham et al. (2003), It makes for an appealing framework that, firstly, provides a useful general intuition to manifold learning, and, secondly, subsumes the other methods in a way that proves beneficial for the important task of out-of-sample extension (Bengio et al., 2004).

Kernel PCA. Kernel PCA builds upon two fundamental concepts in machine learning: it performs *principal component analysis (PCA)* on data transformed by the *kernel trick*. In principle, it undertakes two subsequent steps. First, features of interest are extracted from the data by kernelization. These are taken to capture the intrinsic data structure and may therefore be understood as an approximation to the latent manifold properties. In the end, they constitute a matrix representation. Second, *principal component analyis (PCA)* finds the principal axes along which these intrinsic properties vary, yielding the desired reduction in dimensionality by preserving the most relevant latent dimensions (Schölkopf et al., 1998).

> **Kernelization.** By kernelization, i.e., mapping the data to a space $\mathcal{F}$ of arbitrarily high dimension, features may be obtained that relate to the input in a

possibly non-trivial way[11]. Crucially, the feature map $\phi : \mathbb{R}^D \to \mathcal{F}$ need not be computed explicitly (this might prove prohibitively expensive). Kernelization instead solely relies on inner products $\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ of the transformed inputs. Employing Mercer's theorem of functional analysis, these inner products may be interpreted as performed by a continuous kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ in some space with Hilbert property. Appropriate choice of κ then allows for the data to be represented by a matrix $K \in \mathbb{R}^{N \times N}, K_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$. This matrix is the numerical data representation derived with respect to their latent properties. (Schölkopf et al., 1998). Precisely how it is computed depends on the choice of the kernel function κ and gives rise to different techniques (Ham et al., 2003).

PCA. PCA is a quite powerful technique by itself. It finds the directions of maximum variance through eigenanalysis of the empirical covariance matrix, yielding the most important axes of inter-feature relations that coincide with the principal eigenvectors of the covariance matrix (for comments on eigenanalysis see chapter A.1 of the appendix). The data are projected into the linear subspace spanned by these d eigenvectors, thereby mapping the observations to a coordinate system given by those linear feature combinations that represent the strongest (co)variability. PCA thus performs an orthogonal input transformation that allows for dimensionality reduction at minimal information loss (Cayton, 2005). In kernel PCA, this eigenanalysis is implicitly performed in the feature space $\mathcal{F}$. Algorithmically, it boils down to diagonalizing the kernel matrix K (Schölkopf et al., 1998).

Figure 3 attempts to visualize the idea of kernel PCA. The original data (*left*) are observed in two dimensions but clearly intrinsically one-dimensional, where the latent manifold feature is expressed by coloring. The kernel trick creates a non-linear map, visualized here as a projection of the intrinsic feature to a third coordinate axis (*middle*). Coercing the data to this dimension as the sole axis of variation yields the desired one-dimensional representation (*right*).

[11] Support vector machines use the kernel trick to achieve linear separability. An intuitive example may be given by data observed in two classes that form concentric circles in $\mathbb{R}^2$. While such data are not linearly separable in two dimensions, they are in three: mapping the classes to different heights enables separation by a horizontal hyperplane. This example also hints at the idea of (spectral) clustering to which kernel PCA is indeed intimately related (Bengio et al., 2004).

Figure 3: Schematic idea of kernel PCA: from data observed in two dimensions, but clearly of intrinsic dimensionality one (*left*), create a mapping to a higher-dimensional feature space (*middle*), reduction of which to its principal axes yields the desired one-dimensional representation (*right*). *Source:* own representation, using a subset of `mlbench`'s noise-free `spirals` data. Note that this is but a schematic depiction where the mid and right representation have not been created by an actual implementation of kernel PCA.

3.2 Aspects of Local Graph-Based Techniques

3.2.1 Non-Linearity and Locality

If kernel PCA sounds like a powerful concept, the crux of course lies in finding an appropriate kernel function. The nature of the feature map applied to the input data determines the kind of mapping that may be learned and serves to distinguish the various techniques. As foreshadowed in figure 2, spectral methods decompose into groups using *linear* and *non-linear* kernels, respectively. This distinction now directly translates to the feature map ϕ. Linear methods suffer from the confinement to finding linear subspaces (van der Maaten et al., 2009). PCA in its standard form can be interpreted as kernel PCA by identifying the kernel function with the covariance function. It thus returns the subspace of greatest variability in the original input features (Ham et al., 2003). The closely related *multi-dimensional scaling (MDS)* approach yields the same result, albeit from a different intuition (Saul et al., 2006).

Non-linearity. As extensively discussed above, $\mathcal{X}$ must often be assumed to lie on a non-linear manifold $\mathcal{M} \subset \mathbb{R}^D$, which is precisely why kernelization is usually performed such that the resulting feature space is related to the input space in a non-linear way (Schölkopf et al., 1998). Conceivably, there is no obvious way to arrive at such a mapping. *Graph-based* models therefore approach the problem from an alternative angle. In fact, they do not even perform kernelization explicitly: they transform the data in a way that can be shown to correspond to applying a (data-dependent) kernel function[12], but the fundamental intuition is a different one. The key idea in graph-based learning is to approximate the manifold by a discretized graph representation. Such a graph may be intuitively imagined as a skeletal model of the manifold surface. The graph properties – essentially an approximation of the latent manifold properties – are described by functionals that vary across methods. Eigenanalysis of the associated matrix representation then leads to the sought-for

[12]The report does not discuss the actual kernel function as their illustrative ability is rather limited. For an explicit formulation of kernels in LLE, LEM and HLLE see for example Bengio et al. (2004) and Weinberger et al. (2004).

low-dimensional subspace coordinates (Saul et al., 2006).

Locality. The second desideratum in general manifold learning is the ability to treat highly non-linear manifolds with sufficiently local focus. Non-convexity means $\mathcal{M}$ is isometric to a non-convex subset of Euclidean space (Donoho and Grimes, 2003). Intuitively, such behavior requires careful tracing of the manifold surface. Local graph-based methods therefore focus on solely local manifold properties, and, in doing so, produce sparse matrix representations (Cayton, 2005). They are frequently contrasted to *Isomap*, one of the earliest and most prominent examples of global manifold learning. Isomap retains pairwise distances between points on the manifold surface as measured along graph edges via geodesic curves[13] (Tenenbaum et al., 2000). Its central assumptions are global isometry and convexity of the parameter space (Tenenbaum et al., 2000). While it yields good results in many applications, Isomap does not sufficiently account for the curvature of strongly non-convex manifolds. In order to avoid this drawback, local methods limit isometry to only hold between neighboring points and relax the parameter space condition to open, connected subspaces (Donoho and Grimes, 2003).

3.2.2 Neighborhood Graphs

Besides the common theoretical framework, LLE, LEM and HLLE also share a general algorithmic structure. It might be summarized as follows (Bengio et al., 2003):

1. Construct a neighborhood graph $\mathcal{G}$ from the observed data.
2. Analyze the graph properties with an appropriate functional and derive a matrix representation M thereof.
3. Find the eigenvalues and associated eigenvectors of M.
4. From the principal (top or bottom) eigenvectors, as determined by the ordered eigenvalues, retrieve the low-dimensional coordinates.

Neighborhoods. All local graph-based methods fundamentally build on neighborhood graph approximations of the manifold surface. A neighborhood of $\boldsymbol{x} \in \mathcal{X}$ is a subset of $\mathcal{X}$ containing another, open subset of $\mathcal{X}$ of which $\boldsymbol{x}$ is an element. Members of the neighborhood are called neighbors of $\boldsymbol{x}$. In metric spaces neighborhoods are defined via distances and therefore translate to open balls around each point (Waldmann, 2014). This distance-based construction locally applies to manifolds as a direct consequence of their local isometry to the Euclidean observation space (Ma and Fu, 2011). There are two principal ways to build a neighborhood around $\boldsymbol{x} \in \mathcal{X}$, both of which usually employ the squared Euclidean norm[14] $\|\cdot\|^2$. Let $\mathcal{N} : \mathcal{X} \to \mathcal{X}^\ell, \boldsymbol{x} \mapsto \mathcal{N}(\boldsymbol{x})$ be a constructor that assigns a set of neighbors to $\boldsymbol{x}$. The first possibility is to restrict the size of the neighborhood to the k points with the smallest distance to $\boldsymbol{x}$, such that $\ell = k$ and $\mathcal{N}_k(\boldsymbol{x}) = \{\boldsymbol{x}_j \in \mathcal{X} : \|\boldsymbol{x} - \boldsymbol{x}_j\|^2 \leq \gamma\}$, with $\gamma \in \mathbb{R}$ being the k-th instance of ordered pairwise distances. Alternatively, the neighborhood may be constructed by collecting all points that have a maximum

[13] It is thus a non-linear variant of MDS, which uses standard Euclidean distances (Tenenbaum et al., 2000).

[14] In principle, alternative metrics are equally applicable, for instance such that measure angles (Belkin and Niyogi, 2004).

distance of $\epsilon \in \mathbb{R}$ to $\boldsymbol{x}$, yielding $\mathcal{N}_\epsilon(\boldsymbol{x}) = \{\boldsymbol{x}_j \in \mathcal{X} : \|\boldsymbol{x} - \boldsymbol{x}_j\|^2 \leq \epsilon\}$ and $\ell = |\mathcal{N}_\epsilon(\boldsymbol{x})|$ (He et al., 2005). Both k and ϵ are hyperparameters that must be specified upfront. Their choice reflects beliefs about the topological structure of $\mathcal{M}$ – smaller neighborhoods corresponding to a higher degree of non-linearity – and may affect performance rather strongly (Sudderth, 2002).

Neighborhood graphs. $\mathcal{M}$ can now be characterized by a *neighborhood graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, still assuming it is sampled well by $\mathcal{X}$. Inputs $\boldsymbol{x} \in \mathcal{X}$ form vertices $\mathcal{V}$ and edges $\mathcal{E}$ indicate neighborhood relations (Belkin and Niyogi, 2001). Each vertex is connected to its k nearest neighbors or all points within ϵ-radius, depending on the neighborhood definition. It is easy to see that k-neighborhoods are an asymmetric notion; for one point to be among another's k nearest neighbors the reverse need not be true. k-neighborhoods therefore lead to directed graphs. Conversely, the ϵ-distance boundary holds in both directions and produces undirected graphs (He et al., 2005). A fictional example for a directed graph is given by figure 4, showing k-neighborhoods for seven data points ($k = 2$).

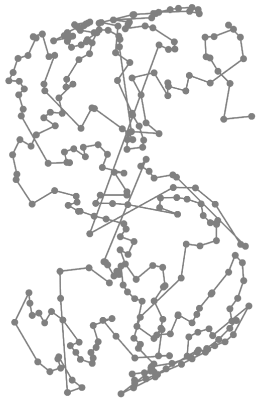

Figure 4: Exemplary neighborhood graph for 300 points sampled from the S-curve manifold. The graph is computed using k-neighborhoods, $k = 1$, in a sequential manner: from a starting point, the respective nearest neighbor is found among the previously unused points. *Source:* own representation.

3.3 Unsupervised Techniques

3.3.1 Laplacian Eigenmaps (LEM)

- Notion of locality
- Laplacian eigenmaps

3.3.2 Locally Linear Embedding (LLE)

- Notion of local linearity
- Approximation of graph Laplacian

3.3.3 Hessian Locally Linear Embedding (HLLE)

- Hessian instead of Laplacian (eigenmaps)
- Hessian instead of LS fit (LLE)

4 Semi-Supervised Locally Linear Embedding (SSLLE)

4.1 Employment of Prior Information

- Why use labels in the first place?
- How will that help?
- How do we even find prior points?
- Exact vs inexact knowledge

test $\mathbf{x}$
Prior points: take minmax approach from sparse MDS (Sparse multidimensional scaling using landmark points Vin de Silva and Joshua B. Tenenbaum 2004). Easy and deterministic after choosing seed value. Instead Euclidean distances, though, take geodesics as estimated in isomap.
can we view the prior info as some kind of active learning? like we choose some points to label in a hopefully cleverish way and then hand them to you (e.g., to look at some pictures instead of all droelf thousand)

4.2 SS-LLE Algorithm

- What is different wrt standard LLE?

4.3 Strengths and Drawbacks

See (van der Maaten et al., 2009) for extensive discussion of manifold learning
Theoretical convergence? (e.g., ISOMAP has this)
Determination of d: actually requires to know d, right? Must be automatically known if prior points are known
Potential shortcoming: what if manifold is not well-sampled? Not a problem with synthetic data, but IRL. But probably problematic with all manifold approaches
This is directly related to the COD – local methods require dense sampling (van der Maaten et al., 2009)
Also: generalization to new points (w/o recomputing everything) neighborhood-preserving propositions

5 Experiment Results

5.1 Experimental Design

5.1.1 Software Implementation

5.1.2 Evaluation Framework

5.2 Replication of Original Results on the Incomplete Tire

5.2.1 Incomplete Tire Data

5.2.2 Choice of Hyperparameter and Prior Point Configuration

5.3 Application on XXX Data

5.3.1 XXX Data

5.3.2 Choice of Hyperparameter and Prior Point Configuration

5.4 Results and Discussion

6 Conclusion

Lorem ipsum

A Appendix

A.1 Eigenanalysis

Eventually, all spectral manifold learning methods boil down to an eigenanalysis of a matrix K believed to capture information about the intrinsic manifold structure. As explained in chapter **??**, PCA finds the principal eigenvectors of empirical covariance, thereby defining a low-dimensional subspace containing most of the data-inherent variability. The very same idea applies when diagonalizing the more general matrix corresponding to the non-linear feature map: the top (or bottom[15]) d eigenvectors of K span a subspace into which the data may be projected under minimal loss of information. More precisely, the representation of $\mathcal{X}$ by the d principal eigenvectors of K incurs is loss-optimal with respect to the least-squares error. Eigenanalysis is thus a very powerful concept with ubiquitous application (Schölkopf et al., 1998).

Eigenvectors and eigenvalues. Formally, eigenanalysis is the decomposition of a square matrix into pairs of *eigenvectors* and *eigenvalues*. Let $A \in \mathbb{R}^{N \times N}$ be a square matrix and $\lambda \in \mathbb{R}$ a scalar value. λ is said to be an eigenvalue to A if there exists $\boldsymbol{v} \in \mathbb{R}^N \setminus \{0\}$ such that $A\boldsymbol{v} = \lambda\boldsymbol{v}$. Then, v is the eigenvector corresponding to the eigenvalue λ, and their tuple is also called an *eigenpair*.

Null spaces. A closely related notion is that of the *null space*, consisting of the vectors that map A to 0 upon multiplication from the right: $\{\boldsymbol{v} \in \mathbb{R}^N : A\boldsymbol{v} = 0\}$. It can be easily seen that the null space consists precisely of those eigenvectors of A that correspond to an eigenvalue of zero and the zero vector itself. For a specific eigenvalue λ of A, the null space of $\lambda I - A$ (with I the N-dimensional identity matrix) constitutes the *eigenspace* of A. (Börm and Mehl, 2012).

Generalized eigenvalue problems. Eigendecomposition of a matrix A can be framed as the solution of a generalized eigenvalue problem. Generalized eigenvalue problems are posed subject to a constraint on a second, also symmetric matrix $B \in \mathbb{R}^{N \times N}$. As the standard eigenvalue problem results immediately from $B = I$, the generalized form subsumes both cases. It is given by

$$A\boldsymbol{V} = B\boldsymbol{V}\boldsymbol{\Lambda},$$

where $\boldsymbol{V} = ([\boldsymbol{v}_1, \boldsymbol{v}_2, ..., \boldsymbol{v}_N]) \in \mathbb{R}^{N \times N}$ is the matrix of eigenvectors of A and $\boldsymbol{\Lambda} = diag([\lambda_1, \lambda_2, ..., \lambda_N]^T) \in \mathbb{R}^{N \times N}$ is the diagonal matrix of the associated eigenvalues. The generalized eigenvalue problem may be stated equivalently as

$$\max_{\boldsymbol{V}} trace(\boldsymbol{V}^T A \boldsymbol{V}), \quad \text{s.t.} \quad \boldsymbol{V}^T B \boldsymbol{V} = I,$$

and translated to the first form with help of the Lagrangian multiplier (Ghojogh et al., 2019). It must be noted that solving eigenvalue problems becomes computationally challenging rather quickly. Therefore, eigendecomposition is performed approximately in virtually all practical applications (Börm and Mehl, 2012).

[15] This differs across methods and shall be made clear later.

B Electronic Appendix

Data, code and figures are provided in electronic form.

References

Belkin, M. and Niyogi, P. (2001). Laplacian eigenmaps and spectral technique for embedding and clustering, *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, p. 585–591.

Belkin, M. and Niyogi, P. (2004). Semi-supervised learning on riemannian manifolds, *Machine Learning* **56**: 209–239.

Bengio, Y., Delalleau, O., Roux, N. L., Paiement, J.-F., Vincent, P. and Ouimet, M. (2004). Learning eigenfunction links spectral embedding and kernel pca, *Neural Computation* **16**: 2197–2219.

Bengio, Y., Paiement, J.-F., Vincent, P., Delalleau, O., Roux, N. L. and Ouimet, M. (2003). Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering, *Proceedings of the 16th International Conference on Neural Information Processing Systems*, MIT Press, p. 177–184.

Brown, R. (2006). *Topology and Groupoids. A Geometric Account of General Topology, Homotopy Types and the Fundamental Groupoid*, 2 edn, Createspace.

Börm, S. and Mehl, C. (2012). *Numerical Methods for Eigenvalue Problems*, De Gruyter.

Cayton, L. (2005). Algorithms for manifold learning, *Technical Report CS2008-0923*, University of California, San Diego (UCSD).

Donoho, D. L. and Grimes, C. (2003). Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data, *Proceedings of the National Academy of Sciences of the United States of America* **100**(10): 5591–5596.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning, *arXiv: Machine Learning* .

Ghojogh, B., Karray, F. and Crowley, M. (2019). Eigenvalue and generalized eigenvalue problems: Tutorial.

Ham, J., Lee, D. D., Mika, S. and Schölkopf, B. (2003). A kernel view of the dimensionality reduction of manifolds, *Technical Report TR-110*, Max-Planck-Institute for Biological Cybernetics.

He, X., Cai, D., Yan, S. and Zhang, H.-J. (2005). Neighborhood preserving embedding, *Proceedings of the Tenth IEEE International Conference on Computer Vision*.

Leist, A., Playne, D. P. and Hawick, K. A. (2009). Exploiting graphical processing units for data-parallel scientific applications, *Concurrency and Computation. Practice and Experience* **21**(18): 2400–2437.

Ma, Y. and Fu, Y. (2011). *Manifold Learning. Theory and Applications*, CRC Press.

McCleary, J. (2006). *A First Course in Topology. Continuity and Dimension*, American Mathematical Society.

Mukherjee, A. (2015). *Differential Topology*, 2 edn, Springer.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research* **12**: 2825–2830.

Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding, *Science* **290**(5500): 2323–2326.

Saul, L. K., Weinberger, K. Q., Sha, F., Ham, J. and Lee, D. D. (2006). Spectral methods for dimensionality reduction, *in* O. Chapelle, B. Scholkopf and A. Zien (eds), *Semi-Supervised Learning*, MIT Press Scholarship Online, chapter 1.

Schölkopf, B., Smola, A. and Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem, *Neural Computation* **10**: 1299–1319.

Sudderth, E. B. (2002). Nonlinear manifold learning part ii 6.454 summary.

Tenenbaum, J. B., de Silva, V. and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction, *Science* **290**(5500): 2319–2322.

van der Maaten, L., Postma, E. and van den Herik, J. (2009). Dimensionality reduction: A comparative review, *Technical Report TiCC TR 2009-005*, Tilburg University.

Verleysen, M. and Francois, D. (2005). The curse of dimensionality in data mining and time series prediction, *in* J. Cabestany, A. Prieto and F. Sandoval (eds), *Computational Intelligence and Bioinspired Systems*, Springer.

Waldmann, S. (2014). *Topology. An Introduction*, Springer.

Weinberger, K. Q., Sha, F. and Saul, L. K. (2004). Learning a kernel matrix for nonlinear dimensionality reduction, *Proceedings of the 21rd International Conference on Machine Learning*.

Wu, H.-T. and Wu, N. (2018). Think globally, fit locally under the manifold setup: Asymptotic analysis of locally linear embedding, *Ann. Stat* **246**(6B): 3805–3837.

Yang, X., Fu, H., Zha, H. and Barlow, J. (2006). Semi-supervised nonlinear dimensionality reduction, *Proceedings of the 23rd International Conference on Machine Learning*.

Declaration of Authorship

I hereby declare that the report submitted is my own unaided work. All direct or indirect sources used are acknowledged as references. I am aware that the Thesis in digital form can be examined for the use of unauthorized aid and in order to determine whether the report as a whole or parts incorporated in it may be deemed as plagiarism. For the comparison of my work with existing sources I agree that it shall be entered in a database where it shall also remain after examination, to enable comparison with future Theses submitted. Further rights of reproduction and usage, however, are not granted here. This paper was not previously presented to another examination board and has not been published.