Pavan K. Turaga · Anuj Srivastava
Editors

Riemannian Computing in Computer Vision

Springer

Riemannian Computing in Computer Vision

Pavan K. Turaga • Anuj Srivastava
Editors

Riemannian Computing in Computer Vision

Springer

Editors
Pavan K. Turaga
Arizona State University
Tempe, AZ, USA

Anuj Srivastava
Florida State University
Tallahassee, FL, USA

ISBN 978-3-319-22956-0 ISBN 978-3-319-22957-7 (eBook)
DOI 10.1007/978-3-319-22957-7

Library of Congress Control Number: 2015954575

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Contents

Chapter 1
Welcome to Riemannian Computing in Computer Vision

Anuj Srivastava and Pavan K. Turaga

Abstract The computer vision community has registered a strong progress over the last few years due to: (1) improved sensor technology, (2) increased computation power, and (3) sophisticated statistical tools. Another important innovation, albeit relatively less visible, has been the involvement of differential geometry in developing vision frameworks. Its importance stems from the fact that despite large sizes of vision data (images and videos), the actual interpretable variability lies on much lower-dimensional manifolds of observation spaces. Additionally, natural constraints in mathematical representations of variables and desired invariances in vision-related problems also lead to inferences on relevant nonlinear manifolds. Riemannian computing in computer vision (RCCV) is the scientific area that integrates tools from Riemannian geometry and statistics to develop theoretical and computational solutions in computer vision. Tools from RCCV has led to important developments in low-level feature extraction, mid-level object characterization, and high-level semantic interpretation of data. In this chapter we provide background material from differential geometry, examples of manifolds commonly encountered in vision applications, and a short summary of past and recent developments in RCCV. We also summarize and categorize contributions of the remaining chapters in this volume.

1.1 Introduction and Motivation

The computer vision community has witnessed a tremendous progress in research and applications over the last decade or so. Besides other factors, this development can be attributed to: (1) improved sensor technology, (2) increased computation power, and (3) sophisticated statistical tools. An important source of innovation

A. Srivastava (✉)
Department of Statistics, Florida State University, Tallahassee, FL, USA
e-mail: anuj@stat.fsu.edu

P.K. Turaga
School of Arts, Media, Engineering & Electrical Engineering,
Arizona State University, Tempe, AZ, USA
e-mail: pturaga@asu.edu

© Springer International Publishing Switzerland 2016
P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_1

has also been the fact that, in addition to statistics, differential geometry has also emerged as a significant source of tools and resources to exploit knowledge and context in computer vision solutions. The usage of geometry in computer vision is relatively recent and sparse. This combination of geometry and statistics is very fundamentally tied to the nature of vision data, characterized by dominance of object structures and their variabilities. Computer vision is mainly about finding structures of interest, often representing physical objects being observed using cameras of different kinds. Since these structures are inherently variable—two images of the same object will exhibit pixel variability—one needs to take this variability into account using statistics. We focus on ideas that borrow basic tools from traditionally disparate branches—statistics and geometry—and blend them together to reach novel frameworks and algorithms in vision.

1.1.1 What Is RCCV?

A large variety of computer vision solutions can be expressed as either inferences under appropriate statistical models or minimizations of certain energy functions. In some cases, the solutions are naturally constrained to lie in subsets that are not linear, i.e., they are neither subspaces nor affine subspaces (translations of subspaces). In fact, in high-level computer vision, it is not difficult to realize that the set of objects of most common kinds do not follow the rules of Euclidean geometry, e.g., adding or subtracting two images of an automobile does not result in a valid image of an automobile. Thus, the set of automobile pictures is not easy to characterize at the raw-pixel level. In some special cases, under certain enabling assumptions, as will be discussed in the later chapters, one can arrive at well-defined characterizations of objects in images. A simple example of imposing a constraint in representation is when a vector is forced to have a unit norm, in order to pursue a scale-invariant solution, and the desirable space is a unit sphere. The nonlinearity of such spaces makes it difficult to apply traditional techniques that primarily rely on Euclidean analysis and multivariate statistics. Consider the challenge of developing statistical tools when even addition or subtraction is not a valid operation on that domain! Thus, even simple statistics, such as sample means and covariances, are no longer defined in a straightforward way. One needs a whole new approach to define and compute statistical inferences and optimization solutions. This motivates the need for RCCV.

Definition 1.1 (Riemannian Computing in Computer Vision—RCCV). Riemannian Computing in Computer Vision, or RCCV, is the scientific area that integrates tools from Riemannian geometry and statistics to develop theoretical and computational solutions for optimization and statistical inference problems driven by applications in computer vision.

In the context of nonlinear spaces, the analysis relies on elements of differential geometry. Here, the chosen space is established as a differentiable manifold and endowed with a smooth metric structure, termed *Riemannian* structure, to enable computation of distances and averages. A Riemannian structure, in principle, is capable of providing most of the tools for calculus—averages, derivatives, integrals, path lengths, and so on. In some cases, the spaces of interest enjoy an additional structure, such as a group structure, that is useful in the analysis. A group implies presence of a binary operation that allows one to compose and invert elements of the set. Such groups, manifolds with a group structure, are called *Lie groups* and play a very important role in RCCV. The examples of Lie groups include the sets of rigid and certain nonrigid transformations on objects and images. Sometimes Lie groups are also used to represent nuisance variability in situations where some transformations are not deemed relevant. For instance, in shape analysis of objects, their pose relative to the camera is treated as a nuisance parameter.

Next, we provide some basic background material for an uninitiated reader. Those with a working knowledge of differential geometry can skip this section.

1.2 Basic Tools from Riemannian Geometry

In this section we introduce some notation and basic concepts that are needed in the development of RCCV. Some of the following chapters define additional notation and ideas that are specific to their own treatments.

The key idea in RCCV is that the underlying space of interest is nonlinear. Let M represent the set of allowable feature values. Being nonlinear implies that for arbitrary points p_1 and p_2 in M, and arbitrary scalars α_1 and α_2, the combination $\alpha_1 p_1 + \alpha_2 p_2$ may not be in M. In fact, even the operation "+" may only be interpretable when we embed M inside a larger vector space V. While such embeddings are possible in most cases, the issue is that there are often many embeddings available and selecting one is a nontrivial problem. Given an embedding, the quantity $\alpha_1 p_1 + \alpha_2 p_2$ will lie in V (since it is a vector space) but not necessarily in M. Since addition (and subtraction) are not valid operations in M, one needs different sets of tools to handle M.

1.2.1 Tangent Spaces, Exponential Map, and Its Inverse

One starts by establishing M as a "differentiable manifold." Loosely speaking, it implies that small neighborhoods can be smoothly (diffeomorphically) mapped to open sets of Euclidean spaces, and such mappings can be composed smoothly. We will avoid precise definitions here and refer the reader to any introductory textbook on differential geometry, such as [7, 55]. Given a manifold, the next step is to impose a metric on it using a Riemannian structure (developed using a sequence of steps as follows). One denotes the set of all vectors that are tangent to M at a point $p \in M$ denote it by $T_p(M)$. (A tangent vector is the derivative of a differentiable curve passing through the point p at that point itself.) $T_p(M)$ is a vector space of the same dimension as M and allows standard statistical procedures such as PCA, discriminant analysis, and regression models, directly in a natural way. The indexed collection of all tangent spaces of M is called the *tangent bundle* of M, $TM = \cup_{p \in M} T_p(M)$.

Since a tangent space is a convenient domain for computational statistics, it is useful to have mappings back and forth between M and $T_p(M)$, for arbitrary $p \in M$. The mapping from $T_p(M)$ to M is called the *exponential map* and is denoted by $\exp_p : T_p(M) \to M$ such that $\exp_p(0) = p$. This mapping is usually a bijection between some neighborhood of 0 in $T_p(M)$ and a neighborhood of p in M. (There are exceptions though, especially when one deals with infinite-dimensional manifolds.)

1.2.2 Riemannian Structure and Geodesics

In order to obtain a metric on M, one imposes a Riemannian structure on M. A *Riemannian metric* is an inner product defined on tangent spaces of M that changes smoothly with p. For any $v_1, v_2 \in T_p(M)$, we will denote the metric by $\langle v_1, v_2 \rangle_p$. If $\alpha : [0,1] \to M$ is a C^1-curve (its derivative is continuous), then we can define the length of α according to

$$L[\alpha] = \int_0^1 \langle \dot{\alpha}(t), \dot{\alpha}(t) \rangle_{\alpha(t)} \, dt.$$

Note that $\dot{\alpha}(t) \in T_{\alpha(t)}(M)$ and the inner product inside the integral is defined using the Riemannian metric at that point. For any two points $p_1, p_2 \in M$, let the curve α be such that $\alpha(0) = p_1$ and $\alpha(1) = p_2$. Then, $L[\alpha]$ is the distance between those two points along the path α. The path with the smallest length is called a *geodesic*: $\alpha^* = \arg\min_{\alpha:[0,1] \to M, \alpha(0)=p_1, \alpha(1)=p_2} L[\alpha]$. The length of the shortest geodesic between any two points defines the distance (or geodesic distance) between those two points, i.e., $d(p_1, p_2) = L[\alpha^*]$. In the case of relatively simpler manifolds, such as a sphere or a matrix Lie group, the formulae for geodesic length are known and can be used directly. In some cases, such as those involving shape manifolds of planar curves, the problem of finding geodesics and geodesic distances is a challenge in itself and has been addressed using computational solutions.

1.2.3 Group Actions, Orbits, and Quotient Spaces

One frequent reason for involving differential geometry in computer vision is the need for invariance in vision applications. For instance, the identity or class of an object is invariant to the relative pose between that object and the camera. The classification of an activity being performed by a human subject in front of a camera is considered invariant to the execution rate of the activity. A natural way to achieve such invariance to certain transformations, it turns out, is through *actions* of corresponding groups on the representation space.

A group G is said to act on a manifold M, if there is a mapping $\phi : G \times M \to M$ that satisfies: (1) $\phi(g_2, \phi(g_1, p)) = \phi(g_2 \cdot g_1, p)$, for all $g_1, g_2 \in G$ and $p \in M$ and (2) $\phi(g_{id}, p) = p$ for all $p \in M$, where g_{id} is the identity element of the group G. The first item says that we can replace a sequence of transformations of an object by a cumulative transformation on that object. The second item simply means that the identity transformation leaves the object unchanged. A consequence of this definition is that for every transformation there is an inverse that can *undo* that transformation. A simple example of the group action is when $G = GL(n)$, the set of all non-singular matrices with matrix multiplication as the group action, and $M = \mathbb{R}^n$, with $\phi(A, x) = Ax$ for any $A \in GL(n)$ and $x \in \mathbb{R}^n$.

It is interesting to note that the group actions define an equivalence relation: two points p_1 and p_2 are said to be equivalent, or $p_1 \sim p_2$, when there exists a $g \in G$ such that $p_2 = \phi(g, p_1)$. That is, if one object can be transformed into another object, under a valid transformation, then they are termed equivalent. This equivalence relation partitions M into equivalence classes that are also called *orbits*. An orbit containing a point $p \in M$ is given by $[p] = \{\phi(g, p) | g \in G\}$, the set of all transformations of p. The set of all equivalence classes, or orbits, is called the quotient space of M under G. As an example, consider a problem where one is interested in strength (or magnitude) of the observed data and ignores their directions. Thus, for any $x \in \mathbb{R}^n$, one is interested only in $\|x\|$, the two-norm of x. Consequently, all the points that lie on the same sphere are indistinguishable. Mathematically, this is accomplished by defining an action of $SO(n)$ on $\mathbb{R}^n$ according to $\phi(O, x) = Ox$, for any $O \in SO(n)$ and $x \in \mathbb{R}^n$. Then, the orbit of an $x \in \mathbb{R}^n$ is given by $[x] = \{Ox | O \in SO(n)\}$. All points in an orbit are deemed equivalent and one compares points only by comparing their orbits. Since points within a rotation are equivalent, the equivalence class is a sphere, and the quotient space of $\mathbb{R}^n$ modulo $SO(n)$ is simply the positive real line consisting of all possible radii.

As hinted earlier, in computer vision the reason for introducing group actions is to achieve invariances to certain transformations. For instance, in shape analysis one usually starts with a base space—such as the set of landmarks—and then one wishes to be invariant to the effects of translation, scaling, and rotation. Preprocessing is usually employed to normalize the effects of translation and scale. The resultant space is also called the pre-shape space. The remaining problem of removing the effect of rotation can be viewed as comparing the orbits of two elements of the pre-shape space—where the orbit is defined as the action of the rotation group on an element of the pre-shape space. Similarly, in activity analysis, one often encounters the problem of comparing feature sequences while being invariant to the rate of execution of the activity. This requires defining an orbit of a sequence under a special kind of diffeomorphism group. Thus, the action of groups on underlying representation spaces is of great importance for devising classification algorithms in computer vision applications.

1.2.4 Probability Models and Learning

The problems in computer vision are naturally expressed as those of statistical inferences under appropriate models and probability distributions. These probability distributions are to be learned from past training data and used in future inferences. In order to perform these two steps—training and testing—one needs appropriate probability models that respect the geometries of underlying relevant manifolds. There are two main directions for defining probability models on metric spaces: (1) a parametric family that is rich enough to capture the desired variability and (2) a nonparametric model defined using a kernel and a bandwidth parameter. Of course, one can use a combination—termed semi-parametric models—also, but we will not discuss that direction here.

The training part in the parametric approach boils down to estimation of model parameters using the training data. One either uses a frequentist approach, seeking a maximum-likelihood estimates of the model parameters, or takes a Bayesian approach, imposes a prior model on the unknown variables, and analyzes their posterior distribution to specify the model. Let $f(p;\theta)$ for a $\theta \in \Theta$ denote a probability density on M parameterized by a θ. For a set of observed points $p_1, p_2, \ldots, p_n$, the maximum-likelihood estimate of θ is given by solving the optimization problem:

$$\hat{\theta}_n = \arg\max_{\theta \in \Theta} \left(\sum_{i=1}^{n} \log(f(p_i;\theta)) \right).$$

Under a Bayesian approach, one assume a prior on $\theta \sim f_0(\theta)$, and solves for θ under the posterior density given by

$$f(\theta | p_1, p_2, \ldots, p_n) = f_0(\theta)(\prod_{i=1}^{n} f(p_i;\theta)).$$

One can use the posterior mean, or some other posterior summary, as an estimate for θ in this setting. In a nonparametric approach [49, 50], the centerpiece is a kernel density estimator of the underlying density function given by

$$f(p) = \frac{1}{nZ} \sum_{i=1}^{n} K(p, p_i; h),$$

where K is a kernel function that is based on the distance between p and p_i, h is the bandwidth, and Z is the integral of K over M. Depending on the choice of K and, more importantly h, the shape of the density function can change drastically. The problem of estimating an optimal bandwidth from the data itself is a difficult issue in standard nonparametric statistics, and some results from the classical theory can be brought over to RCCV manifolds also.

1.3 History, Current Progress, and Remaining Challenges

In this section we summarize progress in use of differential geometry in vision and related areas over the years and outline some current topics of research.

1.3.1 Manifolds and Application Areas in RCCV

Here, we take a brief look at some nonlinear manifolds that are frequently encountered in computer vision applications.

1. **Spheres, Histograms, Bag of Words**: As mentioned previously, fixing the norm of the vectors restricts the relevant space of vectors to a sphere. In case the norm is selected to be one, the corresponding space is a unit sphere $\mathbb{S}^n \subset \mathbb{R}^{n+1}$. The unit spheres arise in situation where one is interested in direction of the vector rather than its magnitude. In statistical analysis, such considerations have led to an interesting area of study called *Directional Statistics* [44]. In computer vision, one often uses normalized histograms as representations or summaries of image or video data. Let a histogram h be a vector of frequency data points in $n+1$ predetermined bins. One way to normalize h is to impose the condition that $\sum_{i=1}^{n+1} h_i = 1$. In other words, h is a probability vector of size $n+1$. If we define $p_i = +\sqrt{h_i}$, then we see that $\|p\| = \sqrt{\sum_{i=1}^{n+1} p_i^2} = \sqrt{\sum_{i=1}^{n+1} h_i} = 1$, or $p \in \mathbb{S}^n$. Thus, one can conveniently represent normalized histograms as elements of a unit sphere, after a square-root transformation, and utilize the geometry of this sphere to analyze histograms [59]. It has also been argued [39] that the popular bag-of-words representation, which is commonplace in vision applications (cf. [48]), is best studied in terms of the statistical manifold of the n-dimensional multinomial family. Lafferty and Lebanon [39] proposed this approach and drew from results in information geometry to derive heat kernels on this statistical family, demonstrating the benefits of a geometric approach over a traditional vector space approach.
2. **Rotation Matrices, Rigid Motions, Structure from Motion**: One of the central problems in computer vision is recognition of objects from their images or videos, invariant to their pose relative to the camera. Related problems include pose tracking, pose-invariant detection of objects, and so on. How is the pose of a 3D object, relative to a chosen coordinate system, represented in analysis? It is most naturally represented as an element of $SO(3)$, the set of all 3×3 orthogonal matrices with determinant $+1$ [29]. This representation of pose enjoys a special structure that has the correct physical interpretations. The set $SO(3)$ is a Lie group with the group operation given by matrix multiplication. If one applies two rotations $O_1, O_2 \in SO(3)$ on a rigid object, in that order, then the cumulative rotation is given by $O_1 O_2 \in SO(3)$. Similarly, the transpose of a rotation $O \in SO(3)$, denoted by O^T, represents the inverse rotation and nullifies the

effect of O since $OO^T = I$. Incidentally, $SO(n)$ is a subset (and subgroup) of a more general transformation group $GL(n)$, the set of all $n \times n$ non-singular matrices. $GL(n)$ is useful in characterizing affine transformation in computer vision. Geometrical techniques for rotation averaging in structure from motion problems have been investigated in computer vision literature [9, 27].

3. **Covariance Tensors, Diffusion Tensor Imaging, Image-Patch Modeling**: In statistics one often represents a random quantity by a collection of its lower order moments. In particular, the use of mean and covariance matrices to characterize random variables is very common, especially under Gaussian models. A covariance matrix is a square, symmetric, and positive-definite matrix. When analyzing variables using their covariance matrices, the geometry of the space of such symmetric, positive-definite matrices (SPDMs) becomes important. This is a nonlinear space whose geometry is naturally studied as a quotient space of a Lie group, $GL(n)$ modulo $SO(n)$. In other words, one forms the action of $SO(n)$ on $GL(n)$ and identify the orbits of $SO(n)$ with individual SPDMs. A field of 3×3 tensors forms an intermediate representation of data in diffusion-tensor MRI, and one needs the geometry of SPDMs for interpolation, denoising, and registrations of image data [15, 47, 51, 54]. This representation has also proved highly successful in modeling textures in patches and its application to pedestrian detection and tracking [66, 67].
4. **Subspaces, Dynamical Models, Projections**: Due to severity of dimensions in vision data sets (images and videos) encountered in computer vision, the task of dimension reduction is a central theme. One idea here is to use a linear projection to a low-dimensional subspace of the original observation space. In order to make these projections optimal, for the given data and the given application, one needs to optimize a certain chosen objective function over the space of subspaces. The space of all d-dimensional orthogonal bases of an n-dimensional space ($d << n$) is called a Stiefel manifold $\mathcal{S}_{n,d}$, while the space of of all d-dimensional subspaces of $\mathbb{R}^n$ is called a Grassmann manifold $\mathcal{G}_{n,d}$. These manifolds are naturally studied as quotient spaces of larger rotations modulo smaller rotations: $\mathcal{S} \equiv SO(n)/SO(n-d)$ and $\mathcal{G}_{n,d} = SO(n)/(SO(n-d) \times SO(d))$. The use of linear projections of image data, optimally related to specific vision tasks, is an important area in itself [42, 58]. The Grassmann manifold also plays an important role in characterizing linear, time-invariant, dynamical systems where the observability matrix of the system is represented as a subspace of appropriate dimensions [64, 65]. The Grassmann manifold also arises as the representation space in face recognition when using a subspace to model a collection of faces. This representation has been used to devise very effective face recognition algorithms [24, 25].
5. **Deformations, Image Registration**: One of the most studied problems in image processing, especially medical image analysis, is the problem of registration of pixels/voxels across images. This task is performed by fixing one image I_1 and deforming the other image I_2 such that the corresponding pixel locations are considered matched. The deformation γ is a process of changing pixel locations,

while keeping the pixel values fixed, in a smooth fashion so that the pixels do not cross each other. In mathematical terms, one defines deformation using diffeomorphic transformations of the set of pixel locations. Thus, the set of diffeomorphisms is synonymous with the set of deformations of images used in image registration [5, 71]. The cost function used for image registration is most commonly of the type

$$\arg\min_{\gamma\in\Gamma}\left(\|I_1 - I_2\circ\gamma\|^2 + \lambda\mathcal{R}(\gamma)\right),$$

where Γ is the set of all deformations and $\mathcal{R}$ is a roughness penalty on γ.

6. **Elastic Functions, Trajectories, Activities**: In experiments involving dynamic systems one studies the evolution of certain feature(s) of interest over an observation period. These variables are mathematically represented as real- or vector-valued functions over time and the area of statistics that deals with such data is called *functional data analysis*. The main challenges in this area come from two sources: (1) the infinite dimensionality of representation spaces and (2) the fact that the systems evolve under different rates in different observations. The latter source, termed the phase variability in functional data, is a nuisance variable and needs to be removed in statistical data analysis. This requires proper metrics and models to be able to formally define the concepts of amplitude and phase components in functions [34]. The metrics that allow comparisons of functions under different phase components are termed *elastic metrics* and the resulting analysis *elastic FDA*. An extension of this problem involves studying temporal evolutions of systems whose representations take values on Riemannian manifolds. The sample paths, or observations, of these systems are studied as trajectories on Riemannian manifolds [35, 62]. For instance, consider a human performing an action or an activity in front of a depth sensor where one forms a skeletal representation of the human body and studies the evolution of its shape to characterize the activity. This activity can be represented as a trajectory on an appropriate shape space of skeletons and one needs metrics/models for classifying activities using noisy depth data (cf. [3]).
7. **Shape Representations**: In the context of detection, tracking, and recognition of objects in image and video data, the shapes of silhouettes of objects play a very important role. Since these silhouettes of objects form contours and surfaces in 2D and 3D images, the area of shape analysis of curves and surfaces has become very active. Even though one starts with Euclidean representations of these objects, shape analysis quickly becomes complicated because shape is a property that is invariant to rigid motion, global scaling, and re-parameterization of the objects being studied. Thus, shape spaces are typically quotient spaces representing original objects (curves or surfaces) modulo the actions of these shape-preserving transformation groups. The shape preserving groups are rotation $SO(n)$, translation $\mathbb{R}^n$, scaling $\mathbb{R}_+$, and re-parameterization.

1.3.2 Historical Perspective

The use of Riemannian geometry has been widespread in several branches of science and engineering, contributing towards developing methodologies, algorithms, and systems. For example, in design of control of systems, Brockett and colleagues [8] developed a system theory for Lie groups and their quotient spaces in the 1970s, followed by many control theoretic solutions on general manifolds [11–13, 41, 72]. Similar use of Riemannian geometry was also seen in mechanical designs, robotics, and human–machine interaction systems.

The explicit use of manifolds, Lie groups, and Lie group actions on manifolds in the area of pattern recognition was pioneered by Ulf Grenander. His formulation of pattern theory for representation of complex systems is based on using combinations of simple elementary units, called *generators*, that under the actions of small groups capture the variability exhibited by the observed systems. This theory was laid out in a series of monographs [17, 18, 20], culminating in the textbook [19]. A confluence of particle filtering and Riemannian geometry, especially for Lie groups and their quotient spaces, can be found in related papers [23, 56, 57]. In terms of problems in image understanding, some applications of Grenander's pattern theory appeared in the joint works with Amit [2], Keenan [22], and Miller [20, 21, 45]. This laid the foundation for a principled approach to the development of deformable template theory that has been used extensively in a variety of applications. Shape analysis of landmark configurations, advanced by David Kendall and colleagues starting in the 1980s [10, 31–33, 40] and continuing to date, represents another major development that is firmly rooted in Riemannian geometry. An innovative direction in shape analysis has been to handle continuous objects, including curves [60, 73, 74], surfaces [36–38], and general trajectories.

Focusing on the area of computer vision and image understanding, Kanatani [29] provided a nice discourse on the use of groups for understanding structures in images. To cite an example, consider the problem of tracking and recognizing objects in video data. The relative pose of objects, with respect to the camera, is conveniently represented as an element of the special orthogonal group [23, 29, 46]. The use of Riemannian geometry has also been prevalent in tracking [52, 66], mean shift algorithm that was used for segmentation and clustering [63], activity recognition [43, 69, 70], basis learning [26], and so on.

In statistics, one of the earliest uses of Riemannian geometry involved imposing metrics on parametric families of probability density functions (pdfs), using the so-called *Fisher–Rao* metric [53]. This work led by Amari [1], Kass [30], Efron [14], and many others, used the parametric version of the Fisher–Rao Riemannian metric seeking geodesic paths and geodesic distances between pdfs restricted to chosen exponential families. In recent years, the nonparametric version of this metric has proven very useful in a variety of vision applications. Although, it was mentioned as early as 1943, by Bhattacharya [6], the use of nonparametric Fisher geometry for comparing probability distribution has also been developed with computer vision applications in mind [59]. A framework for statistical analysis of manifold-valued variables has been explored in a series of papers by Pennec et al. [51].

1.3.3 Current Progress and Remaining Challenges

Despite a steady progress in identifying relevant tools from Riemannian geometry, and their applications in vision problems, there are many directions and novel venues for making progress in RCCV and, consequently, to make a large impact in the area of computer vision. The challenges in posing and solving statistical problems on nonlinear manifolds with applications to computer vision are many. We enumerate a few of those:

1. **Nonlinearity**: The original problem, of course, is to define tools that can overcome the issue of nonlinearity of manifold spaces. There has been a large progress in identifying the basic sets of interest in computer vision and exploiting their geometries, as described earlier in this chapter. Researchers are discovering novel spaces, characterized by nonlinear geometries motivated by vision applications, that will become domains for future advances in RCCV. Examples include vector bundles of known Riemannian manifolds, to help analyze trajectories on these spaces.
2. **Effective and Efficient Metrics**: Unlike Euclidean spaces, where there is naturally a canonical choice of a metric, the case of manifolds can be very different. There may be several choices and the most obvious one may not provide all the desired invariances and computational efficiency. Take the case of functional data analysis where the classical $\mathbb{L}^2$ norm and the associated harmonic analysis have been used for long time for statistical analysis. However, in case the phase variability is present in the observed data, the use of Fisher–Rao metric and its derivatives has proven much more efficient [61]. More recent work has focused on studying different Sobolev metrics for analyzing functional data with phase variability. Similar examples are present in many vision applications. Even though the manifolds of interest have been identified, the choice of metrics that match the goals of the problem becomes important. For instance, in tracking shapes of human silhouettes in video data, it is important to choose shape metrics that impose smaller distances on natural motions and larger distances on improbable deformations of shapes.
3. **Computational Tools**: Beyond appropriate choice of metrics and development of geometrical tools, it is also necessary to have efficient computational solutions to leading problems in computer vision. For any manifold representation, one needs algorithms for computing geodesics, estimating parameters, learning models, evaluating likelihoods, computing likelihood ratios, etc. For instance, while the formulation of geodesics on a Grassmann manifold is known for a while, it is important to exploit tools from numerical analysis to help decrease the computation cost [16]. Similar ideas are needed in nearly all the domains mentioned above—functional data analysis, covariance tracking, image registration, optimal dimension reduction, etc,
4. **Geometry and Machine Learning**: Perhaps the largest area of growth in computer vision and pattern recognition has been in the area of machine learning, particularly the advances made in probabilistic methods using deep learning

frameworks. These and related tools in machine learning seek to learn features and classifiers from the training data, in order to maximize classification of objects and patterns in image data. Since many relevant features are known to lie on nonlinear manifolds, the process of learning and discrimination of data classes can benefit from exploiting geometries of these manifolds. One example of these idea was illustrated in [4] where the authors used Adaboost techniques using geodesic distances on shape manifolds for improving face recognition performances. Another example is the use of kernel-based techniques and the recent research on the choice of kernels suitable for use on nonlinear Riemannian manifolds [25, 28]. However, much work needs to be done on the theoretical front to provide the provable performance guarantees of general machine learning algorithm on manifolds, analogous to the seminal work of Vapnik [68] but generalized to non-Euclidean spaces. Additional work is needed as well on the applied front when dealing with computational complexity issues and trade-offs between approximations and accuracy.

We believe that a combination of tools and ideas from geometry, statistics, and computational science will fuel development of next generation of computer vision algorithms.

1.4 Volume Layout and Chapter Contributions

This edited volume provides a sampling of latest advances in RCCV from different problem areas and perspectives in computer vision. These are invited chapters from experienced researchers with strong, exemplary records of publications in computer vision or related areas. The chapters in the remainder of this volume provide strong instances of progress in RCCV. The volume is divided into four parts as follows. Although these chapters have been arranged according to their main focus, the tools used and results achieved can be reorganized in many different ways.

1.4.1 Statistical Computations on Riemannian Manifolds

The first part of the volume deals with defining or improving tools for computational statistics related to manifold-valued random variables.

1. The second chapter, written by *Cheng et al.*, looks at a very specific task of computing the sample mean, termed Frechet mean, from a given sample of points on a Riemannian manifold with nonpositive curvature. They provide an algorithm for computing this and analyze its asymptotic convergence to the population mean. They also provide three experiments demonstrating advantages of their construction on the Riemannian manifold of SPDMs.

2. The first chapter by *Jayasumana et al.* studies the suitability of kernel functions on Riemannian manifolds. Since positive-definite kernels are critical in improving discriminatory approaches using SVMs, etc., one needs to define such kernels on manifolds of interest also. This chapter provides some interesting constructions—Gaussian RBF kernels and kernel-based classifiers—for some RCCV manifolds of interest.
3. The next chapter by *Kim et al.* studies a similar problem, this time of canonical correlation analysis, on the space of SPDMs. They use the differential geometry of this space to define a Lagrangian method for computing CCA components and demonstrate this idea using synthetic experiments.
4. The chapter by *Fletcher et al.* develops a framework for regression problems in situations where the response variable lies on a Riemannian manifold. It uses a natural analog of Gaussian distributions on manifolds and tools from principal geodesic analysis to define the regression problem and to derive a formal solution that also has accompanying residual analysis.

Together these four chapters bolster the toolbox for computing sample statistics on Riemannian manifolds and treat the case of SPDMs in great details.

1.4.2 Color, Motion, and Stereo

This part of the volume deals with extracting low-level features from images and videos that form raw data for statistical inferences in vision problems.

1. The first chapter in this part, written by *Anand et al.*, provides a nice transition from statistical analysis to feature extraction in vision data sets. It deals with improving estimation of motion features using robust statistics on Grassmann manifolds. They describe the use of the classical mean shift algorithm to Grassmann manifolds and derive M-estimators for improving robustness to noise in the inference process. They also describe the applications of these ideas in a number of vision applications.
2. The second chapter by *Govindu* is on mechanisms for averaging features when they are elements of certain matrix Lie groups. It focuses on the special Euclidean group that represents rigid motions of a camera with respect to the imaged object. The main application here is to perform 3D registrations of multiple views of a scene in order to improve scene reconstruction.
3. The last chapter in this section, written by *Li et al.*, highlights the well-established involvement of Lie group structures in robotics and mechanizations. The main transformation group used here is the special Euclidean group, denoting rigid motions of a robot and applied to localization of multiple robots simultaneously.

1.4.3 Shape, Surfaces, and Trajectories

As discussed previously, differential geometry plays an integral part in analyzing shapes of objects. Since shapes are considered invariant to certain transformations of the underlying objects, shape spaces are quotient spaces of the representation space modulo shape-preserving transformations.

1. The first chapter by *Brignell et al.* continues to advance Kendall's representation of shapes using landmarks. It develops the notion of covariance-weighted Procrustes alignment and applies it to some simulated data sets.
2. The second chapter, written by *Joshi et al.*, summarizes recent progress in the area of elastic shape analysis of curves for several types. It includes data objects such as real-valued functions, curves in Euclidean spaces, and curves on nonlinear manifolds. It provides examples from computer vision, including shape analysis and activity recognition, to demonstrate these ideas.
3. The next chapter, by *Bauer et al.*, focuses on the problem of shape analysis of Euclidean curves and discusses some limitations of the elastic framework discussed in the previous chapter. They suggest the use of higher-order Sobolev metrics, i.e., metrics involving second- or higher-order derivatives of the given curves, as a solution.
4. The last chapter in this section, written by *Kurtek et al.*, studies more complex objects, such as surfaces and images, from the perspective of registration and shape analysis. It introduces an analog of square-root transformations, discussed in the previous two chapters for curves, to surfaces and images and demonstrates the use of these techniques in generating statistical summaries and disease classification.

1.4.4 Objects, Humans, and Activities

The last part of this volume includes chapters that are focused more heavily on high-level vision applications, such as face recognition, object recognition, and activity recognition.

1. The first chapter, authored by *Porikli et al.*, looks at the problem of boosting classification performance using machine learning ideas, but for features that are SPDM valued. They demonstrate the idea of using Logitboost for the covariance descriptors and demonstrate an improvement in human detection performance using this boosting approach.
2. The second chapter, written by *Lui*, develops a framework for performing regression using matrix-valued variables that are constrained to lie on certain manifolds. Specifically, it develops the analog of generalized least squares framework for Euclidean variables, to spaces such as SPDMs and Grassmann manifolds. Finally, it demonstrates these ideas using tracking, recognition, and human interactions.

3. The next chapter, by *Shaw and Chellappa*, tackles an important problem in computer vision that involves transfer of information and statistical models from one domain to another. Using the geometry of Grassmann manifolds, the authors demonstrate domain adaptation in object recognition and age classification.
4. The next chapter, by *Harandi et al.*, is on the topic of coding dictionaries for image/video representations that are manifold valued. Taking the example of SPDMs, the authors provide a new approach, titled *coordinate coding*, using different metric structures on the space of SPDMs, and illustrate the improvements using experimental studies.
5. The last chapter, by *Shroff et al.*, proposes a general framework to enable summarization and search over feature spaces that are manifold valued. The approach extends classical vector space techniques, such as column subset selection and locality-sensitive hashing, to non-Euclidean manifolds. They develop highly efficient algorithms for exemplar selection and approximate nearest-neighbor search and demonstrate the utility of these techniques on shapes and activities.

1.4.5 Potential Pathways Through the Volume

We believe that the volume presents multiple possibilities and approaches for study. The presented sequencing is likely to be helpful for a beginning researcher in computer vision. However, any fixed sequencing is by no means optimal or suitable for everybody. It is worth noting that almost all chapters of the book are rather self-contained—each chapter introduces the proposed problem, the Riemannian formulation, associated algorithms and evaluation—independently of other chapters. While this can result in some redundancy, we feel that such a self-contained treatment makes the book eminently readable from any chapter. We however do highly recommend starting with the introductory chapter and Part I of the book which should lay the general foundational principles. After that readers interested in specific application areas can make segue into individual chapters.

It is also worth noting that many chapters in the later parts, even though categorized into application areas, present novel theoretical material applicable more broadly than the current application. Thus, an open-minded reader would benefit from every chapter—irrespective of which application area it appears under. The partitioning of material as presented should be attributed to the limitations of the printed book format. In the increasingly interdisciplinary world we find ourselves in, some of the most fruitful questions are to be found at the intersections of such boundaries.

Acknowledgements The authors would like to acknowledge the support received from National Science Foundation grants #1320267 and #1319658 during the preparation of this edited volume.

References

1. Amari S (1985) Differential geometric methods in statistics. Lecture notes in statistics, vol 28. Springer, New York
2. Amit Y, Grenander U, Piccioni M (1991) Structural image restoration through deformable templates. J Am Stat Assoc 86(414):376–387
3. Anirudh R, Turaga P, Su J, Srivastava A (2015) Elastic functional coding of human actions: from vector-fields to latent variables. In: IEEE conference on computer vision and pattern recognition, Boston
4. Ballihi L, Amor BB, Daoudi M, Srivastava A, Aboutajdine D (2012) Boosting 3-d-geometric features for efficient face recognition and gender classification. IEEE Trans Inf Forensic Secur 7(6):1766–1779
5. Beg M, Miller M, Trouvé A, Younes L (2005) Computing large deformation metric mappings via geodesic flows of diffeomorphisms. Int J Comput Vision 61:139–157
6. Bhattacharya A (1943) On a measure of divergence between two statistical populations defined by their probability distributions. Bull Calcutta Math Soc 35:99–109
7. Boothby WM (2007) An introduction to differentiable manifolds and Riemannian geometry, Revised, 2nd edn. Academic Press, New York
8. Brockett RW (1972) System theory on group manifolds and coset spaces. SIAM J Control 10(2):265–84
9. Chatterjee A, Govindu VM (2013) Efficient and robust large-scale rotation averaging. In: IEEE international conference on computer vision, ICCV, Sydney, pp 521–528
10. Dryden IL, Mardia KV (1998) Statistical shape analysis. Wiley, Chichester
11. Duncan TE (1977) Some filtering results in Riemannian manifolds. Inf Control 35(3):182–195
12. Duncan TE (1979) Stochastic systems in Riemannian manifolds. J Optim Theory Appl 27(3):399–426
13. Duncan TE (1990) An estimation problem in compact lie groups. Syst Control Lett 10(4):257–63
14. Efron B (1975) Defining the curvature of a statistical problem (with applications to second order efficiency). Ann Stat 3:1189–1242
15. Fletcher P, Joshi S (2007) Riemannian geometry for the statistical analysis of diffusion tensor data. Signal Process 87(2):250–262
16. Gallivan KA, Srivastava A, Liu X, Van Dooren P (2003) Efficient algorithms for inferences on grassmann manifolds. In: IEEE workshop on statistical signal processing, pp 315–318
17. Grenander U (1976/1978) Pattern synthesis: lectures in pattern theory, vol I, II. Springer, New York
18. Grenander U (1981) Regular structures: lectures in pattern theory, vol III. Springer, New York
19. Grenander U (1993) General pattern theory. Oxford University Press, Oxford
20. Grenander U, Miller MI (1994) Representations of knowledge in complex systems. J R Stat Soc 56(3):549–603
21. Grenander U, Miller MI (1998) Computational anatomy: an emerging discipline. Q Appl Math LVI(4):617–694
22. Grenander U, Chow Y, DKeenan (1990) HANDS: a pattern theoretic study of biological shapes. Springer, New York
23. Grenander U, Miller MI, Srivastava A (1998) Hilbert-Schmidt lower bounds for estimators on matrix Lie groups for ATR. IEEE Trans Pattern Anal Mach Intell 20(8):790–802
24. Ham J, Lee DD (2008) Extended grassmann kernels for subspace-based learning. In: Advances in neural information processing systems, vol 21. Proceedings of the twenty-second annual conference on neural information processing systems, Vancouver, British Columbia, pp 601–608
25. Ham J, Lee DD (2008) Grassmann discriminant analysis: a unifying view on subspace-based learning. In: Proceedings of the twenty-fifth international conference on machine learning, (ICML 2008), Helsinki, pp 376–383

26. Hamm J, Lee DD (2008) Grassmann discriminant analysis: a unifying view on subspace-based learning. In: International conference on machine learning, Helsinki, pp 376–383
27. Hartley RI, Trumpf J, Dai Y, Li H (2013) Rotation averaging. Int J Comput Vis 103(3):267–305
28. Jayasumana S, Hartley RI, Salzmann M, Li H, Harandi MT (2013) Kernel methods on the riemannian manifold of symmetric positive definite matrices. In: IEEE conference on computer vision and pattern recognition, pp 73–80
29. Kanatani K (1990) Group-theoretical methods in image understanding. Springer, New York
30. Kass RE, Vos PW (1997) Geometric foundations of asymptotic inference. Wiley, New York
31. Kendall DG (1984) Shape manifolds, procrustean metrics and complex projective spaces. Bull Lond Math Soc 16:81–121
32. Kendall DG, Barden D, Carne TK, Le H (1999) Shape and shape theory. Wiley, Chichester
33. Kent JT, Mardia KV (2001) Shape, Procrustes tangent projections and bilateral symmetry. Biometrika 88:469–485
34. Kneip A, Ramsay JO (2008) Combining registration and fitting for functional models. J Am Stat Assoc 103(483):1155–1165
35. Kume A, Dryden IL, Le H (2007) Shape-space smoothing splines for planar landmark data. Biometrika 94:513–528
36. Kurtek S, Klassen E, Ding Z, Srivastava A (2010) A novel Riemannian framework for shape analysis of 3d objects. In: IEEE computer society conference on computer vision and pattern recognition, pp 1625–1632
37. Kurtek S, Klassen E, Ding Z, Jacobson S, Jacobson JL, Avison MJ, Srivastava A (2011) Parameterization-invariant shape comparisons of anatomical surfaces. IEEE Trans Med Imaging 30(3):849–858
38. Kurtek S, Klassen E, Gore JC, Ding Z, Srivastava A (2012) Elastic geodesic paths in shape space of parametrized surfaces. In: Transactions on pattern analysis and machine intelligence. Accepted for publication. doi:10.1109/TPAMI.2011.233
39. Lafferty J, Lebanon G (2005) Diffusion kernels on statistical manifolds. J Mach Learn Res 6:129–163
40. Le H, Kendall DG (1993) The Riemannian structure of euclidean shape spaces: a novel environment for statistics. Ann Stat 21(3):1225–1271
41. Leonard NE, Krishnaprasad PS (1995) Motion control of drift-free left-invariant systems on lie groups. IEEE Trans Autom Control 40(9):1539–1554
42. Liu X, Srivastava A, Gallivan K (2004) Optimal linear representations of images for object recognition. IEEE Trans Pattern Anal Mach Intell 26(5):662–666
43. Lui YM, Beveridge JR, Kirby M (2010) Action classification on product manifolds. In: IEEE international conference on computer vision and pattern recognition, pp 833–839
44. Mardia KV, Jupp P (2000) Directional statistics, 2nd edn. Wiley, New York
45. Miller MI, Christensen GE, Amit Y, Grenander U (1993) Mathematical textbook of deformable neuroanatomies. Proc Nat Acad Sci 90(24):11944–11948
46. Moakher M (2002) Means and averaging in the group of rotations. SIAM J Matrix Anal Appl 24:1–16
47. Moakher M (2005) A differential geometric approach to the geometric mean of symmetric positive-definite matrices. SIAM J Matrix Anal Appl 26(3):735–747
48. Niebles JC, Wang H, Li F (2008) Unsupervised learning of human action categories using spatial-temporal words. Int J Comput Vis 79(3):299–318
49. Pelletier B (2005) Kernel density estimation on riemannian manifolds. Stat Probab Lett 73(3):297–304
50. Pelletier B (2006) Nonparametric regression estimation on closed riemannian manifolds. J Nonparametr Stat 18(1):57–67
51. Pennec X, Fillard P, Ayache N (2006) A Riemannian framework for tensor computing. Int J Comput Vis 66(1):41–66
52. Porikli F, Tuzel O, Meer P (2006) Covariance tracking using model update based on lie algebra. In: IEEE international conference on computer vision and pattern recognition, New York, pp 728–735

53. Rao CR (1945) Information and accuracy attainable in the estimation of statistical parameters. Bull Calcutta Math Soc 37:81–91
54. Schwartzman A (2006) Random ellipsoids and false discovery rates: statistics for diffusion tensor imaging data. Ph.D. thesis, Stanford
55. Spivak M (1979) A comprehensive introduction to differential geometry, vol I, II. Publish or Perish, Inc., Berkeley
56. Srivastava A (2000) A Bayesian approach to geometric subspace estimation. IEEE Trans Signal Process 48(5):1390–1400
57. Srivastava A, Klassen E (2004) Bayesian and geometric subspace tracking. Adv Appl Probab 136(1):43–56
58. Srivastava A, Liu X (2005) Tools for application-driven dimension reduction. J Neurocomput 67:136–160
59. Srivastava A, Jermyn IH, Joshi SH (2007) Riemannian analysis of probability density functions with applications in vision. In: IEEE conference on computer vision and pattern recognition, CVPR'07, pp 1–8. doi:10.1109/CVPR.2007.383188
60. Srivastava A, Klassen E, Joshi SH, Jermyn IH (2011) Shape analysis of elastic curves in euclidean spaces. Pattern Anal Mach Intell 33(7):1415–1428
61. Srivastava A, Wu W, Kurtek S, Klassen E, Marron JS (2011) Registration of functional data using Fisher-Rao metric. arXiv [arXiv:1103.3817]
62. Su J, Kurtek S, Klassen E, Srivastava A (2014) Statistical analysis of trajectories on riemannian manifolds: bird migration, hurricane tracking, and video surveillance. Ann Appl Stat 8(1):530–552
63. Subbarao R, Meer P (2009) Nonlinear mean shift over Riemannian manifolds. Int J Comput Vis 84(1):1–20
64. Turaga PK, Veeraraghavan A, Chellappa R (2008) Statistical analysis on Stiefel and Grassmann manifolds with applications in computer vision. In: IEEE international conference on computer vision and pattern recognition
65. Turaga P, Veeraraghavan A, Srivastava A, Chellappa R (2010) Statistical computations on grassmann and stiefel manifolds for image and video based recognition. In: IEEE transactions on pattern analysis and machine intelligence. Accepted for publication
66. Tuzel O, Porikli F, Meer P (2006) Region covariance: a fast descriptor for detection and classification. In: European conference on computer vision, Graz, pp 589–600
67. Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on riemannian manifolds. IEEE Trans Pattern Anal Mach Intell 30(10):1713–1727
68. Vapnik V (1998) Statistical learning theory. Wiley, New York
69. Vaswani N, Roy-Chowdhury A, Chellappa R (2005) "shape activity": a continuous-state hmm for moving/deforming shapes with application to abnormal activity detection. IEEE Trans Image Process 14(10):1603–1616
70. Veeraraghavan A, Srivastava A, Roy Chowdhury AK, Chellappa R (2009) Rate-invariant recognition of humans and their activities. IEEE Trans Image Process 18(6):1326–1339
71. Vercautere T, Pennec X, Perchant A, Ayache N (2008) Symmetric log-domain diffeomorphic registration: a demons-based approach. In: MICCAI 2008. Lecture notes in computer science, vol 5241, pp 754–761
72. Walsh G, Sarti A, Sastry S (1993) Algorithms for steering on the group of rotations. In: Proceedings of ACC
73. Younes L (1999) Optimal matching between shapes via elastic deformations. J Image Vis Comput 17(5/6):381–389
74. Younes L, Michor PW, Shah J, Mumford D, Lincei R (2008) A metric on shape space with explicit geodesics. Mat E Appl 19(1):25–57

Part I
Statistical Computing on Manifolds

This part presents a diverse selection of fundamental statistical techniques.

Chapter 2
Recursive Computation of the Fréchet Mean on Non-positively Curved Riemannian Manifolds with Applications

Guang Cheng, Jeffrey Ho, Hesamoddin Salehian, and Baba C. Vemuri

Abstract Computing the Riemannian center of mass or the finite sample Fréchet mean has attracted enormous attention lately due to the easy availability of data that are manifold valued. Manifold-valued data are encountered in numerous domains including but not limited to medical image computing, mechanics, statistics, machine learning. It is common practice to estimate the finite sample Fréchet mean by using a gradient descent technique to find the minimum of the Fréchet function when it exists. The convergence rate of this gradient descent method depends on many factors including the step size and the variance of the given manifold-valued data. As an alternative to the gradient descent technique, we propose a recursive (incremental) algorithm for estimating the Fréchet mean/expectation (iFEE) of the distribution from which the sample data are drawn. The proposed algorithm can be regarded as a geometric generalization of the well-known incremental algorithm for computing arithmetic mean, since it reinterprets this algebraic formula in terms of geometric operations on geodesics in the more general manifold setting. In particular, given known formulas for geodesics, iFEE does not require any optimization in contrast to the non-incremental counterparts and offers significant improvement in efficiency and flexibility. For the case of simply connected, complete and nonpositively curved Riemannian manifolds, we prove that iFEE converges to the true expectation in the limit. We present several experiments demonstrating the efficiency and accuracy of iFEE in comparison to the non-incremental counterpart for computing the finite sample Fréchet mean of symmetric positive definite matrices as well as applications of iFEE to K-means clustering and diffusion tensor image segmentation.

G. Cheng
Google Inc. Kirkland WA, USA
e-mail: seagle99@gmail.com

J. Ho
MediaTek, San Jose, CA 95134
e-mail: jho.jeffrey@gmail.com

H. Salehian • B.C. Vemuri (✉)
University of Florida, Gainesville, FL, USA
e-mail: salehian@cise.ufl.edu; vemuri@cise.ufl.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_2

2.1 Introduction

In computer vision and machine learning, statistical analysis of features often requires them to be considered as random variables. Although features are always represented as collections of real numbers, their idiosyncratic origins and indigenous constraints often can best be interpreted not as vectorial features in $\mathbb{R}^n$ but as manifold features, features belonging to some embedded submanifold $\mathcal{M}$ of $\mathbb{R}^n$. In this sense, manifold-valued random variables abound in vision and machine learning literature: Popular image features such as SIFT and HOG, due to normalization, are often features defined on spheres. For applications involving directional and geometric data, important features can often be found in Lie groups that model their underlying symmetries (e.g., [25]). From Lie groups, one is naturally led to their homogeneous and symmetric spaces, and not surprisingly, the homogeneous and symmetric spaces of classical Lie groups such as (complex) projective spaces, Stiefel manifolds (of which spheres form an important family), and Grassmannians are especially rich domains for generating useful features, making their appearances in a wide range of vision problems, including shape and procrustean analysis [30], subspace clustering [45], dynamic texture classification [19], object recognition [14, 48], and many others. In particular, as a symmetric space of the general linear group $\mathbf{GL}(n)$ (n-by-n non-singular matrices), the manifold $\mathbb{P}(n)$ of n-by-n symmetric positive-definite (SPD) matrices has featured prominently in many vision and learning problems, from the relatively simple image structure tensors [29] that have been the traditional workhorse in vision algorithms to the more refined covariance features used in tracking [17, 50] and recognition [49], and from the application domain of diffusion tensor MRI (DT-MRI) in medical imaging (e.g., [22, 34, 41, 52]) to the more esoteric domain of information geometry using Fisher–Rao metric [2]. While manifolds identify the domains on which the features are defined, probability and metric then furnish the tools for analyzing and characterizing their uncertainty and similarity, respectively. In the general manifold setting, the union of probability and geometry (metric) naturally leads to the notion of Fréchet mean/expectation [23] of a random variable, whose definition requires the specifications of both a distribution and a Riemannian metric. Consequently, an important computational problem is to estimate the Fréchet mean using samples of the distribution. However, in this era of massive and continuous streaming data, samples are often given either as a whole that are difficult to utilize due to their size or in parts with availability depending on other external factors. Therefore, from an application viewpoint, the desired algorithm should be *incremental* in nature in order to maximize computational efficiency and account for data availability, requirements that are seldom addressed in more theoretically oriented domains. In this chapter, we propose incremental Fréchet expectation estimator (iFEE), a novel incremental algorithm for computing the Fréchet mean of manifold-valued random variables, and establish its convergence for simply connected, complete Riemannian manifolds with nonpositive sectional curvature.

Let $\mathcal{M}$ denote a (complete) Riemannian manifold and $d\omega$ its canonical Riemannian volume measure. We will consider a probability measure dP on $\mathcal{M}$ that is absolutely continuous with respect to $d\omega$, i.e., its density function $P(\mathbf{x})$ exists with $dP = P(\mathbf{x})\, d\omega$ as the measure. The Fréchet expectation (or Fréchet mean) of the probability measure dP is defined as

$$\mathbf{m} = \arg\min_{\mathbf{z}\in\mathcal{M}} \int_{\mathcal{M}} \mathbf{d}^2_{\mathcal{M}}(\mathbf{z},\mathbf{x})\, P(\mathbf{x}) d\omega, \tag{2.1}$$

where $\mathbf{d}_{\mathcal{M}}(\mathbf{z},\mathbf{x})$ is the Riemannian geodesic distance between $\mathbf{z},\mathbf{x} \in \mathcal{M}$. Note that the expectation $\mathbf{m}$ is a point in $\mathcal{M}$ (or a set in general), and it requires the specifications of the Riemannian metric $\mathbf{d}^2_{\mathcal{M}}(\mathbf{z},\mathbf{x})$ and the probability measure dP. In general, the existence and uniqueness of Fréchet expectation for an arbitrary probability measure defined on a Riemannian manifold is a subtle and technical issue; however, for simply connected and complete Riemannian manifolds with nonpositive sectional curvature, a theorem of E. Cartan (see Sect. 6.1.5 in [7]) shows that the Riemannian center of mass (Fréchet expectation) always exists and is unique for any probability measure absolutely continuous with respect to ω. Therefore, in this chapter, we will focus exclusively on such Riemannian manifolds, and in particular, we will define an estimator $\mathbf{m}_k$ of $\mathbf{m}$ for any sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$ of i.i.d. samples drawn from the distribution dP and show that $\mathbf{m}_k$ converges asymptotically to $\mathbf{m}$ as $k \to \infty$.

In the Euclidean domain, the estimator $\mathbf{m}_k$ is well known and it is simply the average of the (finite) samples

$$\mathbf{m}_k = \frac{\mathbf{x}_1 + \cdots + \mathbf{x}_k}{k}. \tag{2.2}$$

The validity of the estimator of course is guaranteed by the (weak) law of large numbers, which states that the estimator $\mathbf{m}_k$ converges in probability to the true mean $\mathbf{m}$. For practitioners in computer vision and machine learning (and others), this well-known result is so deeply ingrained that many times we use it without immediate awareness of it. In particular, $\mathbf{m}_k$ can be computed incrementally using the formula

$$\mathbf{m}_{k+1} = \frac{k\,\mathbf{m}_k + \mathbf{x}_{k+1}}{k+1}, \tag{2.3}$$

and in $\mathbb{R}^n$, the two formulas above are in fact equivalent. However, due to their algebraic appearances, the underlying geometry of the two formulas are often overlooked, and on non-Euclidean manifolds where the relations between geometry and algebra are no longer as transparent, proper generalizations of these two formulas must rely on their geometric interpretations rather than their algebraic forms.

Specifically, Eq. (2.2) can be generalized geometrically to any Riemannian manifold $\mathcal{M}$ as the center of mass of the finite samples, according to

$$\mathbf{m}_k = \arg\min_{\mathbf{z}\in\mathcal{M}} \sum_{i=1}^{k} \mathbf{d}^2_{\mathcal{M}}(\mathbf{z}, \mathbf{x}_i), \tag{2.4}$$

and computationally, it can be solved as an optimization problem on $\mathcal{M}$ for each k. On the other hand, the incremental form in Eq. (2.3) involves only two points and in a Euclidean space $\mathbb{R}^n$, we can interpret it geometrically as moving an appropriate distance away from $\mathbf{m}_k$ towards $\mathbf{x}_{k+1}$ on the straight line joining $\mathbf{x}_{k+1}$ and $\mathbf{m}_k$. This geometric procedure can be readily extended to any Riemannian manifold using geodesics, and for classical spaces such as the aforementioned examples, there are often closed-form formulas for geodesics joining two given points. This readily yields an algorithm for computing $\mathbf{m}_k$ that does not require any function optimization, a considerable advantage often realized as gains in computation time of several orders in magnitude over non-incremental algorithms based on Eq. (2.4). However, because of the presence of curvature, generalizations of Eqs. (2.2) and (2.3) are no longer equivalent as in the Euclidean case, and the incremental computation of $\mathbf{m}_k$ will also depend on the ordering of the sequence $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k$ with the latter non-commutative property marking the qualitative divergence between the Euclidean and non-Euclidean cases. In particular, it is not immediately clear that $\mathbf{m}_k$ will indeed converge asymptotically to the true expectation $\mathbf{m}$.

In this chapter, we establish the convergence of the incremental algorithm for simply connected complete Riemannian manifolds with nonpositive sectional curvature, and the proof presented in Sect. 2.2 is geometric in nature and elementary in detail. The basic idea is to use the Riemannian exponential and logarithm maps to transfer the problem from the manifold $\mathcal{M}$ to the tangent space $\mathbf{T}_\mathbf{m}$ at $\mathbf{m}$, and in the tangent space, we can compare the magnitudes of the incremental mean $\mathbf{m}_{k+1}$ computed by iFEE and the Euclidean mean computed by Eq. (2.3). An important geometric consequence of the nonpositive sectional curvature assumption is that the former is always not larger than the latter, and this provides the desired contraction that can be deployed in an inductive argument similar to the one for proving the law of large numbers in the Euclidean domain. In particular, the proof illustrates and illuminates the effect of curvature on the convergence of the incremental estimator, with the convergence in the Euclidean case (zero curvature) implicitly implying the convergence for all negative curvature cases. Furthermore, the convergence result also provides a geometric generalization of the law of large numbers in that the well-known sample average in the Euclidean law of large numbers is now replaced by the geometric operation of moves on geodesics.

In the statistics literature, manifold-valued random variables, or more general metric space-valued random variables, have been studied quite extensively, e.g., [9, 10, 24, 31, 46, 56]. However, the focus has always been on establishing and characterizing convergence of finite-sample means, and the attention is on extending

Eq. (2.2) to more general domains. However, in our more application-oriented context, the focus is instead on generalizing the incremental form in Eq. (2.3), and this provides a context that is qualitatively different from the aforementioned works in statistics. Perhaps the most well-known example is the incremental principal component analysis (PCA) [35] and its various applications in computer vision and image processing (e.g., [42]) that demand and hence motivate the incrementalization of a well-known method originated from pure statistics. In particular, computation of the mean from sample data plays an important role in a variety of applications such as clustering, segmentation, and atlas construction in medical imaging, and the performances of these algorithms are often determined by how efficiently and accurately can the mean be computed. As an incremental algorithm, iFEE provides a far more computationally efficient alternative to the standard non-incremental algorithms that compute the Fréchet mean based on minimizing Eq. (2.4). The (asymptotic) accuracy and efficiency of iFEE have been thoroughly evaluated using experiments with synthetic and real data, and in the experiment section of this chapter, we report the significant gains in running time achieved by iFEE over other non-incremental methods without any noticeable degradation in its accuracy.

2.2 Algorithm and Convergence Analysis

In this section, we present the incremental algorithm for computing the Fréchet expectation on general Riemannian manifolds and provide a convergence proof of the incremental algorithm for simply connected and complete Riemannian manifolds with nonpositive sectional curvature. Although phrased in the context of Riemannian geometry, the incremental algorithm and most of the convergence proof do not require much beyond the elementary notions such as Riemannian exponential and logarithm maps that are familiar to a large section of the computer vision and medical image computing audience. For simplicity of exposition, we will assume $\mathcal{M}$ is a simply connected and complete Riemannian manifold with nonpositive sectional curvature such that the Riemannian exponential map $\mathbf{Exp}_{\mathbf{x}}$ and its inverse, the Riemannian logarithm map $\mathbf{Log}_{\mathbf{x}}$, based at every $\mathbf{x} \in \mathcal{M}$ are diffeomorphisms between the tangent space $\mathbf{T}_{\mathbf{x}}$ and $\mathcal{M}$. In particular, $\mathcal{M}$ is assumed to have the $\mathbb{R}^n$ topology[1] and by Hopf–Rinow Theorem [16], any two points $\mathbf{x}, \mathbf{y}$ in $\mathcal{M}$ can be joined by a unique geodesic whose length gives the Riemannian distance $\mathbf{d}_{\mathcal{M}}(\mathbf{x}, \mathbf{y})$ between $\mathbf{x}$ and $\mathbf{y}$. An important example of a complete Riemannian manifold with nonpositive sectional curvature topologically equivalent to $\mathbb{R}^k$ for some k is the space $\mathbb{P}(n)$ of n-by-n symmetric positive matrices equipped with the affine-invariant Riemannian metric.

[1]This topological assumption is not particularly restrictive since by Cartan–Hadamard Theorem [7], any simply connected complete d-dimensional Riemannian manifold with nonpositive sectional curvature can be constructed topologically from $\mathbb{R}^d$.

Let $\mathbf{x}$ denote an $\mathcal{M}$-valued random variable and dP its associated probability measure (distribution) on $\mathcal{M}$ that is absolutely continuous with respect to the Riemannian volume measure $d\mathbf{x}$ with density function $P(\mathbf{x})$. Since $\mathbf{Exp_x}$ is a diffeomorphism between $\mathbf{T_x}$ and $\mathcal{M}$, we can use $\mathbf{Exp_x}$ to pull the distribution dP (and its density function) back to $\mathbf{T_x}$, essentially using $\mathbf{Exp_x}$ to identify distributions on $\mathcal{M}$ and $\mathbf{T_x}$. When there is no possibility of confusion, we will use the same notation for such a pair of distributions.[2] By a theorem of E. Cartan (see P. 256 and Proposition 60 on P. 234 in [7]), dP has a unique (Fréchet) expectation $\mathbf{m}$ defined by Eq. (2.1), and as the minimum, the first-order stationary condition gives

$$\int_{\mathcal{M}} \mathbf{Log_m}(\mathbf{x})\, P(\mathbf{x}) d\mathbf{x} = 0. \tag{2.5}$$

Using $\mathbf{Exp_m}$ and $\mathbf{Log_m}$ to identify distributions on $\mathcal{M}$ and $\mathbf{T_m}$, the integral above can be transferred onto $\mathbf{T_m}$:

$$\int_{\mathbf{T_m}} x P(x) dx = 0, \tag{2.6}$$

where the distribution $P(x)dx$ on $\mathbf{T_m}$ in Eq. (2.6) is the pull back of the distribution $P(\mathbf{x})d\mathbf{x}$ on $\mathcal{M}$ in Eq. (2.5) using $\mathbf{Exp_m}$.

2.2.1 The Incremental Fréchet Expectation Estimator

Let $\mathbf{x}_1, \mathbf{x}_2, \ldots$ be a sequence of i.i.d. samples of the probability distribution dP. The incremental Fréchet expectation estimator $\mathbf{m}_k$, $k = 1, 2, \ldots$ is defined as follows:

1. $\mathbf{m}_1 = \mathbf{x}_1$.
2. For $k > 1$, let $\gamma_k(t)$ denote the unique geodesic joining $\mathbf{m}_{k-1}$ and $\mathbf{x}_k$ such that $\gamma_k(0) = \mathbf{m}_{k-1}$ and $\gamma_k(1) = \mathbf{x}_k$. $\mathbf{m}_k = \gamma_k(\frac{1}{k})$.

Since the geodesic $\gamma_k(t)$ is constant speed [16], it follows from the definition of $\mathbf{m}_k$ that $\mathbf{d}_{\mathcal{M}}(\mathbf{x}_k, \mathbf{m}_k) = (k-1)\, \mathbf{d}_{\mathcal{M}}(\mathbf{m}_{k-1}, \mathbf{m}_k)$. In particular, when $k = 2$, $\mathbf{m}_2$ is simply the midpoint on the geodesic joining $\mathbf{x}_1$ and $\mathbf{x}_2$. For $k > 2$, the estimator $\mathbf{m}_k$ is closer to the previous estimator $\mathbf{m}_{k-1}$ than to the new sample $\mathbf{x}_k$ by a factor of $k-1$, a direct generalization of the Euclidean formula in Eq. (2.3), interpreted geometrically. The incremental estimator $\mathbf{m}_k$ is also a $\mathcal{M}$-valued random variable, and $\mathbf{x}_i$ being independent samples implies that $\mathbf{m}_k$ is also independent of $\mathbf{x}_j$ for $j > k$. We will denote $dP(\mathbf{m}_k)$ its probability measure and $P(\mathbf{m}_k)\, d\mathbf{m}_k$ its density function with respect to the Riemannian volume measure ($\mathbf{m}_k$ in $d\mathbf{m}_k$ indicates the variable of integration).

[2]Points in $\mathcal{M}$ are denoted by boldface font and their corresponding points in the tangent space are denoted by regular font.

The convergence result established below is a particular type of convergence called the convergence *in probability* of the random variables $\mathbf{m}_k$ to the expectation $\mathbf{m}$ (see [32]). Specifically, for this particular type of convergence, we need to show that for sufficiently large k, the probability that $\mathbf{d}^2_{\mathcal{M}}(\mathbf{m}_k,\ \mathbf{m})$ is larger than any $\epsilon > 0$ can be made arbitrary small: for $\epsilon, \delta > 0$, there exists an integer $K(\epsilon, \delta)$ depending on ϵ, δ such that

$$\mathbf{Pr}\,\{\,\mathbf{d}^2_{\mathcal{M}}(\mathbf{m}_k,\ \mathbf{m}) > \epsilon\,\} < \delta, \tag{2.7}$$

for $k > K(\epsilon, \delta)$. A simple application of the Markov's inequality then shows that it is sufficient to demonstrate that the mean squared error $\mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k)$ of $\mathbf{m}_k$ with respect to $\mathbf{m}$ defined by

$$\mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k) = \int_{\mathcal{M}} \mathbf{d}^2_{\mathcal{M}}(\mathbf{m}_k,\ \mathbf{m}) P(\mathbf{m}_k)\, d\mathbf{m}_k \tag{2.8}$$

converges to zero as $k \to \infty$. Recall that for a nonnegative real-valued random variable $\mathbf{X}$, Markov's inequality states that

$$\mathbf{Pr}\,\{\mathbf{X} > \epsilon\,\} \le \frac{\mathbf{E}[\mathbf{X}]}{\epsilon}, \tag{2.9}$$

where $\mathbf{E}[\mathbf{X}]$ is the expectation of $\mathbf{X}$. In our context, $\mathbf{X} = \mathbf{d}^2_{\mathcal{M}}(\mathbf{m}_k,\ \mathbf{m})$, and $\mathbf{E}[\mathbf{X}] = \mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k)$. Therefore, if $\mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k) \to 0$ as $k \to \infty$, Eq. (2.7) is true for any $\epsilon, \delta > 0$ with $k > K(\epsilon, \delta)$: since $\mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k) \to 0$ as $k \to \infty$, there is a $K(\epsilon, \delta)$ such that for all $k > K(\epsilon, \delta)$, $\mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k) = \mathbf{E}[\mathbf{X}] < \epsilon\delta$, and Eq. (2.7) follows readily from Eq. (2.9).

The convergence proof will also rely on the variance of the random variable $\mathbf{x}$ defined by $\mathbf{Var}(\mathbf{x}) = \int_{\mathcal{M}} \mathbf{d}^2_{\mathcal{M}}(\mathbf{x},\ \mathbf{m}) P(\mathbf{x})\, d\mathbf{x}$. In general, the integrals in $\mathbf{Var}(\mathbf{x})$ and $\mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k)$ are difficult to evaluate directly on $\mathcal{M}$. However, considerable simplification is possible if the integrals are pulled back to the tangent space at $\mathbf{m}$. Specifically, since $\mathbf{d}^2_{\mathcal{M}}(\mathbf{m}_k,\ \mathbf{m})$ can be determined as the squared $\mathbf{L}^2$ norm of the tangent vector $\mathbf{Log}_{\mathbf{m}}(\mathbf{m}_k)$ in $\mathbf{T}_{\mathbf{m}}$, the integral above has a much simpler form on $\mathbf{T}_{\mathbf{m}}$:

$$\mathbf{MSE}_{\mathbf{m}}(\mathbf{m}_k) = \int_{\mathbf{T}_{\mathbf{m}}} \|m_k\|^2 P(m_k)\, dm_k. \tag{2.10}$$

We remind the reader that the measures $P(\mathbf{m}_k)\, d\mathbf{m}_k$, $P(m_k)\, dm_k$ in the two equations above, although defined on different domains, are identified via $\mathbf{Exp}_{\mathbf{m}}$.

2.2.2 The Example of $\mathbb{P}(n)$

From an application viewpoint, $\mathbb{P}(n)$ equipped with the affine ($\mathbf{GL}$) invariant metric is certainly the most important example of complete Riemannian manifold of non-positive sectional curvature. Specifically, the general linear group $\mathbf{GL}(n)$ (n-by-n non-singular matrices) acts on $\mathbb{P}(n)$ according to the formula: $\forall \mathbf{g} \in \mathbf{GL}(n), \forall \mathbf{M} \in \mathbb{P}(n)$, $\mathbf{g}^*(\mathbf{M}) = \mathbf{gMg}^\top$. The action is not only transitive but for any pair of $\mathbf{M}, \mathbf{N} \in \mathbb{P}(n)$, there exists a $\mathbf{g} \in \mathbf{GL}(n)$ such that $\mathbf{gMg}^\top$ is the identity matrix and $\mathbf{gNg}^\top$ is a diagonal matrix, a geometric interpretation of the well-known algebraic fact that a pair of SPD matrices can be simultaneously diagonalized. The tangent space at each point in $\mathbb{P}(n)$ is identified with the vector space of n-by-n symmetric matrices, and a $\mathbf{GL}$-invariant Riemannian metric [26] can be specified by the inner product for the tangent space $\mathbf{T_M}$ of $\mathbf{M} \in \mathbb{P}(n)$: $< \mathbf{U}, \mathbf{V} >_{\mathbf{M}} = \mathbf{tr}(\mathbf{M}^{-1}\mathbf{U}\mathbf{M}^{-1}\mathbf{V})$, where $\mathbf{U}, \mathbf{V}$ are tangent vectors considered as symmetric matrices and $\mathbf{tr}$ denotes the trace of a matrix. The differential geometry of $\mathbb{P}(n)$ has been studied extensively by the differential geometer Helgason [26], and the invariance of the metric under $\mathbf{GL}$-action makes many aspects of its geometry tractable. For example, $\mathbb{P}(n)$ is a complete Riemannian manifold with negative sectional curvature, and there is a closed-form formula for the Riemannian distance between $\mathbf{M}, \mathbf{N} \in \mathbb{P}(n)$:

$$\mathbf{d}^2_{\mathbb{P}(n)}(\mathbf{M}, \mathbf{N}) = \mathbf{tr}(\mathbf{Log}(\mathbf{M}^{-1}\mathbf{N})^2), \tag{2.11}$$

where $\mathbf{Log}$ denotes the matrix logarithm. Furthermore, there is also a closed-form formula for the (unique) geodesic joining any pair of points $\mathbf{M}, \mathbf{N}$ in $\mathbb{P}(n)$. Because of the invariance, the geodesic paths in $\mathbb{P}(n)$ originate from geodesic curves joining the identity matrix to diagonal matrices. Specifically, let $\mathbf{D}$ denote a diagonal matrix with positive diagonal entries $d_1, d_2, \ldots, d_n > 0$. The geodesic path $\gamma(t)$ joining the identity matrix and $\mathbf{D}$ is given by

$$\gamma(t) = \begin{pmatrix} e^{t\log(d_1)} & & 0 \\ & \ddots & \\ 0 & & e^{t\log(d_n)} \end{pmatrix} \quad \text{or} \quad \gamma(t) = \begin{pmatrix} d_1^t & & 0 \\ & \ddots & \\ 0 & & d_n^t \end{pmatrix}. \tag{2.12}$$

It is evident that $\gamma(0) = \mathbf{I}_n$ is the identity matrix and $\gamma(1) = \mathbf{D}$. In particular, $\gamma(t) = \mathbf{D}^t$, as the fractional power of the diagonal matrix. For a general SPD matrix $\mathbf{M}$, its fractional power can be defined knowing its eigen-decomposition: $\mathbf{M} = \mathbf{UDU}^\top$ with $\mathbf{D}$ diagonal and $\mathbf{U}$ orthogonal, its power $\mathbf{M}^t$ for $t \geq 0$ is $\mathbf{M}^t = \mathbf{UD}^t\mathbf{U}^\top$. These matrix operations provide the steps for computing the geodesic joining any pairs of $\mathbf{M}, \mathbf{N} \in \mathbb{P}(n)$: using $\mathbf{g} = \mathbf{M}^{-\frac{1}{2}}$, $(\mathbf{M}, \mathbf{N})$ can be transformed simultaneously to $(\mathbf{I}_n, \mathbf{M}^{-\frac{1}{2}}\mathbf{N}\mathbf{M}^{-\frac{1}{2}})$, and the geodesic path $\gamma(t)$ joining $\mathbf{M}, \mathbf{N}$ is the transform under $\mathbf{M}^{\frac{1}{2}}$ of the geodesic path $\bar{\gamma}(t)$ joining $(\mathbf{I}, \mathbf{M}^{-\frac{1}{2}}\mathbf{N}\mathbf{M}^{-\frac{1}{2}})$. Let $\mathbf{M}^{-\frac{1}{2}}\mathbf{N}\mathbf{M}^{-\frac{1}{2}} = \mathbf{UDU}^\top$ denote the eigen-decomposition of $\mathbf{M}^{-\frac{1}{2}}\mathbf{N}\mathbf{M}^{-\frac{1}{2}}$, and the geodesic path joining $(\mathbf{I}, \mathbf{M}^{-\frac{1}{2}}\mathbf{N}\mathbf{M}^{-\frac{1}{2}})$ is the transform under $\mathbf{U}$ of the geodesic path joining $\mathbf{I}, \mathbf{D}$, with the latter being $\mathbf{D}^t$. Working backward, this gives the simple formulas $\overline{\gamma(t)} = (\mathbf{M}^{-\frac{1}{2}}\mathbf{N}\mathbf{M}^{-\frac{1}{2}})^t$, and $\gamma(t) = \mathbf{M}^{\frac{1}{2}}(\mathbf{M}^{-\frac{1}{2}}\mathbf{N}\mathbf{M}^{-\frac{1}{2}})^t\mathbf{M}^{\frac{1}{2}}$. Using the above formula, the iFEE has a particularly simple form: given the i.i.d. samples $\mathbf{x}_1, \mathbf{x}_2, \ldots.$ in $\mathbb{P}(n)$, the incremental estimator is given by

$$\mathbf{m}_1 = \mathbf{x}_1 \tag{2.13}$$

$$\mathbf{m}_{k+1} = \mathbf{m}_k^{\frac{1}{2}}(\mathbf{m}_k^{-\frac{1}{2}}\mathbf{x}_{k+1}\mathbf{m}_k^{-\frac{1}{2}})^{\frac{1}{k+1}}\mathbf{m}_k^{\frac{1}{2}}. \tag{2.14}$$

We remark that the computation of $\mathbf{m}_{k+1}$ requires the eigen-decomposition of the SPD matrix $\mathbf{m}_k^{-\frac{1}{2}}\mathbf{x}_{k+1}\mathbf{m}_k^{-\frac{1}{2}}$ that can be done using many efficient and robust numerical algorithms.

2.2.3 Convergence Analysis

The analysis of the estimator's convergence rests on the asymptotic behavior of $\mathbf{MSE_m}(\mathbf{m}_k)$ defined in Eq. (2.8) and its equivalent form defined on the tangent space $\mathbf{T_m}$ in Eq. (2.10). In particular, $\mathbf{m}_k$ *converges to* $\mathbf{m}$ *in the L2 norm* if $\mathbf{MSE_m}(\mathbf{m}_k) \to 0$ as $k \to \infty$ [32]. An important element in the proof of the latter is a crucial upper bound for the geodesic distance $\mathbf{d}_{\mathcal{M}}(\mathbf{m}_{k+1}, \mathbf{m})$ between $\mathbf{m}_{k+1}$ and $\mathbf{m}$ using the Euclidean distance in $\mathbf{T_m}$, which is a consequence of the nonpositive curvature assumption. The specific detail is illustrated in Fig. 2.1 and given in the Proposition below. The basic idea is that by lifting $\mathbf{m}_k$ and $\mathbf{x}_{k+1}$ back to $\mathbf{T_m}$ using the Riemannian

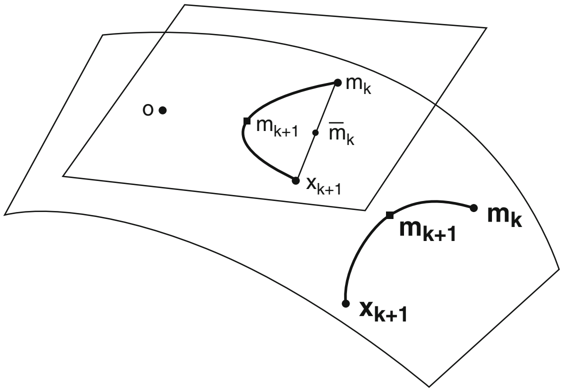


Fig. 2.1 Geometric consequences of nonpositive curvature assumption. The three points $\mathbf{m}_k, \mathbf{x}_{k+1}, \mathbf{m}_{k+1}$ are on a geodesic $\gamma(t)$ in $\mathcal{M}$. After lifting back to the tangent space $\mathbf{T_m}$ at $\mathbf{m}$, the point $m_{k+1} = \mathbf{Log_m}(\mathbf{m}_{k+1})$ is closer to the origin than the corresponding point $\overline{m}_k$ on the straight line joining m_k and x_{k+1}. Note that the Euclidean distance between the origin and m_{k+1} is the same as the geodesic distance $\mathbf{d}_{\mathcal{M}}(\mathbf{m}_{k+1}, \mathbf{m})$

logarithm map $\mathbf{Log_m}$, a triangle Σ can be formed in $\mathbf{T_m}$ using the three points m_k, x_{k+1}, and $\mathbf{o}$, the origin. The geodesic distance $\mathbf{d}_{\mathcal{M}}(\mathbf{m}_{k+1}, \mathbf{m})$ is then bounded by the Euclidean distance between the origin $\mathbf{o}$ and the corresponding point $\overline{m}_k$ on Σ. This upper bound in terms of the Euclidean length of a vector in $\mathbf{T_m}$ is useful because the resulting integral that gives the desired upper bound for the MSE in Eq. (2.10) can be easily evaluated in an inductive argument using the distributions $P(m_{k-1})$ and $P(x)$ instead of $P(m_k)$. Specifically, we have

Proposition 2.1. *Let* $\mathbf{x}, \mathbf{y}, \mathbf{z}$ *be three points on a complete Riemannian manifold* $\mathcal{M}$ *with nonpositive sectional curvature and* $\gamma(t)$ *the unique geodesic path joining* $\mathbf{x}, \mathbf{y}$ *such that* $\gamma(0) = \mathbf{x}, \gamma(1) = \mathbf{y}$. *Furthermore, let* $x = \mathbf{Log_z}(\mathbf{x}), y = \mathbf{Log_z}(\mathbf{y})$, *and* $\overline{\gamma}(t)$ *denote the straight line joining* x, y *such that* $\overline{\gamma(0)} = x, \overline{\gamma(1)} = y$. *Then,* $\mathbf{d}_{\mathcal{M}}(\gamma(t), \mathbf{z}) \le \|\overline{\gamma(t)}\|$.

The proposition is a consequence of the nonpositive assumption on curvature. Geometrically, it asserts that the geodesic distance between $\mathbf{m}$ and any point on the geodesic $\gamma(t)$ cannot be greater than the Euclidean distance in $\mathbf{T_m}$ between the origin and the corresponding point on the line joining x and y. The proof of the proposition is relegated to the appendix.

Armed with the above proposition, the proof of the convergence of the iFEE is straightforward and essentially follows from the law of large numbers in the Euclidean case.

Theorem 2.1. *The incremental Fréchet expectation estimator* $\mathbf{m}_k$ *converges to the true Fréchet expectation* $\mathbf{m}$ *in probability, i.e., as* $k \to \infty$,

$$\mathbf{MSE_m}(\mathbf{m}_k) \to 0.$$

Proof. We will inductively show that

$$\mathbf{MSE_m}(\mathbf{m}_k) \le \frac{1}{k}\mathbf{Var}(\mathbf{x}). \tag{2.15}$$

Since $\mathbf{MSE_m}(\mathbf{m}_1) = \mathbf{Var}(\mathbf{x})$, the inequality clearly holds for $k = 1$. For $k + 1 > 1$, we have, by Proposition 2.1,

$$\mathbf{MSE_m}(\mathbf{m}_{k+1}) \le \int_{\mathbf{T_m}} \| \frac{k\, m_k + x}{k+1} \|^2 \, P(m_k)P(x) dm_k dx.$$

The integral on the right can be evaluated as

$$\begin{aligned} &\int_{\mathbf{T_m}} \| \frac{k\, m_k + x}{k+1} \|^2 \, P(m_k)P(x)\, dm_k dx \\ &\quad = \int_{\mathbf{T_m}} \frac{k^2}{(k+1)^2} \, \|m_k\|^2 P(m_k)\, dm_k \end{aligned}$$

$$+ 2\frac{k}{(k+1)^2}\int_{\mathbf{T_m}}\int_{\mathbf{T_m}} m_k^\top x\, P(m_k)P(x)\, dm_k dx$$
$$+ \int_{\mathbf{T_m}} \frac{1}{(k+1)^2}\, \|x\|^2\, P(x)\, dx$$
$$= \frac{k^2}{(k+1)^2}\mathbf{MSE_m}(\mathbf{m}_k) + \frac{1}{(k+1)^2}\mathbf{Var(x)} \leq \frac{1}{k+1}\mathbf{Var(x)},$$

where the last inequality follows from the induction hypothesis, and

$$\int_{\mathbf{T_m}}\int_{\mathbf{T_m}} m_k^\top x\, P(m_k)P(x)\, dm_k dx = 0$$

follows from Eq. (2.6), although $\int_{\mathbf{T_m}} m_k P(m_k) dm_k$ is not guaranteed to be zero.

We remark that although Proposition 2.1 is valid for any other point $\mathbf{z} \neq \mathbf{m}$ in $\mathcal{M}$, the cross-term in the sum above $\int_{\mathbf{T_m}}\int_{\mathbf{T_m}} m_k^\top x\, P(m_k)P(x)\, dm_k dx$, when evaluated at $\mathbf{T_z}$, is generally nonzero. In particular, the above argument only works for the unique Fréchet expectation $\mathbf{m}$.

2.3 Related Work

Manifold-valued random variables or, more generally, random variables taking values on general metric spaces have been studied quite extensively in probability and statistics literature since the seminal work of Fréchet [23]. In the context of Riemannian manifolds, the notion of center of mass that is equivalent to the Fréchet expectation as defined by Eq. (2.1) was initially introduced by E. Cartan (see Sect. 6.1.5 in [7]), who established, among many other things, the uniqueness of Riemannian center of mass for complete manifolds of nonpositive sectional curvature. For general Riemannian manifolds, the Fréchet expectation is unique only for distributions with some special properties, for example, when their supports are contained in convex geodesic balls. In statistics literature, the primary interest and focus are on establishing the convergence of finite-sample means to the true expectation. With random variables taking values in general metric spaces and the non-uniqueness of expectation, characterizations of convergence require more elaborate machinery to account for the marked increase in the topological and geometric complexity. The basic form of the general law of large numbers for metric space-valued random variables was first established in [46, 56] under different assumptions on the metric spaces and the types of convergence. This abstract framework has been applied to study concrete statistical problems such as procrustean shape analysis [33], and several recent works [24, 31] have substantially extended the scope of these earlier works, both in abstraction and in application. In the context of Riemannian manifolds, the pair of papers [9, 10] provide some of the basic results, including a very general central limit theorem and several concrete examples concerning both the intrinsic and extrinsic means of several classical manifolds such as the complex projective spaces used in procrustean analysis. We remark that the idea of computing the mean incrementally and the question of its convergence do not seem to have been discussed nor studied in these works.

In the conference version of this chapter [18] we proved the convergence of the incremental estimator under the assumptions that the distribution is symmetric and the Riemannian manifold $\mathcal{M}$ is the space of SPD matrices $\mathbb{P}(n)$ equipped with the *GL*-invariant Riemannian metric. This result and its proof were later extended to general distributions on $\mathbb{P}(n)$ in our second conference paper [28]. The proofs presented in both papers rely on an inequality that is valid only for Riemannian manifolds with nonpositive sectional curvature, and this makes the method ill suited for further extension to manifolds with positive curvature that includes important examples in computer vision applications such as Grassmannians, Stiefel manifolds, and most compact Lie groups. In particular, the connection between the Euclidean and non-Euclidean cases were not made explicit in these two earlier approaches, and the proof presented in this chapter that uses geometric comparisons is considerably more flexible in its potential for future extensions.

After the publication of [18, 28], we were made aware of the paper by Sturm [44] wherein he formulated and proved a substantially stronger convergence result for length spaces, a more general class of spaces than Riemannian manifolds. Specifically, length spaces are metric spaces that determine the distance between two points using the minimal length of a path joining them, and compared with Riemannian manifolds, length spaces retain the notion of geodesics (distance-minimizing paths) but forsake the manifold structure as well as the exponential and logarithm maps. Surprisingly, it is still possible, in the absence of a manifold structure, to define a useful notion of nonpositive curvature for length spaces, and Strum [44] has formulated and proved a convergence result for length spaces of nonpositive curvature, of which complete Riemannian manifolds of nonpositive curvature are special cases. Although Sturm's result subsumes ours, the convergence theorem and its proof presented in [44] require considerably more machinery and longer exposition to compensate for the loss of familiar structures such as the Riemannian exponential and logarithm maps. From this perspective, our present exposition offers three important contributions: First, we present a significantly shorter and more accessible convergence proof using only elementary Riemannian geometry that is familiar to general readership of the computer vision and medical image computing communities. Second, our proof provides a more transparent and clear connection between the convergence in the Euclidean domain (the law of large numbers) and the non-Euclidean manifolds. Third, and most importantly, our proof method that relies heavily on the linearization provided by the Riemannian exponential and logarithm maps should be better adapted for studying the convergence problems for more general Riemannian manifolds and analyzing other related issues. Although the generality provided by [44] is both reassuring and welcoming, applications in computer vision and machine learning often require a specialized instead of generalized mathematical context, and in particular, general results must be sharpened and improved for special cases, perhaps in terms of shorter proofs or more precise characterizations, for which our current work serves as an example.

There are several articles in literature on computing the finite sample Fréchet mean on P_n. Moakher [38], Bhatia and Holbrook [8], presented methods for computing this intrinsic mean. Independently, there was work by Fletcher and Joshi [21] that presents a gradient descent algorithm for computing the finite sample Fréchet mean of diffusion tensors (matrices in P_n). Further, Ando, Li, and Mathias (ALM) [3] presented a technique to compute the geometric mean of $n \geq 3$ SPD matrices and listed 10 properties (now called the ALM axioms) that their mean satisfied. It is to be noted that their mean distinct from the Fréchet mean of the given sample set. In [11] Bini et al. present an extension of the unweighted geometric mean of two SPD matrices to higher number of SPD matrices by using "symmetrization methods" and induction, which satisfies the ALM axioms. This however is not the Fréchet mean of the given sample set. More recently, Lim and Pálfia [36] presented results on computing a weighted inductive mean of a finite set of SPD matrices. In [36], authors restrict themselves to discrete probability densities on P_n unlike the work presented in this chapter where we consider the continuous densities. For other types of means, we refer the reader to the excellent discussion in Pennec [40] wherein a gradient descent algorithm is presented for finding the finite sample Fréchet mean for simply connected nonpositively curved Riemannian manifolds with curvature bounded from below. In [1], Afsari et al. present a comprehensive set of results on the convergence of the gradient descent algorithm for finding the Riemannian center of mass (a.k.a. finite sample Fréchet mean) on Riemannian manifolds.

In Medical Image Computing, an impetus for developing efficient algorithms for computing means from samples is provided by the prominent role played by the mean tensor (using various kinds of distances/divergences) in solving a wide range of important problems that include diffusion tensor image (DTI) as well as structure tensor field segmentation, interpolation, clustering, and atlas construction. For example, authors in [51] generalize the geometric active contour-based piecewise constant segmentation [15, 47] to segmentation of DTIs using the Euclidean distance to measure the distance between two SPD tensors. Authors in [20] present a geometric active contour-based approach [13, 37] for tensor field segmentation that used information from the diffusion tensors to construct the so-called structure tensor which was a sum of structure tensors formed from each component of the diffusion tensor. A Riemannian metric on the manifold of SPD matrices was used in [5, 39, 41] for DTI segmentation and for computing the mean interpolant of diffusion tensors, respectively. In [52, 53, 57] and [39], the symmetrized KL-divergence (KL_s) was used for DTI segmentation and interpolation, respectively. Other applications of computing mean diffusion tensor field from a finite sample of diffusion tensor fields (structure tensor fields) can be found in [27, 55]. However, none of the above methods for computing the mean of SPD matrices (or SPD fields) used within the segmentation or the dictionary learning algorithms or in their own right for interpolation are in recursive form, even though it is evident that a recursive formulation would be more desirable in designing a more efficient algorithm.

2.4 Experiments

In this section, we empirically demonstrate the accuracy and efficiency of the proposed iFEE algorithm using a set of experiments. The distributions studied in these experiments are all defined on $\mathbb{P}(n)$, the space of n-by-n SPD matrices equipped with the GL-invariant Riemannian metric. All experiments reported in this section were performed on an Apple laptop with a 2.5GHz Intel Core $i5$ CPU and 4GB DDR3. All reported timings are on the aforementioned CPU.

2.4.1 Performance of the Incremental Fréchet Expectation Estimator

Symmetric Distributions We illustrate the performance of iFEE on a set of random samples on $\mathbb{P}(n)$ drawn from a symmetric distribution and compare the accuracy and computational efficiency of iFEE and the non-incremental (batch-mode) gradient descent algorithm for computing the finite sample Fréchet mean (FM) of the given data set. To this end, a set of 100 i.i.d samples from a log-normal distribution [43] on $\mathbb{P}(6)$ are generated, and the Fréchet mean is computed using iFEE as well as the batch-mode (gradient descent) method. We set the expectation and the variance of log-normal distribution to the identity matrix and one, respectively. The error in estimation is measured by the geodesic distance from each estimated point to the identity. Further, for each new sample, the computation time for each method is recorded. Figure 2.2a illustrates the significant difference in running time between iFEE and the *batch mode method denoted using the legend, FM, in this figure and figures to follow*. It can be seen that the time taken by iFEE is considerably shorter than the batch-mode (FM) method.

The accuracy errors of the two estimators are shown in Fig. 2.2b. It can be seen that the incremental estimator provides roughly the same accuracy as the batch-mode counterpart. Furthermore, for large numbers of samples, the incremental estimation error clearly converges to zero. Therefore, the algorithm performs more accurately as the number of data samples grows.

Asymmetric Distributions For asymmetric distributions, we use a mixture of two log-normal distributions on $\mathbb{P}(4)$ and repeat the same experiment as above. The first distribution in the mixture is centered at the identity matrix with the variance 0.1, and the second component is centered at a randomly chosen matrix with variance 0.2. A set of 500 samples are drawn from this distribution for the experiment. To measure the error, we compute the gradient vector of the objective function in Eq. (2.4) ($k = 500$) and its norm. Figure 2.2c depicts the timing plot for convergence of the two algorithms for samples drawn from this asymmetric distribution. The plot shown is an average over 500 runs. As evident, iFEE shows superior computational

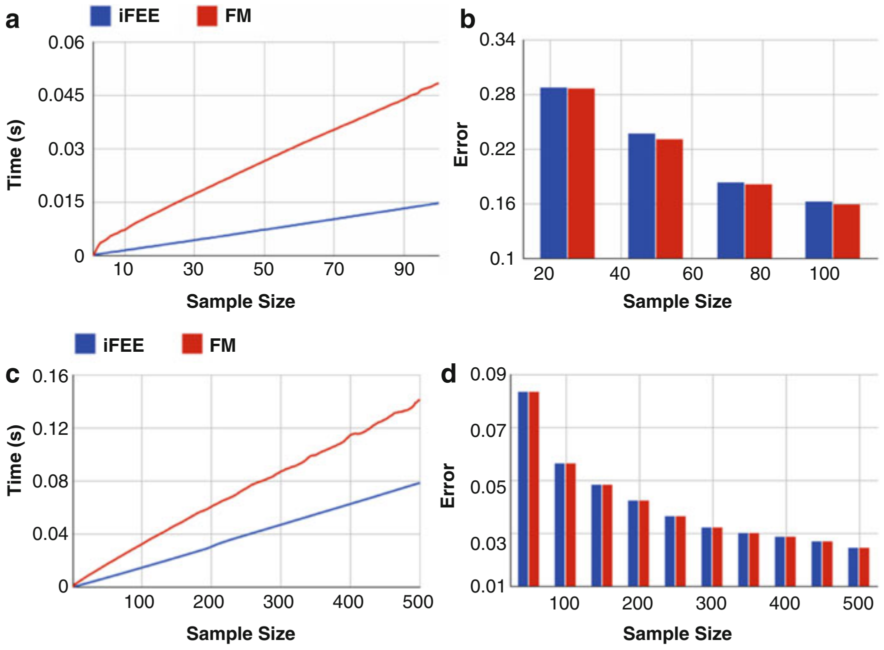


Fig. 2.2 (**a**) and (**b**): Time and error comparison of iFEE (*blue*) vs. batch-mode (*red*) Fréchet mean computation for data from a symmetric distribution on $\mathbb{P}(6)$. (**c**) and (**d**): Similar experiment for data from an asymmetric distribution on $\mathbb{P}(4)$

efficiency. Figure 2.2d depicts the error of iFEE vs. its counterpart batch-mode algorithm for this asymmetric distribution. Note that the accuracies of both the algorithms are as expected similar.

2.4.2 Application to K-Means Clustering

In this section, we evaluate the performance of our proposed incremental algorithm within the K-means clustering framework. K-means clustering is of fundamental importance for many applications in computer vision and machine learning. Due to the lack of a closed-form formula for computing the Fréchet mean, mean computation is the most time consuming step in applying K-means to SPD matrices, since at the end of each iteration the mean for each estimated cluster needs to be recomputed. The experimental results in this section demonstrate that, for SPD matrices, our iFEE algorithm can significantly speed up the clustering process—when compared with the batch mode—without any observable degradation in its accuracy.

For comparisons, we use the two different ways to compute the cluster centers: (1) the iFEE algorithm and (2) the batch-mode algorithm for computing the Fréchet mean (FM). iFEE is applied to the K-means clustering for SPD matrices as follows. At the end of each iteration of the K-means algorithm, we only consider matrices whose cluster assignments have changed. For each of these "moving" samples, the source cluster center is updated by removing the sample, and the destination cluster center is updated by adding the new sample matrix. Both these updates can be efficiently performed using our incremental formula given in Eq. (2.14), with appropriate weights. A set of experiments are presented here using different scenarios to illustrate the effectiveness of our method. In each experiment, a set of random samples from mixtures of log-normal distributions on $\mathbb{P}(n)$ are generated and used as inputs to the K-means algorithm. In the first experiment, we increase the number of samples and compare the accuracy and running time of incremental and batch-mode estimates for each case. In the second experiment, we evaluate the performance of each algorithm with respect to the matrix dimension. To measure the clustering error, the geodesic distance between each estimated cluster center and its true value is computed and these are summed over all cluster centers and reported.

Figure 2.3a, b, respectively, compare the running time and the clustering accuracy of each method with increasing number of samples. It is evident that the iFEE outperforms the batch-mode method, while the accuracy for both methods are very similar. Moreover, as the number of samples increases, iFEE improves in accuracies. Also Fig. 2.3c illustrates a significant difference in running time between these two methods, while Fig. 2.3d shows that the accuracies for both methods are roughly the same. These experiments verify that the proposed iFEE is far more computationally efficient than the batch-mode algorithm for K-means clustering applied to SPD matrices.

2.4.3 *Application to Diffusion Tensor Image Segmentation*

In this section, we present results of applying our iFEE algorithm to real data segmentation, specifically, the DTI segmentation problem. Diffusion tensors are SPD (coefficient) matrices in a quadratic approximation to the diffusivity function characterizing the water molecule diffusion in sample tissue being imaged using a medical imaging technique called diffusion magnetic resonance imaging [6]. Standard MRI acquisition is modified via the application of diffusion sensitizing magnetic field gradients at each voxel of an image grid to acquire the signal along the applied magnetic field gradient directions. One acquires these signals over a sphere of directions and at each voxel of an image grid, the diffusivity function is then approximated by a zero-mean local Gaussian, whose covariance matrix is proportional to the diffusion tensor. For more details on DTI, we refer the reader to [6].

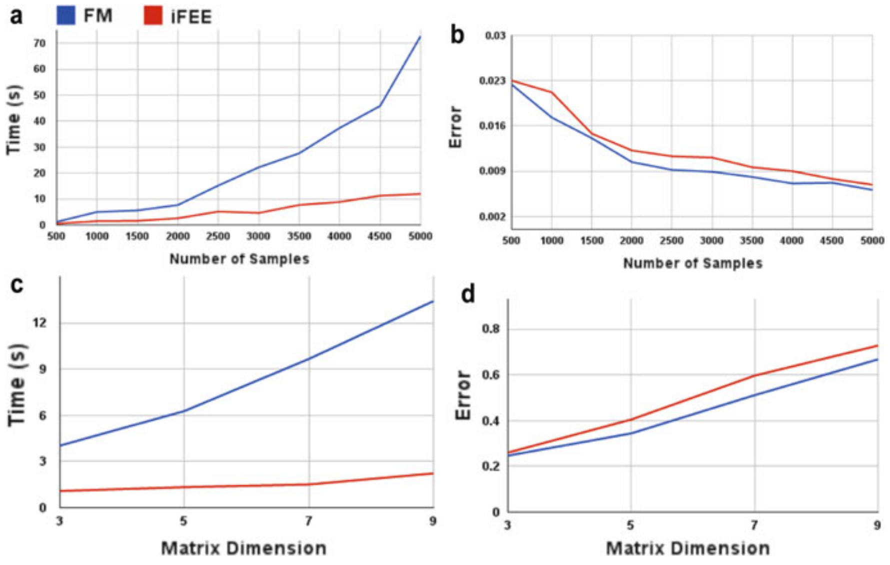


Fig. 2.3 Comparison of running times and accuracy for K-means clustering based on iFEE and batch-mode estimators for: (**a**) and (**b**) varying number of samples from three clusters on $\mathbb{P}(4)$; (**c**) and (**d**) 1000 samples from three clusters with varying sizes

In [52], the classical level-set-based (piecewise constant model) segmentation algorithm [15] was generalized to cope with a field of diffusion tensors. The constant in the piecewise constant model employed here is the mean of the tensor-valued voxels in a region. In this section, we use this algorithm to segment DTIs and use different (tensor field) mean estimation techniques within this algorithm for comparison purposes. We applied six different methods to compute the mean diffusion tensor (SPD matrices) and compared their accuracies and computation speed in the task of DTI segmentation. The first two methods used here are the proposed *iFEE* and the batch-mode Fréchet mean (*FM*) obtained from Eq. (2.4). The next two methods denoted henceforth by *KLS* and *RKLS*, respectively, are the mean computed using the symmetrized KL-divergence [52] and its recursive counterpart, reported in [18]. The last two techniques are the Log-Euclidean mean (*LEM*) [4] and its recursive version (*RLEM*) introduced in [54].

The diffusion tensors at each grid point of the image field are estimated (using the method described in [52]) from a diffusion MR scan of a rat spinal cord. The data were acquired using a 17.6-T Brucker scanner, along 21 directions with a b-value of 1000 s/mm^2. Each voxel size in the scan was $35 \times 35 \times 300\,\mu$m; and the image resolution was 128×128. Our goal here is to segment the gray matter from the DTI of the rat spinal cord. We used the same initialization for all the methods. We applied all of the six methods (incremental and batch-mode versions for each of the three "distance" measures) to perform this experiment. In order to compare the time efficiency, we report the run times for the entire gray matter segmentation

Table 2.1 Time (in seconds) for segmentation of the gray matter in a rat spinal cord

	iFEE	*FM*	*RKLS*	*KLS*	*RLEM*	*LEM*
MT	1.85	92.05	2.89	4.20	1.45	26.59
TT	52.42	147.79	54.23	59.41	66.34	117.58

MT and *TT* denote the mean computation time and total segmentation time, respectively

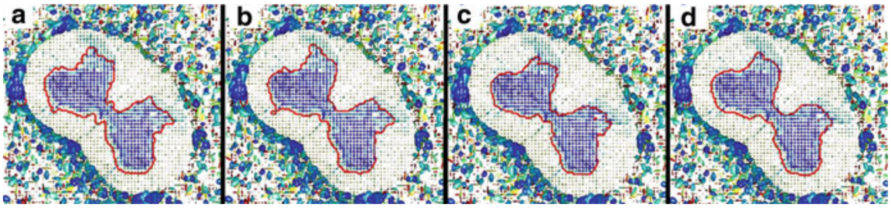


Fig. 2.4 Segmentation results of gray matter in a rat spinal cord. (**a**)–(**d**) Results from *iFEE*, *FM*, *RKLS*, and *RLEM*, respectively

process, including the total time required to compute the means. Table 2.1 shows the result of this comparison, from which we can see that FM takes nearly two thirds of the total reported segmentation time to compute the Fréchet mean, whereas using the iFEE makes the computation much faster and also significantly reduces the total segmentation time.

The segmentation results are shown in Fig. 2.4 for each method. For the sake of space, we present only the segmentation results from *iFEE*, *FM*, *RKLS*, and *RLEM* algorithms, as the results from *RKLS* and *RLEM* are visually similar to their non-incremental counterparts. The segmented region is the gray matter in the rat spinal cord. The region surrounding the entire spinal cord shown in blue is water in which the excised spinal cord was suspended for ex vivo image acquisition. Each $(3, 3)$ diffusion tensor in the DTI data is depicted using an ellipsoid, whose axis directions and lengths correspond to the eigenvectors and eigenvalues of the tensor, respectively. The color of each ellipsoid ranges from blue to red, demonstrating the lowest to highest degree of anisotropy, respectively. Moreover, the segmentation result is depicted as a curve in red overlaid on the ellipsoidal visualization of the diffusion tensor field. From the figure, we can see that the segmentation results are visually similar to each other, while the iFEE takes far less computation time, which would be very useful in practice.

2.5 Conclusions

In this chapter, we have presented a novel incremental Fréchet expectation estimator-dubbed iFEE that incrementally computes the Fréchet expectation of a distribution defined on a Riemannian manifold and presented a proof for the algorithm's convergence for simply connected and complete Riemannian manifolds with nonpositive sectional curvature. In iFEE, the intrinsic mean update is done by moving the current mean towards the new sample on the geodesic joining them; therefore, provided that the geodesics are accessible, iFEE does not require optimization and is computationally very efficient. The asymptotic accuracy of iFEE is guaranteed by the convergence analysis, and it provides an example of *geometric generalizations of the law of large numbers* in that the well-known sample average in the Euclidean law of large numbers is now replaced by the geometric operation of moves on geodesics. We have presented several experiments demonstrating the efficiency and accuracy of the iFEE algorithm.

Acknowledgements This research was supported in part by the NIH grant NS066340 to BCV.

Appendix

In this appendix, we prove Proposition 2.1 presented in Sect. 2.2. The proof is entirely elementary if we assume the general property of $\mathbf{CAT}(0)$-metric spaces [12] and Toponogov's comparison theorem (specifically, the easier half of the theorem on manifolds of nonpositive sectional curvature) [7].

Complete Riemannian manifolds of nonpositive sectional curvature form an important subclass of $\mathbf{CAT}(0)$-metric spaces [12]. For our purpose, the detailed definition of $\mathbf{CAT}(0)$-metric spaces is not necessary; instead, we will recall only the features that are used in the proof. A geodesic triangle $\mathit{\Gamma}$ on a complete Riemannian manifold $\mathcal{M}$ is the union of three geodesic segments joining three points $\mathbf{p}, \mathbf{q}, \mathbf{r} \in \mathcal{M}$: $\gamma_1(0) = \gamma_3(1) = \mathbf{p}, \gamma_1(1) = \gamma_2(0) = \mathbf{q}, \gamma_2(1) = \gamma_3(0) = \mathbf{r}$. Its comparison triangle $\mathit{\Delta}$ is a triangle in $\mathbb{R}^2$ with vertices p, q, r such that the lengths of the sides $\overline{pq}, \overline{qr}, \overline{rp}$ are equal to the lengths of the geodesic segments $\gamma_1, \gamma_2, \gamma_3$, respectively. Such comparison triangle always exists for any geodesic triangle in $\mathcal{M}$, and it is unique up to a rigid transform in $\mathbb{R}^2$. The correspondence between the three sides and segments extends naturally to points on the triangles as well: a point $\mathbf{x} \in \mathit{\Gamma}$ corresponds to a point $x \in \mathit{\Delta}$ if their associated sides correspond and their distances to the corresponding two endpoints are the same. For example, if $\mathbf{x} \in \gamma_1$ and $x \in \overline{pq}$, $\mathbf{x}$ corresponds to x if $\mathbf{d}_{\mathcal{M}}(\mathbf{x}, \mathbf{p}) = \mathbf{d}_{\mathbb{R}^2}(x, p)$, and hence $\mathbf{d}_{\mathcal{M}}(\mathbf{x}, \mathbf{q}) = \mathbf{d}_{\mathbb{R}^2}(x, q)$ as well. An important property enjoyed by any $\mathbf{CAT}(0)$-metric space is that for any pair of points $\mathbf{x}, \mathbf{y}$ on $\mathit{\Gamma}$ and their corresponding points x, y on $\mathit{\Delta}$, we have (see Fig. 2.5) $\mathbf{d}_{\mathcal{M}}(\mathbf{x}, \mathbf{y}) \leq \mathbf{d}_{\mathbb{R}^2}(x, y)$. The importance of this inequality is the upper bound given by the Euclidean distance in $\mathbb{R}^2$, and it allows us to bound the integral of the squared distance function on $\mathcal{M}$ by an integral involving squared Euclidean distance that is considerably easier to manage. Finally, for the pair of triangles $\mathit{\Gamma}, \mathit{\Delta}$, Toponogov's comparison theorem asserts the angle $\angle(rpq)$ on $\mathit{\Delta}$ no smaller than $\angle(\mathbf{rpq})$ on $\mathit{\Gamma}$.

Armed with these results, the proof of Proposition 2.1 is straightforward and it involves comparing two triangles in the tangent space $\mathbf{T_m}$. See Fig. 2.1. We restate Proposition 2.1 for convenience below and now present its proof.

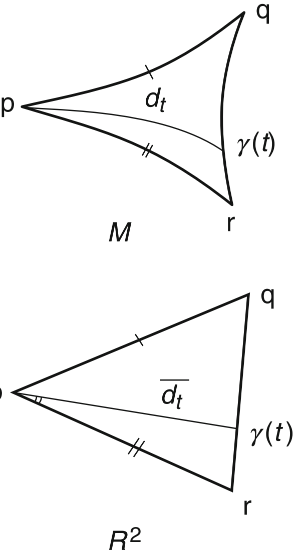


Fig. 2.5 A geodesic triangle in $\mathcal{M}$ and its comparison triangle in $\mathbb{R}^2$. Corresponding sides on the triangles have the same length. By Toponogov's comparison theorem, the angle $\angle(qpr)$ is not less than the angle $\angle(\mathbf{qpr})$ due the nonpositive sectional curvature of $\mathcal{M}$. Furthermore, if $\gamma(t), \overline{\gamma}(t)$ denote the geodesic and straight line joining $\mathbf{p}, \mathbf{q}$ and p, q, respectively, then the geodesic distance d_t between $\mathbf{p}$ and $\gamma(t)$ is not greater than the Euclidean distance $\overline{d_t}$ between p and $\overline{\gamma}(t)$, i.e., $d_t \leq \overline{d_t}$

Proposition 2.1. *Let* $\mathbf{x}, \mathbf{y}, \mathbf{z}$ *be two points on a complete Riemannian manifold* $\mathcal{M}$ *with nonpositive sectional curvature and* $\gamma(t)$ *the unique geodesic path joining* $\mathbf{x}, \mathbf{y}$ *such that* $\gamma(0) = \mathbf{x}, \gamma(1) = \mathbf{y}$. *Furthermore, let* $x = \mathbf{Log_z}(\mathbf{x}), y = \mathbf{Log_z}(\mathbf{y})$, *and* $\overline{\gamma}(t)$ *denote the straight line joining* x, y *such that* $\overline{\gamma(0)} = x, \overline{\gamma(1)} = y$. *Then,* $\mathbf{d}_{\mathcal{M}}(\gamma(t), \mathbf{z}) \leq \|\overline{\gamma(t)}\|$.

Proof. Given $\mathbf{m}_k$, $\mathbf{x}_{k+1}$ in $\mathcal{M}$, and $\mathbf{m}_{k+1}$ determined according to the iFEE algorithm, we will denote m_k, x_{k+1} and m_{k+1}, their corresponding points in $\mathbf{T_m}$ under the Riemannian logarithm map $\mathbf{Log_m}$. Without loss of generality, we will prove the proposition using $\mathbf{z} = \mathbf{m}, \mathbf{x} = \mathbf{x}_{k+1}, \mathbf{y} = \mathbf{m}_k$. Let $a = \mathbf{d}_{\mathcal{M}}(\mathbf{x}_{k+1}, \mathbf{m})$ and $b = \mathbf{d}_{\mathcal{M}}(\mathbf{m}_k, \mathbf{m})$. On $\mathbf{T_m}$, we have the first triangle Σ formed by the three vertices: x_{k+1}, m_k, and $\mathbf{o}$ the origin with the side lengths $|\overline{x_{k+1}\mathbf{o}}| = a, |\overline{m_k\mathbf{o}}| = b$. The geodesic triangle Γ on $\mathcal{M}$ spanned by $\mathbf{m}, \mathbf{x}_{k+1}, \mathbf{m}_k$ has a comparison triangle Δ in $\mathbf{T_m}$ spanned by $\mathbf{o}, p, q$ with $|\overline{p\mathbf{o}}| = a, |\overline{q\mathbf{o}}| = b$, and by Toponogov's comparison theorem,

$$\theta_\sigma \equiv \angle(x_{k+1}\mathbf{o}m_k) \leq \angle(p\mathbf{o}q) \equiv \theta_\delta,$$

since, by definition, $\angle(x_{k+1}\mathbf{o}m_k) = \angle(\mathbf{x}_{k+1}\mathbf{m}\mathbf{m}_k)$.

For completing the proof, we need to show that the distance between any point on the side $\overline{pq}$ of Δ and the origin is not greater than the distance between its corresponding point on the side $\overline{x_{k+1}m_k}$ of Σ and the origin. Specifically, a point $u \in \overline{pq}$ can be written as $u = tp + (1-t)q$, for some $0 \leq t \leq 1$ and its corresponding point v on $\overline{x_{k+1}m_k}$ is the point $v = tx_{k+1} + (1-t)m_k$. Since the triangle Δ is unique up to a rigid transform, we can, without loss of generality, assume that the two triangles Δ, Σ are contained in a two-dimensional subspace of $\mathbf{T_m}$ such that (using the obvious coordinates) they are spanned by the following two sets of three points:

$$\begin{aligned} \Delta : &\qquad (0,0), (a,0),\ (b\cos\theta_\delta,\ b\sin\theta_\delta), \\ \Sigma : &\qquad (0,0), (a,0),\ (b\cos\theta_\sigma,\ b\sin\theta_\sigma), \end{aligned}$$

with $\theta_\sigma \le \theta_\delta$. Consequently, $u = (ta + (1-t)b\cos\theta_\delta, (1-t)b\sin\theta_\delta)$ and $v = (ta + (1-t)b\cos\theta_\sigma, (1-t)b\sin\theta_\sigma)$, and their lengths are, respectively,

$$\begin{aligned} \|u\| &= \sqrt{t^2a^2 + 2t(1-t)ab\cos\theta_\delta + (1-t)^2}, \\ \|v\| &= \sqrt{t^2a^2 + 2t(1-t)ab\cos\theta_\sigma + (1-t)^2}. \end{aligned}$$

Since $\theta_\sigma \le \theta_\delta$, it then follows that $\|u\| \le \|v\|$.

References

1. Afsari B, Tron R, Vidal R (2013) On the convergence of gradient descent for finding the riemannian center of mass. SIAM J Control Optim 51(3):2230–2260
2. Amari S (2001) Information geometry. American Mathematical Society, Providence
3. Ando T, Li CK, Mathias R (2004) Geometric means. Linear Algebra Appl 385(2):305–334
4. Arsigny V, Fillard P, Pennec X, Ayache N (2006) Log-Euclidean metrics for fast and simple calculus on diffusion tensors. Magn Reson Med 56:411–421
5. Barmpoutis A, Vemuri BC, Shepherd TM, Forder JR (2007) Tensor splines for interpolation and approximation of DT-MRI with applications to segmentation of isolated rat hippocampi. IEEE Trans Med Imaging 26:1537–1546
6. Basser PJ, Mattiello J, Lebihan D (1994) Estimation of the effective self-diffusion tensor from NMR spin echo. J Magn Reson 103(3):247–254
7. Berger M (2007) A panoramic view of Riemannian geometry. Springer, Berlin
8. Bhatia R, Holbrook J (2006) Riemannian geometry and matrix geometric means. Linear Algebra Appl 413(2):594–618
9. Bhattacharya R, Patrangenaru V (2003) Large sample theory of intrinsic and extrinsic sample means on manifolds - I. Ann Stat 31(1):1–29
10. Bhattacharya R, Patrangenaru V (2005) Large sample theory of intrinsic and extrinsic sample means on manifolds - II. Ann Stat 33(3):1225–1259
11. Bini DA, Meini B, Poloni F (2010) An effective matrix geometric mean satisfying the ando-li-mathias properties. Math Comput 79(269):437–452
12. Bridson M, Haefliger (1999) A metric spaces of non-positive curvature. Springer, Berlin
13. Caselles V, Kimmel R, Sapiro G (1997) Geodesic active contours. Int J Comput Vis 22 (1):61–79
14. Cetingul H, Vidal R (2009) Intrinsic mean shift for clustering on Stiefel and Grassmann manifolds. In: Proceedings of IEEE international conference on computer vision and pattern recognition, pp 1896–1902
15. Chan T, Vese L (2001) Active contours without edges. IEEE Trans Image Process 10(2): 266–277
16. Cheeger J, Ebin D (2008) Comparison theorems in Riemannian geometry. American Mathematical Society, Providence
17. Cheng G, Vemuri BC (2013) A novel dynamic system in the space of SPD matrices with applications to appearance tracking. SIAM J Imag Sci 6(1):592–615

18. Cheng G, Salehian H, Vemuri BC (2012) Efficient recursive algorithms for computing the mean diffusion tensor and applications to DTI segmentation. In: ECCV, vol 7, pp 390–401
19. Doretto G, Chiuso A, Wu YN, Soatto S (2003) Dynamic textures. Int J Comput Vis 51(2): 91–109
20. Feddern C, Weickert J, Burgeth B (2003) Level-set methods for tensor-valued images. In: Proceedings of 2nd IEEE workshop on variational, geometric and level set methods in computer vision, pp 65–72
21. Fletcher T, Joshi S (2004) Principal geodesic analysis on symmetric spaces: statistics of diffusion tensors. In: Computer vision and mathematical methods in medical and biomedical image analysis, pp 87–98
22. Fletcher P, Joshi S (2007) Riemannian geometry for the statistical analysis of diffusion tensor data. Signal Process 87(2):250–262
23. Fréchet M (1948) Les éléments aléatoires de nature quelconque dans un espace distancié. Ann Inst Henri Poincare 10(4):215–310
24. Ginestet C (2012) Strong and weak laws of large numbers for Frechet sample means in bounded metric spaces. arXiv:1204.3183
25. Hartley R, Trumpf J, Dai Y, Li H (2013) Rotation averaging. Int J Comput Vis 103(3):267–305
26. Helgason S (2001) Differential geometry, lie groups, and symmetric spaces. American Mathematical Society, Providence
27. Ho J, Xie Y, Vemuri BC (2013) On A nonlinear generalization of sparse coding and dictionary learning. In: ICML, pp 1480–1488
28. Ho J, Cheng G, Salehian H, Vemuri B (2013) Recursive karcher expectation estimators and geometric law of large numbers. In: Proceedings of the 16th international conference on artificial intelligence and statistics, pp 325–332
29. Horn B (1886) Robot vision. MIT Press, Cambridge
30. Kendall D (1984) Shape manifolds, procrustean metrics, and complex projective spaces. Bull Lond Math Soc 16:18–121
31. Kendall W, Le H (2011) Limit theorems for empirical Frechet means of independent and non-identically distributed manifold-valued random variables. Braz J Probab Stat 25(3):323–352
32. Khoshnevisan D (2007) Probability. Graduate studies in mathematics, vol 80. American Mathematical Society, Providence
33. Le H (2001) Locating Féchet means with application to shape spaces. Adv Appl Probab 33(2):324–338
34. Lenglet C, Rousson M, Deriche R, Faugeras O (2006) Statistics on the manifold of multivariate normal distributions: theory and application to diffusion tensor MRI processing. J Math Imaging Vision 25:423–444. doi:10.1007/s10851-006-6897-z. http://www.dx.doi.org/10.1007/s10851-006-6897-z
35. Levy A, Lindenbaum M (2000) Sequential karhunen–loeve basis extraction and its application to images. IEEE Trans Image Process 9(8):1371–1374
36. Lim Y, Pálfia M (2014) Weighted inductive means. Linear Algebra Appl 453:59–83
37. Malladi R, Sethian J, Vemuri BC (1995) Shape modeling with front propagation: a level set approach. IEEE Trans Pattern Anal Mach Intell 17(2):158–175
38. Moakher M (2005) A differential geometric approach to the geometric mean of symmetric positive-definite matrices. SIAM J Matrix Anal Appl 26(3):735–747
39. Moakher M, Batchelor PG (2006) Symmetric positive-definite matrices: from geometry to applications and visualization. Visualization and processing of tensor fields. Springer, Berlin
40. Pennec X (2006) Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. J. Math. Imaging Vision 25(1):127–154
41. Pennec X, Fillard P, Ayache N (2006) A Riemannian framework for tensor computing. Int J Comput Vis 66(1):41–66
42. Ross D, Lim J, Lin RS, Yang MH (2008) A Riemannian framework for tensor computing. Int J Comput Vis 77(1–1):125–141
43. Schwartzman A (2006) Random ellipsoids and false discovery rates: Statistics for diffusion tensor imaging data. Ph.D. thesis, Stanford University

44. Sturm KT (2003) Probability measures on metric spaces of nonpositive curvature. In: Auscher P, Coulhon T, Grigoryan A (eds) Heat kernels and analysis on manifolds, graphs, and metric spaces, vol 338. American Mathematical Society, Providence
45. Subbarao R, Meer P (2009) Nonlinear mean shift over Riemannian manifolds. Int J Comput Vis 84(1):1–20
46. Sverdrup-Thygeson H (1981) Strong law of large numbers for measures of central tendency and dispersion of random variables in compact metric spaces. Ann Stat 9(1):141–145
47. Tsai A, Yezzi AJ, Willsky A (2001) Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation, and magnification. IEEE Trans Image Process 10(8):1169–1186
48. Turaga P, Veeraraghavan A, Srivastava A, Chellappa R (2011) Statistical computations on Grassmann and Stiefel manifolds for image and video-based recognition. IEEE Trans Pattern Anal Mach Intell 33(11):2273–2286
49. Tuzel O, Porikli F, Meer P (2007) Human detection via classification on Riemannian manifolds. In: Proceedings of IEEE international conference on computer vision and pattern recognition
50. Tyagi A, Davis J (2008) A recursive filter for linear systems on Riemannian manifolds. In: Proceedings of IEEE international conference on computer vision and pattern recognition
51. Wang Z, Vemuri B (2004) Tensor field segmentation using region based active contour model. In: European conferene on computer vision (ECCV), pp 304–315
52. Wang Z, Vemuri B (2005) DTI segmentation using an information theoretic tensor dissimilarity measure. IEEE Trans Med Imaging 24(10):1267–1277
53. Weldeselassie Y, Hamarneh G (2007) DT-MRI segmentation using graph cuts. In: SPIE Medical Imaging, vol 6512
54. Wu Y, Wang J, Lu H (2009) Real-time visual tracking via incremental covariance model update on log-euclidean Riemannian manifold. In: CCPR
55. Xie Y, Vemuri BC, Ho J (2010) Statistical analysis of tensor fields. In: MICCAI, pp 682–689
56. Ziezold H (1977) On expected figures and a strong law of large numbers for random elements in quasi-metric spaces. In: Transactions of the 7th prague conference on information theory, statistical decision functions, random processes and of the 1974 European meeting of statisticians
57. Ziyan U, Tuch D, Westin C (2006) Segmentation of thalamic nuclei from DTI using spectral clustering. In: MICCAI, pp 807–814

Chapter 3
Kernels on Riemannian Manifolds

Sadeep Jayasumana, Richard Hartley, and Mathieu Salzmann

Abstract We discuss an approach to exploiting kernel methods with manifold-valued data. In many computer vision problems, the data can be naturally represented as points on a Riemannian manifold. Due to the non-Euclidean geometry of Riemannian manifolds, usual Euclidean computer vision and machine learning algorithms yield inferior results on such data. We define positive definite kernels on manifolds that permit us to embed a given manifold with a corresponding metric in a reproducing kernel Hilbert space. These kernels make it possible to utilize algorithms developed for linear spaces on nonlinear manifold-valued data.

We primarily work with Gaussian radial basis function (RBF)-type kernels. Since the Gaussian RBF defined with any given metric is not always positive definite, we present a unified framework for analyzing the positive definiteness of the Gaussian RBF on a generic metric space. We then use the proposed framework to identify positive definite kernels on three specific manifolds commonly encountered in computer vision: the Riemannian manifold of symmetric positive definite matrices, the Grassmann manifold, and Kendall's manifold of 2D shapes. We show that many popular algorithms designed for Euclidean spaces, such as support vector machines, discriminant analysis, and principal component analysis can be generalized to Riemannian manifolds with the help of such positive definite Gaussian kernels.

S. Jayasumana (✉)
University of Oxford, Oxford, UK

50b High Street, Wheatley, Oxfordshire OX33 1XT, United Kingdom
e-mail: sadeep@robots.ox.ac.uk

R. Hartley
Australian National University, Canberra, ACT, Australia
e-mail: richard.hartley@anu.edu.au

M. Salzmann
NICTA, Sydney, Australia
e-mail: mathieu.salzmann@nicta.com.au

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_3

3.1 Introduction

Mathematical entities that do not form Euclidean spaces but lie on nonlinear manifolds are often encountered in computer vision. Examples include normalized histograms that form the unit n-sphere S^n, symmetric positive definite (SPD) matrices, linear subspaces of a Euclidean space, and 2D shapes. Recently, the latter three manifolds have drawn significant attention in the computer vision community due to their widespread applications. SPD matrices, which form a Riemannian manifold when endowed with an appropriate metric [28], are encountered in computer vision in the forms of covariance region descriptors [36, 37], diffusion tensors [28], and structure tensors [13]. Linear subspaces of a Euclidean space, known to form a Riemannian manifold named the Grassmann manifold, are commonly used to model image sets [14, 15] and videos [35]. Shape descriptors can be used as strong cues in object recognition [4].

Since manifolds lack a vector space structure and other Euclidean structures such as norm and inner product, many popular computer vision and machine learning algorithms including support vector machines (SVM), principal component analysis (PCA), and mean-shift clustering cannot be applied in their original forms on manifolds. One way of dealing with this difficulty is to neglect the nonlinear geometry of manifold-valued data and apply Euclidean methods directly. As intuition suggests, this approach often yields poor accuracy and undesirable effects, as was demonstrated, for instance, in [3, 28] for SPD matrices.

When the manifold under consideration is Riemannian, another common approach used to cope with its nonlinearity consists in approximating the manifold-valued data with its projection to a tangent space at a particular point on the manifold, for example, the mean of the data. Such tangent space approximations are often calculated successively as the algorithm proceeds [37]. However, mapping data to a tangent space only yields a first-order approximation of the data that can be distorted, especially in regions far from the origin of the tangent space. Moreover, iteratively mapping back and forth to the tangent spaces significantly increases the computational cost of the algorithm. It is also difficult to decide the origin of the tangent space, which strongly affects the accuracy of this approximation.

In Euclidean spaces, the success of many computer vision algorithms arises from the use of kernel methods [31, 32]. Therefore, one could think of following the idea of kernel methods in $\mathbb{R}^n$ and embed a manifold in a high-dimensional reproducing kernel Hilbert space (RKHS), where linear geometry applies. Many Euclidean algorithms can be directly generalized to an RKHS, which is a vector space that possesses an important structure: the inner product. Such an embedding, however, requires a kernel function defined on the manifold, which, according to Mercer's theorem [31], should be positive definite. While many positive definite kernels are known for Euclidean spaces, such knowledge remains limited for manifolds.

In this chapter, we present a generalization of successful and powerful kernel methods to manifold-valued data. To this end, we analyze the Gaussian RBF kernel on a generic manifold and provide necessary and sufficient conditions for the

Gaussian RBF kernel generated by a distance function on any nonlinear manifold to be positive definite. This lets us generalize kernel methods to manifold-valued data while preserving the favorable properties of the original algorithms.

We apply the theory to analyze the positive definiteness of Gaussian kernels defined on three specific manifolds: the Riemannian manifold of SPD matrices, the Grassmann manifold, and the shape manifold. Given the resulting positive definite kernels, we discuss different kernel methods on these three manifolds, including kernel SVM, multiple kernel learning (MKL), and kernel PCA. We present experiments on a variety of computer vision tasks, such as pedestrian detection, segmentation, face recognition, and action recognition, to show that manifold kernel methods outperform the corresponding Euclidean algorithms that neglect the manifold geometry, as well as other state-of-the-art techniques specifically designed for manifolds.

3.2 Riemannian Manifolds of Interest

A *topological manifold*, generally referred to as simply a *manifold*, is a topological space that is locally homeomorphic to the n-dimensional Euclidean space $\mathbb{R}^n$, for some n. Here, n is referred to as the dimensionality of the manifold. A *differentiable manifold* is a topological manifold that has a globally defined differential structure. The tangent space at a given point on a differentiable manifold is a vector space that consists of the tangent vectors of all possible curves passing through the point.

A *Riemannian manifold* is a differentiable manifold equipped with a smoothly varying inner product on each tangent space. The family of inner products on all tangent spaces is known as the *Riemannian metric* of the manifold. It enables us to define various geometric notions on the manifold such as the angle between two curves, or the length of a curve. The *geodesic distance* between two points on the manifold is defined as the length of the shortest curve connecting them. Such shortest curves are known as *geodesics* and are analogous to straight lines in $\mathbb{R}^n$.

The geodesic distance induced by the Riemannian metric is the most natural measure of dissimilarity between two points lying on a Riemannian manifold. However, in practice, many other nonlinear distances or *metrics* which do not necessarily arise from Riemannian metrics can also be useful for measuring dissimilarity on manifolds. It is worth noting that the term *Riemannian metric* refers to a family of inner products while the term *metric* refers to a distance function that satisfies the four metric axioms. A nonempty set endowed with a metric is known as a *metric space* which is a more abstract space than a Riemannian manifold.

In the following, we provide a brief overview of the three Riemannian manifolds discussed in this chapter.

3.2.1 The Riemannian Manifold of SPD Matrices

A $d \times d$, real SPD matrix S has the property: $\mathbf{x}^T S \mathbf{x} > 0$ for all nonzero $\mathbf{x} \in \mathbb{R}^d$. The space of $d \times d$ SPD matrices, which we denote by Sym_d^+, is clearly not a vector space since an SPD matrix when multiplied by a negative scalar is no longer SPD. Instead, Sym_d^+ forms a convex cone in the d^2-dimensional Euclidean space.

The geometry of the space Sym_d^+ is best explained with a Riemannian metric that induces an infinite distance between an SPD matrix and a non-SPD matrix [2, 28]. Therefore, the geodesic distance induced by such a Riemannian metric is a more accurate distance measure on Sym_d^+ than the Euclidean distance in the d^2-dimensional Euclidean space it is embedded in. Two popular Riemannian metrics proposed on Sym_d^+ are the affine-invariant Riemannian metric [28] and the log-Euclidean Riemannian metric [2, 3]. They result in the affine-invariant geodesic distance and the log-Euclidean geodesic distance, respectively. These two distances are so far the most widely used metrics on Sym_d^+.

Apart from these two geodesic distances, a number of other metrics have been proposed for Sym_d^+ to capture its nonlinearity [33]. For a review of metrics on Sym_d^+, we refer the reader to [11].

3.2.2 The Grassmann Manifold

A point on the (n, r) Grassmann manifold, where $n > r$, is an r-dimensional linear subspace of the n-dimensional Euclidean space. Such a point is generally represented by an $n \times r$ matrix Y whose columns store an orthonormal basis of the subspace. With this representation, the point on the Grassmann manifold is the linear subspace spanned by the columns of Y or $\mathrm{span}(Y)$ which we denote by $[Y]$. We use $\mathcal{G}_n^r$ to denote the (n, r) Grassmann manifold.

The set of $n \times r$ $(n > r)$ matrices with orthonormal columns forms a manifold known as the (n, r) *Stiefel manifold*. From its embedding in the nr-dimensional Euclidean space, the (n, r) Stiefel manifold inherits a canonical Riemannian metric [12]. Points on $\mathcal{G}_n^r$ are equivalence classes of $n \times r$ matrices with orthonormal columns, where two matrices are equivalent if their columns span the same r-dimensional subspace. Thus, the orthogonal group $\mathcal{O}_r$ acts via isometries (change of orthogonal basis) on the Stiefel manifold by multiplication on the right, and $\mathcal{G}_n^r$ can be identified as the set of orbits of this action. Since this action is both free and proper, $\mathcal{G}_n^r$ forms a manifold, and it is given a Riemannian structure by equipping it with the standard *normal* Riemannian metric derived from the metric on the Stiefel manifold.

A geodesic distance on the Grassmann manifold, called the arc length distance, can be derived from its canonical geometry described above. The arc length distance between two points on the Grassmann manifold turns out to be the l^2 norm of the vector formed by the principal angles between the two corresponding subspaces.

Several other metrics on the Grassmann manifold can be derived from the principal angles. We refer the reader to [12] for more details and properties of different Grassmann metrics.

3.2.3 The Shape Manifold

A number of different representations have been proposed to capture shapes, which are geometric information invariant to translation, scale, and rotation. In this chapter, we use Kendall's formalism where a 2D shape is initially represented as an n-dimensional complex vector, where n is the number of landmarks that denote the shape [21]. The real and imaginary parts of each element of the vector encode the x and y coordinates of a landmark, respectively. Translation and scale invariances are achieved by subtracting the mean from the vector and scaling it to have unit norm. The vector $\mathbf{z}$ obtained in this manner is dubbed *pre-shape* as it is not invariant to rotation yet. Pre-shapes lie on the complex unit sphere $\mathbb{C}S^{n-1}$. To remove rotation, all rotated versions of $\mathbf{z}$ are identified and the final shape is obtained as the resulting equivalence class of pre-shapes, denoted by $[\mathbf{z}]$. Since rotations correspond to multiplication by complex numbers of unit magnitude, the shape manifold is in fact the complex projective space $\mathbb{C}P^{n-2}$. In the following, we use SP^n to denote the manifold of 2D shapes defined by n landmarks.

Arguably, the most popular distance on the shape manifold is the full Procrustes distance [21]. A detailed analysis of different metrics on the shape manifold is given in [10].

3.3 Hilbert Space Embedding of Manifolds

An *inner product space* is a vector space equipped with an inner product. A *Hilbert space* is an (often infinite-dimensional) inner product space which is complete with respect to the norm induced by the inner product. A Reproducing Kernel Hilbert Space (RKHS) is a special kind of Hilbert space of functions on some nonempty set $\mathcal{X}$ in which all evaluation functionals are bounded and hence continuous [1]. The inner product of an RKHS of functions on $\mathcal{X}$ can be defined by a bivariate function on $\mathcal{X} \times \mathcal{X}$, known as the *reproducing kernel* of the RKHS.

Many useful computer vision and machine learning algorithms developed for Euclidean spaces depend only on the notion of inner product, which allows us to measure angles and also distances. Therefore, such algorithms can be extended to Hilbert spaces without effort. A notable special case arises with RKHSs where the inner product of the Hilbert space can be evaluated using a kernel function without computing the actual vectors. This concept, known as the *kernel trick*, is commonly employed in machine learning in the following setting: input data in some n-dimensional Euclidean space $\mathbb{R}^n$ are mapped to a high-dimensional RKHS where some learning algorithm, which requires only the inner product, is applied. We never need to calculate actual vectors in the RKHS since the learning algorithm only requires the inner product of the RKHS, which can be calculated by means of a kernel function defined on $\mathbb{R}^n \times \mathbb{R}^n$. A variety of algorithms can be used with the kernel trick, such as support vector machines (SVM), principal component analysis (PCA), Fisher discriminant analysis (FDA), k-means clustering, and ridge regression.

Embedding lower-dimensional data in a higher-dimensional RKHS is commonly and successfully employed with data that lie in a Euclidean space. The theoretical concepts of such embeddings can directly be extended to manifolds. Points on a manifold $\mathcal{M}$ are mapped to elements in a high (possibly infinite)-dimensional Hilbert space $\mathcal{H}$, a subspace of the space spanned by real-valued functions[1] on $\mathcal{M}$. A kernel function $k : (\mathcal{M} \times \mathcal{M}) \to \mathbb{R}$ is used to define the inner product on $\mathcal{H}$, thus making it an RKHS. The technical difficulty in utilizing Hilbert space embeddings with manifold-valued data arises from the fact that, according to Mercer's theorem, the kernel function must be positive definite to define a valid RKHS. While many positive definite kernel functions are known for $\mathbb{R}^n$, generalizing them to manifolds is not straightforward.

Identifying such positive definite kernel functions on manifolds would, however, be greatly beneficial. Indeed, embedding a manifold in an RKHS has two major advantages: First, the mapping transforms the nonlinear manifold into a (linear) Hilbert space, thus making it possible to utilize algorithms designed for linear spaces with manifold-valued data. Second, as evidenced by the theory of kernel methods in Euclidean spaces, it yields a much richer high-dimensional representation of the original data distribution, making tasks such as classification easier.

In the next two sections, we build a generic framework that enables us to define provably positive definite Gaussian RBF-type kernels on manifolds. Our main focus is on Gaussian RBF kernels since they have proven extremely useful and versatile in Euclidean spaces. We discuss other types of kernels in Sect. 3.6.

3.4 Theory of Positive and Negative Definite Kernels

In this section, we present some general results on positive and negative definite kernels. These results will be useful for our derivations in later sections. We start with the definition of real-valued positive and negative definite kernels on a nonempty set [5]. Note that by *kernel on* $\mathcal{X}$ or *kernel on* $\mathcal{X} \times \mathcal{X}$, we mean a real-valued function on $\mathcal{X} \times \mathcal{X}$ hereafter.

Definition 1. Let $\mathcal{X}$ be a nonempty set. A kernel $f : (\mathcal{X} \times \mathcal{X}) \to \mathbb{R}$ is called **positive definite** if it is symmetric (i.e., $f(x, y) = f(y, x)$ for all $x, y \in \mathcal{X}$) and

[1]We limit the discussion to real Hilbert spaces and real-valued kernels, since they are the most useful kind in learning algorithms. However, the theory holds for complex Hilbert spaces and complex-valued kernels as well.

$$\sum_{i,j=1}^{m} c_i c_j f(x_i, x_j) \geq 0$$

for all $m \in \mathbb{N}, \{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ and $\{c_1, \ldots, c_m\} \subseteq \mathbb{R}$. The kernel f is called **negative definite** if it is symmetric and

$$\sum_{i,j=1}^{m} c_i c_j f(x_i, x_j) \leq 0$$

for all $m \in \mathbb{N}, \{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ and $\{c_1, \ldots, c_m\} \subseteq \mathbb{R}$ with $\sum_{i=1}^{m} c_i = 0$.

It is important to note the additional constraint on $\sum c_i$ for the negative definite case. Due to this constraint, some authors refer to this latter kind as *conditionally negative definite*. However, in this chapter, we stick to the most common terminology used in the literature.

Given a finite set of points $\{x_i\}_{i=1}^{m} \subseteq \mathcal{X}$ and a kernel k on $\mathcal{X}$, consider the $m \times m$ matrix K defined by $K_{ij} = k(x_i, x_j)$. If the kernel k is positive definite, the matrix K turns out to be positive semi-definite, whereas if k is negative definite, K is conditionally negative semi-definite. Note that there is an unfortunate confusion of terminology here due to different conventions in literature on kernels and linear algebra.

We now present the following theorem, which plays a central role in this chapter. It was introduced by Schoenberg in 1938 [29], well before the theory of reproducing kernel Hilbert spaces was established in 1950 [1].

Theorem 2. *Let $\mathcal{X}$ be a nonempty set and $f : (\mathcal{X} \times \mathcal{X}) \to \mathbb{R}$ be a kernel. The kernel* $\exp(-\gamma f(x, y))$ *is positive definite for all $\gamma > 0$ if and only if f is negative definite.*

Proof. We refer the reader to Theorem 3.2.2 of [5] for a detailed proof of this theorem.

Note that this theorem describes Gaussian RBF—like exponential kernels that are positive definite for all $\gamma > 0$. One might also be interested in exponential kernels that are positive definite for only some values of γ. Such kernels, for instance, were exploited in [16]. To our knowledge, there is no general result characterizing this kind of kernels. However, the following result can be obtained from the above theorem.

Theorem 3. *Let $\mathcal{X}$ be a nonempty set and $f : (\mathcal{X} \times \mathcal{X}) \to \mathbb{R}$ be a kernel. If the kernel* $\exp(-\gamma f(x, y))$ *is positive definite for all $\gamma \in (0, \delta)$ for some $\delta > 0$, then it is positive definite for all $\gamma > 0$.*

Proof. If $\exp(-\gamma f(x, y))$ is positive definite for all $\gamma \in (0, \delta)$, it directly follows from Definition 1 that $1 - \exp(-\gamma f(x, y))$ is negative definite for all $\gamma \in (0, \delta)$. Therefore, the pointwise limit

$$f(x, y) = \lim_{\gamma \to 0+} \frac{1 - \exp(-\gamma f(x, y))}{\gamma}$$

is also negative definite. Now, since $f(x, y)$ is negative definite, it follows from Theorem 2 that $\exp(-\gamma f(x, y))$ is positive definite for all $\gamma > 0$.

Next, we highlight an interesting property of negative definite kernels. It is well known that a positive definite kernel represents the inner product of an RKHS [1]. Similarly, a negative definite kernel represents the squared norm of a Hilbert space under some conditions stated by the following theorem.

Theorem 4. *Let $\mathcal{X}$ be a nonempty set and $f(x, y) : (\mathcal{X} \times \mathcal{X}) \to \mathbb{R}$ be a negative definite kernel. Then, there exists a Hilbert space $\mathcal{H}$ and a mapping $\psi : \mathcal{X} \to \mathcal{H}$ such that*

$$f(x, y) = \|\psi(x) - \psi(y)\|^2 + h(x) + h(y),$$

where $h : \mathcal{X} \to \mathbb{R}$ is a function which is nonnegative whenever f is. Furthermore, if $f(x, x) = 0$ for all $x \in \mathcal{X}$, then $h = 0$.

Proof. The proof of a more general version of this theorem can be found in Proposition 3.3.2 of [5].

Now, we state and prove a lemma that will be useful for the proof of our main theorem.

Lemma 5. *Let $\mathcal{V}$ be an inner product space. The squared l^2 distance in $\mathcal{V}$ defined by $f : (\mathcal{V} \times \mathcal{V}) \to \mathbb{R} : f(x, y) = \|x - y\|^2_{\mathcal{V}}$ is a negative definite kernel.*

Proof. The kernel f is obviously symmetric. Based on Definition 1, we then need to prove that $\sum_{i,j=1}^m c_i c_j f(x_i, x_j) \leq 0$ for all $m \in \mathbb{N}$, $\{x_1, \ldots, x_m\} \subseteq \mathcal{X}$ and $\{c_1, \ldots, c_m\} \subseteq \mathbb{R}$ with $\sum_{i=1}^m c_i = 0$. Now,

$$\begin{aligned}
\sum_{i,j=1}^m c_i c_j f(x_i, x_j) &= \sum_{i,j=1}^m c_i c_j \|x_i - x_j\|^2_{\mathcal{V}} \\
&= \sum_{i,j=1}^m c_i c_j \langle x_i - x_j, x_i - x_j\rangle_{\mathcal{V}} \\
&= \sum_{j=1}^m c_j \sum_{i=1}^m c_i \langle x_i, x_i\rangle_{\mathcal{V}} + \sum_{i=1}^m c_i \sum_{j=1}^m c_j \langle x_j, x_j\rangle_{\mathcal{V}} - 2 \sum_{i,j=1}^m c_i c_j \langle x_i, x_j\rangle_{\mathcal{V}}.
\end{aligned}$$

The first two terms vanish since $\sum_{i=1}^m c_i = 0$. Therefore,

$$\sum_{i,j=1}^m c_i c_j f(x_i, x_j) = -2 \sum_{i,j=1}^m c_i c_j \langle x_i, x_j\rangle_{\mathcal{V}} = -2 \left\| \sum_{i=1}^m c_i x_i \right\|^2_{\mathcal{V}} \leq 0.$$

3.5 Gaussian RBF Kernels on Manifolds

A number of well-known kernels exist for Euclidean spaces including the linear kernel, polynomial kernels, and the Gaussian RBF kernel. The key challenge in generalizing kernel methods from Euclidean spaces to manifolds lies in defining appropriate positive definite kernels on the manifold. There is no straightforward way to generalize Euclidean kernels such as the linear kernel and polynomial kernels to nonlinear manifolds, since these kernels depend on the linear geometry of Euclidean spaces. However, we show that the popular Gaussian RBF kernel can be generalized to manifolds under certain conditions.

In this section, we first introduce a general theorem that provides necessary and sufficient conditions to define a positive definite Gaussian RBF kernel on a given manifold and then show that some popular metrics on Sym_d^+, $\mathcal{G}_n^r$, and SP^n yield positive definite Gaussian RBFs on the respective manifolds.

3.5.1 *The Gaussian RBF on Metric Spaces*

The Gaussian RBF kernel has proven very effective in Euclidean spaces for a variety of kernel-based algorithms. It maps the data points to an infinite-dimensional Hilbert space, which, intuitively, yields a very rich representation of the data. In $\mathbb{R}^n$, the Gaussian kernel can be expressed as $k_G(\mathbf{x}, \mathbf{y}) := \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$, which makes use of the Euclidean distance between two data points $\mathbf{x}$ and $\mathbf{y}$. To define a kernel on a manifold, we would like to replace the Euclidean distance by a more accurate distance measure on the manifold. However, not all geodesic distances yield positive definite kernels. For example, in the case of the unit n-sphere embedded in $\mathbb{R}^{n+1}$, $\exp(-\gamma\, d_g^2(x, y))$, where d_g is the usual geodesic distance or the great-circle distance on the manifold, is not positive definite.

We now state our main theorem which provides the sufficient and necessary conditions to obtain a positive definite Gaussian kernel from a given distance function defined on a generic space.

Theorem 6. *Let* (M, d) *be a metric space and define* $k : (M \times M) \to \mathbb{R}$ *by* $k(x, y) := \exp(-\gamma\, d^2(x, y))$. *Then,* k *is a positive definite kernel for all* $\gamma > 0$ *if and only if there exists an inner product space* $\mathcal{V}$ *and a function* $\psi : M \to \mathcal{V}$ *such that,* $d(x, y) = \|\psi(x) - \psi(y)\|_{\mathcal{V}}$.

Proof. We first note that positive definiteness of $k(.,.)$ for all γ and negative definiteness of $d^2(.,.)$ are equivalent conditions according to Theorem 2.

To prove the forward direction of the present theorem, let us first assume that $\mathcal{V}$ and ψ exist such that $d(x, y) = \|\psi(x) - \psi(y)\|_{\mathcal{V}}$. Then, from Lemma 5, d^2 is negative definite and hence k is positive definite for all γ.

On the other hand, if k is positive definite for all γ, then d^2 is negative definite. Furthermore, $d(x, x) = 0$ for all $x \in M$ since d is a metric. According to Theorem 4, then $\mathcal{V}$ and ψ exist such that $d(x, y) = \|\psi(x) - \psi(y)\|_{\mathcal{V}}$.

3.5.2 Geodesic Distances and the Gaussian RBF

A Riemannian manifold, when considered with its geodesic distance, forms a metric space. Given Theorem 6, it is then natural to wonder under which conditions would a geodesic distance on a manifold yield a Gaussian RBF kernel. We now present and prove the following theorem, which answers this question for complete Riemannian manifolds.

Theorem 7. *Let $\mathcal{M}$ be a complete Riemannian manifold and d_g be the geodesic distance induced by its Riemannian metric. The Gaussian RBF kernel $k_g : (\mathcal{M} \times \mathcal{M}) \to \mathbb{R} : k_g(x, y) := \exp(-\gamma\, d_g^2(x, y))$ is positive definite for all $\gamma > 0$ if and only if $\mathcal{M}$ is isometric (in the Riemannian sense) to some Euclidean space $\mathbb{R}^n$.*

Proof. If $\mathcal{M}$ is isometric to some $\mathbb{R}^n$, the geodesic distance on $\mathcal{M}$ is simply the Euclidean distance in $\mathbb{R}^n$, which can be trivially shown to yield a positive definite Gaussian RBF kernel by setting ψ in Theorem 6 to the identity function.

On the other hand, if k_g is positive definite, from Theorem 6, there exists an inner product space $\mathcal{V}_g$ and a function $\psi_g : \mathcal{M} \to \mathcal{V}_g$ such that $d_g(x, y) = \|\psi_g(x) - \psi_g(y)\|_{\mathcal{V}_g}$. Let $\mathcal{H}_g$ be the completion of $\mathcal{V}_g$. Therefore, $\mathcal{H}_g$ is a Hilbert space, in which $\mathcal{V}_g$ is dense.

Now, take any two points x_0, x_1 in $\mathcal{M}$. Since the manifold is complete, from the Hopf–Rinow theorem, there exists a geodesic $\delta(t)$ joining them with $\delta(0) = x_0$ and $\delta(1) = x_1$, and realizing the geodesic distance. By definition, $\delta(t)$ has a constant speed $d_g(x_0, x_1)$. Therefore, for all $x_t = \delta(t)$ where $t \in [0, 1]$, the following equality holds:

$$d_g(x_0, x_t) + d_g(x_t, x_1) = d_g(x_0, x_1).$$

This must also be true for images $\psi(x_t)$ in $\mathcal{H}_g$ for $t \in [0, 1]$. However, since $\mathcal{H}_g$ is a Hilbert space, this is only possible if all the points $\psi(x_t)$ lie on a straight line in $\mathcal{H}_g$. Let $\psi(\mathcal{M})$ be the range of ψ. From the previous argument, for any two points in $\psi(\mathcal{M}) \subseteq \mathcal{H}_g$, the straight line segment joining them is also in $\psi(\mathcal{M})$. Therefore, $\psi(\mathcal{M})$ is a convex set in $\mathcal{H}_g$. Now, since $\mathcal{M}$ is complete, any geodesic must be extensible indefinitely. Therefore, the corresponding line segment in $\psi(\mathcal{M})$ must also be extensible indefinitely. This proves that $\psi(\mathcal{M})$ is an affine subspace of $\mathcal{H}_g$, which is isometric to $\mathbb{R}^n$, for some n. Since $\mathcal{M}$ is isometric to $\psi(\mathcal{M})$, this proves that $\mathcal{M}$ is isometric to the Euclidean space $\mathbb{R}^n$.

According to Theorem 7, it is possible to obtain a positive definite Gaussian kernel from the geodesic distance on a Riemannian manifold only when the manifold is made essentially equivalent to some $\mathbb{R}^n$ by the Riemannian metric that defines the geodesic distance. Although this is possible for some Riemannian manifolds, such as the Riemannian manifold of SPD matrices, for some others, it is theoretically impossible.

In particular, if the manifold is compact, it is impossible to find an isometry between the manifold and $\mathbb{R}^n$, since $\mathbb{R}^n$ is not compact. Therefore, it is not possible to obtain a positive definite Gaussian kernel from the geodesic distance of a compact manifold. In such cases, the best hope is to find a different non-geodesic distance on the manifold that does not differ much from the geodesic distance but still satisfies the conditions of Theorem 6.

3.5.3 Kernels on Sym_d^+

We next discuss the different metrics on Sym_d^+ that can be used to define positive definite Gaussian kernels. In particular, we focus on the log-Euclidean distance, which is a true geodesic distance on Sym_d^+ [3].

In the log-Euclidean framework, a geodesic connecting $S_1, S_2 \in \mathrm{Sym}_d^+$ is defined as $\gamma(t) = \exp((1-t)\log(S_1) + t\log(S_2))$ for $t \in [0, 1]$. The log-Euclidean geodesic distance between S_1 and S_2 can be expressed as

$$d_{\mathrm{LE}}(S_1, S_2) = \|\log(S_1) - \log(S_2)\|_F \,, \tag{3.1}$$

where $\|\cdot\|_F$ denotes the Frobenius matrix norm induced by the Frobenius matrix inner product $\langle .,. \rangle_F$.

The main reason to exploit the log-Euclidean distance in our experiments is that it defines a true geodesic distance that has proven an effective distance measure on Sym_d^+. Furthermore, it yields a positive definite Gaussian kernel as stated in the following corollary to Theorem 6:

Corollary 8 (Theorem 6). *The* ***Log-Euclidean Gaussian kernel*** $k_{\mathrm{LE}} : (\mathrm{Sym}_d^+ \times \mathrm{Sym}_d^+) \to \mathbb{R} : k_{\mathrm{LE}}(S_1, S_2) := \exp(-\gamma\, d_{\mathrm{LE}}^2(S_1, S_2))$*, where* $d_{\mathrm{LE}}(S_1, S_2)$ *is the log-Euclidean distance between* S_1 *and* S_2*, is a positive definite kernel for all* $\gamma > 0$.

Proof. The matrix logarithm log(.) maps $d \times d$ SPD matrices to $d \times d$ symmetric matrices, which form an inner product space when endowed with the Frobenius matrix inner product $\langle .,. \rangle_F$. The result then directly follows from Theorem 6 with $\psi = (S \mapsto \log(S))$.

A number of other metrics have been proposed for Sym_d^+ [11]. The definitions and properties of these metrics are summarized in Table 3.1. Note that only some of them were derived by considering the Riemannian geometry of the manifold and hence define true geodesic distances. Similar to the log-Euclidean metric, from Theorem 6, it directly follows that the Cholesky and power-Euclidean metrics also define positive definite Gaussian kernels for all values of γ. Some metrics may yield

Table 3.1 Properties of different metrics on the three manifolds

Metric name	Formula	Geodesic distance	PD Gaussian kernel for all $\gamma > 0$
SPD matrices			
Log-Euclidean	$\|\log(\mathbf{S}_1) - \log(\mathbf{S}_2)\|_F$	**Yes**	**Yes**
Affine invariant	$\|\log(\mathbf{S}_1^{-1/2}\mathbf{S}_2\mathbf{S}_1^{-1/2})\|_F$	Yes	No
Cholesky	$\|\mathrm{chol}(\mathbf{S}_1) - \mathrm{chol}(\mathbf{S}_2)\|_F$	No	Yes
Power Euclidean	$\frac{1}{\alpha}\|\mathbf{S}_1^\alpha - \mathbf{S}_2^\alpha\|_F$	No	Yes
Root Stein divergence	$\left[\log\det\left(\frac{1}{2}\mathbf{S}_1 + \frac{1}{2}\mathbf{S}_2\right) - \frac{1}{2}\log\det(\mathbf{S}_1\mathbf{S}_2)\right]^{1/2}$	No	No
Grassmann manifold			
Projection	$2^{-1/2}\|Y_1Y_1^T - Y_2Y_2^T\|_F = (\sum_i \sin^2\theta_i)^{1/2}$	No	**Yes**
Arc length	$(\sum_i \theta_i^2)^{1/2}$	Yes	No
Fubini study	$\arccos\|\det(Y_1^TY_2)\| = \arccos(\prod_i \cos\theta_i)$	No	No
Chordal 2-norm	$\|Y_1U - Y_2V\|_2 = 2\max_i \sin\frac{1}{2}\theta_i$	No	No
Chordal F-norm	$\|Y_1U - Y_2V\|_F = 2(\sin^2\frac{1}{2}\theta_i)^{1/2}$	No	No
Shape manifold			
Full Procrustes	$\left(1 - \|\langle\mathbf{z}_1, \mathbf{z}_2\rangle\|^2\right)^{\frac{1}{2}}$	**No**	**Yes**
Partial Procrustes	$(1 - \|\langle\mathbf{z}_1, \mathbf{z}_2\rangle\|)^{\frac{1}{2}}$	No	No
Arc length	$\arccos(\|\langle\mathbf{z}_1, \mathbf{z}_2\rangle\|)$	Yes	No

We analyze the positive definiteness of the Gaussian kernels generated by different metrics. Theorem 6 applies to the metrics claimed to generate positive definite Gaussian kernels. For the other metrics, examples of nonpositive definite Gaussian kernels exist

a positive definite Gaussian kernel for some value of γ only. This, for instance, was shown in [33] for the root Stein divergence metric. Constraints on γ are nonetheless undesirable, since one should be able to freely tune γ to reflect the data distribution, and automatic model selection algorithms require kernels to be positive definite for continuous values of $\gamma > 0$ [8].

3.5.4 Kernels on $\mathcal{G}_n^r$

Similarly to Sym_d^+, different metrics can be defined on $\mathcal{G}_n^r$. Many of these metrics are related to the principal angles between two subspaces. Given two $n \times r$ matrices Y_1 and Y_2 with orthonormal columns, representing two points on $\mathcal{G}_n^r$, the principal angles between the corresponding subspaces are obtained from the singular value decomposition of $Y_1^TY_2$ [12]. More specifically, if USV^T is the singular value decomposition of $Y_1^TY_2$, then the entries of the diagonal matrix S are the cosines of the principal angles between $[Y_1]$ and $[Y_2]$. Let $\{\theta_i\}_{i=1}^r$ represent those principal angles. Then, the geodesic distance derived from the cannonical geometry of the Grassmann manifold, called the arc length, is given by $(\sum_i \theta_i^2)^{1/2}$ [12].

Unfortunately, as can be shown with a counterexample, this distance squared is not negative definite and hence does not yield a positive definite Gaussian for all $\gamma > 0$.

Nevertheless, there exists another widely used metric on the Grassmann manifold, namely the projection metric, which gives rise to a positive definite Gaussian kernel. The projection distance between two subspaces $[Y_1], [Y_2]$ is given by

$$d_P([Y_1], [Y_2]) = 2^{-1/2} \| Y_1 Y_1^T - Y_2 Y_2^T \|_F. \tag{3.2}$$

We now formally introduce the corresponding Gaussian RBF kernel on the Grassmann manifold.

Corollary 9 (Theorem 6). *The* ***Projection Gaussian kernel*** $k_P : (\mathcal{G}_n^r \times \mathcal{G}_n^r) \to \mathbb{R} : k_P([Y_1], [Y_2]) := \exp(-\gamma\, d_P^2([Y_1], [Y_2]))$*, where* $d_P([Y_1], [Y_2])$ *is the projection distance between* $[Y_1]$ *and* $[Y_2]$*, is a positive definite kernel for all* $\gamma > 0$.

Proof. The proof follows from Theorem 6 with $\psi = ([Y] \mapsto YY^T)$ and $\mathcal{V}$ being the space of $n \times n$ matrices endowed with the Frobenius inner product.

As shown in Table 3.1, none of the other popular metrics on Grassmann manifolds have this property.

3.5.5 Kernels on SP^n

While an ideal metric on shape descriptors should be fully invariant to translation, scale, and rotation, not all metrics commonly used on SP^n satisfy this requirement [10]. The full Procrustes metric, however, is fully invariant to all these three transformations. Therefore, although it is not a geodesic distance on the shape manifold, it has become a popular shape distance [10]. Given two pre-shapes $\mathbf{z}_1$ and $\mathbf{z}_2$, the full Procrustes distance between the corresponding shapes is given by Kendall [21] and Dryden and Mardia [10]

$$d_{FP}([\mathbf{z}_1], [\mathbf{z}_2]) = \left(1 - |\langle \mathbf{z}_1, \mathbf{z}_2 \rangle|^2\right)^{\frac{1}{2}}, \tag{3.3}$$

where $\langle .,. \rangle$ and $|.|$ denote the usual inner product in $\mathbb{C}^m$ and the absolute value of a complex number, respectively. In the following corollary, we prove that the full Procrustes distance yields a positive definite Gaussian kernel on the shape manifold.

Corollary 10 (Theorem 6). *The* ***Procrustes Gaussian kernel*** $k_{FP} : (\mathrm{SP}^n \times \mathrm{SP}^n) \to \mathbb{R} : k_{FP}([\mathbf{z}_1], [\mathbf{z}_2]) := \exp(-\gamma\, d_{FP}^2([\mathbf{z}_1], [\mathbf{z}_2]))$*, where* d_{FP} *is the full Procrustes distance between two shapes* $[\mathbf{z}_1]$ *and* $[\mathbf{z}_2]$*, is a positive definite kernel for all* $\gamma > 0$.

Proof. Consider the map $\mathbb{C}S^{n-1} \to \mathbb{C}^{n \times n} : \mathbf{z} \mapsto \mathbf{z}\mathbf{z}^*$ from pre-shapes to Hermitian matrices. It is clear that any rotation of the pre-shape $\mathbf{z}$ is also mapped to the same point under this map. Therefore, $\psi_1 : \mathrm{SP}^n \to \mathbb{C}^{n \times n} : [\mathbf{z}] \mapsto \mathbf{z}\mathbf{z}^*$ turns out to be a function that maps the shape manifold to the space of $n \times n$ Herminitan matrices, which, when endowed with the Frobenius inner product, forms a real inner product space. Furthermore, it can be shown that,

$$d_{FP}([\mathbf{z}_1], [\mathbf{z}_2]) = \left\| \frac{1}{2}\psi_1([\mathbf{z}_1]) - \frac{1}{2}\psi_1([\mathbf{z}_2]) \right\|_F .$$

Therefore, by applying Theorem 6 with $\psi = \frac{1}{2}\psi_1$, we have that k_{FP} is positive definite for all $\gamma > 0$.

Properties of other metrics on the shape manifold are summarized in Table 3.1. Note that it is possible to define a Riemannian metric on the shape manifold that induces the arc length distance. Hence it can be considered as a geodesic distance on the shape manifold.

3.6 Non-Gaussian Kernels on Manifolds

Although the Gaussian RBF is the predominant kernel used with Euclidean-valued data, other kinds of kernels are also used in practice. Similarly, a number of non-Gaussian kernels have been proposed for manifolds. In particular, kernels that are analogous to linear kernels in Euclidean spaces were proposed for the Grassmann manifold in [14] and for *SPD* matrices in [40]. These kernels exploit the inner product structure in the linear spaces to which these two manifolds are isometrically (in the distance-preserving sense) mapped under the Projection metric and the log-Euclidean metric, respectively. It was empirically shown in [20] that the Gaussian RBF kernels introduced in the previous section usually outperforms these linear-type kernels, agreeing with the common observations in Euclidean spaces.

In [19], families of positive definite radial kernels were proposed for the unit n-sphere, the Grassmann manifold, and the shape manifold. They also proposed an optimization algorithm to automatically learn the best suited radial kernel from these families in a classification setting. These radial kernels take the form of conic combinations of monomial kernels (integer powers of linear kernels). The optimization algorithm essentially optimizes the weights in the conic combination. Interestingly, the Gaussian RBF kernels defined in Theorems 9 and 10 belong to the family of radial kernels introduced there.

It is known that the heat kernel is positive definite on any Riemannian manifold [6]. However, evaluating the heat kernel on a nontrivial Riemannian manifold is mathematically cumbersome. In [6], the authors relied on a number of approximation techniques to show that this kernel can nevertheless be estimated on a number of manifolds commonly encountered in computer vision.

3.7 Kernel-Based Algorithms on Manifolds

A major advantage of being able to compute positive definite kernels on manifolds is that it directly allows us to make use of algorithms developed for $\mathbb{R}^n$, while still accounting for the geometry of the manifold. In this section, we discuss the use of five kernel-based algorithms on manifolds. The resulting algorithms can be thought of as generalizations of the original Euclidean kernel methods to manifolds. In the following, we use $\mathcal{M}$, $k(.,.)$, $\mathcal{H}$ and $\phi(x)$ to denote a d-dimensional manifold, a positive definite kernel function defined on $\mathcal{M} \times \mathcal{M}$, the RKHS generated by k, and the feature vector in $\mathcal{H}$ to which $x \in \mathcal{M}$ is mapped, respectively. Although we use $\phi(x)$ for explanation purposes, following the *kernel trick*, it never needs to be explicitly computed in any of the algorithms.

3.7.1 Support Vector Machines on Manifolds

We first consider the binary classification problem on a manifold. To this end, we propose to extend the popular Euclidean kernel SVM algorithm to manifold-valued data. Given a set of training examples $\{(x_i, y_i)\}_{i=1}^m$, where $x_i \in \mathcal{M}$ and the label $y_i \in \{-1, 1\}$, kernel SVM searches for a hyperplane in $\mathcal{H}$ that separates the feature vectors of the positive and negative classes with maximum margin. The class of a test point $x \in \mathcal{M}$ is determined by the position of the feature vector $\phi(x)$ in $\mathcal{H}$ relative to the separating hyperplane. Classification with kernel SVM can be done very fast, since it only requires to evaluate the kernel at the *support vectors*.

The above procedure is equivalent to solving the standard kernel SVM problem with kernel matrix generated by k. Thus, any existing SVM software package can be utilized for training and classification. Convergence of standard SVM optimization algorithms is guaranteed since k is positive definite.

Kernel SVM on manifolds is much simpler to implement and less computationally demanding in both training and testing phases than the current state-of-the-art binary classification algorithms on manifolds, such as LogitBoost on a manifold [37], which involves iteratively combining weak learners on different tangent spaces. Weighted mean calculation in LogitBoost on a manifold involves an extremely expensive gradient descent procedure at each boosting iteration, which makes the algorithm scale poorly with the number of training samples. Furthermore, while LogitBoost learns classifiers on tangent spaces used as first-order Euclidean approximations to the manifold, our approach works in a rich high-dimensional feature space. As will be shown in our experiments, this yields better classification results.

With manifold-valued data, extending the current state-of-the-art binary classification methods to multi-class classification is not straightforward and requires additional work [34]. In contrast, our manifold kernel SVM classification method can easily be extended to the multi-class case with standard one-vs-one or one-vs-all procedures.

3.7.2 *Multiple Kernel Learning on Manifolds*

We next tackle the problem of combining multiple manifold-valued descriptors via a MKL approach. The core idea of MKL is to combine kernels computed from different descriptors (e.g., image features) to obtain a kernel that optimally separates two classes for a given classifier. Here, we follow the formulation of [39] and make use of an SVM classifier. As a feature selection method, MKL has proven more effective than conventional techniques such as wrappers, filters, and boosting [38].

More specifically, given training examples $\{(x_i, y_i)\}_1^m$, where $x_i \in \mathcal{X}$ (some nonempty set), $y_i \in \{-1, 1\}$, and a set of descriptor generating functions $\{g_j\}_1^N$ where $g_j : \mathcal{X} \to \mathcal{M}$, we seek to learn a binary classifier $f : \mathcal{X} \to \{-1, 1\}$ by selecting and optimally combining the different descriptors generated by $g_1, \dots, g_N$. Let $K^{(j)}$ be the kernel matrix generated by g_j and k as $K_{pq}^{(j)} = k(g_j(x_p), g_j(x_q))$. The combined kernel can be expressed as $K^* = \sum_j \lambda_j K^{(j)}$, where $\lambda_j \geq 0$ for all j guarantees the positive definiteness of K^*. The weights $\boldsymbol{\lambda}$ can be learned using a min-max optimization procedure with an l^1 regularizer on $\boldsymbol{\lambda}$ to obtain a sparse combination of kernels. The algorithm has two steps in each iteration: First it solves a conventional SVM problem with $\boldsymbol{\lambda}$, hence K^*, fixed. Then, it updates $\boldsymbol{\lambda}$ with SVM parameters fixed. These two steps are repeated until convergence. For more details, we refer the reader to [39] and [38]. Note that convergence of MKL is only guaranteed if all the kernels are positive definite, which is satisfied in this setup since k is positive definite.

We also note that MKL on manifolds gives a convenient method to combine manifold-valued descriptors with Euclidean descriptors, which is otherwise a difficult task due to their different geometries.

3.7.3 *Kernel Principal Component Analysis on Manifolds*

We next study the extension of kernel PCA to nonlinear dimensionality reduction on manifolds. The usual Euclidean version of kernel PCA has proven successful in many applications [30, 31]. On a manifold, kernel PCA proceeds as follows: All points $x_i \in \mathcal{M}$ of a given data set $\{x_i\}_{i=1}^m$ are mapped to feature vectors in $\mathcal{H}$, thus yielding the transformed set $\{\phi(x_i)\}_{i=1}^m$. The covariance matrix of this transformed set is then computed, which really amounts to computing the kernel matrix of the original data using the function k. An l-dimensional representation of the data is obtained by computing the eigenvectors of the kernel matrix. This representation can be thought of as a Euclidean representation of the original manifold-valued data. However, owing to our kernel, it was obtained by accounting for the geometry of $\mathcal{M}$.

Once the kernel matrix is calculated with k, the implementation details of the algorithm are similar to that of the Euclidean kernel PCA algorithm. We refer the reader to [30] for more details on the implementation.

3.7.4 Kernel k-Means on Manifolds

For clustering problems on manifolds, we propose to make use of kernel k-means. Kernel k-means maps points to a high-dimensional Hilbert space and performs k-means in the resulting feature space [30]. In a manifold setting, a given data set $\{x_i\}_{i=1}^m$, with each $x_i \in \mathcal{M}$, is clustered into a predefined number of groups in $\mathcal{H}$, such that the sum of the squared distances from each $\phi(x_i)$ to the nearest cluster center is minimum. The resulting clusters can then act as classes for the $\{x_i\}_{i=1}^m$.

The unsupervised clustering method on Riemannian manifolds proposed in [13] clusters points in a low-dimensional space after dimensionality reduction on the manifold. In contrast, our method performs clustering in a high-dimensional RKHS which, intuitively, better represents the data distribution.

3.7.5 Kernel Fisher Discriminant Analysis on Manifolds

The kernelized version of linear discriminant analysis, known as Kernel Fisher discriminant analysis (kernel FDA), can also be extended to manifolds on which a positive definite kernel can be defined. In particular, given $\{(x_i, y_i)\}_{i=1}^m$, with each $x_i \in \mathcal{M}$ having class label y_i, manifold kernel FDA maps each point x_i on the manifold to a feature vector in $\mathcal{H}$ and finds a new basis in $\mathcal{H}$ where the class separation is maximized. The output of the algorithm is a Euclidean representation of the original manifold-valued data, but with a larger separation between class means and a smaller within-class variance. Up to $(l - 1)$ dimensions can be extracted via kernel FDA, where l is the number of classes. We refer the reader to [31] for implementation details. In Euclidean spaces, kernel FDA has become an effective pre-processing step to perform nearest-neighbor classification in the highly discriminative, reduced dimensional space.

3.8 Applications and Experiments

In this section, we present a set of experiments on the material presented in this chapter using the positive definite kernels introduced in Sect. 3.5 and the algorithms described in Sect. 3.7.

3.8.1 Pedestrian Detection

The pedestrian detection experiment published in [17] evaluates the performance of the manifold SVM/MKL framework described in this chapter. Covariance

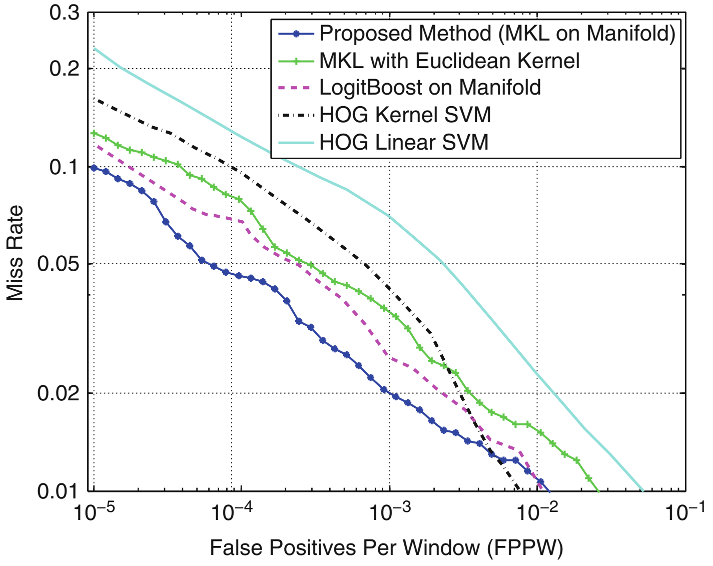


Fig. 3.1 Pedestrian detection. Detection-error trade-off curves for the proposed manifold MKL approach and state-of-the-art methods on the INRIA data-set. Manifold MKL outperforms existing manifold methods and Euclidean kernel methods

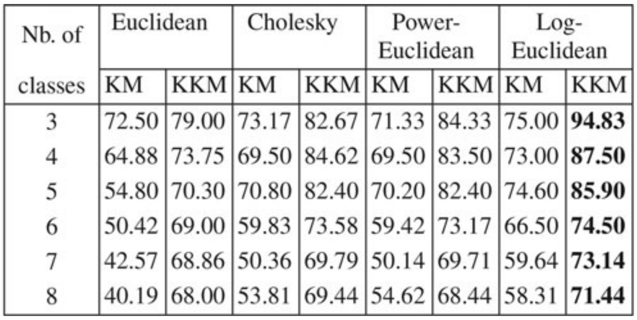
descriptors [37] that lie on Sym_8^+ and the log-Euclidean Gaussian kernel defined on them were used in this experiment. The MKL setup described in Sect. 3.7.2 was utilized with 100 kernels, each generated from covariance descriptors calculated in a fixed subwindow inside the detection window.

The INRIA person data-set [9] was used to compare the performance of manifold SVM/MKL with other state-of-the-art classification algorithms on manifolds. The resulting detection-error trade-off (DET) curves are shown in Fig. 3.1. As can be seen from the graph, manifold MKL performs better than the Euclidean MKL method that neglects the nonlinear geometry of Sym_d^+, and the LogitBoost on manifolds method that uses tangent space approximations. This demonstrates the benefits of accounting for the nonlinearity of the manifold using an appropriate positive definite kernel. It is also worth noting that LogitBoost on the manifold is significantly more complex and harder to implement than the manifold MKL method.

3.8.2 *Visual Object Categorization*

We next demonstrate the use of kernels on manifolds in the context of unsupervised object categorization. This experiment uses the ETH-80 data set [24], which contains 8 object categories with 10 objects each and 41 images per object.

Table 3.2 Object categorization



Nb. of classes	Euclidean		Cholesky		Power-Euclidean		Log-Euclidean	
	KM	KKM	KM	KKM	KM	KKM	KM	KKM
3	72.50	79.00	73.17	82.67	71.33	84.33	75.00	**94.83**
4	64.88	73.75	69.50	84.62	69.50	83.50	73.00	**87.50**
5	54.80	70.30	70.80	82.40	70.20	82.40	74.60	**85.90**
6	50.42	69.00	59.83	73.58	59.42	73.17	66.50	**74.50**
7	42.57	68.86	50.36	69.79	50.14	69.71	59.64	**73.14**
8	40.19	68.00	53.81	69.44	54.62	68.44	58.31	**71.44**

Sample images and percentages of correct clustering on the ETH-80 data set using k-means (KM) and kernel k-means (KKM) with different metrics on Sym_d^+. The proposed KKM method with the log-Euclidean metric achieves best results in all tests

The goal was to find object categories (clusters) in an unsupervised setting. A single covariance descriptor was used to describe each image, narrowing down the problem to a clustering problem in Sym_5^+. Consequently, the kernel k-means algorithm on Sym_5^+ described in Sect. 3.7.4 was used to obtain object categories.

To set a benchmark, the performance of both k-means and kernel k-means on Sym_5^+ was evaluated with different metrics that generate positive definite Gaussian kernels (see Table 3.1). For the power-Euclidean metric, α was set to 0.5, which achieved the best results in [11]. For all non-Euclidean metrics with (non-kernel) k-means, the Karcher mean [11] was used to compute the centroid. The results of the different methods are summarized in Table 3.2. Manifold kernel k-means with the log-Euclidean Gaussian kernel performs significantly better than all other methods in all test cases.

3.8.3 *Video-Based Face Recognition*

Face recognition from video, which uses an image set for identification of a person, is a rapidly developing area in computer vision. For this task, the videos are often modeled as linear subspaces which lie on a Grassmann manifold [14, 15]. To demonstrate the use of the projection Gaussian kernel in video-based face recognition, we used the YouTube Celebrity data set [23], which contains 1910 video clips of 47 different people.

We extracted face regions from videos and resized them to have a common size of 96×96. Each face image was then represented by a histogram of local binary patterns [27] having 232 equally spaced bins. We next represented each image set corresponding to a single video clip by a linear subspace of order 5. We randomly chose 70 % of the data set for training and the remaining 30 % for testing. We report the classification accuracy averaged over 10 different random splits.

Table 3.3 Face recognition

Method	Classification accuracy
DCC [22]	60.21 ± 2.9
KAHM [7]	67.49 ± 3.5
GDA [14]	58.72 ± 3.0
GGDA [15]	61.05 ± 2.2
Linear SVM	64.76 ± 2.1
Manifold kernel FDA	65.32 ± 1.4
Manifold kernel SVM	$\mathbf{71.78 \pm 2.4}$

Recognition accuracies on the YouTube celebrity data set. Manifold kernel SVM and FDA with the Gaussian kernel achieve the best results

We employed both kernel SVM on a manifold and kernel FDA on a manifold with our projection Gaussian kernel for classification. With kernel SVM, a one-vs-all procedure was used for multiclass classification. For kernel FDA, the training data were projected to an $(l-1)$-dimensional space, where $l = 47$ is the number of classes, and we used a 1-nearest-neighbor method to predict the class of a test sample projected to the same space. We determined the hyperparameters of both methods using cross-validation on the training data.

We compared our approach with several state-of-the-art image set classification methods: discriminant analysis of canonical correlations (DCC) [22], kernel affine hull method (KAHM) [7], Grassmann discriminant analysis (GDA) [14], and graph-embedding Grassmann discriminant analysis (GGDA) [15]. As can be seen in Table 3.3, kernel FDA on a manifold and kernel SVM on a manifold both outperform the other methods significantly, with kernel SVM achieving the best accuracy.

3.8.4 Shape Retrieval

Finally, we exploit the full Procrustes Gaussian kernel for the task of shape retrieval. State-of-the-art shape retrieval methods [4, 25] perform exhaustive nearest-neighbor search over the entire database using nonlinear distances between the shapes, which does not scale with the database size. Here, instead, we made use of the kernel PCA algorithm of Sect. 3.7.3 to obtain a real-valued Euclidean representation of the data set while preserving the important shape variances. We then performed shape retrieval on the resulting Euclidean space, which can be done more efficiently using an algorithm such as k-d trees. To validate our shape retrieval approach, we used the SQUID Fish data set [26] which consists of 1100 unlabeled fish contours. We obtained 100 landmarks from each contour. We set aside 10 different fish shapes from the data set as query images and used the remaining shapes to obtain a $d = 50$-dimensional real Euclidean representation of the data set. For retrieval, we projected

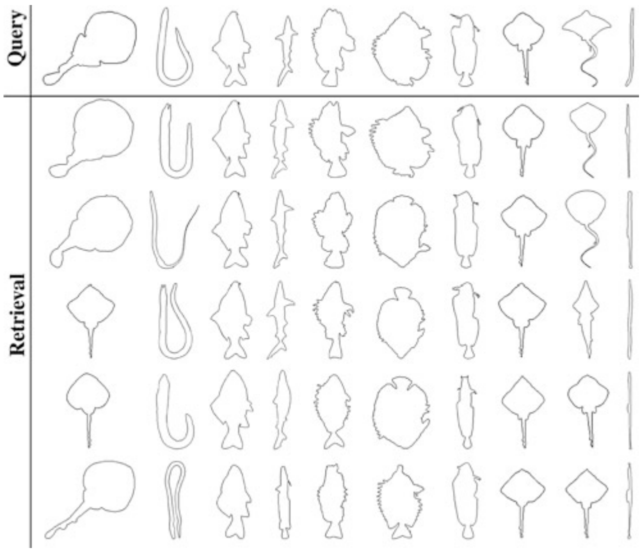


Fig. 3.2 Shape retrieval. Retrieval results from the SQUID fish data set using our kernel PCA approach. The nearest neighbors are ordered according to their distance to the query

each query shape to this 50-dimensional space and found its 5 nearest neighbors using the standard Euclidean distance. Figure 3.2 depicts the retrieval results. Note that the retrieved shapes match the query ones very accurately.

3.9 Conclusion

In this chapter, we have discussed kernels on Riemannian manifolds, which is a growing research topic in computer vision [18, 19]. We have showed that powerful kernel methods from Euclidean spaces can be generalized to manifold-valued data with the help of an appropriately defined Gaussian kernel on the manifold of interest. We have introduced a unified framework to analyze the positive definiteness of the Gaussian RBF kernel defined on a manifold or a more general metric space. We have then used the same framework to derive provably positive definite kernels on the Riemannian manifold of SPD matrices, the Grassmann manifold, and the shape manifold. These kernels were then utilized to extend popular learning algorithms designed for Euclidean spaces, such as SVM and FDA, to manifolds. Experimental results on several challenging computer vision tasks, such as pedestrian detection, object categorization, and face recognition, have evidenced that identifying positive definite kernel functions on manifolds can be greatly beneficial when working with manifold-valued data.

References

1. Aronszajn N (2006) Theory of reproducing kernels. Trans Am Math Soc, Magn Reson Med Aug;56(2):411–21
2. Arsigny V, Fillard P, Pennec X, Ayache N (2005) Fast and simple computations on tensors with log-Euclidean metrics. Research report RR-5584, INRIA
3. Arsigny V, Fillard P, Pennec X, Ayache N (2006) Log-Euclidean metrics for fast and simple calculus on diffusion tensors. Magn Reson Med 56(2):411–421
4. Bai X, Yang X, Latecki L, Liu W, Tu Z (2010) Learning context-sensitive shape similarity by graph transduction. IEEE Trans Pattern Anal Mach Intell 32(5):861–874
5. Berg C, Christensen JPR, Ressel P (1984) Harmonic analysis on semigroups. Springer, New York
6. Caseiro R, Henriques JF, Martins P, Batista J (2012) Semi-intrinsic mean shift on Riemannian manifolds. In: European conference on computer vision (ECCV)
7. Cevikalp H, Triggs B (2010) Face recognition based on image sets. In: CVPR
8. Chapelle O, Vapnik V, Bousquet O, Mukherjee S (2002) Choosing multiple parameters for support vector machines. Mach Learn 46(1-3):131–159
9. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: CVPR
10. Dryden I, Mardia K (1998) Statistical shape analysis. Wiley, Chichester
11. Dryden IL, Koloydenko A, Zhou D (2009) Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging. Ann Appl Stat
12. Edelman A, Arias TA, Smith ST (1998) The geometry of algorithms with orthogonality constraints. SIAM J Matrix Anal Appl 20(2):303–353
13. Goh A, Vidal R (2008) Clustering and dimensionality reduction on Riemannian manifolds. In: CVPR
14. Hamm J, Lee DD (2008) Grassmann discriminant analysis: a unifying view on subspace-based learning. In: ICML
15. Harandi MT, Sanderson C, Shirazi S, Lovell BC (2011) Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching. In: CVPR
16. Harandi M, Sanderson C, Hartley R, Lovel B (2012) Sparse coding and dictionary learning for symmetric positive definite matrices: a kernel approach. In: European conference on computer vision (ECCV)
17. Jayasumana S, Hartley R, Salzmann M, Li H, Harandi M (2013) Kernel methods on the Riemannian manifold of symmetric positive definite matrices. In: CVPR
18. Jayasumana S, Salzmann M, Li H, Harandi M (2013) A framework for shape analysis via Hilbert space embedding. In: ICCV
19. Jayasumana S, Hartley R, Salzmann M, Li H, Harandi M (2014) Optimizing over radial kernels on compact manifolds. In: CVPR
20. Jayasumana S, Hartley R, Salzmann M, Li H, Harandi M (2015) Kernel methods on Riemannian manifolds with Gaussian RBF kernels. Pattern Anal Mach Intell
21. Kendall DG (1984) Shape manifolds, Procrustean metrics, and complex projective spaces. Bull Lond Math Soc 16(2):81–121
22. Kim TK, Kittler J, Cipolla R (2007) Discriminative learning and recognition of image set classes using canonical correlations. IEEE Trans Pattern Anal Mach Intell 29(2):1005–1018
23. Kim M, Kumar S, Pavlovic V, Rowley HA (2008) Face tracking and recognition with visual constraints in real-world videos. IEEE Comput Vis Pattern Recogn 29(2):286–299
24. Leibe B, Schiele B (2003) Analyzing appearance and contour based methods for object categorization. In: CVPR
25. Ling H, Jacobs D (2007) Shape classification using the inner-distance. IEEE Trans Pattern Anal Mach Intell
26. Mokhtarian F, Abbasi S, Kittler J (1996) Efficient and robust retrieval by shape content through curvature scale space. In: BMVC

27. Ojala T, Pietikainen M, Maenpaa T (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans Pattern Anal Mach Intell 24(7):971–987
28. Pennec X, Fillard P, Ayache N (2006) A Riemannian framework for tensor computing. Int J Comput Vis, 66(1):41–66
29. Schoenberg IJ (1938) Metric spaces and positive definite functions. Trans Am Math Soc, 44:522–536
30. Schölkopf B, Smola A, Müller KR (1998) Nonlinear component analysis as a kernel eigenvalue problem. Neural Comput 10(5):1299–1319
31. Schölkopf B, Smola AJ (2002) Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT Press, Cambridge
32. Shawe-Taylor J, Cristianini N (2004) Kernel methods for pattern analysis. Cambridge University Press, Cambridge
33. Sra S (2012) Positive definite matrices and the symmetric Stein divergence. Preprint [arXiv:1110.1773]
34. Tosato D, Farenzena M, Cristani M, Spera M, Murino V (2010) Multi-class classification on Riemannian manifolds for video surveillance. In: European conference on computer vision (ECCV)
35. Turaga P, Veeraraghavan A, Srivastava A, Chellappa R (2011) Statistical computations on Grassmann and Stiefel manifolds for image and video-based recognition. IEEE Trans Pattern Anal Mach Intell 33(11):2273–2286
36. Tuzel O, Porikli F, Meer P (2006) Region covariance: a fast descriptor for detection and classification. In: European conference on computer vision (ECCV)
37. Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on Riemannian manifolds. IEEE Trans Pattern Anal Mach Intell 30(10):1713–1727
38. Varma M, Babu BR (2009) More generality in efficient multiple kernel learning. In: ICML
39. Varma M, Ray D (2007) Learning the discriminative power-invariance trade-off. In: ICCV
40. Wang R, Guo H, Davis LS, Dai Q (2012) Covariance discriminative learning: a natural and efficient approach to image set classification. In: CVPR

Chapter 4
Canonical Correlation Analysis on SPD(n) Manifolds

Hyunwoo J. Kim, Nagesh Adluru, Barbara B. Bendlin, Sterling C. Johnson, Baba C. Vemuri, and Vikas Singh

Abstract Canonical correlation analysis (CCA) is a widely used statistical technique to capture correlations between two sets of multivariate random variables and has found a multitude of applications in computer vision, medical imaging, and machine learning. The classical formulation assumes that the data live in a pair of *vector spaces* which makes its use in certain important scientific domains problematic. For instance, the set of symmetric positive definite matrices (SPD), rotations, and probability distributions all belong to certain curved Riemannian manifolds where vector-space operations are in general not applicable. Analyzing the space of such data via the classical versions of inference models is suboptimal. Using the space of SPD matrices as a concrete example, we present a principled generalization of the well known CCA to the Riemannian setting. Our CCA algorithm operates on the product Riemannian manifold representing SPD matrix-valued fields to identify meaningful correlations. As a proof of principle, we present experimental results on a neuroimaging data set to show the applicability of these ideas.

4.1 Introduction

Canonical correlation analysis (CCA) is a powerful statistical technique to extract linear components that capture correlations between two multivariate random variables [25]. CCA provides an answer to the following question: suppose we are given data of the form, $(\boldsymbol{x}_i \in \mathcal{X}, \mathbf{y}_i \in \mathcal{Y})_{i=1}^N \subset \mathcal{X} \times \mathcal{Y}$ where $\boldsymbol{x}_i \in \mathbf{R}^m$ and $\mathbf{y}_i \in \mathbf{R}^n$, find a model that explains *both* of these observations. More precisely, CCA identifies a pair of directions where the projections (namely, u and v) of the random variables $\boldsymbol{x}$

H.J. Kim (✉) • N. Adluru • B.B. Bendlin • S.C. Johnson • V. Singh
University of Wisconsin-Madison, Madison, WI, USA
e-mail: hwkim@cs.wisc.edu; adluru@wisc.edu; bbb@medicine.wisc.edu; scj@medicine.wisc.edu; vsingh@biostat.wisc.edu

B.C. Vemuri
University of Florida, Gainesville, FL, USA
e-mail: vemuri@cise.ufl.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_4

and $\mathbf{y}$ yield maximum correlation $\rho_{u,v} = \text{COV}(u,v)/\sigma_u\sigma_v$. Here, $\text{COV}(u,v)$ denotes the covariance function and $\sigma.$ gives the standard deviation. During the last decade, the CCA formulation has been broadly applied to various unsupervised learning problems in computer vision and machine learning including image retrieval [21], face/gait recognition [54], super-resolution [29], and action classification [34].

Beyond the applications described above, a number of works have recently investigated the use of CCA in analyzing neuroimaging data [6]. Here, for each participant in a study, one acquires different types of images such as magnetic resonance (MRI), computed tomography (CT), and functional MRI. It is expected that each imaging modality captures a unique aspect of the disease pathology. Therefore, given a group of N subjects and their brain images, we may want to identify relationships (e.g., anatomical/functional correlations) across different image types. When performed across different diseases, such an analysis will reveal insights into what is similar and what is different across diseases even when their symptomatic presentation may be similar. Alternatively, CCA may serve a feature extraction role. That is, the brain regions found to be strongly correlated can be used directly in downstream statistical analysis. In a study of a large number of subjects, rather than performing a hypothesis test on *all* brain voxels independently for each imaging modality, restricting the number of tests only to the set of "relevant" voxels (found via CCA) can improve statistical power.

The classical version of CCA described above concurrently seeks two linear subspaces in *vector spaces* $\mathbf{R}^m$ and $\mathbf{R}^n$ for the two multivariate random variables $\boldsymbol{x}$ and $\mathbf{y}$. The projection onto the straight line (linear subspace) is obtained by an inner product. This formulation is broadly applicable but encounters problems for manifold-valued data that are becoming increasingly important in current research. For example, diffusion tensor magnetic resonance images (DTI) allow one to infer the diffusion tensor characterizing the anisotropy of water diffusion at each voxel in an image volume [9]. This tensorial feature can be visualized as an ellipsoid and represented by a 3×3 symmetric positive definite (SPD) matrix at each voxel in the acquired image volume. Neither the individual SPD matrices nor the field of these SPD matrices lies in a vector space but instead are elements of a negatively curved Riemannian manifold where standard vector space operations are not valid [10, 24]. Classical CCA is not applicable in this setting. For T1-weighted magnetic resonance images (MRIs), we are frequently interested in analyzing not just the 3D intensity image on its own, but rather a quantity that captures the deformation field between each image and a *population template*. A registration between the image and the template yields the deformation field required to align the image pairs, and the determinant of the Jacobian J of this deformation at each voxel is a commonly used feature that captures local volume changes [12, 28]. Quantities such as the Cauchy deformation tensor defined as $\sqrt{J^TJ}$ have been reported in literature for use in morphometric analysis [27]. The input to the statistical analysis is a 3D image of voxels, where each voxel corresponds to a matrix $\sqrt{J^TJ} \succ 0$ (the Cauchy deformation tensor). Another example of manifold-valued fields is derived from high angular resolution diffusion images (HARDI) and can be used to compute the ensemble average propagators (EAPs) at each voxel of the given HARDI data.

The EAP is a probability density function that is related to the diffusion-sensitized MR signal via the Fourier transform [11]. Since an EAP is a probability density function, by using a square root parameterization of this density function, it is possible to identify it with a point on the unit Hilbert sphere. Once again, to perform any statistical analysis of these data-derived features, we cannot apply standard vector-space operations since the unit Hilbert sphere is a positively curved manifold. When analyzing real brain imaging data, it is entirely possible that no meaningful correlations exist in the data. The key difficulty is that we do not know whether the experiment (i.e., inference) failed because there is in fact no statistically meaningful signal in the data set or if the algorithms being used are suboptimal.

Related Work There are two somewhat distinct bodies of work that are related to and motivate this work. The first one pertains to the extensive study of the classical CCA and its nonlinear variants. These include various interesting results based on kernelization [2, 8, 20], neural networks [26, 36], and deep architectures [3]. Most, if not all of these strategies extend CCA to arbitrary nonlinear spaces. However, this flexibility brings with it the associated issues of model selection (and thereby, regularization), controlling the complexity of the neural network structure, choosing an appropriate activation function, and so on. On the other hand, a second line of work incorporates the specific geometry of the data directly within the estimation problem. Various statistical constructs have been generalized to Riemannian manifolds: these include regression [45, 57], classification [52], kernel methods [31], margin-based and boosting classifiers [37], interpolation, convolution, filtering [19], and dictionary learning [23, 38]. Among the most closely related are ideas dealing with projective dimensionality reduction methods, for instance, the generalization of principal components analysis (PCA) via the so-called principal geodesic analysis (PGA) [17], geodesic PCA [30], exact PGA [48], horizontal dimension reduction [47] with frame bundles, and an extension of PGA to the product space of Riemannian manifolds, namely, tensor fields [52]. Except the nonparametric method of [49], most of these strategies focus on one rather than two sets of random variables (as is the case in CCA). Even in this setting, the first results on successful generalization of parametric regression models to Riemannian manifolds are relatively recent: geodesic regression [16, 41] and polynomial regression [22]. It should be noted that the adaptive CCA formulation in [53] seems related to this work but is in fact not designed for manifold-valued input data.

We describe a parametric model between two different tensor fields on a Riemannian manifold, which falls beyond these recent works. The CCA formulation we present requires the optimization of functions over either a single product manifold or a pair of product manifolds (of different dimensions) concurrently. The latter problem involving product manifolds of different dimensions will not be addressed here. Note that in general, on manifolds the projection operation does not have a nice closed form solution. So, we need to perform projections via an optimization scheme on the two manifolds and find the best pair of geodesic subspaces. We provide a solution to this problem.

4.2 Canonical Correlation in Euclidean Space

First, we will briefly review the classical CCA in Euclidean space to motivate the rest of our presentation. Recall that Pearson's product–moment correlation coefficient is a quantity to measure the relationship of two random variables, $x \in \mathbf{R}$ and $y \in \mathbf{R}$. For one dimensional random variables,

$$\rho_{x,y} = \frac{\mathrm{COV}(x,y)}{\sigma_x \sigma_y} = \frac{\mathbb{E}[(x-\mu_x)(y-\mu_y)]}{\sigma_x \sigma_y} = \frac{\sum_{i=1}^N (x_i-\mu_x)(y_i-\mu_y)}{\sqrt{\sum_{i=1}^N (x_i-\mu_x)^2}\sqrt{\sum_{i=1}^N (y_i-\mu_y)^2}} \tag{4.1}$$

where μ_x and μ_y are the means of $\{x_i\}_{i=1}^N$ and $\{y_i\}_{i=1}^N$. For high-dimensional data, $\boldsymbol{x} \in \mathbf{R}^m$ and $\mathbf{y} \in \mathbf{R}^n$, we cannot perform a direct calculation as above. So, we need to project each set of variables onto a special axis in each space $\mathcal{X}$ and $\mathcal{Y}$. CCA generalizes the concept of correlation to random vectors (potentially of different dimensions). It is convenient to think of CCA as a measure of correlation between two multivariate data based on the *best* projection which maximizes their mutual correlation.

Canonical correlation for $\boldsymbol{x} \in \mathbf{R}^m$ and $\mathbf{y} \in \mathbf{R}^n$ is given by

$$\max_{\boldsymbol{w}_x,\boldsymbol{w}_y} \mathrm{corr}(\pi_{\boldsymbol{w}_x}(\boldsymbol{x}), \pi_{\boldsymbol{w}_y}(\mathbf{y})) = \max_{\boldsymbol{w}_x,\boldsymbol{w}_y} \frac{\sum_{i=1}^N \boldsymbol{w}_x^T(\boldsymbol{x}_i-\boldsymbol{\mu}_x)\boldsymbol{w}_y^T(\mathbf{y}_i-\boldsymbol{\mu}_y)}{\sqrt{\sum_{i=1}^N \left(\boldsymbol{w}_x^T(\boldsymbol{x}_i-\boldsymbol{\mu}_x)\right)^2}\sqrt{\sum_{i=1}^N \left(\boldsymbol{w}_y^T(\mathbf{y}_i-\boldsymbol{\mu}_y)\right)^2}} \tag{4.2}$$

where $\pi_{\boldsymbol{w}_x}(\boldsymbol{x}) := \arg\min_{t\in\mathbf{R}} \ \mathrm{d}(t\boldsymbol{w}_x, \boldsymbol{x}-\boldsymbol{\mu}_x)^2 = \arg\min_{t\in\mathbf{R}} \ \mathrm{d}(\boldsymbol{\mu}_x + t\boldsymbol{w}_x, \boldsymbol{x})^2$ and $\boldsymbol{\mu}_x, \boldsymbol{w}_x \in \mathbf{R}^m$, $\boldsymbol{\mu}_y, \boldsymbol{w}_y \in \mathbf{R}^n$. We will call $\pi_{\boldsymbol{w}_x}(\boldsymbol{x})$ the *projection coefficient* for $\boldsymbol{x}$ (similarly for $\mathbf{y}$). Define $S_{\boldsymbol{w}_x}$ as the subspace which is the span of $\boldsymbol{w}_x$. The projection of $\boldsymbol{x}$ onto $S_{\boldsymbol{w}_x}$ is given by $\Pi_{S_{\boldsymbol{w}_x}}(\boldsymbol{x})$. Without loss of generality, assume that $\boldsymbol{x}, \mathbf{y}$ are centered. We can then verify that the relationship between the projection and the projection coefficient is

$$\Pi_{S_{\boldsymbol{w}_x}}(\boldsymbol{x}) := \arg\min_{\boldsymbol{x}'\in S_{\boldsymbol{w}_x}} \mathrm{d}(\boldsymbol{x},\boldsymbol{x}')^2 = \frac{\boldsymbol{w}_x^T\boldsymbol{x}}{\|\boldsymbol{w}_x\|}\frac{\boldsymbol{w}_x}{\|\boldsymbol{w}_x\|} = \frac{\boldsymbol{w}_x^T\boldsymbol{x}}{\|\boldsymbol{w}_x\|^2}\boldsymbol{w}_x = \pi_{\boldsymbol{w}_x}(\boldsymbol{x})\boldsymbol{w}_x \tag{4.3}$$

In the Euclidean space, $\Pi_{S_{\boldsymbol{w}_x}}(\boldsymbol{x})$ has a closed form solution. In fact, it is obtained by an inner product, $\boldsymbol{w}_x^T\boldsymbol{x}$. Hence, by replacing the projection coefficient $\pi_{\boldsymbol{w}_x}(\boldsymbol{x})$ with $\boldsymbol{w}_x^T\boldsymbol{x}/\|\boldsymbol{w}_x\|^2$ and after a simple calculation, one obtains the form in (4.2). Using a matrix representation, the optimization problem can be written as

$$\max_{\boldsymbol{w}_x,\boldsymbol{w}_y} \quad \boldsymbol{w}_x^T X^T Y \boldsymbol{w}_y \text{ subject to } \boldsymbol{w}_x^T X^T X \boldsymbol{w}_x = \boldsymbol{w}_y^T Y^T Y \boldsymbol{w}_y = 1 \tag{4.4}$$

where $\boldsymbol{x}, \boldsymbol{w}_x \in \mathbf{R}^m$, $\mathbf{y}, \boldsymbol{w}_y \in \mathbf{R}^n$, $X = [\boldsymbol{x}_1 \ldots \boldsymbol{x}_N]^T$ and $Y = [\mathbf{y}_1 \ldots \mathbf{y}_N]^T$. The only difference here is that we remove the denominator. Instead, we have two equality constraints (note that correlation is scale invariant).

4.3 Preliminaries

We now briefly summarize certain basic concepts [14] which we will use shortly.

Riemannian Manifolds A *differentiable manifold* [14] of dimension n is a set $\mathcal{M}$ and a family of *injective* mappings $\varphi_i : U_i \subset \mathbf{R}^n \to \mathcal{M}$ of open sets U_i of $\mathbf{R}^n$ into $\mathcal{M}$ such that (1) $\cup_i \varphi_i(U_i) = \mathcal{M}$; (2) for any pair i, j with $\varphi_i(U_i) \cap \varphi_j(U_j) = W \neq \phi$, the sets $\varphi_i^{-1}(W)$ and $\varphi_j^{-1}(W)$ are open sets in $\mathbf{R}^n$ and the mappings $\varphi_j^{-1} \circ \varphi_i$ are differentiable, where $\circ$ denotes function composition.

In other words, a differentiable manifold $\mathcal{M}$ is a topological space that is locally homeomorphic to a Euclidean space and has a globally defined differential structure. The tangent space at a point p on the manifold, $T_p\mathcal{M}$, is a vector space that consists of the tangent vectors of *all* possible curves passing through p.

A Riemannian manifold is equipped with a smoothly varying inner product. The family of inner products on all tangent spaces is known as the *Riemannian metric* of the manifold. The *geodesic distance* between two points on $\mathcal{M}$ is the length of the shortest *geodesic* curve connecting the two points, analogous to straight lines in $\mathbf{R}^n$. The geodesic curve from x_i to x_j can be parameterized by a tangent vector in the tangent space at y_i with an exponential map $\mathrm{Exp}(y_i, \cdot) : T_{y_i}\mathcal{M} \to \mathcal{M}$. The inverse of the exponential map is the logarithm map, $\mathrm{Log}(y_i, \cdot) : \mathcal{M} \to T_{y_i}\mathcal{M}$. Separate from these notations, matrix exponential and logarithm are given as $\exp(\cdot)$ and $\log(\cdot)$.

Intrinsic Mean Given $\boldsymbol{x}_i \in \mathcal{M}$, for $i = 1, \ldots, N$ with a given Riemannian metric d, the finite sample Fréchet mean is defined as the minimizer of sum of squared geodesic distances, given explicitly by,

$$\boldsymbol{\mu}^* = \arg\min_{\boldsymbol{\mu} \in \mathcal{M}} \sum_{i=1}^{N} \mathrm{d}^2(\boldsymbol{x}_i, \boldsymbol{\mu}) \tag{4.5}$$

Uniqueness of Fréchet mean in general is not guaranteed. We refer the reader to [1] for a detailed exposition on the same.

Geodesically Convex A subset C of $\mathcal{M}$ is said to be a *geodesically convex set* if there is a minimizing geodesic curve in C between *any* two points in C. This assumption is commonly used [16] and essential to ensure that the Riemannian operations such as the exponential and logarithm maps are well defined.

4.4 A Model for CCA on Riemannian Manifolds

We now present a step-by-step derivation of our Riemannian CCA model. Classical CCA finds the mean of each data modality. Then, it maximizes correlation between projected data on each subspace translated to each mean. Similarly, CCA on manifolds must first compute the intrinsic mean (i.e., Fréchet mean) of each data

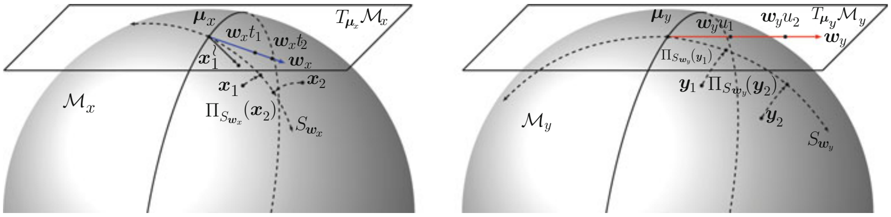


Fig. 4.1 CCA on Riemannian manifolds. CCA searches geodesic submanifolds (subspaces), S_{w_x} and S_{w_y} at the Fréchet mean of data on each manifold. Correlation between projected points $\{\Pi_{S_{w_x}}(\boldsymbol{x}_i)\}_{i=1}^N$ and $\{\Pi_{S_{w_y}}(\mathbf{y}_i)\}_{i=1}^N$ is equivalent to the correlation between *projection coefficients* $\{t_i\}_{i=1}^N$ and $\{u_i\}_{i=1}^N$. Although $\boldsymbol{x}$ and $\mathbf{y}$ belong to the same manifold we show them in different plots for ease of explanation

set. It then identifies a "generalized" version of a subspace at each Fréchet mean to maximize the correlation of projected data. The generalized form of a subspace on Riemannian manifolds has been studied in the literature [17, 30, 37, 48]. The so-called *geodesic submanifold* [17, 33, 52] which has been used for geodesic regression serves our purpose well and is defined as $S = \text{Exp}(\boldsymbol{\mu}, \text{span}(\{\boldsymbol{v}_i\}) \cap U)$, where $U \subset T_{\boldsymbol{\mu}}\mathcal{M}$, and $\boldsymbol{v}_i \in T_{\boldsymbol{\mu}}\mathcal{M}$ [17]. When S has only one tangent vector $\boldsymbol{v}$, then the geodesic submanifold is simply a geodesic curve, see Fig. 4.1.

We can now proceed to formulate the precise form of projection onto a geodesic submanifold. Recall that when given a point, its projection on a set is the closest point in the set. So, the projection onto a geodesic submanifold (S) must be a function satisfying this behavior. This is given by

$$\Pi_S(\boldsymbol{x}) = \arg\min_{\boldsymbol{x}' \in S} \quad \text{d}(\boldsymbol{x}, \boldsymbol{x}')^2 \tag{4.6}$$

In Euclidean space, the projection on a convex set (e.g., subspace) is unique. It is also unique on some manifolds under special conditions, e.g., quaternion sphere [44]. However, the uniqueness of the projection on geodesic submanifolds in general cannot be ensured. Like other methods [32, 48], we assume that given the specific manifold and the data, the projection is well posed.

Finally, the correlation of points (*after* projection) can be measured by the distance from the mean to the projected points. To be specific, the projection on a geodesic submanifold corresponding to $\boldsymbol{w}_x$ is given by

$$\Pi_{S_{w_x}}(\boldsymbol{x}) \quad := \arg\min_{\boldsymbol{x}' \in S_{w_x}} \|\text{Log}(\boldsymbol{x}, \boldsymbol{x}')\|_{\boldsymbol{x}}^2 \tag{4.7}$$

$S_{w_x} := \text{Exp}(\boldsymbol{\mu}_x, \text{span}\{\boldsymbol{w}_x\} \cap U)$ where $\boldsymbol{w}_x$ is a basis tangent vector and $U \subset T_{\boldsymbol{\mu}_x}\mathcal{M}_x$ is a small neighborhood of $\boldsymbol{\mu}_x$. Then, the expression for *projection coefficients* is

$$t_i = \pi_{w_x}(\boldsymbol{x}_i) \quad := \arg\min_{t_i' \in (-\epsilon, \epsilon)} \|\text{Log}(\text{Exp}(\boldsymbol{\mu}_x, t_i' \boldsymbol{w}_x), \boldsymbol{x}_i)\|_{\boldsymbol{\mu}_x}^2 \tag{4.8}$$

where $\boldsymbol{x}_i, \boldsymbol{\mu}_x \in \mathcal{M}_x, \boldsymbol{w}_x \in T_{\boldsymbol{\mu}_x}\mathcal{M}_x, t_i \in \mathbf{R}$. The term $u_i = \pi_{\boldsymbol{w}_y}(\mathbf{y})$ is defined analogously. Here, t_i is a real value to obtain the point $\Pi_{S_{\boldsymbol{w}_x}}(\boldsymbol{x}) = \mathrm{Exp}(\boldsymbol{\mu}_x, t_i\boldsymbol{w}_x)$. As mentioned above, $\boldsymbol{x}$ and $\mathbf{y}$ belong to the same manifold. However, we use two different notations $\mathcal{M}_x$ and $\mathcal{M}_y$ to show that they are differently distributed for ease of discussion.

Notice that we have

$$\mathrm{d}(\boldsymbol{\mu}_x, \Pi_{S_{\boldsymbol{w}_x}}(\boldsymbol{x}_i)) = \|\mathrm{Log}(\boldsymbol{\mu}_x, \mathrm{Exp}(\boldsymbol{\mu}_x, \boldsymbol{w}_x t_i))\|_{\boldsymbol{\mu}_x} = t_i\|\boldsymbol{w}_x\|_{\boldsymbol{\mu}_x} \tag{4.9}$$

By inspection, this shows that the projection coefficient is proportional to the length of the geodesic curve from the base point $\boldsymbol{\mu}_x$ to the projection of $\boldsymbol{x}$, $\Pi_{S_{\boldsymbol{w}_x}}(\boldsymbol{x})$. Correlation is scale invariant, as expected. Therefore, the correlation between projected points $\{\Pi_{S_{\boldsymbol{w}_x}}(\boldsymbol{x}_i)\}_{i=1}^N$ and $\{\Pi_{S_{\boldsymbol{w}_y}}(\mathbf{y}_i)\}_{i=1}^N$ reduces to the correlation between the quantities that serve as projection coefficients here, $\{t_i\}_{i=1}^N$ and $\{u_i\}_{i=1}^N$.

Putting these pieces together, we obtain our generalized formulation for CCA:

$$\rho_{\boldsymbol{x},\mathbf{y}} = \mathrm{corr}(\pi_{\boldsymbol{w}_x}(\boldsymbol{x}), \pi_{\boldsymbol{w}_y}(\mathbf{y})) = \max_{\boldsymbol{w}_x,\boldsymbol{w}_y,\boldsymbol{t},\boldsymbol{u}} \frac{\sum_{i=1}^N (t_i - \bar{t})(u_i - \bar{u})}{\sqrt{\sum_{i=1}^N (t_i - \bar{t})^2}\sqrt{\sum_{i=1}^N (u_i - \bar{u})^2}} \tag{4.10}$$

where $t_i = \pi_{\boldsymbol{w}_x}(\boldsymbol{x}_i)$, $\boldsymbol{t} := \{t_i\}$, $u_i = \pi_{\boldsymbol{w}_y}(\mathbf{y}_i)$, $\boldsymbol{u} := \{u_i\}$, $\bar{t} = \frac{1}{N}\sum_{i=1}^N t_i$, and $\bar{u} = \frac{1}{N}\sum_{i=1}^N u_i$. Expanding out components in (4.10) further, it takes the form

$$\begin{aligned}\rho_{\boldsymbol{x},\mathbf{y}} = \max_{\boldsymbol{w}_x,\boldsymbol{w}_y,\boldsymbol{t},\boldsymbol{u}} & \frac{\sum_{i=1}^N (t_i - \bar{t})(u_i - \bar{u})}{\sqrt{\sum_{i=1}^N (t_i - \bar{t})^2}\sqrt{\sum_{i=1}^N (u_i - \bar{u})^2}} \\ \text{subject to } \ & t_i = \arg\min_{t_i \in (-\epsilon,\epsilon)} \|\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\mu}_x, t_i\boldsymbol{w}_x), \boldsymbol{x}_i)\|^2, \forall i \in \{1,\ldots,N\} \\ & u_i = \arg\min_{u_i \in (-\epsilon,\epsilon)} \|\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\mu}_y, u_i\boldsymbol{w}_y), \mathbf{y}_i)\|^2, \forall i \in \{1,\ldots,N\}\end{aligned} \tag{4.11}$$

Directly, we see that (4.11) is a multilevel optimization and solutions from nested suboptimization problems may be needed to solve the higher level problem. However, deriving the first-order optimality conditions leads to a cleaner formulation:

Define functions as the following.

$$\begin{aligned} f(\boldsymbol{t},\boldsymbol{u}) &:= \frac{\sum_{i=1}^N (t_i - \bar{t})(u_i - \bar{u})}{\sqrt{\sum_{i=1}^N (t_i - \bar{t})^2}\sqrt{\sum_{i=1}^N (u_i - \bar{u})^2}}, \\ g(t_i, \boldsymbol{w}_x) &:= \|\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\mu}_x, t_i\boldsymbol{w}_x), \boldsymbol{x}_i)\|^2, \\ g(u_i, \boldsymbol{w}_y) &:= \|\mathrm{Log}(\mathrm{Exp}(\boldsymbol{\mu}_y, u_i\boldsymbol{w}_y), \mathbf{y}_i)\|^2. \end{aligned} \tag{4.12}$$

Then, we can replace the equality constraints in (4.11) with optimality conditions rather than another optimization problem for each i. Using this idea, we have

$$\begin{aligned}\rho_{\boldsymbol{x},\mathbf{y}} = \max_{\boldsymbol{w}_x,\boldsymbol{w}_y,\boldsymbol{t},\boldsymbol{u}} & f(\boldsymbol{t},\boldsymbol{u}) \\ \text{subject to } & \nabla_{t_i} g(t_i,\boldsymbol{w}_x) = 0, \forall i \in \{1,\dots,N\} \\ & \nabla_{u_i} g(u_i,\boldsymbol{w}_y) = 0, \forall i \in \{1,\dots,N\}\end{aligned} \tag{4.13}$$

4.5 Optimization Schemes

We present two different algorithms to solve the problem of computing CCA on Riemannian manifolds. The first algorithm is based on a numerical optimization scheme for (4.13). We explain the main model here and provide technical details. Later, we present another approach which is based on an approximation for computational efficiency.

4.5.1 An Augmented Lagrangian Method

Due to the nature of our formulation, especially the constraints, our options for numerical optimization scheme are limited. In particular, to avoid dealing with the second-order derivatives of the constraints, we use first-order methods. One option here is a gradient projection method. However, we will need to define distance metric over the decision variables and projection are on the feasible set accordingly. In this case, efficient projections on the feasible set are not available.

The other option is a quadratic penalty algorithm. Given a constrained optimization problem $\max f(\boldsymbol{x})$ s.t. $c_i(\boldsymbol{x}) = 0, \forall i$, such an algorithm optimizes the quadratic penalty function, i.e., $\max f(x) - \nu^k \sum_i c_i(x)^2$. Classical penalty algorithms iteratively solve a sequence of models above while increasing ν^k to infinity. Here, we chose a well-known variation of the penalty method called augmented Lagrangian technique (ALM). It is generally preferred to the classical quadratic penalty method since there is little extra computational cost. In particular, by avoiding $\mu \to \infty$, we reduce the possibility of ill conditioning by introducing explicit Lagrange multiplier estimates into the function to be minimized [42].

The augmented Lagrangian method solves a sequence of the following models while increasing ν_k:

$$\max f(\boldsymbol{x}) + \sum_i \lambda_i c_i(\boldsymbol{x}) - \nu^k \sum_i c_i(\boldsymbol{x})^2 \tag{4.14}$$

Algorithm 1 Riemannian CCA based on the Augmented Lagrangian method

```
1: $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{M}_x, \mathbf{y}_1, \ldots, \mathbf{y}_N \in \mathcal{M}_y$
2: Given $\nu^0 > 0, \tau^0 > 0,$ starting points $(\boldsymbol{w}_x^0, \boldsymbol{w}_y^0, \boldsymbol{t}^0, \boldsymbol{u}^0)$ and $\boldsymbol{\lambda}^0$
3: for $k = 0, 1, 2 \ldots$ do
4:     Start at $(\boldsymbol{w}_x^k, \boldsymbol{w}_y^k, \boldsymbol{t}^k, \boldsymbol{u}^k)$
5:     Find an approximate minimizer $(\boldsymbol{w}_x^k, \boldsymbol{w}_y^k, \boldsymbol{t}^k, \boldsymbol{u}^k)$ of $\mathcal{L}_A(\cdot, \boldsymbol{\lambda}^k; \nu^k)$, and terminate when
       $\|\nabla \mathcal{L}_A(\boldsymbol{w}_x^k, \boldsymbol{w}_y^k, \boldsymbol{t}^k, \boldsymbol{u}^k, \boldsymbol{\lambda}^k; \nu^k)\| \leq \tau^k$
6:     if a convergence test for (4.13) is satisfied then
7:         Stop with approximate feasible solution
8:     end if
9:     $\lambda_{t_i}^{k+1} = \lambda_{t_i}^k - \nu^k \nabla_{t_i} g(t_i, \boldsymbol{w}_x), \forall i$
10:    $\lambda_{u_i}^{k+1} = \lambda_{u_i}^k - \nu^k \nabla_{u_i} g(u_i, \boldsymbol{w}_y), \forall i$
11:    Choose new penalty parameter $\nu^{k+1} \geq \nu^k$
12:    Set starting point for the next iteration
13:    Select tolerance $\tau^{k+1}$
14: end for
```

The augmented Lagrangian formulation for our CCA formulation is given by

$$\max_{\boldsymbol{w}_x, \boldsymbol{w}_y, \boldsymbol{t}, \boldsymbol{u}} \mathcal{L}_A(\boldsymbol{w}_x, \boldsymbol{w}_y, \boldsymbol{t}, \boldsymbol{u}, \boldsymbol{\lambda}^k; \nu^k) = \max_{\boldsymbol{w}_x, \boldsymbol{w}_y, \boldsymbol{t}, \boldsymbol{u}} f(\boldsymbol{t}, \boldsymbol{u}) + \sum_i^N \lambda_{t_i}^k \nabla_{t_i} g(t_i, \boldsymbol{w}_x)$$
$$+ \sum_i^N \lambda_{u_i}^k \nabla_{u_i} g(u_i, \boldsymbol{w}_y) - \frac{\nu^k}{2} \left(\sum_{i=1}^N \nabla_{t_i} g(t_i, \boldsymbol{w}_x)^2 + \nabla_{u_i} g(u_i, \boldsymbol{w}_y)^2 \right) \tag{4.15}$$

The pseudocode for our algorithm is summarized in Algorithm 1.

Remarks. Note that for Algorithm 1, we need the second derivative of g, in particular, for $\frac{d^2}{dwdt} g$ and $\frac{d^2}{dt^2} g$. The literature does not provide a great deal of guidance on second derivatives of functions involving Log($\cdot$) and Exp($\cdot$) maps on general Riemannian manifolds. However, depending on the manifold, it can be obtained analytically or numerically (see "Appendix 1").

Approximate Strategies The difficulty in deriving the algorithm above was the lack of a closed form solution to projections onto geodesic submanifolds. If however, an approximate form of the projection can lead to significant gains in computational efficiency with little sacrifice in accuracy, it may be useful in practice. The simplest approximation is to use a Log-Euclidean model. But it is well known that the Log-Euclidean is only reasonable for data that are tightly clustered on the manifold and not otherwise. Further, the Log-Euclidean metric lacks the important property of affine invariance. We can obtain a more accurate projection using the submanifold expression given in [52]. The form of projection is

$$\Pi_S(\boldsymbol{x}) \approx \text{Exp}(\boldsymbol{\mu}, \sum_{i=1}^d \boldsymbol{v}_i \langle \boldsymbol{v}_i, \text{Log}(\boldsymbol{\mu}, \boldsymbol{x}) \rangle_{\boldsymbol{\mu}}) \tag{4.16}$$

Algorithm 2 CCA with approximate projection

```
1: Input $X_1, \ldots, X_N \in \mathcal{M}_y$, $Y_1, \ldots, Y_N \in M_y$
2: Compute intrinsic mean $\boldsymbol{\mu}_x, \boldsymbol{\mu}_y$ of $\{X_i\}, \{Y_i\}$
3: Compute $X_i^{\wr} = \text{Log}(\boldsymbol{\mu}_x, X_i)$, $Y_i^{\wr} = \text{Log}(\boldsymbol{\mu}_y, Y_i)$
4: Transform (using group action) $\{X_i^{\wr}\}, \{Y_i^{\wr}\}$ to the $T_I\mathcal{M}_x, T_I\mathcal{M}_y$
5: Perform CCA between $T_I\mathcal{M}_x, T_I\mathcal{M}_y$ and get axes $W_a \in T_I\mathcal{M}_x$, $W_b \in T_I\mathcal{M}_y$
6: Transform (using group action) $W_a, W_b$ to $T_{\boldsymbol{\mu}_x}\mathcal{M}_x$, $T_{\boldsymbol{\mu}_y}\mathcal{M}_y$
```

where $\{\boldsymbol{v}_i\}$ denotes an *orthonormal basis* at $T_{\boldsymbol{\mu}}\mathcal{M}$. The CCA algorithm with this approximation for the projection is summarized as Algorithm 2.

Algorithm 2 finds a globally optimal solution to the approximate problem, i.e., the classical version of CCA between two tangent spaces $T_I\mathcal{M}_x$ and $T_I\mathcal{M}_y$. It does not require any initialization. On the other hand, Algorithm 1 is a first-order optimization scheme. It converges to a local minimum. Different intializations may lead to different local solutions. In our experiments, for Algorithm 1, we initialized $\mathbf{w}_x$ and $\mathbf{w}_y$ by Algorithm 2. Further, $\boldsymbol{t}$ and $\boldsymbol{u}$ are initialized by the corresponding projection coefficients to $\mathbf{w}_x$ and $\mathbf{w}_y$ using the iterative method minimizing (4.8).

Finally, we provide a brief remark on one remaining issue. This relates to the question of why we use the group action rather than other transformations such as parallel transport. Observe that Algorithm 2 sends the data from the tangent space at the Fréchet mean of the samples to the tangent space at identity I. The purpose of the transformation is to put all measurements at the identity of the SPD manifold, to obtain a more accurate projection, which can be understood by inspecting (4.16). The projection and inner product depend on the anchor point μ. If μ is identity, then there is no discrepancy between the Euclidean and the Riemannian inner products. Of course, one may use a parallel transport, if desired. However, a group action may be substantially more efficient relative to parallel transport since the former does not require computing a geodesic curve (which *is* needed for parallel transport). Interestingly, it turns out that on SPD manifolds with a GL-invariant metric, parallel transport from an arbitrary point p to Identity I is *equivalent* to the transform using a group action. For this reason, one can parallel transport tangent vectors from p to I using the group action much more efficiently [50]. This is formalized in the following two results.

Theorem 4.1. *On SPD manifold, let $\Gamma_{p\to I}(w)$ denote the parallel transport of $w \in T_p\mathcal{M}$ along the geodesic from $p \in \mathcal{M}$ to $I \in \mathcal{M}$. The parallel transport is equivalent to a group action by $p^{-1/2}wp^{-T/2}$, where the inner product $\langle u, v\rangle_p = tr(p^{-1/2}up^{-1}vp^{-1/2})$.*

Proof. Parallel transport Γ from p to q is given by [15]

$$\Gamma_{p\to q}(w) = p^{1/2}rp^{-1/2}wp^{-1/2}rp^{1/2}$$

where $r = \exp(p^{-1/2}\frac{v}{2}p^{-1/2})$ and $v = \text{Log}(p, q) = p^{1/2}\log(p^{-1/2}qp^{-1/2})p^{1/2}$

Let us transform the tangent vector w at $T_p\mathcal{M}$ to I by setting $q = I$:

$$\Gamma_{p\to I}(w) = p^{1/2}rp^{-1/2}wp^{-1/2}rp^{1/2} \quad \text{where } r = \exp(p^{-1/2}\frac{v}{2}p^{-1/2}) \quad \text{and}$$

$$v = \mathrm{Log}(p, I) = p^{1/2}\log(p^{-1/2}Ip^{-1/2})p^{1/2} = p^{1/2}\log(p^{-1})p^{1/2} \qquad \text{(a)}$$

Then, r is given as

$$\begin{aligned} r &= \exp(p^{-1/2}\frac{v}{2}p^{-1/2}) \\ &= \exp(p^{-1/2}p^{1/2}\log(p^{-1})p^{1/2}p^{-1/2}/2), \text{ by (a)} \\ &= \exp(\log(p^{-1})/2) \\ &= p^{-1/2} \end{aligned} \qquad \text{(b)}$$

Also,

$$\begin{aligned} \Gamma_{p\to I}(w) &= p^{1/2}rp^{-1/2}wp^{-1/2}rp^{1/2} \\ &= p^{1/2}p^{-1/2}p^{-1/2}wp^{-1/2}p^{-1/2}p^{1/2}, \text{ by (b)} \\ &= p^{-1/2}wp^{-1/2} \\ &= p^{-1/2}wp^{-T/2} \text{ since } p^{-1/2} \text{ is SPD.} \end{aligned}$$

Theorem 4.2. *On SPD manifolds, let $\Gamma_{I\to q}(w)$ denote the parallel transport of $w \in T_I\mathcal{M}$ along the geodesic from $I \in \mathcal{M}$ to $q \in \mathcal{M}$. The parallel transport is equivalent to a group action by $q^{1/2}wq^{T/2}$.*

Proof. The proof is similar to that of Theorem 4.1. By substitution, the parallel transport is given by $\Gamma_{I\to q}(w) = rwr$, where $r = \exp(\frac{v}{2})$ and $v = \mathrm{Log}(I, q) = \log(q)$. Then, r is $q^{1/2}$. Hence, $\Gamma_{I\to q}(w) = q^{1/2}wq^{1/2} = q^{1/2}wq^{T/2}$ since $q^{1/2}$ is SPD.

Remarks. Theorems 4.1 and 4.2 show that the parallel transport from or to I is replaceable with group actions. However, in general, the parallel transport of $w \in T_p\mathcal{M}$ from p to q is *not* equivalent to the composition of group actions to transform from p to I and from I to q. This is consistent with the fact that parallel transport from a to c may not be same as two parallel transports: from a to b and then from b to c. The following is an example for SPD(2) manifold.

Example 4.5.1. When p and q are given as

$$p = \begin{pmatrix} 2 & 3 \\ 3 & 5 \end{pmatrix} \quad q = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} \quad w = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

The parallel transport of w from p to q is

$$\Gamma_{p\to q}(w) \approx \begin{pmatrix} -8 & -1 \\ -1 & 0 \end{pmatrix}$$

Transform of w by two group actions ($p \to I \to q$) is

$$q^{1/2}p^{-1/2}wp^{-T/2}q^{T/2} \approx \begin{pmatrix} -4 & 1 \\ 1 & 0 \end{pmatrix}$$

The transform by group actions above is identical to the composition of two parallel transports $\Gamma_{I\to q}(\Gamma_{p\to I}(w))$. However, it is different from $\Gamma_{p\to q}(w)$.

Note We use the first-order optimality condition in (4.13). In general, the first order optimality condition is necessary but not a sufficient condition. So, is (4.13) a relaxed version of (4.5.2)? Interestingly, on SPD manifolds, the first order condition is sufficient for the optimality of projection. To see this, we introduce the concept of *geodesic convexity* [43].

Definition 4.1. A set $A \subset \mathcal{M}$ is *geodesically convex* (g-convex) if any two points of A are joined by a geodesic belonging to A.

Definition 4.2. Let $A \subset \mathcal{M}$ be a g-convex set. Then, a function $f : A \to R$ is g-convex if its restrictions to all geodesic arcs belonging to A are convex in the arc length parameter, i.e., if $t \mapsto f(t) \equiv f(\text{Exp}(x, tu))$ is convex over its domain for all $x \in \mathcal{M}$ and $u \in T_x\mathcal{M}$, where $\text{Exp}(x, \cdot)$ is the exponential map at x [39].

Lemma 4.1. *The function $d(Exp(\mu, tu), S)$ on SPD manifolds is convex with respect to t where $\mu, S \in \mathcal{M}$ and $u \in T_\mu\mathcal{M}$.*

Proof. This can be shown by the definition of the geodesic convexity of the function and the fact that the real-valued function defined on $SPD(n)$ by $P \mapsto d(P, S)$ is geodesically convex, where $S \in SPD(n)$ is fixed and $d(\cdot, \cdot)$ is the geodesic distance [39, 40].

Lemma 4.1 shows that the projection to a geodesic curve on SPD manifolds is a convex problem and the first-order condition for projection coefficients is sufficient.

4.5.2 Extensions to the Product Riemannian Manifold

For applications of this algorithm, we study the problem of statistical analysis of an entire population of images (of multiple types). For such data, each image must be treated as a single entity, which necessitates extending the formulation above to a Riemannian product space.

In other words, our CCA will be performed on product manifolds, i.e., $\mathcal{M}_x := \mathcal{M}_x^1 \times \ldots \times \mathcal{M}_x^m$ and $\mathcal{M}_y := \mathcal{M}_y^1 \times \ldots \times \mathcal{M}_y^n$. We seek $W_x := (W_x^1, \ldots, W_x^m) \in T_{\boldsymbol{\mu}_x}\mathcal{M}_x$, $W_y := (W_y^1, \ldots, W_y^m) \in T_{\boldsymbol{\mu}_y}\mathcal{M}_y$, where $T_{\mu_x}\mathcal{M}_x := T_{\mu_x^1}\mathcal{M}_x^1 \times \ldots T_{\mu_x^m}\mathcal{M}_x^m$ and $T_{\mu_y}\mathcal{M}_y := T_{\mu_y^1}\mathcal{M}_y^1 \times \ldots T_{\mu_y^n}\mathcal{M}_y^n$. We will discuss a Riemannian metric on the product space and *projection coefficients*. Finally, we will offer the extended formulation of our method.

First, let us define a Riemannian metric on the product space $\mathcal{M} = \mathcal{M}^1 \times \ldots \times \mathcal{M}^m$. A natural choice is the following idea from [52]:

$$\langle \boldsymbol{X}_1, \boldsymbol{X}_2 \rangle_{\boldsymbol{P}} = \sum_{j=1}^{m} \langle X_1^j, X_2^j \rangle_{P^j} \tag{4.17}$$

where $\boldsymbol{X}_1 = \left(X_1^1, \ldots, X_1^m\right) \in \mathcal{M}$, and $\boldsymbol{X}_2 = \left(X_2^1, \ldots, X_2^m\right) \in \mathcal{M}$ and $\boldsymbol{P} = \left(P^1, \ldots, P^m\right) \in \mathcal{M}$. Once we have the exponential and logarithm maps, CCA on a Riemannian product space can be directly performed by Algorithm 2. The exponential map $\mathrm{Exp}(\boldsymbol{P}, \boldsymbol{V})$ and logarithm map $\mathrm{Log}(\boldsymbol{P}, \boldsymbol{X})$ are given by

$$(\mathrm{Exp}(P^1, V^1), \ldots, \mathrm{Exp}(P^m, V^m)) \text{ and } (\mathrm{Log}(P^1, X^1), \ldots, \mathrm{Log}(P^m, X^m)) \tag{4.18}$$

respectively, where $\boldsymbol{V} = (V^1, \ldots, V^m) \in T_{\boldsymbol{P}}\mathcal{M}$. The length of tangent vector is $\|\boldsymbol{V}\| = \sqrt{\|V^1\|_{P^1}^2 + \cdots + \|V^m\|_{P^m}^2}$, where $V^i \in T_{P^i}\mathcal{M}_i$. The geodesic distance between two points $\mathrm{d}(\boldsymbol{X}_1, \boldsymbol{X}_2)$ on Riemannian product space is also measured by the length of tangent vector from one point to the other. So, we have

$$\mathrm{d}(\boldsymbol{\mu}_x, \boldsymbol{X}) = \sqrt{\mathrm{d}(\mu_x^1, X^1)^2 + \cdots + \mathrm{d}(\mu_x^m, X^m)^2} \tag{4.19}$$

From our previous discussion of the relationship between *projection coefficients* and distance from the mean to points (after *projection*) in Sect. 4.4, we have $t_i = \mathrm{d}(\boldsymbol{\mu}_x, \Pi_{S_{\boldsymbol{W}_x}}(\boldsymbol{X}_i))/\|\boldsymbol{W}_x\|_{\boldsymbol{\mu}_x}$ and $t_i^j = \mathrm{d}(\mu_x^j, \Pi_{S_{W_x^j}}(X_i^j))/\|W_x^j\|_{\mu_x^j}$. By substitution, the *projection coefficients* on Riemannian product space are given by

$$t_i = \mathrm{d}(\boldsymbol{\mu}_x, \Pi_{S_{\boldsymbol{W}_x}}(\boldsymbol{X}_i))/\|\boldsymbol{W}_x\|_{\boldsymbol{\mu}_x} = \sqrt{\sum_{j}^{m} \left(t_i^j \left\| W_x^j \right\|_{\mu_x^j} \right)^2 \Big/ \sum_{j=1}^{m} \left\| W_x^j \right\|_{\mu_x^j}^2} \tag{4.20}$$

We can now mechanically substitute these "product space" versions of the terms in (4.20) to derive a CCA on a Riemannian product space.

Our formulation is

$$\rho_{X,Y} = \max_{\boldsymbol{W}_x, \boldsymbol{W}_y, \boldsymbol{t}, \boldsymbol{u}} \frac{\sum_{i=1}^{N}(t_i - \bar{t})(u_i - \bar{u})}{\sqrt{\sum_{i=1}^{N}(t_i - \bar{t})^2}\sqrt{\sum_{i=1}^{N}(u_i - \bar{u})^2}}$$

$$s.t. \;\; t_i^j = \arg\min_{t_i^j \in (-\epsilon, \epsilon)} \|\mathrm{Log}(\mathrm{Exp}(\mu_x^j, t_i^j W_x^j), X_i^j)\|^2, \forall i, \forall j$$

$$u_i^k = \arg \min_{u_i^k \in (-\epsilon, \epsilon)} \|\mathrm{Log}(\mathrm{Exp}(\mu_y^k, u_i^k W_y^k), Y_i^k)\|^2, \forall i, \forall k \tag{4.21}$$

$$t_i = \frac{\sqrt{\sum_{j=1}^m \left(t_i^j \|W_x^j\|_{\mu_x^j}\right)^2}}{\sqrt{\sum_{j=1}^m \|W_x^j\|^2_{\mu_x^j}}}, u_i = \frac{\sqrt{\sum_{k=1}^n \left(u_i^k \|W_y^k\|_{\mu_y^k}\right)^2}}{\sqrt{\sum_{k=1}^n \|W_y^k\|^2_{\mu_y^k}}}, \bar{t} = \frac{1}{N}\sum_i^N t_i, \bar{u} = \frac{1}{N}\sum_i^N u_i \forall i$$

where $i \in \{1, \ldots, N\}, j \in \{1, \ldots, m\}$, and $\forall k \in \{1, \ldots, n\}$. This can be optimized by constrained optimization algorithms similar to those described in Sect. 4.5 with relatively minor changes.

4.6 Experiments

4.6.1 CCA on SPD Manifolds

Diffusion tensors are symmetric positive definite matrices at each voxel in a diffusion tensor image (DTI). Let SPD(n) be a manifold for symmetric positive definite matrices of size $n \times n$. This forms a quotient space $GL(n)/O(n)$, where $GL(n)$ denotes the general linear group and $O(n)$ is the orthogonal group. The inner product of two tangent vectors $u, v \in T_p\mathcal{M}$ is given by $\langle u, v\rangle_p = \mathrm{tr}(p^{-1/2}up^{-1}vp^{-1/2})$. Here, $T_p\mathcal{M}$ is a tangent space at p, which is the $(n+1)n/2$ dimensional vector space of symmetric matrices. The geodesic distance is $d(p,q)^2 = \mathrm{tr}(\log^2(p^{-1/2}qp^{-1/2}))$.

Here, the exponential map and logarithm map are defined as

$$\mathrm{Exp}(p, v) = p^{1/2}\exp(p^{-1/2}vp^{-1/2})p^{1/2}, \quad \mathrm{Log}(p, q) = p^{1/2}\log(p^{-1/2}qp^{-1/2})p^{1/2} \tag{4.22}$$

and the first derivative of g in (4.13) on SPD(n) is given by

$$\begin{aligned}\frac{d}{dt_i}g(t_i, \boldsymbol{w}_x) &= \frac{d}{dt_i}\|\mathrm{Log}(\mathrm{Exp}(\mu_x, t_iW_x), X_i)\|^2 = \frac{d}{dt_i}\mathrm{tr}[\log^2(X_i^{-1}S(t_i))] \\ &= 2\mathrm{tr}[\log(X_i^{-1}S(t_i))S(t_i)^{-1}\dot{S}(t_i)], \text{ according to Proposition 2.1 in [39]}\end{aligned} \tag{4.23}$$

where

$$\begin{aligned}S(t_i) &= \mathrm{Exp}(\mu_x, t_iW_x) = \mu_x^{1/2}\exp^{t_iA}\mu_x^{1/2} \\ \dot{S}(t_i) &= \mu_x^{1/2}A\exp^{t_iA}\mu_x^{1/2} \\ A &= \mu_x^{-1/2}W_x\mu_x^{-1/2}\end{aligned} \tag{4.24}$$

Note that the derivative of the equality constraints in (4.13), namely, $\frac{d^2}{dWdt}g$, $\frac{d^2}{dt^2}g$, are calculated numerically. The numerical differentiation requires an orthonormal basis of the tangent space. This can be easily accomplished using an embedding described in more detail in the appendix.

4.6.2 Synthetic Experiments

In this section, we provide experimental results using a synthetic data set to evaluate the performance of Riemannian CCA. To simplify presentation, we introduce two operations vec($\cdot$) and mat($\cdot$) that will be used in the following description. We use "vec" to give the operation of embedding tangent vectors in $T_I\mathcal{M}$ into $\mathbf{R}^6$; "mat" refers to the inversion of "vec." By construction, we have $\langle S_1, S_2\rangle_I = \langle v_1, v_2\rangle$, where $v_i = \text{vec}(S_i)$. In other words, the distance from a base point/origin to each point is identical in the two spaces by the construction. Using group actions and these subroutines, points can be mapped from an arbitrary tangent space $T_p\mathcal{M}$ to $\mathbf{R}^6$ or vice versa, where $p \in \mathcal{M}$. On SPD(3), the two operations are given by

$$\text{vec}(S) := [s_{11}, \sqrt{2}s_{12}, \sqrt{2}s_{13}, s_{22}, \sqrt{2}s_{23}, s_{33}]^T, \text{where } S = \begin{bmatrix} s_{11} & s_{12} & s_{13} \\ s_{21} & s_{22} & s_{23} \\ s_{31} & s_{32} & s_{33} \end{bmatrix} \quad \text{and}$$

$$\text{mat}(\boldsymbol{v}) := \begin{bmatrix} v_1 & \frac{1}{\sqrt{2}}v_2 & \frac{1}{\sqrt{2}}v_3 \\ \frac{1}{\sqrt{2}}v_2 & v_4 & \frac{1}{\sqrt{2}}v_5 \\ \frac{1}{\sqrt{2}}v_3 & \frac{1}{\sqrt{2}}v_5 & v_6 \end{bmatrix}, \text{where } \boldsymbol{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_6 \end{bmatrix}^T \tag{4.25}$$

Our CCA algorithm with approximate projections, namely, Algorithm 2, can be implemented by these two subroutines with group actions.

We now discuss the synthetic data generation. The samples are generated to be spread far apart on the manifold $\mathcal{M}(\equiv \text{SPD}(3))$—observe that if the data are closely clustered, a manifold algorithm will behave similar to its non-manifold counterpart. We generate data around two well-separated means $\mu_{x_1}, \mu_{x_2} \in \mathcal{X}$, $\mu_{y_1}, \mu_{y_2} \in \mathcal{Y}$ by perturbing the data randomly in the corresponding tangent spaces, i.e., adding Gaussian-like noise in each tangent space at cluster mean μ_{x_j} and μ_{y_j}, where $j \in \{1, 2\}$ is the index for cluster. The procedure is described in Algorithm 3.

We plot data projected onto the CCA axes (P_X and P_Y) and compute the correlation coefficients. In our experiments, we see that the algorithm offers improvements. For example, by inspecting the correlation coefficients $\rho_{x,y}$ in Fig. 4.2, we see that manifold CCA yields significantly better correlation relative to other baselines.

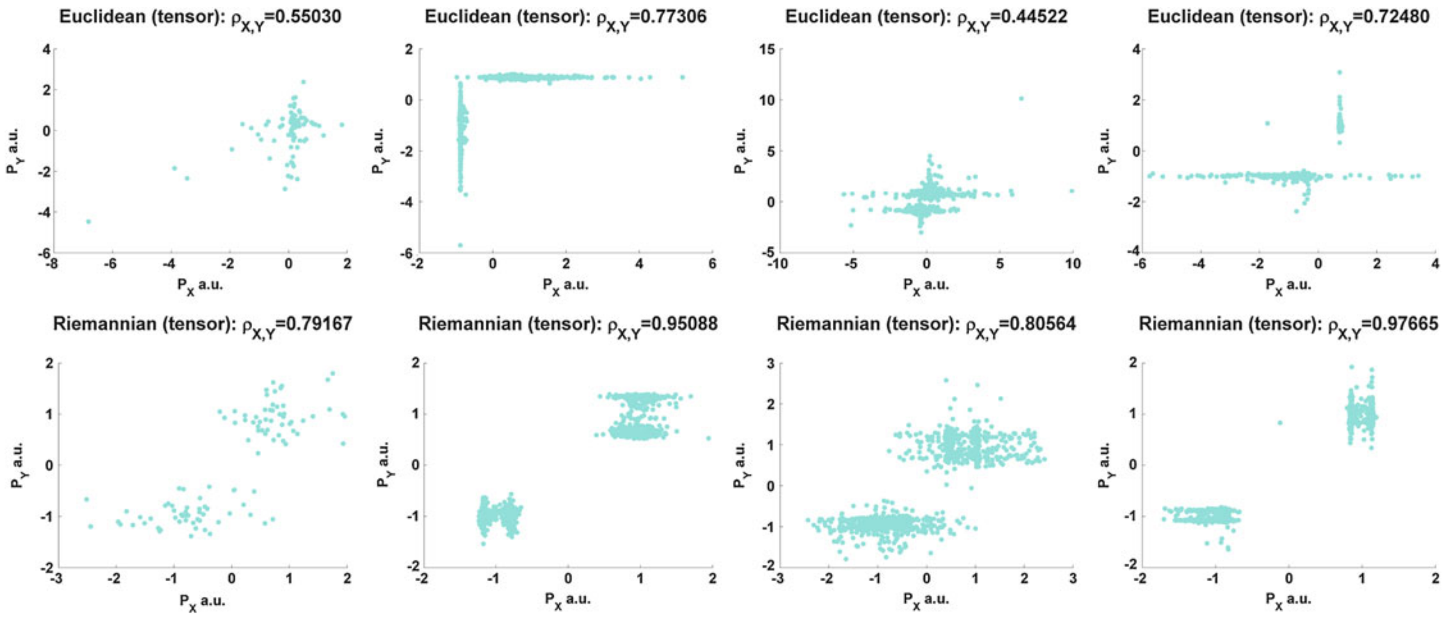


Fig. 4.2 Synthetic experiments showing the benefits of Riemannian CCA. The *top row* shows the projected data using the Euclidean CCA and the *bottom* using Riemannian CCA. P_X and P_Y denote the projected axes. Each column represents a synthetic experiment with a specific set of $\{\mu_{x_j}, \epsilon_{x_j}; \mu_{y_j}, \epsilon_{y_j}\}$. The first column has 100 samples while the three columns on the *right* have 1000 samples. The improvements in the correlation coefficients $\rho_{x,y}$ can be seen from the corresponding titles

Algorithm 3 The procedure simulates truncated-Gaussian-like noise. The second step (safeguard) in the pseudocode ensures that the data live in a reasonably small neighborhood to avoid numerical issues. We define subroutines *mat* for mapping from $\mathbf{R}^{n(n+1)/2}$ to SPD(n) and *vec* for the inversion

```
1: $\boldsymbol{\epsilon}' \in \mathbf{R}^{n(n+1)/2} \sim \mathcal{N}(0, \sigma I)$
2: $\boldsymbol{\epsilon}' \leftarrow \boldsymbol{\epsilon}' \min(\|\boldsymbol{\epsilon}'\|, c_1)/\|\boldsymbol{\epsilon}'\|$,        ▷ $c_1$ is a parameter for a safeguard
3: $\epsilon_I \leftarrow \mathrm{mat}(\boldsymbol{\epsilon}')$,        ▷ tangent vector at $I$
4: Transform (using group action) $\epsilon_I$ to $T_\mu\mathcal{M}$
5: Perturb data $X \leftarrow \mathrm{Exp}(\mu, \epsilon_\mu)$, where $\epsilon_\mu \in T_\mu\mathcal{M}$
```

4.6.3 CCA for Multimodal Risk Analysis

Motivation We collected multimodal magnetic resonance imaging (MRI) data to investigate the effects of risk for Alzheimer's disease (AD) on the white and gray matter in the brain. One of the goals in analyzing this data set is to find statistically significant relationships between AD risk and the brain structure. We can adopt many different ways of modeling these relationships but a potentially useful way is to analyze multimodality imaging data simultaneously, using CCA.

In the current experiments, we include a subset of 343 subjects and first investigate the effects of age and gender in a multimodal fashion since these variables are also important factors in healthy aging. Brain structure is characterized by diffusion-weighted images (DWI) for white matter and T1-weighted (T1W) image data for the gray matter. DWI data provide us information about the microstructure of the white matter. We use diffusion tensor ($\mathcal{D} \in$ SPD(3)) model to represent the diffusivity in the microstructure. T1W data can be used to obtain volumetric properties of the gray matter [18]. The volumetric information is obtained from Jacobian matrices (J) of the diffeomorphic mapping to a population-specific template. These Jacobian matrices can be used to obtain the Cauchy deformation tensors ($\sqrt{J^T J}$) which also belong to SPD(3).

We first focus our analysis using two important regions called hippocampus and cingulum bundle (shown in Fig. 4.3) which are significantly affected by Alzheimer's disease. However, the statistical power of detecting changes in the brain structures *before* the memory/cognitive function is impaired is difficult due to several factors such as noise in the data, small sample, and effect sizes. One approach to improving statistical power in such a setting is to perform only a few number of tests using average properties of the substructures. This procedure reduces both noise and the number of tests. However, taking averages will also dampen the signal of interest which is already weak in certain studies. CCA can take the multimodal information from the imaging data and project the voxels into a space where the signal of interest is likely to be stronger.

Experimental Design The main idea is to detect age and gender effects on the gray and white matter interactions. Hence the key multimodal linear relations we examine for the purpose are

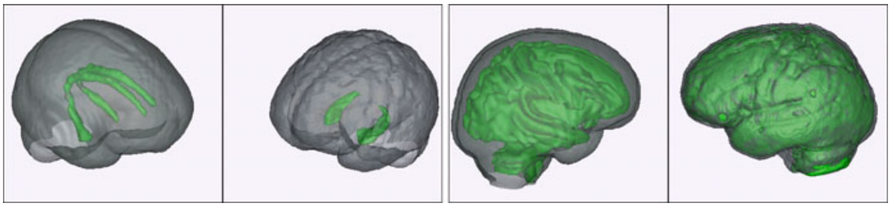


Fig. 4.3 Shown on the *left* are the bilateral cingulum bundles (*green*) inside a brain surface obtained from a population DTI template. Similarly, on the *right* are the bilateral hippocampi. The full gray matter and white matter are shown on the *right*

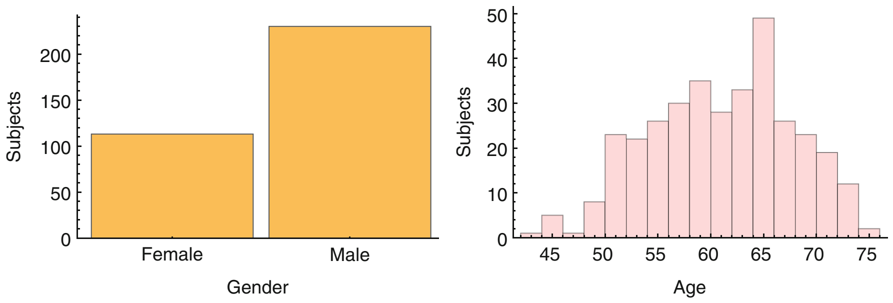


Fig. 4.4 The sample characteristics in terms of gender and age distributions

$$Y_{\text{DTI}} = \beta_0 + \beta_1 \text{Gender} + \beta_2 X_{\text{T1W}} + \beta_3 X_{\text{T1W}} \cdot \text{Gender} + \varepsilon$$

$$Y_{\text{DTI}} = \beta'_0 + \beta'_1 \text{AgeGroup} + \beta'_2 X_{\text{T1W}} + \beta'_3 X_{\text{T1W}} \cdot \text{AgeGroup} + \varepsilon$$

where the AgeGroup is defined as a categorical variable with 0 (middle aged) if the age of the subject ≤ 65 and 1 (old) otherwise. The sample under investigation is between 43 and 75 years of age (see Fig. 4.4 for the full distributions of age and gender). The statistical tests we ask are if the null hypotheses $\beta_3 = 0$ and $\beta'_3 = 0$ can be rejected using our data at $\alpha = 0.05$. The gender and age distributions of the sample are shown in Fig. 4.4.

We report the results from the following four sets of analyses: (i) Classical ROI-average analysis: This is a standard type of setting where the brain measurements in an ROI are averaged. Here, $Y_{\text{DTI}} = \overline{\text{MD}}$, i.e., the average mean diffusivity in the cingulum bundle. $X_{\text{T1W}} = \overline{\log |J|}$, i.e., the average volumetric change (relative to the population template) in the hippocampus. (ii) Euclidean CCA using scalar measures (MD and $\log |J|$) in the ROIs: Here, the voxel data are projected using the classical CCA approach [51], i.e., $Y_{\text{DTI}} = \mathbf{w}_{\text{MD}}^T \text{MD}$ and $X_{\text{T1W}} = \mathbf{w}_{\log |J|}^T \log |J|$. (iii) Euclidean CCA using $\mathcal{D}$ and $\mathcal{J}$ in the ROIs: This setting is an improvement to the setting in **(ii)** above in that the projections are performed using the full tensor data [51]. Here, $Y_{\text{DTI}} = \mathbf{w}_{\mathcal{D}}^T \mathcal{D}$ and $X_{\text{T1W}} = \mathbf{w}_{\mathcal{J}}^T \mathcal{J}$. (iv) Riemannian CCA using $\mathcal{D}$ and $\mathcal{J}$ in the ROIs: Here $Y_{\text{DTI}} = \langle \mathbf{w}_{\mathcal{D}}, \mathcal{D} \rangle_{\mu_{\mathcal{D}}}$ and $X_{\text{T1W}} = \langle \mathbf{w}_{\mathcal{J}}, \mathcal{J} \rangle_{\mu_{\mathcal{J}}}$.

We first show the statistical sensitivities of the four approaches in Fig. 4.5. We can see that the performance of CCA using the full tensor information improves the statistical significance for both Euclidean and Riemannian approaches. The weight vectors in the different settings for both Euclidean and Riemannian CCA are shown in Fig. 4.6, top row. We would like to note that there are several different approaches of using the data from CCA and we performed additional experiments with full gray matter and white matter regions in the brain. We only show and discuss briefly the representative weight vectors in the bottom row of Fig. 4.6 and refer the interested reader to "Appendix 2" for more comprehensive details of the full brain experiments. Interestingly, the weight vectors are spatially cohesive even without enforcing any spatial constraints. What is even more interesting is that the regions picked between the DTI and T1W modalities are complimentary in a biological sense. Specifically, when performing our CCA on the ROIs, although the cingulum bundle extends into the superior midbrain regions, the weights are nonzero in its hippocampal projections. In the case of entire white and gray matter regions, the volumetric difference (from the population template) in the inferior part of the corpus callosum seems to be highly cross-correlated to the diffusivity in the corpus callosum. Our CCA finds these projections without any a priori constraints in the optimization suggesting that performing CCA on the native data can reveal biologically meaningful patterns.

We have presented in these experiments (including those in the appendix) evidence that CCA when performed using the intrinsic properties of the MRI data can reveal biologically meaningful patterns without any a priori biological input to the model. We showed that we can perform various types of multimodal hypothesis testing of linear relationships using the projection vectors from the CCA, which can be easily extended to discriminant analysis (predicting gender and age group using the multimodal brain data) using the CCA projection vectors. CCA can be applied to settings beyond multimodal imaging data, where one can try to directly maximize the correlation between imaging and nonimaging data using a cross-validation technique [7]. Our Riemannian CCA can provide a starting point for such studies.

4.7 Conclusion

The classical CCA assumes that data live in a pair of vector spaces. However, many modern scientific disciplines require the analysis of data which belong to *curved* spaces where classical CCA is no longer applicable. Motivated by the properties of imaging data from neuroimaging studies, we generalize CCA to Riemannian manifolds. We employ differential geometry tools to extend operations in CCA to the manifold setting. Such a formulation results in a multilevel optimization problem. We derive solutions using the first-order condition of projection and an augmented Lagrangian method. In addition, we also develop an efficient single path algorithm with approximate projections. Finally, we propose a generalization to the

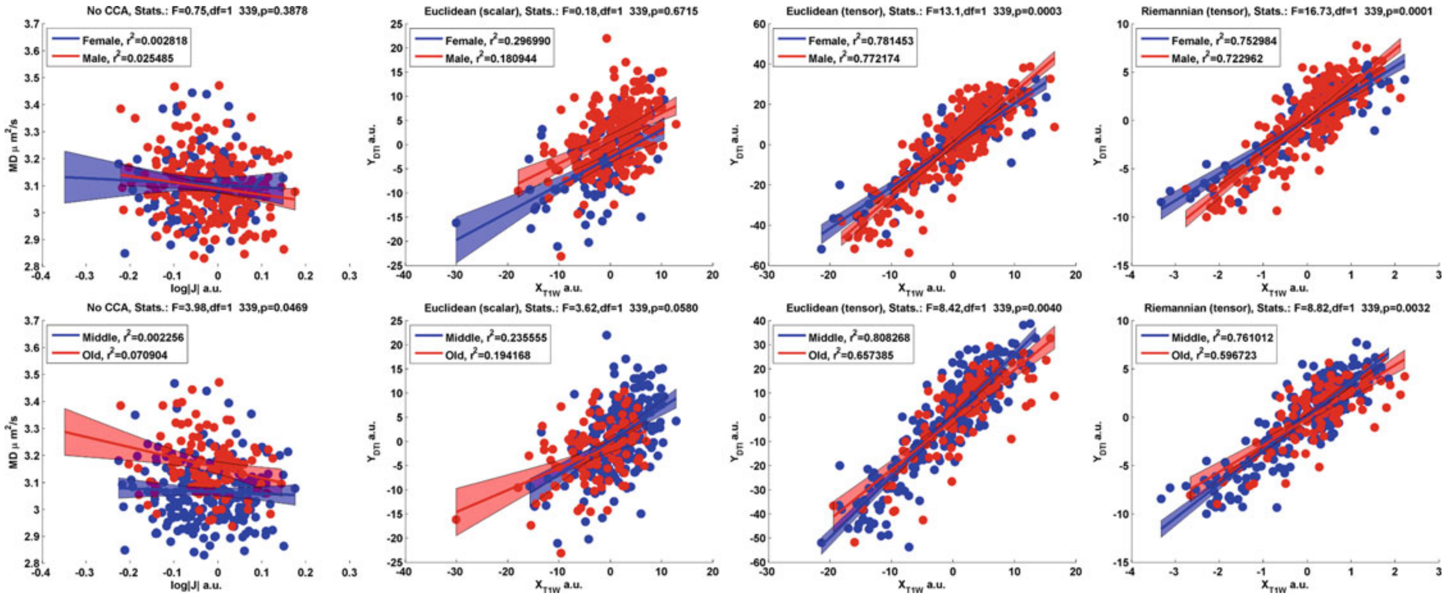


Fig. 4.5 Experimental evidence showing the improvements in statistical significance of finding the multi-modal risk-brain interaction effects. *Top row* shows the gender, volume, and diffusivity interactions. *Second row* shows the interaction effects of the middle/old age groups

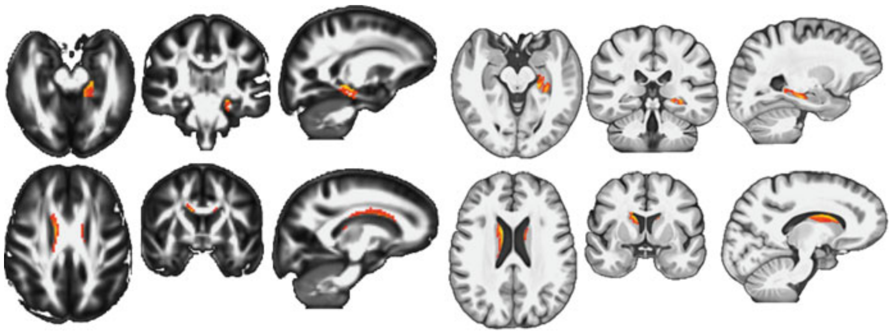


Fig. 4.6 Weight vectors (in *red-yellow color*) obtained from our Riemannian CCA approach. The weights are in arbitrary units. The *top row* is from applying Riemannian CCA on data from the cingulum and hippocampus structures (Fig. 4.3) while the *bottom row* is obtained using data from the entire white and gray matter regions of the brain. On the *left* (three columns) block we show the results in orthogonal view for DTI and on the *right* for T1W. The corresponding underlays are the population averages of the fractional anisotropy and T1W contrast images, respectively

product space of SPD(n), namely, tensor fields allowing us to treat a full brain image as a point on the product manifold. Experiments show the applicability of these ideas for the analysis of neuroimaging data.

Acknowledgements This work was supported in part by NIH grants AG040396 (VS), AG037639 (BBB), AG021155 (SCJ), AG027161 (SCJ), NS066340 (BCV), NSF CAREER award 1252725 (VS). Partial support was also provided by UW ADRC (AG033514), UW ICTR, Waisman Core grant (P30 HD003352), and the Center for Predictive Computational Phenotyping (CPCP) at UW-Madison (AI117924). The contents do not represent views of the Dept. of Veterans Affairs or the United States Government.

Appendix 1

The iterative method Algorithm 1 for Riemannian CCA with exact projection needs first and second derivatives of g in (4.13). We provide more details here.

First Derivative of g for SPD

Given SPD(n), the gradient of g with respect to t is obtained by the following proposition in [39].

Proposition 4.1. *Let $F(t)$ be a real matrix-valued function of the real variable t. We assume that, for all t in its domain, $F(t)$ is an invertible matrix which does not have eigenvalues on the closed negative real line. Then,*

$$\frac{d}{dt} tr[\log^2 F(t)] = 2tr[\log F(t) F(t)^{-1} \frac{d}{dt} F(t)] \tag{4.26}$$

The derivation of $\frac{d}{dt_i} g(t_i, \boldsymbol{w}_x)$ proceeds as

$$\begin{aligned} \frac{d}{dt_i} g(t_i, \boldsymbol{w}_x) &= \frac{d}{dt_i} \|\text{Log}(\text{Exp}(\mu_x, t_i W_x), X_i)\|^2 \\ &= \frac{d}{dt_i} \text{tr}[\log^2(X_i^{-1} S(t_i))] \end{aligned} \tag{4.27}$$

where $S(t_i) = \text{Exp}(\mu_x, t_i W_x) = \mu_x^{1/2} \exp^{t_i A} \mu_x^{1/2}$ and $A = \mu_x^{-1/2} W_x \mu_x^{-1/2}$.

In our formulation, $F(t) = X_i^{-1} S(t_i)$. Then, we have $F(t)^{-1} = S(t_i)^{-1} X_i$ and $\frac{d}{dt} F(t) = X_i^{-1} \dot{S}(t_i)$. Hence, the derivative of g with respect to t_i is given by

$$\begin{aligned} \frac{d}{dt_i} g(t_i, \boldsymbol{w}_x) &= 2\text{tr}[\log(X_i^{-1} S(t_i)) S(t_i)^{-1} X_i X_i^{-1} \dot{S}(t_i)], \text{ according to Proposition 4.1} \\ &= 2\text{tr}[\log(X_i^{-1} S(t_i)) S(t_i)^{-1} \dot{S}(t_i)] \end{aligned} \tag{4.28}$$

where $\dot{S}(t_i) = \mu_x^{1/2} A \exp^{t_i A} \mu_x^{1/2}$.

Numerical Expression for the Second Derivative of g

Riemannian CCA with exact projection can be optimized by Algorithm 1. Observe that the objective function of the proposed augmented Lagrangian method $\mathcal{L}_A$ includes the term ∇g in (4.13). The gradient of $\mathcal{L}_A$ involves the second derivative of g. More precisely, we need $\frac{d^2}{dwdt} g$ and $\frac{d^2}{dt^2} g$. These can be estimated by a finite difference method

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \tag{4.29}$$

Obviously, $\frac{d^2}{dt^2} g$ can be obtained by the expression above using the analytical first derivative $\frac{d}{dt} g$. For $\frac{d^2}{dwdt} g$, we use the orthonormal basis in $T_{\mu_x} \mathcal{M}$ to approximate the derivative. By definition of directional derivative, we have

$$\lim_{h \to 0} \sum_i^d \left(\frac{f(x + hu_i) - f(x)}{h} \right) u_i = \sum_i^d \langle \nabla_x f(x), u_i \rangle u_i = \nabla_x f(x) \tag{4.30}$$

where $x \in \mathcal{X}$, d is dimension of $\mathcal{X}$, and $\{u_i\}$ is orthonormal basis of $\mathcal{X}$. Hence, perturbation along the orthonormal basis enables us to approximate the gradient. For example, on SPD(n) manifolds, the orthonormal basis in arbitrary tangent space $T_p\mathcal{M}$ can be obtained by following three steps:

Step a) Pick an orthonormal basis $\{e_i\}$ of $R^{n(n+1)/2}$,
Step b) Convert $\{e_i\}$ into n-by-n symmetric matrices $\{u_i\}$ in $T_I\mathcal{M}$, i.e., $\{u_i\}$ =mat($\{e_i\}$),
Step c) Transform basis $\{u_i\}$ from $T_I\mathcal{M}$ to $T_p\mathcal{M}$.

Appendix 2

MR Image Acquisition and Processing All the MRI data were acquired on a GE 3.0 Tesla scanner Discovery MR750 MRI system with an 8-channel head coil and parallel imaging (ASSET). The DWI data were acquired using a diffusion-weighted, spin-echo, single-shot, echo planar imaging radiofrequency (RF) pulse sequence with diffusion weighting in 40 noncollinear directions at $b = 1300\,\text{s/mm}^2$ in addition to 8 $b = 0$ (nondiffusion-weighted or T2-weighted) images. The cerebrum was covered using contiguous 2.5-mm-thick axial slices, FOV $= 24\,\text{cm}$, TR $=$ $8000\,\text{ms}$, TE $= 67.8\,\text{ms}$, matrix $= 96 \times 96$, resulting in isotropic $2.5\,\text{mm}^3$ voxels. High-order shimming was performed prior to the DTI acquisition to optimize the homogeneity of the magnetic field across the brain and to minimize EPI distortions. The brain region was extracted using the first $b = 0$ image as input to the brain extraction tool (BET), also part of the FSL software.

Eddy current-related distortion and head motion of each data set were corrected using FSL software package [46]. The b-vectors were rotated using the rotation component of the transformation matrices obtained from the correction process. Geometric distortion from the inhomogeneous magnetic field applied was corrected with the $b = 0$ field map and PRELUDE (phase region expanding labeler for unwrapping discrete estimates) and FUGUE (FMRIBs utility for geometrically unwarping EPIs) from FSL. Twenty-nine subjects did not have field maps acquired during their imaging session. Because these participants did not differ on APOE4 genotype, sex, or age compared to the participants who had field map correction, they were included in order to enhance the final sample size. The diffusion tensors were then estimated from the corrected DWI data using nonlinear least squares estimation using the Camino library [13].

Individual maps were registered to a population-specific template constructed using diffusion tensor imaging toolkit (DTI-TK[1]), which is an optimized DTI spatial normalization and atlas construction tool that has been shown to perform superior registration compared to scalar-based registration methods [55]. The template is constructed in an unbiased way that captures both the average diffusion features (e.g., diffusivities and anisotropy) and anatomical shape features (tract size) in the population. A subset of 80 diffusion tensor maps was used to create a common space template. All diffusion tensor maps were normalized to the template with first rigid followed by affine and then symmetric diffeomorphic transformations. The diffeomorphic coordinate deformations themselves are smooth and invertible, that is, neuroanatomical neighbors remain neighbors under the mapping. At the same time, the algorithms used to create these deformations are *symmetric* in that they are not biased towards the reference space chosen to compute the mappings. Moreover, these topology-preserving maps capture the large deformation necessary to aggregate populations of images in a common space. The spatially normalized data were interpolated to 2×2×2 mm voxels for the final CCA analysis.

Along with the DWI data the T1-weighted images were acquired using BRAVO pulse sequence which uses 3D inversion recovery (IR) prepared fast spoiled gradient recalled echo (FSPGR) acquisition to produce isotropic images at 1×1×1 mm resolution. We extract the brain regions again using BET. We compute an optimal template space, i.e., a population-specific, unbiased average shape and appearance image derived from our population [4]. We use the openly available advanced normalization tools (ANTS[2]) to develop our template space and also perform the registration of the individual subjects to that space [5]. ANTS encodes current best practice in image registration, optimal template construction, and segmentation [35]. Once we perform the registrations we extract the Jacobian matrices (J) per voxel per subject from these deformation fields and compute the Cauchy deformation tensors ($\sqrt{J^T J}$) for performing CCA analysis.

Additional Experiments As described in the main text (Sect. 4.6.3), we examine the following multimodal linear relations but this time by performing CCA on the entire gray and white matter structures rather than just the hippocampi and cinguli:

$$Y_{\mathrm{DTI}} = \beta_0 + \beta_1 \mathrm{Gender} + \beta_2 X_{\mathrm{T1W}} + \beta_3 X_{\mathrm{T1W}} \cdot \mathrm{Gender} + \varepsilon$$

$$Y_{\mathrm{DTI}} = \beta_0' + \beta_1' \mathrm{AgeGroup} + \beta_2' X_{\mathrm{T1W}} + \beta_3' X_{\mathrm{T1W}} \cdot \mathrm{AgeGroup} + \varepsilon$$

To enable this analysis, the gray matter region was defined as follows: First, we performed a three tissue segmentation of each of the spatially normalized

[1] http://dti-tk.sourceforge.net/pmwiki/pmwiki.php.

[2] http://www.picsl.upenn.edu/ANTS/.

T1-weighted images into gray, white, and cerebral spinal fluid using FAST segmentation algorithm [56] implemented in FSL. Then, we take the average of the gray matter probabilities using all the subjects and threshold it to obtain the final binary mask resulting in ∼700,000 voxels. The white matter region is simply defined as the region with fractional anisotropy (FA) obtained from the diffusion tensors > 0.2 which resulted in about 50,000 voxels. We would like to contrast this with the region-specific analyses where the number of voxels was much smaller. To address this in our CCA we imposed an L_1-norm penalty to the weight vectors (similar to Euclidean CCA [51]) in our CCA objective function with a tuning parameter λ.

In addition to the above linear relationships, CCA can also facilitate testing the following relationships by using the weight-vectors as substructures of interest and taking the average mean diffusivity ($\overline{MD}$) and average volumetric change ($\overline{\log |J|}$) in those structures as the outcome measures:

$$\overline{MD} = \beta_0 + \beta_1 \delta_{\text{Female}} + \beta_2 \delta_{\text{Male}} + \varepsilon$$

$$\overline{\log |J|} = \beta_0' + \beta_1' \delta_{\text{Female}} + \beta_2' \delta_{\text{Male}} + \varepsilon$$

where the null hypotheses to be tested are $\beta_1 = \beta_2$ and $\beta_1' = \beta_2'$. Similarly, we can find statistically significant AgeGroup differences by examining the following two models:

$$\overline{MD} = \beta_0 + \beta_1 \delta_{\text{Middle}} + \beta_2 \delta_{\text{Old}} + \varepsilon$$

$$\overline{\log |J|} = \beta_0' + \beta_1' \delta_{\text{Middle}} + \beta_2' \delta_{\text{Old}} + \varepsilon$$

The scatter and bar plots for the testing linear relationships using both Riemannian and Euclidean CCA are shown in Figs. 4.7 and 4.8.

We now present a comprehensive set of montages of all the slices of the brain overlaid by the weight vectors obtained from CCA from Figs. 4.9, 4.10, 4.11, and 4.12.

We can observe that the voxels with nonzero weights (highlighted in red-boxes) are spatially complimentary in DTI and T1W. Even more interestingly, CCA finds the cingulum regions in the white matter for the DTI modality. In our experiments, we observe the same regions for various settings of a sparsity parameter (used heuristically to obtain more interpretable regions). Similarly, Figs. 4.11 and 4.12 show the results using the Euclidean CCA performed using the full tensor information [51]. We can observe that the results are similar to those in Figs. 4.9 and 4.10 but the regions are thinner in the Euclidean version.

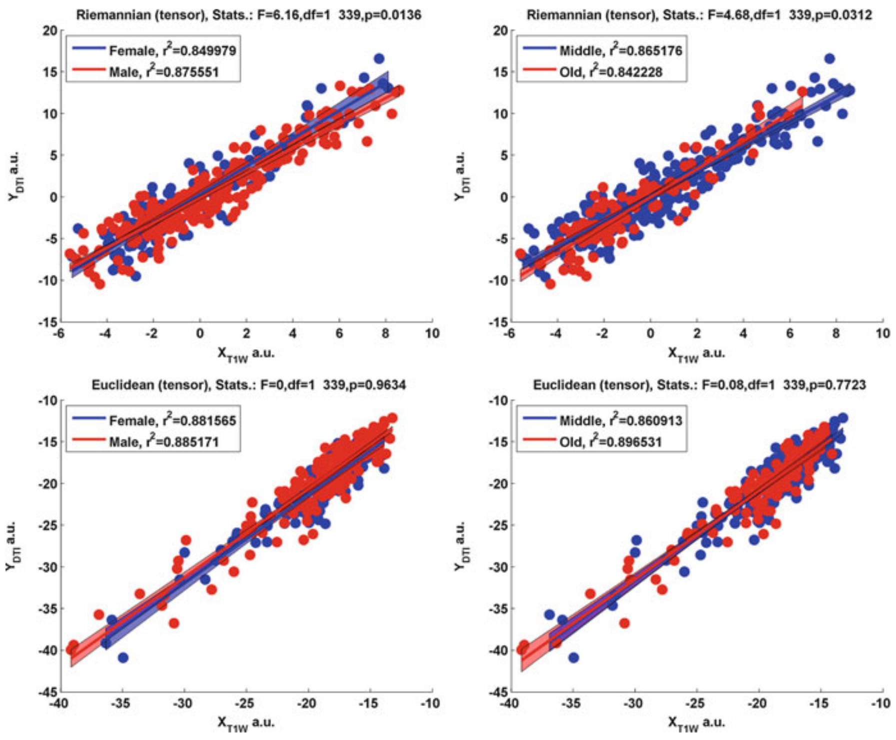


Fig. 4.7 CCA projections revealing statistically significant volume and diffusivity interactions with gender (*left*) and age group (*right*). *Top row* shows the results using Riemannian CCA and the *bottom row* using the Euclidean CCA with vectorization. We can clearly see improvements in the statistical confidence (smaller *p*-values) for rejecting the null hypotheses when using Riemannian CCA

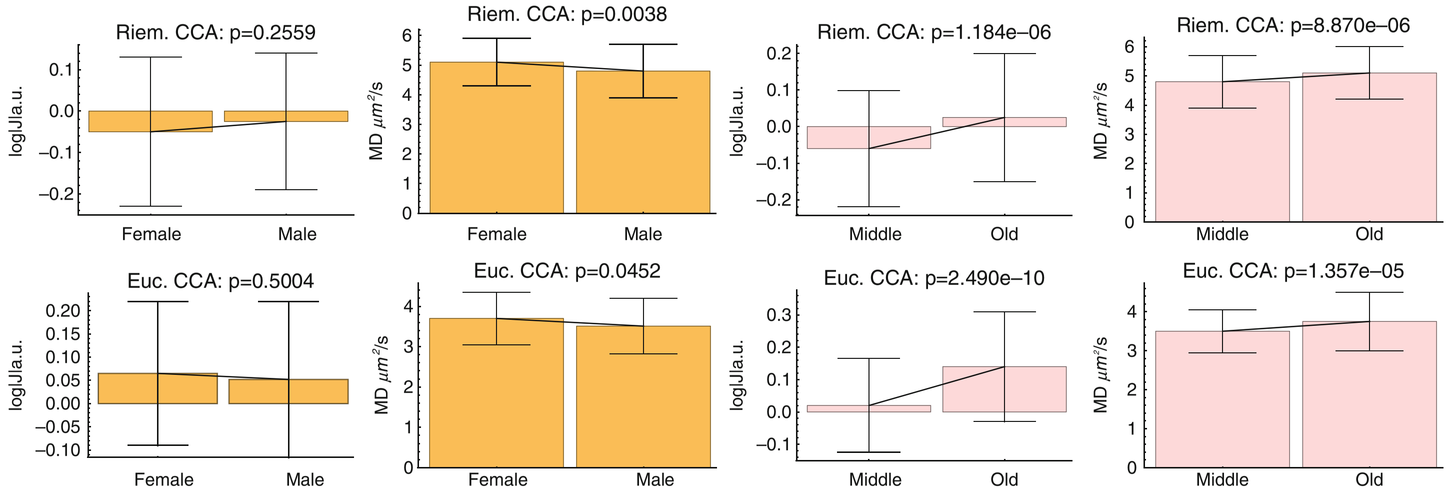


Fig. 4.8 Effect of gender and age group on the average mean diffusivity (MD) and the average log Jacobian determinant. *Top row* shows the results using Riemannian CCA and the *bottom row* using Euclidean CCA with vectorization. Again our approach produces smaller or similar p-values in rejecting the null hypotheses

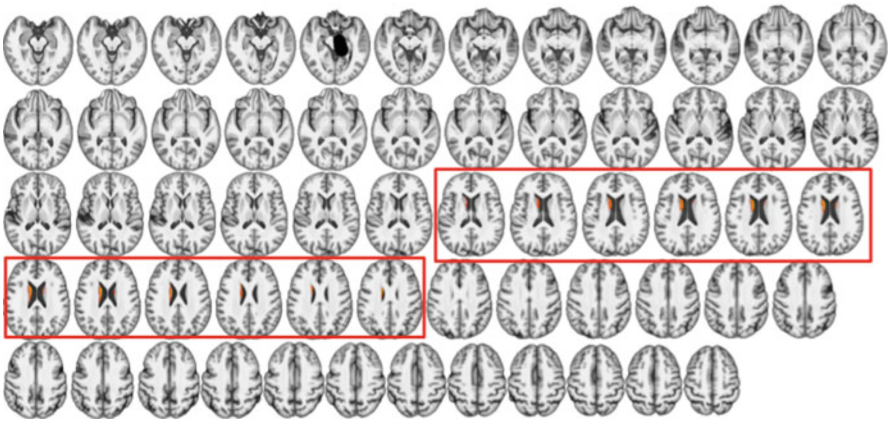


Fig. 4.9 Weight vector (obtained by Riemannian (tensor) CCA) visualization of T1W **axial** slices

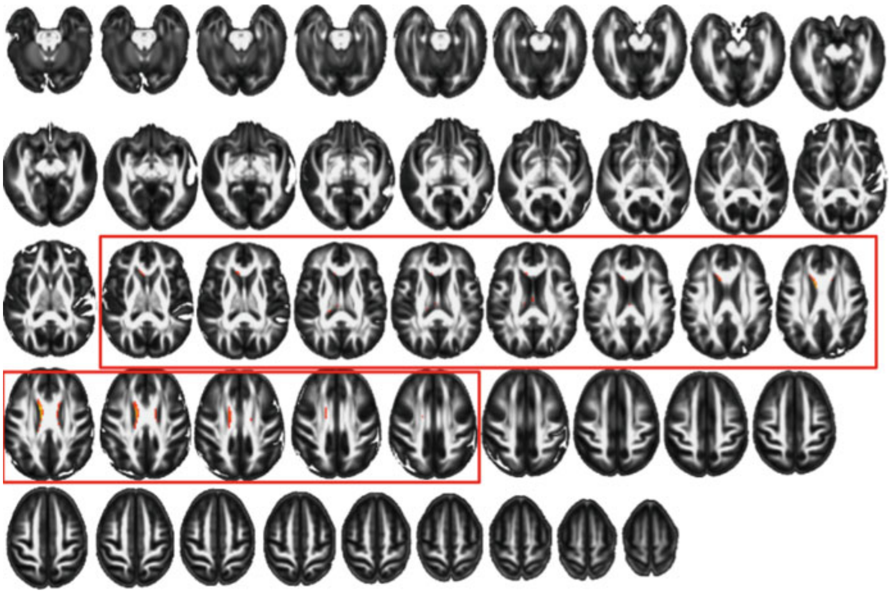


Fig. 4.10 Weight vector (obtained by Riemannian (tensor) CCA) visualization of DTI **axial** slices

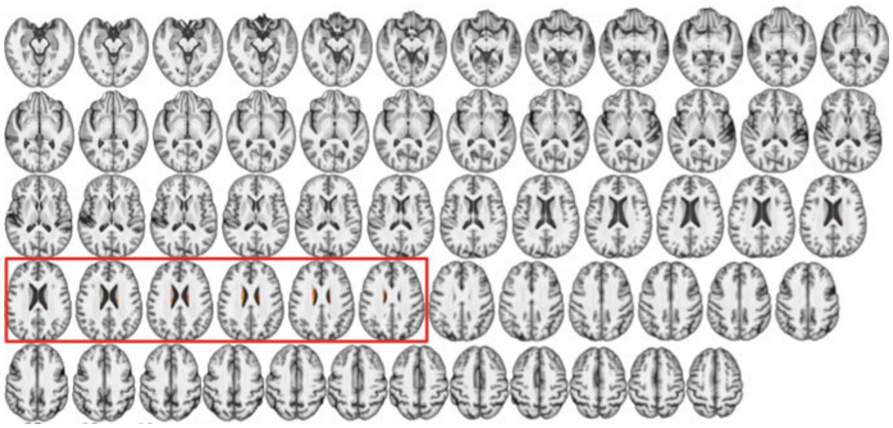


Fig. 4.11 Weight vector [obtained by Euclidean CCA (tensors)] visualization of T1W **axial** slices

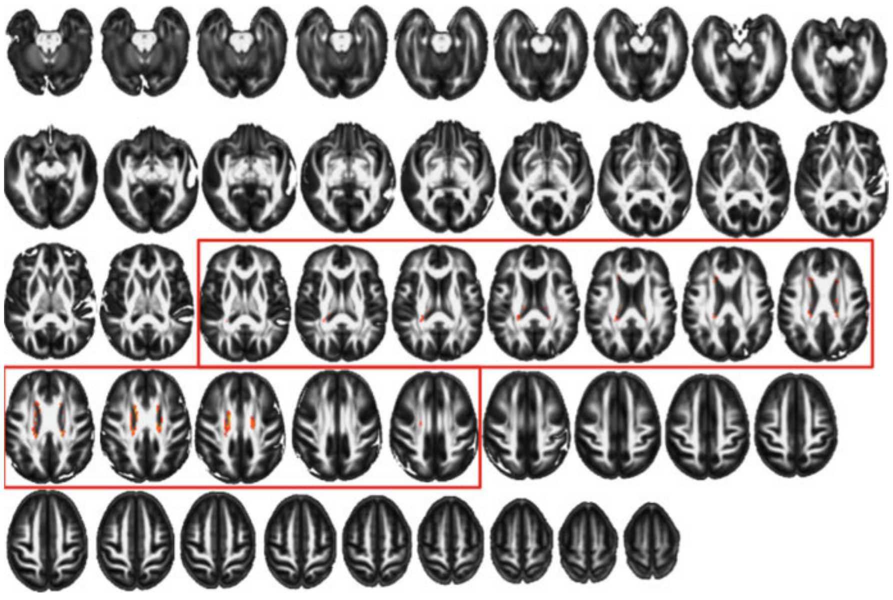


Fig. 4.12 Weight vector [obtained by Euclidean CCA (tensors)] visualization of DTI **axial** slices

References

1. Afsari B (2011) Riemannian l_p center of mass: existence, uniqueness, and convexity. Proc Am Math Soc 139(2):655–673
2. Akaho A (2001) A kernel method for canonical correlation analysis. In: International meeting on psychometric society
3. Andrew G, Arora R, Bilmes J, Livescu K (2013) Deep canonical correlation analysis. In: ICML
4. Avants BB, Gee JC (2004) Geodesic estimation for large deformation anatomical shape and intensity averaging. NeuroImage 23:S139–150
5. Avants BB, Epstein CL, Grossman M, Gee JC (2008) Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. Med Image Anal 12:26–41
6. Avants BB, Cook PA, Ungar L, Gee JC, Grossman M (2010) Dementia induces correlated reductions in white matter integrity and cortical thickness: a multivariate neuroimaging study with SCCA. NeuroImage 50(3):1004–1016
7. Avants BB, Libon DJ, Rascovsky K, Boller A, McMillan CT, Massimo L, Coslett H, Chatterjee A, Gross RG, Grossman M (2014) Sparse canonical correlation analysis relates network-level atrophy to multivariate cognitive measures in a neurodegenerative population. NeuroImage 84(1):698–711
8. Bach FR, Jordan MI (2003) Kernel independent component analysis. J Mach Learn Res 3:1–48
9. Basser PJ, Mattiello J, LeBihan D (1994) MR diffusion tensor spectroscopy and imaging. Biophys J 66(1):259
10. Bhatia R (2009) Positive definite matrices. Princeton University Press, Princeton
11. Callaghan PT (1991) Principles of nuclear magnetic resonance microscopy. Oxford University Press, Oxford
12. Chung MK, Worsley KJ, Paus T, Cherif C, Collins DL, Giedd JN, Rapoport JL, Evans AC (2001) A unified statistical approach to deformation-based morphometry. NeuroImage 14(3):595–606
13. Cook PA, Bai Y, Nedjati-Gilani S, Seunarine KK, Hall MG, Parker GJ, Alexander DC (2006) Camino: open-source diffusion-MRI reconstruction and processing. In: ISMRM, pp 2759
14. Do Carmo MP (1992) Riemannian geometry. Birkhäuser, Boston
15. Ferreira R, Xavier J, Costeira JP, Barroso V (2006) Newton method for Riemannian centroid computation in naturally reductive homogeneous spaces. In: ICASSP
16. Fletcher PT (2013) Geodesic regression and the theory of least squares on riemannian manifolds. Int J Comput Vis 105(2):171–185
17. Fletcher PT, Lu C, Pizer SM, Joshi S (2004) Principal geodesic analysis for the study of nonlinear statistics of shape. Med. Imaging 23(8):995–1005
18. Garrido L, Furl N, Draganski B, Weiskopf N, Stevens J, Chern-Yee Tan G, Driver J, Dolan RJ, Duchaine B (2009) Voxel-based morphometry reveals reduced grey matter volume in the temporal cortex of developmental prosopagnosics. Brain 132(12):3443–3455
19. Goh A, Lenglet C, Thompson PM, Vidal R (2009) A nonparametric Riemannian framework for processing high angular resolution diffusion images (HARDI). In: CVPR, pp 2496–2503
20. Hardoon DR et al (2007) Unsupervised analysis of fMRI data using kernel canonical correlation. NeuroImage 37(4):1250–1259
21. Hardoon DR, Szedmak S, Shawe-Taylor J (2004) CCA: an overview with application to learning methods. Neural Comput 16(12):2639–2664
22. Hinkle J, Fletcher PT, Joshi S (2014) Intrinsic polynomials for regression on Riemannian manifolds. J Math Imaging Vis 50:1–21
23. Ho J, Xie Y, Vemuri B (2013) On a nonlinear generalization of sparse coding and dictionary learning. In: ICML, pp 1480–1488
24. Ho J, Cheng G, Salehian H, Vemuri B (2013) Recursive Karcher expectation estimators and geometric law of large numbers. In: Proceedings of the Sixteenth international conference on artificial intelligence and statistics, pp 325–332

25. Hotelling H (1936) Relations between two sets of variates. Biometrika 28(3/4):321–377
26. Hsieh WW (2000) Nonlinear canonical correlation analysis by neural networks. Neural Netw 13(10):1095–1105
27. Hua X, Leow AD, Parikshak N, Lee S, Chiang MC, Toga AW, Jack CR Jr, Weiner MW, Thompson PM (2008) Tensor-based morphometry as a neuroimaging biomarker for Alzheimer's disease: an MRI study of 676 AD, MCI, and normal subjects. NeuroImage 43(3):458–469
28. Hua X, Gutman B et al (2011) Accurate measurement of brain changes in longitudinal MRI scans using tensor-based morphometry. NeuroImage 57(1):5–14
29. Huang H, He H, Fan X, Zhang J (2010) Super-resolution of human face image using canonical correlation analysis. Pattern Recogn 43(7):2532–2543
30. Huckemann S, Hotz T, Munk A (2010) Intrinsic shape analysis: geodesic PCA for Riemannian manifolds modulo isometric Lie group actions. Stat Sin 20:1–100
31. Jayasumana S, Hartley R, Salzmann M, Li H, Harandi M (2013) Kernel methods on the Riemannian manifold of symmetric positive definite matrices. In: CVPR, pp 73–80
32. Karcher H (1977) Riemannian center of mass and mollifier smoothing. Commun Pure Appl Math 30(5):509–541
33. Kim HJ, Adluru N, Collins MD, Chung MK, Bendlin BB, Johnson SC, Davidson RJ, Singh V (2014) Multivariate general linear models (MGLM) on Riemannian manifolds with applications to statistical analysis of diffusion weighted images. In: CVPR
34. Kim T-K, Cipolla R (2009) CCA of video volume tensors for action categorization and detection. PAMI 31(8):1415–1428
35. Klein A, Andersson J, Ardekani B, Ashburner J, Avants BB, Chiang M, Christensen G, Collins L, Hellier P, Song J, Jenkinson M, Lepage C, Rueckert D, Thompson P, Vercauteren T, Woods R, Mann J, Parsey R (2009) Evaluation of 14 nonlinear deformation algorithms applied to human brain MRI registration. NeuroImage 46:786–802
36. Lai PL, Fyfe C (1999) A neural implementation of canonical correlation analysis. Neural Netw 12(10):1391–1397
37. Lebanon G (2005) Riemannian geometry and statistical machine learning. PhD thesis, Carnegie Mellon University
38. Li P, Wang Q, Zuo W, Zhang L (2013) Log-Euclidean kernels for sparse representation and dictionary learning. In: ICCV, pp 1601–1608
39. Moakher M (2005) A differential geometric approach to the geometric mean of symmetric positive-definite matrices. SIAM J Matrix Anal Appl 26(3):735–747
40. Mostow GD (1973) Strong rigidity of locally symmetric spaces, vol 78. Princeton University Press, Princeton
41. Niethammer M, Huang Y, Vialard F (2011) Geodesic regression for image time-series. In: MICCAI, pp 655–662
42. Nocedal J, Wright SJ (2006) Numerical optimization, 2nd edn. Springer, New York
43. Rapcsák T (1991) Geodesic convexity in nonlinear optimization. J Opt Theory Appl 69(1):169–183
44. Said S, Courty N, Le Bihan N, Sangwine SJ et al (2007) Exact principal geodesic analysis for data on SO(3). In: Proceedings of the 15th European signal processing conference, pp 1700–1705
45. Shi X, Styner M, Lieberman J, Ibrahim JG, Lin W, Zhu H (2009) Intrinsic regression models for manifold-valued data. In: MICCAI, pp 192–199
46. Smith SM, Jenkinson M, Woolrich MW, Beckmann CF, Behrens TE, Johansen-Berg H, Bannister PR, De Luca M, Drobnjak I, Flitney DE, Niazy RK, Saunders J, Vickers J, Zhang Y, De Stefano N, Brady JM, Matthews PM (2004) Advances in functional and structural MR image analysis and implementation as FSL. NeuroImage 23:208–219
47. Sommer S (2013) Horizontal dimensionality reduction and iterated frame bundle development. In: Geometric science of information. Springer, Heidelberg, pp 76–83
48. Sommer S, Lauze F, Nielsen M (2014) Optimization over geodesics for exact principal geodesic analysis. Adv Comput Math 40(2):283–313

49. Steinke F, Hein M, Schölkopf B (2010) Nonparametric regression between general Riemannian manifolds. SIAM J Imaging Sci 3(3):527–563
50. Taniguchi H et al (1984) A note on the differential of the exponential map and jacobi fields in a symmetric space. Tokyo J Math 7(1):177–181
51. Witten DM, Tibshirani R, Hastie T (2009) A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. Biostatistics 10(3):515–534
52. Xie Y, Vemuri BC, Ho J (2010) Statistical analysis of tensor fields. In: MICCAI, pp 682–689
53. Yger F, Berar M, Gasso G, Rakotomamonjy A (2012) Adaptive canonical correlation analysis based on matrix manifolds. In: ICML
54. Yu S, Tan T, Huang K, Jia K, Wu X (2009) A study on gait-based gender classification. IEEE Trans Image Process 18(8):1905–1910
55. Zhang H, Yushkevich PA, Alexander DC, Gee JC (2006) Deformable registration of diffusion tensor MR images with explicit orientation optimization. Med Image Anal 10:764–785
56. Zhang Y, Brady M, Smith S (2001) Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm. IEEE Trans Med Imging 20(1):45–57
57. Zhu H, Chen Y, Ibrahim JG, Li Y, Hall C, Lin W (2009) Intrinsic regression models for positive-definite matrices with applications to diffusion tensor imaging. J Am Stat Assoc 104(487):1203–1212

Chapter 5
Probabilistic Geodesic Models for Regression and Dimensionality Reduction on Riemannian Manifolds

P. Thomas Fletcher and Miaomiao Zhang

Abstract We present a probabilistic formulation for two closely related statistical models for Riemannian manifold data: geodesic regression and principal geodesic analysis. These models generalize linear regression and principal component analysis to the manifold setting. The foundation of the approach is the particular choice of a Riemannian normal distribution law as the likelihood model. Under this distributional assumption, we show that least-squares fitting of geodesic models is equivalent to maximum-likelihood estimation when the manifold is a homogeneous space. We also provide a method for maximum-likelihood estimation of the dispersion of the noise, as well as a novel method for Monte Carlo sampling from the Riemannian normal distribution. We demonstrate the inference procedures on synthetic sphere data, as well as in a shape analysis of the corpus callosum, represented in Kendall's shape space.

5.1 Introduction

Linear models are often the first choice in a statistical analysis of Euclidean data for several reasons. First, the relative simplicity of linear models makes them less prone to problems of overfitting. Second, interpreting linear relationships between variables or linear variability within multivariate data is often easier than interpreting nonlinear alternatives. Third, linear models typically give rise to computationally efficient inference algorithms. When dealing with data that are inherently manifold valued, linear methods are not applicable. However, by using geodesic curves on a manifold as the natural generalization of straight lines, geodesic models also provide simple and interpretable analysis of manifold data.

Much of the literature on manifold statistics focuses on model fitting as a purely geometric optimization problem, rather than as a problem of inferring the parameters of an underlying probability model. Model fitting is typically done

P.T. Fletcher (✉) • M. Zhang
University of Utah, Salt Lake City, UT 84112, USA
e-mail: fletcher@sci.utah.edu; miaomiao@sci.utah.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_5

by minimizing the sum-of-squared geodesic distances from the model to the data. Of course, in traditional multivariate Euclidean statistics, such least-squares fitting is equivalent to maximum-likelihood estimation under a Gaussian likelihood model. The goal of this chapter is to present a similar foundation for probabilistic modeling of manifold data. At the heart of the approach is the definition of a Riemannian normal distribution law. We show how this distribution provides a unifying framework for probabilistic interpretation of several models of manifold data, including the Fréchet mean, geodesic regression, and principal geodesic analysis (PGA).

5.2 Related Work on Manifold Statistics

Possibly the simplest manifold statistic is the *Fréchet mean* of a set of points, $y_1, \ldots, y_N \in M$, which we will denote by μ_{Fr}. It is defined as the minimizer of the sum-of-squared distance function

$$\mu_{\text{Fr}} = \arg\min_{y \in M} \sum_{i=1}^{N} d(y, y_i)^2, \tag{5.1}$$

where d denotes the geodesic distance on M. Fréchet actually introduced a much more general concept of expectation of a probability measure on a metric space [11] of which the sample Fréchet mean on a manifold is a special case. Note that there may be multiple solutions to the above minimization problem. Karcher [18] provided conditions guaranteeing the existence and uniqueness of the Fréchet mean, which were later improved by Kendall [19].

Several works have studied regression models on manifolds, where the goal is to fit a curve on a manifold that models the relationship between a scalar parameter and data on the manifold. This is typically done by a least squares fit, similar to the Fréchet mean definition in (5.1), except now the optimization is over a certain class of curves on the manifold rather than a point. That is, given manifold data $y_1, \ldots, y_N \in M$ with corresponding real data $x_1, \ldots, x_N \in \mathbb{R}$, the regression problem is to find a curve $\hat{\gamma}(x) \in M$ such that

$$\hat{\gamma} = \arg\min_{\gamma \in \Gamma} \sum_{i=1}^{N} d(\gamma(x_i), y_i)^2, \tag{5.2}$$

where Γ is a space of curves on M. In this chapter we will focus on geodesic regression [7, 8], i.e., where Γ is the space of parameterized geodesics on M. Niethammer et al. [22] independently proposed geodesic regression for the case of diffeomorphic transformations of image time series. Hinkle et al. [15] use constant higher-order covariant derivatives to define intrinsic polynomial curves on a Riemannian manifold for regression. Nonparametric kernel regression on

Riemannian manifolds has also been proposed by Davis et al. [3]. Shi et al. [27] proposed a semi-parametric model for manifold response data, which also has the ability to handle multiple covariates.

A closely related problem to the regression problem is that of fitting smoothing splines to manifold data. The typical objective function for smoothing splines is a combination of a data matching term and a regularization term for the spline curve. For example, Su et al. [29] proposed a smoothing spline where the data matching is the same least squares objective as the regression problem (5.2), leading to a smoothing splines optimization of the form

$$\hat{\gamma} = \arg\min_{\gamma \in \Gamma} \sum_{i=1}^{N} d(\gamma(x_i), y_i)^2 + \lambda \mathcal{R}(\gamma), \tag{5.3}$$

where $\mathcal{R}$ is some regularization functional and $\lambda > 0$ is a weighting between regularization and data fitting. In this case, the search space may be the space of all continuous curve segments, $\Gamma = C([0,1], M)$. Jupp and Kent [17] proposed solving the smoothing spline problem on a sphere by unrolling onto the tangent space. This unrolling method was later extended to shape spaces by Kume [20]. Smoothing splines on the group of diffeomorphisms have been proposed as growth models by Miller et al. [21] and as second-order splines by Trouvé et al. [31]. A similar paradigm is used by Durrleman et al. [5] to construct spatiotemporal image atlases from longitudinal data. Yet another related problem is the spline *interpolation* problem, where the data matching term is dropped and the regularization term is optimized subject to constraints that the curve pass through specific points. The pioneering work of Noakes et al. [23] introduced the concept of a cubic spline on a Riemannian manifold for interpolation. Crouch and Leite [2] investigated further variational problems for these cubic splines and for specific classes of manifolds, such as Lie groups and symmetric spaces. Buss and Fillmore [1] defined interpolating splines on the sphere via weighted Fréchet averaging.

Dimensionality reduction is closely related to regression, in that we are seeking a lower-dimensional parameterization of data, with the exception that we are not given the corresponding explanatory variables. PGA [9, 10] generalizes principal component analysis (PCA) to nonlinear manifolds. It describes the geometric variability of manifold data by finding lower-dimensional geodesic subspaces that minimize the residual sum-of-squared geodesic distances to the data. While [9] originally proposed an approximate estimation procedure for PGA, recent contributions [26, 28] have developed algorithms for exact solutions to PGA. Related work on manifold component analysis has introduced variants of PGA. This includes relaxing the constraint that geodesics pass through the mean of the data [13] and, for spherical data, replacing geodesic subspaces with nested spheres of arbitrary radius [16].

All of these methods are based on geometric, least-squares estimation procedures, i.e., they find subspaces that minimize the sum-of-squared geodesic distances to the data. Much like the original formulation of PCA, current component analysis methods on manifolds lack a probabilistic interpretation. The following sections present a unified framework for two recently introduced probabilistic models on manifolds, geodesic regression [8], and probabilistic principal geodesic analysis (PPGA) [32].

5.3 Normal Densities on Manifolds

Throughout, we will let M denote a connected and complete Riemannian manifold. Before defining geodesic models, we first consider a basic definition of a manifold-valued normal distribution and give procedures for maximum-likelihood estimation of its parameters. There is no standard definition of a normal distribution on manifolds, mainly because different properties of the multivariate normal distribution in $\mathbb{R}^n$ may be generalized to manifolds by different definitions. Grenander [12] defines a generalization of the normal distribution to Lie groups and homogeneous spaces as a solution to the heat equation. Pennec [24] defines a generalization of the normal distribution in the tangent space to a mean point via the Riemannian **Log** map. The definition that we use here, introduced in [8], and also used in [15, 32], generalizes the connection between least-squares estimation of statistical models and maximum-likelihood estimation under normally distributed errors.

Consider a random variable y taking values on a Riemannian manifold M, defined by the probability density function (pdf)

$$p(y; \mu, \tau) = \frac{1}{C(\mu, \tau)} \exp\left(-\frac{\tau}{2} d(\mu, y)^2\right), \tag{5.4}$$

$$C(\mu, \tau) = \int_M \exp\left(-\frac{\tau}{2} d(\mu, y)^2\right) dy, \tag{5.5}$$

where $C(\mu, \tau)$ is a normalizing constant. We term this distribution a *Riemannian normal distribution* and use the notation $y \sim N_M(\mu, \tau^{-1})$ to denote it. The parameter $\mu \in M$ acts as a location parameter on the manifold, and the parameter $\tau \in \mathbb{R}_+$ acts as a dispersion parameter, similar to the precision of a Gaussian. This distribution has the advantages that (a) it is applicable to any Riemannian manifold, (b) it reduces to a multivariate normal distribution (with isotropic covariance) when $M = \mathbb{R}^n$, and (c) much like the Euclidean normal distribution, maximum-likelihood estimation of parameters gives rise to least-squares methods when M is a Riemannian homogeneous space, as shown next.

5.3.1 Maximum-Likelihood Estimation of μ

Returning to the Riemannian normal density in (5.4), the maximum-likelihood estimate of the mean parameter, μ, is given by

$$\begin{aligned}\hat{\mu} &= \arg\max_{\mu \in M} \sum_{i=1}^{N} \ln p(y_i; \mu, \tau) \\ &= \arg\min_{\mu \in M} N \ln C(\mu, \tau) + \frac{\tau}{2} \sum_{i=1}^{N} d(\mu, y_i)^2.\end{aligned}$$

This minimization problem clearly reduces to the least-squares estimate, or Fréchet mean in (5.1), if the normalizing constant $C(\mu, \tau)$ does not depend on the μ parameter. As shown in [8], this occurs when the manifold M is a Riemannian homogeneous space, which means that for any two points $x, y \in M$, there exists an isometry that maps x to y. This is because the integral in (5.5) is invariant under isometries. More precisely, given any two points $\mu, \mu' \in M$, there exists an isometry $\phi : M \to M$, with $\mu' = \phi(\mu)$, and we have

$$\begin{aligned}C(\mu, \tau) &= \int_M \exp\left(-\frac{\tau}{2} d(\mu, y)^2\right) dy \\ &= \int_M \exp\left(-\frac{\tau}{2} d(\phi(\mu), \phi(y))^2\right) d\phi(y) \\ &= C(\mu', \tau).\end{aligned}$$

Thus, in the case of a Riemannian homogeneous space, the normalizing constant can be written as

$$C(\tau) = \int_M \exp\left(-\frac{\tau}{2} d(\mu, y)^2\right) dy, \tag{5.6}$$

and we have equivalence of the MLE and Fréchet mean, i.e., $\hat{\mu} = \mu_{\mathrm{Fr}}$.

Two properties of the Riemannian normal distribution are worth emphasizing at this point. First, the requirement that M be a Riemannian homogeneous space is important. Without this, the normalizing constant $C(\mu, \tau)$ may be a function of μ, and if so, the MLE will not coincide with the Fréchet mean. For example, a Riemannian normal distribution on an anisotropic ellipsoid (which is not a homogeneous space) will have a normalizing constant that depends on μ. Second, it is also important that the Riemannian normal density be isotropic, unlike the normal law in [24], which includes a covariance matrix in the tangent space to the mean. Again, a covariance tensor field would need to be a function of the mean point, μ, which would cause the normalizing constant to change with μ, that is, unless the covariant derivative of the covariance field was zero everywhere.

Unfortunately, such tensor fields are not always possible on general homogeneous spaces. For example, the only symmetric, second-order tensor fields with zero covariant derivatives on S^2 are isotropic.

5.3.2 Estimation of the Dispersion Parameter, τ

Maximum-likelihood estimation of the dispersion parameter, τ, can also be done using gradient ascent. Unlike the case for estimation of the μ parameter, now the normalizing constant is a function of τ, and we must evaluate its derivative. We can rewrite the integral in (5.6) in normal coordinates, which can be thought of as a polar coordinate system in the tangent space, $T_\mu M$. The radial coordinate is defined as $r = d(\mu, y)$, and the remaining $n - 1$ coordinates are parameterized by a unit vector v, i.e., a point on the unit sphere $S^{n-1} \subset T_\mu M$. Thus we have the change of variables, $\phi(rv) = \mathbf{Exp}(\mu, rv)$. Now the integral for the normalizing constant becomes

$$C(\tau) = \int_{S^{n-1}} \int_0^{R(v)} \exp\left(-\frac{\tau}{2} r^2\right) |\det(d\phi(rv))| \, dr \, dv, \tag{5.7}$$

where $R(v)$ is the maximum distance that $\phi(rv)$ is defined. Note that this formula is only valid if M is a complete manifold, which guarantees that normal coordinates are defined everywhere except possibly a set of measure zero on M.

The integral in (5.7) is difficult to compute for general manifolds, due to the presence of the determinant of the Jacobian of ϕ. However, for symmetric spaces this change-of-variables term has a simple form. If M is a symmetric space, there exists a orthonormal basis $u_1, \ldots, u_n$, with $u_1 = v$, such that

$$|\det(d\phi(rv))| = \prod_{k=2}^{n} f_k(r), \tag{5.8}$$

where $\kappa_k = K(u_1, u_k)$ denotes the sectional curvature, and f_k is defined as

$$f_k(r) = \begin{cases} \frac{1}{\sqrt{\kappa_k}} \sin(\sqrt{\kappa_k} r) & \text{if } \kappa_k > 0, \\ \frac{1}{\sqrt{-\kappa_k}} \sinh(\sqrt{-\kappa_k} r) & \text{if } \kappa_k < 0, \\ r & \text{if } \kappa_k = 0. \end{cases}$$

Notice that with this expression for the Jacobian determinant there is no longer a dependence on v inside the integral in (5.7). Also, if M is simply connected, then $R(v) = R$ does not depend on the direction v, and we can write the normalizing constant as

$$C(\tau) = A_{n-1} \int_0^R \exp\left(-\frac{\tau}{2} r^2\right) \prod_{k=2}^{n} |\kappa_k|^{-1/2} f_k(\sqrt{|\kappa_k|}r) dr,$$

where A_{n-1} is the surface area of the $n-1$ hypersphere, S^{n-1}. While this formula works only for simply connected symmetric spaces, other symmetric spaces could be handled by lifting to the universal cover, which is simply connected, or by restricting the definition of the Riemannian normal pdf in (5.4) to have support only up to the injectivity radius, i.e., $R = \min_v R(v)$.

The derivative of the normalizing constant with respect to τ is

$$C'(\tau) = A_{n-1} \int_0^R \frac{r^2}{2} \exp\left(-\frac{\tau}{2} r^2\right) \prod_{k=2}^{n} |\kappa_k|^{-1/2} f_k(\sqrt{|\kappa_k|}r) dr. \tag{5.9}$$

Both $C(\tau)$ and $C'(\tau)$ involve only a one-dimensional integral, which can be quickly and accurately approximated by numerical integration. Finally, the derivative of the log-likelihood needed for gradient ascent is given by

$$\frac{d}{d\tau} \sum_{i=1}^{N} \ln p(y_i; \mu, \tau) = -N \frac{C'(\tau)}{C(\tau)} - \frac{1}{2} \sum_{i=1}^{N} d(\mu, y_i)^2.$$

5.3.3 *Sampling from a Riemannian Normal Distribution*

In this section, we describe a Markov Chain Monte Carlo (MCMC) method for sampling from a Riemannian normal distribution with given mean and dispersion parameters, (μ, τ). From (5.7) we see that the Riemannian normal density is proportional to an isotropic Gaussian density in $T_\mu M$ times a change-of-variables term. This suggests using an independence sampler with an isotropic Gaussian as the proposal density.

More specifically, let $y \sim N_M(\mu, \tau^{-1})$, and let $\phi(rv) = \mathbf{Exp}(\mu, rv)$ be normal coordinates in the tangent space $T_\mu M$. Then the density in (r, v) is given by

$$f(r, v) \propto \begin{cases} \exp\left(-\frac{\tau}{2} r^2\right) |\det(d\phi(rv))| & r \leq R(v), \\ 0 & \text{otherwise.} \end{cases}$$

Notice that the density is zero beyond the cut locus. For the independence sampler, we will not need to compute the normalization constant. We will then use an isotropic (Euclidean) Gaussian in $T_\mu M$ as the proposal density, which in polar coordinates is given by

$$g(r, v) \propto r \exp\left(-\frac{\tau}{2} r^2\right).$$

Algorithm 1 Independence sampler for the Riemannian normal distribution

```
Input: Parameters $\mu, \tau$
Draw initial sample $(r, v)$ from $g$
for $i = 1$ to $S$ do
    Sample proposal $(\tilde{r}, \tilde{v})$ from $g$
    Compute the acceptance probability $\alpha((\tilde{r}, \tilde{v}), (r, v))$ using (5.10)
    Draw a uniform random number $u \in [0, 1]$
    if $\tilde{r} \leq R(\tilde{v})$ and $u \leq \alpha((\tilde{r}, \tilde{v}), (r, v))$ then
        Accept: Set $y_i = \mathbf{Exp}(\mu, \tilde{r}\tilde{v})$, and set $(r, v) = (\tilde{r}, \tilde{v})$
    else
        Reject: Set $y_i = \mathbf{Exp}(\mu, rv)$
    end if
end for
```

An iteration of the independence sampler begins with the previous sample (r, v) and generates a proposal sample $(\tilde{r}, \tilde{v})$ from g, which is accepted with probability

$$\begin{aligned} \alpha((\tilde{r}, \tilde{v}), (r, v)) &= \min\left\{1, \frac{f(\tilde{r}, \tilde{v})g(r, v)}{f(r, v)g(\tilde{r}, \tilde{v})}\right\} \\ &= \min\left\{1, \left|\frac{r \det(d\phi(\tilde{r}\tilde{v}))}{\tilde{r} \det(d\phi(rv))}\right|\right\}. \end{aligned} \tag{5.10}$$

So the acceptance probability reduces to simply a ratio of the **Log** map change-of-variables factors, which for symmetric spaces can be computed using (5.8). The final MCMC procedure is given by Algorithm 1.

5.3.4 Sphere Example

We now demonstrate the above procedures for sampling from Riemannian normal densities and ML estimation of parameters on the two-dimensional sphere, S^2. Figure 5.1 shows example samples generated using the independence sampler in Algorithm 1 for various levels of τ. Notice that the sampler is efficient (high acceptance rate) for larger values of τ, but less efficient for smaller τ as the distribution approaches a uniform distribution on the sphere. This is because the proposal density matches the true density well, but the sampler rejects points beyond the cut locus, which happens more frequently when τ is small and the distribution is approaching the uniform distribution on the sphere.

Next, to test the ML estimation procedures, we used the independence sampler to repeatedly generate $N = 100$ random points on S^2 from a $N_{S^2}(\mu, \tau)$ density, where $\mu = (0, 0, 1)$ was the north pole, and again we varied $\tau = 1, 20, 50$. Then we computed the MLEs, $\hat{\mu}, \hat{\tau}$, using the gradient ascent procedures above. Each experiment was repeated 1000 times, and the results are summarized in Fig. 5.2. For the $\hat{\mu}$ estimates, we plot a kernel density estimate of the points $\mathbf{Log}_{\mu}\hat{\mu}$. This is a Monte Carlo simulation of the sampling distribution of the $\hat{\mu}$ statistic, mapped

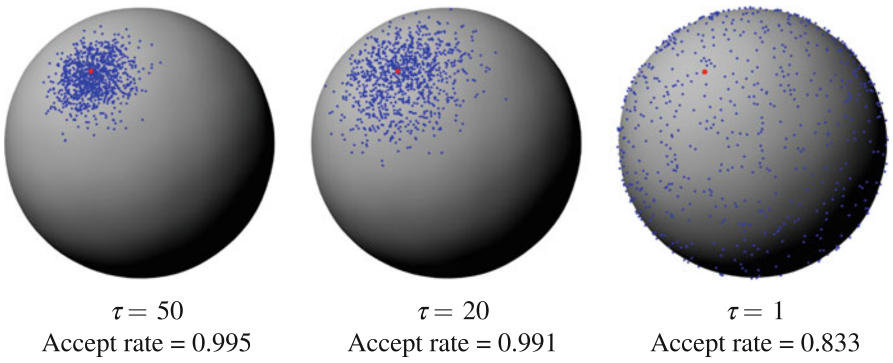


Fig. 5.1 Samples from a Riemannian normal density on S^2 for various levels of τ. Samples are in *blue*, and the mean parameter, μ, is shown in *red*

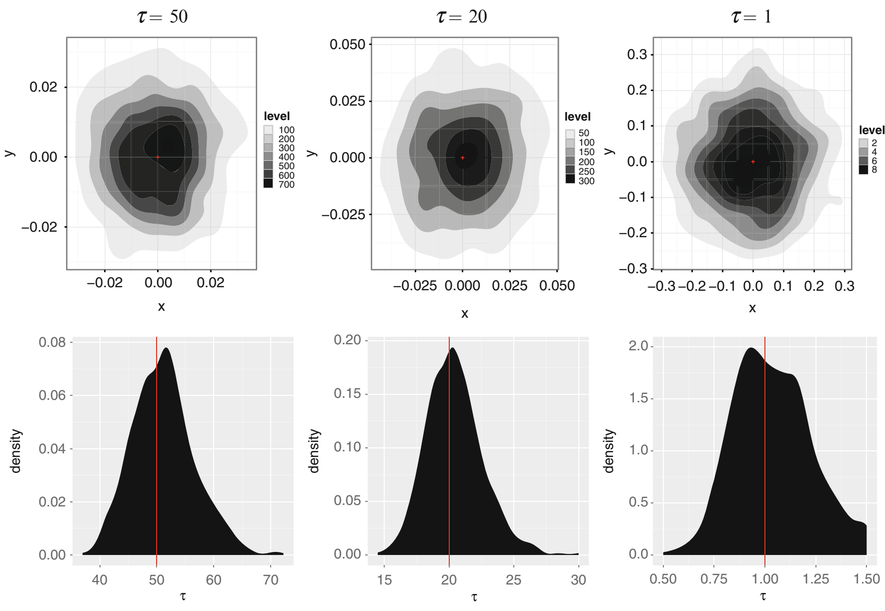


Fig. 5.2 Monte Carlo simulation of the MLEs, $\hat{\mu}$ (*top row*), and $\hat{\tau}$ (*bottom row*). The true parameter values are marked in *red*

into the tangent space of the true mean, $T_\mu M$, via the **Log** map. Similarly, the corresponding empirical sampling distribution of the $\hat{\tau}$ statistics are plotted as kernel density estimates. While the true sampling distributions are unknown, the plots demonstrate that the MLEs have reasonable behavior, i.e., they are distributed about the true parameter values, and their variance decreases as τ increases.

5.4 Probabilistic Geodesic Regression

Let $y_1, \ldots, y_N$ be points on a smooth Riemannian manifold M, with associated scalar values $x_1, \ldots, x_N \in \mathbb{R}$. The goal of geodesic regression is to find a geodesic curve γ on M that best models the relationship between the x_i and the y_i, as shown in Fig. 5.3. Just as in linear regression, the speed of the geodesic will be proportional to the independent parameter corresponding to the x_i. The tangent bundle, TM, is the space of all possible tangent vectors to points in M. Because a geodesic on a complete manifold is uniquely defined by its initial point and velocity, the tangent bundle serves as a convenient parameterization of the set of possible geodesics on M. An element $(p, v) \in TM$ represents an intercept, or initial point, $\gamma(0) = p$, and a slope, or initial velocity, $\gamma'(0) = v$. Such a geodesic may also be written in terms of the Riemannian **Exp** map: $\gamma(x) = \mathbf{Exp}(p, xv)$. Given this setup, we are ready to define the probabilistic model for geodesic regression as

$$y \sim N_M(\mathbf{Exp}(p, xv), \tau^{-1}). \tag{5.11}$$

As is customary in regression models, y is a random variable, while the x values are observations that we do not model as random. We now have model parameters, $(p, v) \in TM$ and $\tau \in \mathbb{R}_+$, which we want to estimate. Notice that for Euclidean space, the exponential map is simply addition, that is, $\mathbf{Exp}(p, v) = p + v$. Thus, when $M = \mathbb{R}^n$, the geodesic model reduces to multiple linear regression with isotropic Gaussian errors, and ML inference is the standard ordinary least squares.

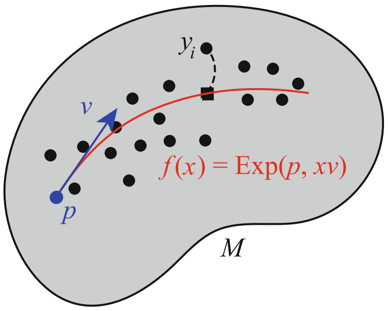


Fig. 5.3 Schematic of the geodesic regression model

5.4.1 Maximum-Likelihood Estimation

Given data $(x_i, y_i) \in \mathbb{R} \times M$, for $i = 1, \ldots, N$, the joint log-likelihood for the geodesic regression model, using (5.4) and (5.11), is given by

$$\log \prod_{i=1}^{N} p(y; x, (p, v), \tau) = -\sum_{i=1}^{N} \left\{ \log C(\gamma(x_i), \tau) + \frac{\tau}{2} d(\gamma(x_i), y_i)^2 \right\}, \tag{5.12}$$
$$\gamma(x_i) = \mathbf{Exp}(p, x_i v).$$

Again, for general manifolds, the normalization constant may be a function of the point $\gamma(x_i)$, and thus a function of (p, v). However, if we restrict our attention to homogeneous spaces, the same argument used before in Sect. 5.3 applies to show that the normalizing constant is independent of the point $\gamma(x_i)$. Thus, for homogeneous spaces, ML estimation is equivalent to the minimization problem

$$((\hat{p}, \hat{v}), \hat{\tau}) = \arg \min_{\substack{(p,v) \in TM \\ \tau \in \mathbb{R}_+}} E((p, v), \tau),$$

$$E((p, v), \tau) = N \log C(\tau) + \frac{\tau}{2} \sum_{i=1}^{N} d(\mathbf{Exp}(p, x_i v), y_i)^2.$$

As before, we solve this problem using gradient descent.

Gradient for (p, v)

The gradient for the intercept and slope parameters require that we take the derivative of the exponential map, $\mathbf{Exp}(p, x_i v)$, with respect to the initial point p and velocity v. To do this, first consider a variation of geodesics given by $c_1(s, t) = \mathbf{Exp}(\mathbf{Exp}(p, su_1), tv(s))$, where $u_1 \in T_pM$ defines a variation of the initial point along the geodesic $\eta(s) = \mathbf{Exp}(p, su_1)$. Here we have also extended $v \in T_pM$ to a vector field $v(s)$ along η via parallel translation. Next consider a variation of geodesics $c_2(s, t) = \mathbf{Exp}(p, su_2 + tv)$, where $u_2 \in T_pM$. (Technically, u_2 is a tangent to the tangent space, i.e., an element of $T_v(T_pM)$, but there is a natural isomorphism $T_v(T_pM) \cong T_pM$.) Taking the derivatives with respect to s at $s = 0$, these variations lead to Jacobi fields along the geodesic γ, giving the derivatives of the $\mathbf{Exp}$ map:

$$d_p\mathbf{Exp}(p, x_i v) \cdot u_1 = \frac{d}{ds} c_1(s, x_i)\Big|_{s=0}$$
$$d_v\mathbf{Exp}(p, x_i v) \cdot u_2 = \frac{d}{ds} c_2(s, x_i)\Big|_{s=0}.$$

Finally, the gradients of the negative log-likelihood function are given by

$$\nabla_p E = -\sum_{i=1}^{N} d_p \mathbf{Exp}(p, x_i v)^\dagger \epsilon_i, \tag{5.13}$$

$$\nabla_v E = -\sum_{i=1}^{N} x_i \, d_v \mathbf{Exp}(p, x_i v)^\dagger \epsilon_i, \tag{5.14}$$

where we have defined $\epsilon_i = \mathbf{Log}(\mathbf{Exp}(p, x_i v), y_i)$, and we have taken the adjoint of the **Exp** derivative, defined by

$$\langle d_p \mathbf{Exp}(p, v) u, w \rangle = \langle u, d_p \mathbf{Exp}(p, v)^\dagger w \rangle.$$

Formulas for Jacobi fields and their respective adjoint operators can often be derived analytically for many useful manifolds. We will give examples of the computations for spheres and Kendall's shape space in Sect. 5.6.

Gradient for τ

The gradient for τ is similar to the normal distribution case. Plugging in the ML estimate for the geodesic parameters, $(\hat{p}, \hat{v})$, we have

$$\nabla_\tau E = N \frac{C'(\tau)}{C(\tau)} + \frac{1}{2} \sum_{i=1}^{N} d(\mathbf{Exp}(\hat{p}, x_i \hat{v}), y_i)^2.$$

5.5 Probabilistic Principal Geodesic Analysis

PCA [14] has been widely used to analyze highdimensional Euclidean data. Tipping and Bishop proposed probabilistic PCA (PPCA) [30], which is a latent variable model for PCA. A similar formulation was independently proposed by Roweis [25]. The main idea of PPCA is to model an n-dimensional Euclidean random variable y as

$$y = \mu + Bx + \epsilon, \tag{5.15}$$

where μ is the mean of y, x is a q-dimensional latent variable, with $x \sim N(0, I)$, B is an $n \times q$ factor matrix that relates x and y, and $\epsilon \sim N(0, \sigma^2 I)$ represents error. We will find it convenient to model the factors as $B = W\Lambda$, where the columns of W are mutually orthogonal, and Λ is a diagonal matrix of scale factors. This removes the rotation ambiguity of the latent factors and makes them analogous to the eigenvectors and eigenvalues of standard PCA (there is still of course an ambiguity of the ordering of the factors). We now generalize this model to random variables on Riemannian manifolds.

5.5.1 Probability Model

The PPGA model for a random variable y on a smooth Riemannian manifold M is

$$y|x \sim N_M\left(\mathbf{Exp}(\mu, z), \tau^{-1}\right), z = W\Lambda x, \tag{5.16}$$

where $x \sim N(0, 1)$ are again latent random variables in $\mathbb{R}^q$, μ here is a base point on M, W is a matrix with q columns of mutually orthogonal tangent vectors in $T_\mu M$, Λ is a $q \times q$ diagonal matrix of scale factors for the columns of W, and τ is a scale parameter for the noise. In this model, a linear combination of $W\Lambda$ and the latent variables x forms a new tangent vector $z \in T_\mu M$. Next, the exponential map shoots the base point μ by z to generate the location parameter of a *Riemannian normal distribution*, from which the data point y is drawn. Note that in Euclidean space, the exponential map is an addition operation, $\mathbf{Exp}(\mu, z) = \mu + z$. Thus, our model coincides with (5.15), the standard PPCA model, when $M = \mathbb{R}^n$.

5.5.2 Inference

We develop a maximum likelihood procedure to estimate the parameters $\theta = (\mu, W, \Lambda, \tau)$ of the PPGA model defined in (5.16). Given observed data $y_i \in \{y_1, \ldots, y_N\}$ on M, with associated latent variable $x_i \in \mathbb{R}^q$, and $z_i = W\Lambda x_i$, we formulate an expectation maximization (EM) algorithm. Since the expectation step over the latent variables does not yield a closed-form solution, we develop an HMC method to sample x_i from the posterior $p(x|y; \theta)$, the log of which is given by

$$\log \prod_{i=1}^{N} p(x_i|y_i; \theta) \propto -N \log C - \sum_{i=1}^{N} \frac{\tau}{2} d\left(\mathbf{Exp}(\mu, z_i), y_i\right)^2 - \frac{\|x_i\|^2}{2}, \tag{5.17}$$

and use this in a Monte Carlo expectation maximization (MCEM) scheme to estimate θ. The procedure contains two main steps:

E-Step: HMC

For each x_i, we draw a sample of size S from the posterior distribution (5.17) using HMC with the current estimated parameters θ^k. Denoting x_{ij} as the jth sample for x_i, the Monte Carlo approximation of the Q function is given by

$$Q(\theta|\theta^k) = E_{x_i|y_i;\theta^k}\left[\prod_{i=1}^{N} \log p(x_i|y_i;\theta^k)\right] \approx \frac{1}{S}\sum_{j=1}^{S}\sum_{i=1}^{N} \log p(x_{ij}|y_i;\theta^k). \tag{5.18}$$

Hamiltonian Monte Carlo (HMC) [4] is a powerful gradient-based Markov Chain Monte Carlo sampling method that is applicable to a wide array of continuous probability distributions. It rigorously explores the entire space of a target distribution by utilizing Hamiltonian dynamics as a Markov transition probability. The gradient information of the log probability density is used to efficiently sample from the higher probability regions.

Next, we derive an HMC procedure to draw a random sample from the posterior distribution of the latent variables x. The first step to sample from a distribution $f(x)$ using HMC is to construct a Hamiltonian system $H(x, m) = U(x) + V(m)$, where $U(x) = -\log f(x)$ is a "potential energy", and $V(m) = -\log g(m)$ is a "kinetic energy", which acts as a proposal distribution on an auxiliary momentum variable, m. An initial random momentum m is drawn from the density $g(m)$. Starting from the current point x and initial random momentum m, the Hamiltonian system is integrated forward in time to produce a candidate point, x^*, along with the corresponding forward-integrated momentum, m^*. The candidate point x^* is accepted as a new point in the sample with probability

$$P(\text{accept}) = \min(1, \exp(-U(x^*) - V(m^*) + U(x) + V(m)).$$

This acceptance–rejection method is guaranteed to converge to the desired density $f(x)$ under fairly general regularity assumptions on f and g.

In our HMC sampling procedure, the potential energy of the Hamiltonian $H(x_i, m) = U(x_i) + V(m)$ is defined as $U(x_i) = -\log p(x_i|y_i;\theta)$, and the kinetic energy $V(m)$ is a typical isotropic Gaussian distribution on a q-dimensional auxiliary momentum variable, m. This gives us a Hamiltonian system to integrate: $\frac{dx_i}{dt} = \frac{\partial H}{\partial m} = m$ and $\frac{dm}{dt} = -\frac{\partial H}{\partial x_i} = -\nabla_{x_i} U$. Due to the fact that x_i is a Euclidean variable, we use a standard "leap-frog" numerical integration scheme, which approximately conserves the Hamiltonian and results in high acceptance rates. Now, the gradient with respect to each x_i is

$$\nabla_{x_i} U = x_i - \tau \Lambda W^T \{d_{z_i}\mathbf{Exp}(\mu, z_i)^\dagger \mathbf{Log}(\mathbf{Exp}(\mu, z_i), y_i)\}. \tag{5.19}$$

M-Step: Gradient Ascent

In this section, we derive the maximization step for updating the parameters $\theta = (\mu, W, \Lambda, \tau)$ by maximizing the HMC approximation of the Q function in (5.18). This turns out to be a gradient ascent scheme for all the parameters since there are no closed-form solutions.

Gradient for τ

The gradient term for estimating τ is

$$\nabla_\tau Q = -N\frac{C'(\tau)}{C(\tau)} - \frac{1}{S}\sum_{i=1}^{N}\sum_{j=1}^{S} d(\mathbf{Exp}(\mu, z_{ij}), y_i)^2,$$

where the derivative $C'(\tau)$ is given in (5.9).

Gradient for μ

From (5.17) and (5.18), the gradient term for updating μ is

$$\nabla_\mu Q = \frac{1}{S}\sum_{i=1}^{N}\sum_{j=1}^{S} \tau d_\mu \mathbf{Exp}(\mu, z_{ij})^\dagger \mathbf{Log}\left(\mathbf{Exp}(\mu, z_{ij}), y_i\right).$$

Gradient for Λ

For updating Λ, we take the derivative w.r.t. each ath diagonal element Λ^a as

$$\frac{\partial Q}{\partial \Lambda^a} = \frac{1}{S}\sum_{i=1}^{N}\sum_{j=1}^{S} \tau (W^a x_{ij}^a)^T \{d_{z_{ij}} \mathbf{Exp}(\mu, z_{ij})^\dagger \mathbf{Log}(\mathbf{Exp}(\mu, z_{ij}), y_i)\},$$

where W^a denotes the ath column of W, and x_{ij}^a is the ath component of x_{ij}.

Gradient for W

The gradient w.r.t. W is

$$\nabla_W Q = \frac{1}{S}\sum_{i=1}^{N}\sum_{j=1}^{S} \tau d_{z_{ij}} \mathbf{Exp}(\mu, z_{ij})^\dagger \mathbf{Log}(\mathbf{Exp}(\mu, z_{ij}), y_i) x_{ij}^T \Lambda. \tag{5.20}$$

Algorithm 2 Monte Carlo expectation maximization for probabilistic principal geodesic analysis

```
Input: Data set $Y$, reduced dimension $q$.
Initialize $\mu, W, \Lambda, \sigma$.
repeat
    Sample $X$ according to (5.19),
    Update $\mu, W, \Lambda, \sigma$ by gradient ascent in Section 3.2.2.
until convergence
```

To preserve the mutual orthogonality constraint on the columns of W, we project the gradient in (5.20) onto the tangent space at W and then update W by shooting the geodesic on the Stiefel manifold in the negative projected gradient direction, see the detail in [6].

The MCEM algorithm for PPGA is an iterative procedure for finding the subspace spanned by q principal components, shown in Algorithm 2. The computation time per iteration depends on the complexity of exponential map, log map, and Jacobi field which may vary for different manifold. Note the cost of the gradient ascent algorithm also linearly depends on the data size, dimensionality, and the number of samples drawn. An advantage of MCEM is that it can run in parallel for each data point. Since the posterior distribution (5.17) is estimated by HMC sampling, to diagnose the convergence of our algorithm, we run parallel independent chains to obtain univariate quantities of the full distribution.

5.6 Experiments

In this section, we demonstrate the effectiveness of PPGA and our ML estimation using both simulated data on the 2D sphere and a real corpus callosum data set. Before presenting the experiments of PPGA, we briefly review the necessary computations for the specific types of manifolds used, including the Riemannian exponential map, log map, and Jacobi fields.

5.6.1 Simulated Sphere Data

Sphere Geometry Overview

Let p be a point on an n-dimensional sphere embedded in $\mathbb{R}^{n+1}$, and let v be a tangent at p. The inner product between tangents at a base point p is the usual Euclidean inner product. The exponential map is given by a 2D rotation of p by an angle given by the norm of the tangent, i.e.,

$$\mathbf{Exp}(p, v) = \cos\theta \cdot p + \frac{\sin\theta}{\theta} \cdot v, \quad \theta = \|v\|. \tag{5.21}$$

The log map between two points p, q on the sphere can be computed by finding the initial velocity of the rotation between the two points. Let $\pi_p(q) = p \cdot \langle p, q \rangle$ denote the projection of the vector q onto p. Then

$$\mathbf{Log}(p, q) = \frac{\theta \cdot \left(q - \pi_p(q)\right)}{\|q - \pi_p(q)\|}, \quad \theta = \arccos(\langle p, q \rangle). \tag{5.22}$$

All sectional curvatures for S^n are equal to one. The adjoint derivatives of the exponential map are given by

$$d_p\mathbf{Exp}(p, v)^\dagger w = \cos(\|v\|)w^\perp + w^\top, \quad d_v\mathbf{Exp}(p, v)^\dagger w = \frac{\sin(\|v\|)}{\|v\|}w^\perp + w^\top,$$

where $w^\perp, w^\top$ denote the components of w that are orthogonal and tangent to v, respectively. An illustration of geodesics and the Jacobi fields that give rise to the exponential map derivatives is shown in Fig. 5.4.

Parameter Estimation on the Sphere

Using our generative model for PGA (5.16), we forward simulated a random sample of 100 data points on the unit sphere S^2, with known parameters $\theta = (\mu, W, \Lambda, \tau)$, shown in Table 5.1. Next, we ran our maximum likelihood estimation procedure to test whether we could recover those parameters. We initialized μ from a random uniform point on the sphere. We initialized W as a random Gaussian matrix, to which we then applied the Gram–Schmidt algorithm to ensure its columns were orthonormal. Figure 5.4 compares the ground truth principal geodesics and MLE principal geodesic analysis using our algorithm. A good overlap between the first principal geodesic shows that PPGA recovers the model parameters.

One advantage that our PPGA model has over the least-squares PGA formulation is that the mean point is estimated jointly with the principal geodesics. In the standard PGA algorithm, the mean is estimated first (using geodesic least squares),

Table 5.1 Comparison between ground truth parameters for the simulated data and the MLE of PPGA, non-probabilistic PGA, and standard PCA

	μ	w	Λ	τ
Ground truth	$(-0.78, 0.48, -0.37)$	$(-0.59, -0.42, 0.68)$	0.40	100
PPGA	$(-0.78, 0.48, -0.40)$	$(-0.59, -0.43, 0.69)$	0.41	102
PGA	$(-0.79, 0.46, -0.41)$	$(-0.59, -0.38, 0.70)$	0.41	N/A
PCA	$(-0.70, 0.41, -0.46)$	$(-0.62, -0.37, 0.69)$	0.38	N/A

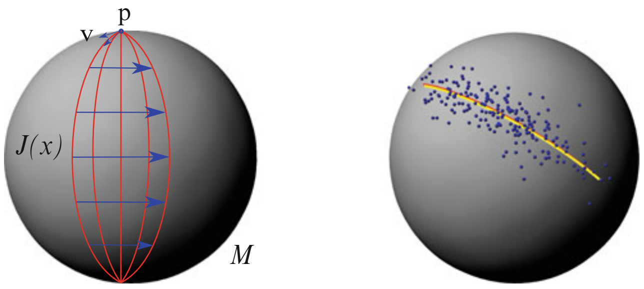


Fig. 5.4 *Left*: Jacobi fields; *Right*: the principal geodesic of random generated data on unit sphere. *Blue dots*: random generated sphere data set. *Yellow line*: ground truth principal geodesic. *Red line*: estimated principal geodesic using PPGA

then the principal geodesics are estimated second. This does not make a difference in the Euclidean case (principal components must pass through the mean), but it does in the nonlinear case. To demonstrate this, we give examples where data can be fit better when jointly estimating mean and PGA than when doing them sequentially. We compared our model with PGA and standard PCA (in the Euclidean embedding space). The noise variance τ was not valid to be estimated in both PGA and PCA. The estimation error of principal geodesics turned to be larger in PGA compared to our model. Furthermore, the standard PCA converges to an incorrect solution due to its inappropriate use of a Euclidean metric on Riemannian data. A comparison of the ground truth parameters and these methods is given in Table 5.1.

5.6.2 *Shape Analysis of the Corpus Callosum*

Shape Space Geometry

A configuration of k points in the 2D plane is considered as a complex k-vector, $z \in \mathbb{C}^k$. Removing translation, by requiring the centroid to be zero, projects this point to the linear complex subspace $V = \{z \in \mathbb{C}^k : \sum z_i = 0\}$, which is equivalent to the space $\mathbb{C}^{k-1}$. Next, points in this subspace are deemed equivalent if they are a rotation and scaling of each other, which can be represented as multiplication by a complex number, $\rho e^{i\theta}$, where ρ is the scaling factor and θ is the rotation angle. The set of such equivalence classes forms the complex projective space, $\mathbb{C}P^{k-2}$.

We think of a centered shape $p \in V$ as representing the complex line $L_p = \{z \cdot p : z \in \mathbb{C} \backslash \{0\}\}$, i.e., L_p consists of all point configurations with the same shape as p. A tangent vector at $L_p \in V$ is a complex vector, $v \in V$, such that $\langle p, v \rangle = 0$. The exponential map is given by rotating (within V) the complex line L_p by the initial velocity v, that is,

$$\mathbf{Exp}(p, v) = \cos\theta \cdot p + \frac{\|p\| \sin\theta}{\theta} \cdot v, \quad \theta = \|v\|. \tag{5.23}$$

Likewise, the log map between two shapes $p, q \in V$ is given by finding the initial velocity of the rotation between the two complex lines L_p and L_q. Let $\pi_p(q) = p \cdot \langle p, q\rangle / \|p\|^2$ denote the projection of the vector q onto p. Then the log map is given by

$$\mathbf{Log}(p, q) = \frac{\theta \cdot \big(q - \pi_p(q)\big)}{\|q - \pi_p(q)\|}, \quad \theta = \arccos \frac{|\langle p, q\rangle|}{\|p\| \, \|q\|}. \tag{5.24}$$

The sectional curvatures of $\mathbb{C}P^{k-2}$, $\kappa_i = K(u_i, v)$, used in (5.8), can be computed as follows. Let $u_1 = i \cdot v$, where we treat v as a complex vector and $i = \sqrt{-1}$. The remaining $u_2, \ldots, u_n$ can be chosen arbitrarily to construct an orthonormal frame with v and u_1. Then we have $K(u_1, v) = 4$ and $K(u_i, v) = 1$ for $i > 1$. The adjoint derivatives of the exponential map are given by

$$d_p\mathbf{Exp}(p, v)^\dagger w = \cos(\|v\|) w_1^\perp + \cos(2\|v\|) w_2^\perp + w^\top,$$

$$d_v\mathbf{Exp}(p, v)^\dagger w = \frac{\sin(\|v\|)}{\|v\|} w_1^\perp + \frac{\sin(2\|v\|)}{2\|v\|} + w_2^\top,$$

where $w_1^\perp$ denotes the component of w parallel to u_1, i.e., $w_1^\perp = \langle w, u_1\rangle u_1$, $u_2^\top$ denotes the remaining orthogonal component of w, and $w^\top$ denotes the component tangent to v.

Shape Variability of Corpus Callosum Data

As a demonstration of PPGA on Kendall shape space, we applied it to corpus callosum shape data derived from the OASIS database (www.oasis-brains.org). The data consisted of magnetic resonance images (MRI) from 32 healthy adult subjects. The corpus callosum was segmented in a midsagittal slice using the ITK SNAP program (www.itksnap.org). An example of a segmented corpus callosum in an MRI is shown in Fig. 5.5. The boundaries of these segmentations were sampled with 64 points using ShapeWorks (www.sci.utah.edu/software.html). This algorithm generates a sampling of a set of shape boundaries while enforcing correspondences between different point models within the population. Figure 5.5 displays the first two modes of corpus callosum shape variation, generated from the as points along the estimated principal geodesics: $\mathbf{Exp}(\mu, \alpha_i w_i)$, where $\alpha_i = -3\lambda_i, -1.5\lambda_i, 0, 1.5\lambda_i, 3\lambda_i$, for $i = 1, 2$.

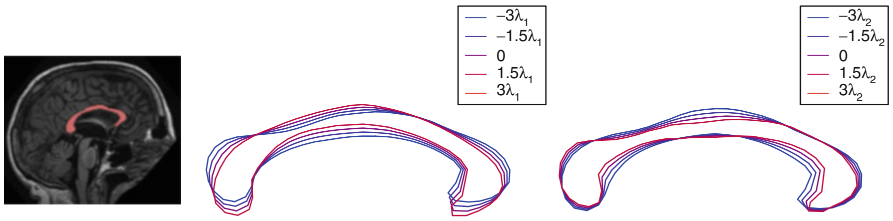


Fig. 5.5 *Left*: example corpus callosum segmentation from an MRI slice. *Middle to right*: first and second PPGA mode of shape variation with -3, -1.5, 1.5, and $3 \times \lambda$

Acknowledgements This work was supported in part by NSF CAREER Grant 1054057.

References

1. Buss SR, Fillmore JP (2001) Spherical averages and applications to spherical splines and interpolation. ACM Trans Graph 20(2):95–126
2. Crouch P, Leite FS (1995) The dynamic interpolation problem: on Riemannian manifolds, Lie groups, and symmetric spaces. J Dyn Control Syst 1(2):177–202
3. Davis B, Fletcher PT, Bullitt E, Joshi S (2007) Population shape regression from random design data. In: Proceedings of IEEE international conference on computer vision
4. Duane S, Kennedy A, Pendleton B, Roweth D (1987) Hybrid Monte Carlo. Phys Lett B 195:216–222
5. Durrleman S, Pennec X, Trouvé A, Gerig G, Ayache N (2009) Spatiotemporal atlas estimation for developmental delay detection in longitudinal datasets. In: Medical image computing and computer-assisted intervention, pp 297–304
6. Edelman A, Arias TA, Smith ST (1998) The geometry of algorithms with orthogonality constraints. SIAM J Matrix Anal Appl 20(2):303–353
7. Fletcher PT (2011) Geodesic regression on Riemannian manifolds. In: MICCAI workshop on mathematical foundations of computational anatomy, pp 75–86
8. Fletcher PT (2012) Geodesic regression and the theory of least squares on Riemannian manifolds. Int J Comput Vis 105:1–15
9. Fletcher PT, Lu C, Joshi S (2003) Statistics of shape via principal geodesic analysis on Lie groups. In: Computer vision and pattern recognition, pp 95–101
10. Fletcher PT, Lu C, Pizer SM, Joshi S (2004) Principal geodesic analysis for the study of nonlinear statistics of shape. IEEE Trans Med Imaging 23(8):995–1005
11. Fréchet M (1948) Les éléments aléatoires de nature quelconque dans un espace distancié. Ann Inst Henri Poincaré 10(3):215–310
12. Grenander U (1963) Probabilities on algebraic structures. Wiley, New York
13. Huckemann S, Ziezold H (2006) Principal component analysis for Riemannian manifolds, with an application to triangular shape spaces. Adv Appl Probab 38(2):299–319
14. Jolliffe IT (1986) Principal component analysis, vol 487. Springer, New York
15. Joshi SC, Hinkle J, Fletcher PT (2014) Intrinsic polynomials for regression on riemannian manifolds. J. Math. Imaging Vision 50(1–2):32–52
16. Jung S, Dryden IL, Marron JS (2012) Analysis of principal nested spheres. Biometrika 99(3):551–568
17. Jupp PE, Kent JT (1987) Fitting smooth paths to spherical data. Appl Stat 36(1):34–46

18. Karcher H (1977) Riemannian center of mass and mollifier smoothing. Commun Pure Appl Math 30:509–541
19. Kendall WS (1990) Probability, convexity, and harmonic maps with small image I: uniqueness and fine existence. Proc Lond Math Soc 3(61):371–406
20. Kume A, Dryden IL, Le H (2007) Shape-space smoothing splines for planar landmark data. Biometrika 94(3):513–528
21. Miller M (2004) Computational anatomy: shape, growth, and atrophy comparison via diffeomorphisms. NeuroImage 23:S19–S33
22. Niethammer M, Huang Y, Viallard F-X (2011) Geodesic regression for image time-series. In: Proceedings of medical image computing and computer assisted intervention
23. Noakes L, Heinzinger G, Paden B (1989) Cubic splines on curved spaces. IMA J Math Control Inf 6(4):465–473
24. Pennec X (2006) Intrinsic statistics on Riemannian manifolds: basic tools for geometric measurements. J Math Imaging Vision 25(1):127–154
25. Roweis S (1998) EM algorithms for PCA and SPCA. In: Advances in neural information processing systems, pp 626–632
26. Said S, Courty N, Le Bihan N, Sangwine SJ (2007) Exact principal geodesic analysis for data on SO(3). In: Proceedings of the 15th European signal processing conference, pp 1700–1705
27. Shi X, Styner M, Lieberman J, Ibrahim J, Lin W, Zhu H (2009) Intrinsic regression models for manifold-valued data. J Am Stat Assoc 5762:192–199
28. Sommer S, Lauze F, Hauberg S, Nielsen M (2010) Manifold valued statistics, exact principal geodesic analysis and the effect of linear approximations. In: Proceedings of the European conference on computer vision, pp 43–56
29. Su J, Dryden IL, Klassen E, Le H, Srivastava A (2012) Fitting smoothing splines to time-indexed, noisy points on nonlinear manifolds. Image Vis Comput 30(6):428–442
30. Tipping ME, Bishop CM (1999) Probabilistic principal component analysis. J R Stat Soc Ser B (Stat Methodol) 61(3):611–622
31. Trouvé A, Vialard F-X (2010) A second-order model for time-dependent data interpolation: Splines on shape spaces. In: MICCAI STIA workshop
32. Zhang M, Fletcher PT (2013) Probabilistic principal geodesic analysis. In: Neural information processing systems (NIPS)

Part II
Color, Motion, and Stereo

This part presents theory and applications in problems such as chromatic filtering, 3D motion estimation, and fundamental matrix estimation.

Chapter 6
Robust Estimation for Computer Vision Using Grassmann Manifolds

Saket Anand, Sushil Mittal, and Peter Meer

Abstract Real-world visual data are often corrupted and require the use of estimation techniques that are robust to noise and outliers. Robust methods are well studied for Euclidean spaces and their use has also been extended to Riemannian spaces. In this chapter, we present the necessary mathematical constructs for Grassmann manifolds, followed by two different algorithms that can perform robust estimation on them. In the first one, we describe a nonlinear mean shift algorithm for finding modes of the underlying kernel density estimate (KDE). In the second one, a user-independent robust regression algorithm, the generalized projection-based M-estimator (gpbM), is detailed. We show that the gpbM estimates are significantly improved if KDE optimization over the Grassmann manifold is also included. The results for a few real-world computer vision problems are shown to demonstrate the importance of performing robust estimation using Grassmann manifolds.

6.1 Introduction

Estimation problems in geometric computer vision often require dealing with orthogonality constraints in the form of linear subspaces. Since orthogonal matrices representing linear subspaces of Euclidean space can be represented as points on Grassmann manifolds, understanding the geometric properties of these manifolds can prove very useful for solving many vision problems. Usually, the estimation process involves optimizing an objective function to find the regression coefficients that best describe the underlying constraints. Alternatively, given a distribution of

S. Anand (✉)
IIIT-Delhi, New Delhi, India
e-mail: anands@iiitd.ac.in

S. Mittal
Scibler Corporation, Santa Clara, CA, USA
e-mail: mittal@scibler.com

P. Meer
Department of ECE, Rutgers University, Piscataway, NJ, USA
e-mail: meer@cronos.rutgers.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_6

sampled hypotheses of linear solutions, it could also be formulated as finding the cluster centers of those distributions as the dominant solutions to the underlying observations.

A typical regression problem in computer vision involves discovering multiple, noisy inlier structures present in the data corrupted with gross outliers. Usually, very little or no information is available about the number of inlier structures, the nature of the noise corrupting each one of them, and the amount of gross outliers. The original RAndom SAmple Consesus (RANSAC) [5] and its several variants like MLESAC, LO-RANSAC, PROSAC, and QDEGSAC [15] were designed for problems with single inlier structure where either a *fixed* estimate of the scale of inlier noise is provided by the user beforehand or it is estimated using a simple heuristic. As the complexity of the problems grow, a scale estimate becomes harder to determine. Although Grassmann manifolds provide a continuous parameter space to optimize and refine the estimate returned by a robust algorithm, it is still essential for the original algorithm to perform well in situations where the data deviate from the underlying model in a variety of different ways. Accurate estimation of the scale of inlier noise, especially for multidimensional problems, is an important step for all robust algorithms.

Robust methods applied to the more general Riemannian manifolds have appeared in computer vision literature. A short introduction to Riemannian manifolds, mainly from a computer vision point of view, and some applications can be seen in [12] and [19], as well as the references therein. More recent work in this area could be found in [7, 8, 10, 16, 20–22]. In case of Grassmann manifolds, such techniques amount to clustering and finding the modes of the subspace distributions obtained from the data.

This chapter is organized as follows. We introduce in Sect. 6.2, the necessary tools for solving subspace estimation and clustering problems on Grassmann manifolds. Section 6.3 describes two robust subspace estimation algorithms—the nonlinear mean shift and the generalized projection-based M-estimator (gpbM). A few applications of the two algorithms are presented in Sect. 6.4. We conclude with a discussion in Sect. 6.5.

6.2 Grassmann Manifolds

A point $\mathbf{X}$ on the Grassmann manifold, $\mathbf{G}_{m,k}$, represents a k-dimensional linear subspace in $\mathbb{R}^m$, where $m > k$. The point $\mathbf{X}$ is represented by an $m \times k$ orthogonal matrix, i.e., $\mathbf{X}^\top \mathbf{X} = \mathbf{I}_{k \times k}$ and is independent of the choice of any particular basis vectors. Hence, points on the Grassmann manifold are *equivalence classes* of $m \times k$ orthogonal matrices, where two matrices are equivalent if their columns span the same k-dimensional subspace in $\mathbb{R}^m$ [4].

Tangent vectors at $\mathbf{X}$ to $\mathbf{G}_{m,k}$ are also represented as $m \times k$ matrices. Given a real-valued function $f : \mathbf{G}_{m,k} \rightarrow \mathbb{R}$ on the manifold, $\Delta(f)$ is the magnitude of the derivative of f in the tangent direction Δ at $\mathbf{X}$. Intuitively, the tangent vector can be thought of as velocity of a point constrained to move on the manifold. The tangent

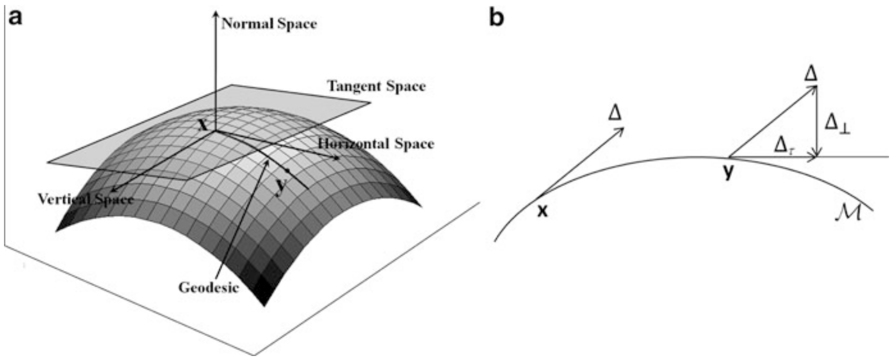


Fig. 6.1 (**a**) Tangent and normal spaces of Grassmann (Riemannian) manifold with the geodesic along the direction of the horizontal space. (**b**) Parallel transport of tangent Δ from **X** to **Y** on the manifold by removing the normal component $\Delta_\perp$

space can be further divided into complementary horizontal and vertical spaces. The space normal to the tangent space is called the normal space. See Fig. 6.1a for an illustration of this decomposition. A 3×3 matrix of $\mathbf{G}_{3,1}$ has two-dimensional horizontal, one-dimensional vertical, and six-dimensional normal space.

A *curve* in a Riemannian manifold $\mathcal{M}$ is a smooth mapping α from an open interval $\mathbf{T}$ of $\mathbb{R}$ to $\mathcal{M}$. For a particular $t \in \mathbf{T}$, $\mathbf{X}(t)$ lies on the manifold and $\mathbf{X}'(t)$ is a tangent vector at $\mathbf{X}(t)$. Given points $\mathbf{X}, \mathbf{Y}$ on $\mathcal{M}$, the shortest curve connecting $\mathbf{X}$ and $\mathbf{Y}$ is called the geodesic. The length of the geodesic is defined to be the *Riemannian distance* between the two points. It can be shown that for Grassmann manifold $\mathbf{G}_{m,k}$, a geodesic from $\mathbf{X}$ in the direction of the tangent $\boldsymbol{\Lambda}$ ($m \times k$ matrix) can be written as

$$\mathbf{X}(t) = [\mathbf{XV} \quad \mathbf{U}] \begin{bmatrix} \cos \boldsymbol{\Sigma} t \\ \sin \boldsymbol{\Sigma} t \end{bmatrix} \mathbf{V}^\top \tag{6.1}$$

where $\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ is the compact SVD of $\boldsymbol{\Lambda}$ (only $\mathbf{U}_k$, the first k columns of U are computed) and the operators *sin* and *cos* act element by element along the diagonal of $\boldsymbol{\Sigma}$.

In Euclidean space, a vector can be moved parallel to itself by just moving the base of the vector. Moving a tangent vector Δ from a point $\mathbf{X}$ to $\mathbf{Y}$ on the manifold $\mathcal{M}$ also accumulates a normal component $\Delta_\perp$ at $\mathbf{Y}$, which is subtracted from the transported vector. This is called *parallel translation* and is illustrated in Fig. 6.1b. A tangent vector $\Delta \in \Delta_{\mathbf{X}}$ at $\mathbf{X} = \mathbf{X}(0)$ can be parallel-translated to another point $\mathbf{Y} \in \mathcal{M}$ by infinitesimally removing the normal component of the translated vector, $\Delta_\perp$ along the path between $\mathbf{X}$ and $\mathbf{Y}$ on the manifold. For Grassmann manifold $\mathbf{G}_{m,k}$, the parallel-translation of Δ along the geodesic in direction $\boldsymbol{\Lambda}$ is given by

$$\Delta(t) = \left([\mathbf{XV} \quad \mathbf{U}] \begin{bmatrix} -\sin \boldsymbol{\Sigma} t \\ \cos \boldsymbol{\Sigma} t \end{bmatrix} \mathbf{U}^\top + [\mathbf{I} - \mathbf{U}\mathbf{U}^\top] \right) \Delta \tag{6.2}$$

where $\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ is the compact SVD of $\boldsymbol{\Lambda}$, and $\Delta(0) = \Delta$.

The horizontal space of the tangent vectors and geodesics on the manifold are closely related. There is a unique geodesic curve $\alpha : [0,1] \to \mathbf{G}_{m,k}$, starting at $\mathbf{X}$ with tangent vector Δ, which has the initial velocity $\alpha'(0) = \Delta$. The *exponential map*, $\exp_{\mathbf{X}}$, maps Δ from the tangent space to the point on the manifold reached by this geodesic

$$\exp_{\mathbf{X}}(\Delta) = \alpha(1) = \mathbf{X}(1) \tag{6.3}$$

where $\mathbf{X}(1)$ is computed using (6.1). The origin of the tangent space is mapped to the point itself, $\exp_{\mathbf{X}}(\mathbf{0}) = \mathbf{X}(0)$. For each point $\mathbf{X} \in \mathbf{G}_{m,k}$, there exists a neighborhood around the origin of $\Delta_{\mathbf{X}}$ that can be uniquely mapped to a neighborhood of $\mathbf{X}$ via $\exp_{\mathbf{X}}$. The inverse mapping is achieved by the *logarithm map*, $\log_{\mathbf{X}} = \exp_{\mathbf{X}}^{-1}$.

Given two points $\mathbf{X}, \mathbf{Y} \in \mathbf{G}_{m,k}$, the logarithm map finds a tangent direction $\boldsymbol{\Lambda}$ such that the geodesic from $\mathbf{X}$ along $\boldsymbol{\Lambda}$ reaches $\mathbf{Y}$ in unit time. With $[\mathbf{X} \quad \mathbf{X}_{\perp}]^{\top}[\mathbf{X} \quad \mathbf{X}_{\perp}] = \mathbf{I}_m$, let $\mathcal{C}$ and $\mathcal{S}$ represent the generalized singular values of $\mathbf{X}^{\top}\mathbf{Y}$ and $\mathbf{X}_{\perp}^{\top}\mathbf{Y}$ such that $\mathcal{C}^{\top}\mathcal{C} + \mathcal{S}^{\top}\mathcal{S} = \mathbf{I}_k$. With some computation, it can be shown that for $\mathbf{G}_{m,k}$, the logarithm operator can be written as

$$\log_{\mathbf{X}}(\mathbf{Y}) = \boldsymbol{\Lambda} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\top} = \mathbf{U}\cos^{-1}(\mathcal{C}_1)\,\mathbf{V}^{\top} = \mathbf{U}\sin^{-1}(\mathcal{S}_1)\,\mathbf{V}^{\top} \tag{6.4}$$

where $\mathcal{C}_1 = \begin{bmatrix} \mathcal{C} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{(m-k)} \end{bmatrix}$ and $\mathcal{S}_1 = \begin{bmatrix} \mathcal{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}_{(m-k)\times(m-k)} \end{bmatrix}$ and $\cos^{-1}$ and $\sin^{-1}$ act element by element along the diagonals of $\mathcal{C}_1$ and $\mathcal{S}_1$. The exponential and logarithm operators vary as the point $\mathbf{X}$ moves on $\mathbf{G}_{m,k}$ which is made explicit above by the subscript.

The distance between two points on the Grassmann manifold using (6.4) is given by

$$d(\mathbf{X}, \mathbf{Y}) = ||\log_{\mathbf{X}}(\mathbf{Y})||_F \tag{6.5}$$

where $||\cdot||_F$ is the matrix Frobenius norm. The gradient of the squared Riemannian distance for Grassmann manifolds [19] is

$$\nabla_{\mathbf{X}} d^2(\mathbf{X}, \mathbf{Y}) = -2\log_{\mathbf{X}}(\mathbf{Y}) \tag{6.6}$$

For a real-valued, scalar function $f : \mathbf{G}_{m,k} \to \mathbb{R}$ on $\mathbf{G}_{m,k}$, let $\partial \mathbf{f}_{\mathbf{X}}$ be the $m \times k$ Jacobian of f w.r.t. $\mathbf{X}$ such that $\partial \mathbf{f}_{\mathbf{X}}(i,j) = \partial f / \partial \mathbf{X}(i,j)$, $i = 1, \ldots, m$, and $j = 1, \ldots, k$. The jth column vector in $\partial \mathbf{f}_{\mathbf{X}}$ gives the partial differential of f w.r.t. the jth basis vector of $\mathbf{X}$. Since each entry of $\partial \mathbf{f}_{\mathbf{X}}$ is computed independently, in general, $\partial \mathbf{f}_{\mathbf{X}}$ *does not* lie in the tangent space $\boldsymbol{\Delta}_{\mathbf{X}}$. The *gradient* of f at $\mathbf{X}$ is the tangent vector $\boldsymbol{\nabla}\mathbf{f}_{\mathbf{X}}$ obtained by subtracting from $\partial \mathbf{f}_{\mathbf{X}}$ its component in subspace spanned by the columns of $\mathbf{X}$ yielding

$$\nabla \mathbf{f}_{\mathbf{X}} = \partial \mathbf{f}_{\mathbf{X}} - \mathbf{X}\mathbf{X}^{\top}\partial \mathbf{f}_{\mathbf{X}} = \mathbf{X}_{\perp}\mathbf{X}_{\perp}^{\top}\partial \mathbf{f}_{\mathbf{X}} \tag{6.7}$$

It is easy to verify that $\mathbf{X}^{\top}(\nabla \mathbf{f}_{\mathbf{X}}) = \mathbf{0}$. See [12] for proofs of most of the above equations.

6.3 Robust Estimation Using Grassmann Manifolds

First, we describe the nonlinear mean shift algorithm, which takes a clustering-based approach to identify dominant subspace hypotheses over the Grassmann manifold. Next, we describe the generalized projection-based M-estimator (gpbM), which improves the subspace estimate over the Grassmann manifold by using conjugate gradient optimization method.

6.3.1 Nonlinear Mean Shift on Grassmann Manifolds

The mean shift algorithm [3] takes an iterative approach for identifying local modes of the underlying kernel density estimate (KDE). In Euclidean space, this is achieved by computing weighted means of sample points in a local neighborhood. As opposed to this, the mean of points lying on a Grassmann manifold itself may not lie on the manifold. However, the tangent vectors of these points exist in a vector space, where their weighted mean can be computed and used to update the mode estimate. This method has been generalized to many Riemannian manifolds [19].

Given n points on the Grassmann manifold, $\mathbf{X}_i, i = 1, \ldots, n$, the kernel density estimate with a kernel profile κ and bandwidth h is

$$\hat{f}_{\kappa}(\mathbf{Y}) = \frac{c_{\kappa,h}}{n}\sum_{i=1}^{n}\kappa\left(\frac{d^2(\mathbf{Y},\mathbf{X}_i)}{h^2}\right) \tag{6.8}$$

where $c_{\kappa,h}$ is the normalization constant and $d(\cdot,\cdot)$ represents the Riemannian distance for Grassmann manifolds computed using (6.5). The kernel profile κ is related to the kernel function as $K_h(u) = c_{\kappa,h}\kappa(u^2)$ [3]. The bandwidth h, which is a tuning parameter in the application, is provided by the user. It can also be thought of as the scale parameter in the distance function.

The gradient of $\hat{f}_\kappa$ at $\mathbf{Y}$ is calculated as

$$\begin{aligned}\nabla\hat{f}_\kappa(\mathbf{Y}) &= \frac{1}{n}\sum_{i=1}^{n}\nabla\kappa\left(\frac{d^2(\mathbf{Y},\mathbf{X}_i)}{h^2}\right)\\ &= -\frac{1}{n}\sum_{i=1}^{n} g\left(\frac{d^2(\mathbf{Y},\mathbf{X}_i)}{h^2}\right)\frac{\nabla d^2(\mathbf{Y},\mathbf{X}_i)}{h^2}\\ &= \frac{2}{n}\sum_{i=1}^{n} g\left(\frac{d^2(\mathbf{Y},\mathbf{X}_i)}{h^2}\right)\frac{\log_{\mathbf{Y}}(\mathbf{X}_i)}{h^2}\end{aligned} \tag{6.9}$$

where $g(\cdot) = -\kappa'(\cdot)$, and the relation (6.6) is used in the last equation of (6.9). The gradient of the distance function is taken here with respect to $\mathbf{Y}$. Like in the Euclidean case, the updated nonlinear mean shift vector is then computed as

$$\delta\mathbf{Y} = \frac{\sum_{i=1}^{n} g\left(\frac{d^2(\mathbf{Y},\mathbf{X}_i)}{h^2}\right)\log_{\mathbf{Y}}(\mathbf{X}_i)}{\sum_{i=1}^{n} g\left(\frac{d^2(\mathbf{Y},\mathbf{X}_i)}{h^2}\right)} \tag{6.10}$$

This expression is defined in a vector space since $\log_{\mathbf{Y}}(\mathbf{X}_i)$ terms lie in the tangent space $\Delta_{\mathbf{Y}}$ and the kernel terms $g(d^2(\mathbf{Y},\mathbf{X}_i)/h^2)$ are scalars. Since the mean shift vector is computed in the tangent space, i.e., not on the manifold intrinsically, this algorithm was referred to as the *extrinsic* mean shift in [1]. On the other hand, the intrinsic mean shift (Int-MS) proposed in [1] operates directly on the manifold. As shown in Sect. 6.4.4, there is little difference in performance of the two mean shift procedures. The mean shift vector (6.10) computed in $\Delta_{\mathbf{Y}}$ is projected back to the Grassmann manifold using the exponential map for the jth iteration

$$\mathbf{Y}^{(j+1)} = \exp_{\mathbf{Y}^{(j)}}\left(\delta\mathbf{Y}^{(j)}\right) \tag{6.11}$$

Each iteration of (6.11) updates the current mode estimate $\mathbf{Y}^{(j)}$ by moving along the geodesic defined by the mean shift vector to get the next estimate, $\mathbf{Y}^{(j+1)}$. A mean shift iteration is initialized at each data point by initializing $\mathbf{X} = \mathbf{X}_i$ and repeatedly updated until convergence. The complete nonlinear mean shift algorithm is shown in Algorithm 1.

Algorithm 1 Nonlinear mean shift over Grassmann manifolds

```
Given: Points on Grassmann manifold $\mathbf{X}_i, i = 1, \ldots, n$
for $i \leftarrow 1 \ldots n$
$\mathbf{Y} \leftarrow \mathbf{X}_i$
repeat
$\delta\mathbf{Y} \leftarrow \frac{\sum_{i=1}^{n} g\left(d^2(\mathbf{Y}, \mathbf{X}_i)/h^2\right) log_{\mathbf{Y}}(\mathbf{X}_i)}{\sum_{i=1}^{n} g\left(d^2(\mathbf{Y}, \mathbf{X}_i)/h^2\right)}$
$\mathbf{Y} \leftarrow exp_{\mathbf{Y}}(\delta\mathbf{Y})$
until $\|\delta\mathbf{Y}\| < \epsilon$
Retain $\mathbf{Y}$ as a local mode
Report distinct local modes.
```

The mean shift procedure is initiated from every hypothesis point on the manifold. Points whose iterations converge into a particular mode belong to its basin of attraction. Dominant modes are identified as the ones having high kernel density estimates and a large number of points in their basins of attraction. Spurious modes are characterized by low kernel density estimates and fewer points in their basins of attraction and can be easily pruned.

6.3.2 Generalized Projection-Based M-Estimators

The generalized projection-based M-estimator (gpbM) [13] is a robust subspace estimation algorithm that works on the hypothesize and test principle. The scale of the inlier noise, the number of inlier structures, and the associated model hypotheses are automatically estimated by gpbM without any user intervention. The gpbM can also handle heteroscedastic data for single or multiple carrier problems in a unified framework. The original pbM algorithm [17] performed scale estimation for each newly chosen elemental subset-based hypothesis by computing the median absolute deviation (MAD) estimate in each dimension separately. More recently, a completely different scale estimation strategy was presented in [13] showing superior performance over competing methods. However, the model parameters were estimated by optimization over a discrete set of parameter matrices. In [12], these estimates were refined over the Grassmann manifold as a continuous space of parameter matrices, which led to significant performance improvements.

Given n_1 measurements of inlier variables $\mathbf{y}_i \in \mathbb{R}^p$, let $\mathbf{x}_i \in \mathbb{R}^m, i = 1, \ldots, n_1$ represent the corresponding *carrier* vectors that are usually monomials in a subset of the variables. For example, in the case of fitting an ellipse to measured data $\mathbf{y}_i = [y_1 \ y_2]^\top \in \mathbb{R}^2$, the corresponding carrier vector is given by $\mathbf{x} = [y_1 \ y_2 \ y_1^2 \ y_1 y_2 \ y_2^2]^\top \in \mathbb{R}^5$. In this setting, robust subspace estimation corresponds to performing a linear estimation in the carrier space of the data corrupted with outliers and containing an unknown number of noisy inlier points. We have n ($> n_1$) points $\mathbf{x}_i$, $i = 1, \ldots, n$, where $n - n_1$ points are outliers. The parameter matrix $\boldsymbol{\Theta}$ is an $m \times k$, $(k < m)$, orthonormal matrix such that $\boldsymbol{\Theta}^\top \boldsymbol{\Theta} = \mathbf{I}_{k\times k}$ and therefore can be represented as a point on the Grassmann manifold $\mathbf{G}_{m,k}$. Geometrically, it is a basis of the k-dimensional *null* space of the inlier data representing the k constraints imposed on them. The $\boldsymbol{\alpha} \in \mathbb{R}^k$ is the corresponding k-dimensional vector of intercepts. Therefore,

$$\boldsymbol{\Theta}^\top \mathbf{x}_{io} - \boldsymbol{\alpha} = \mathbf{0}_k \tag{6.12}$$

where $\mathbf{x}_{io}$, $i = 1, \ldots, n_1$, are the *unknown* true values of the inlier carrier points. The multiplicative ambiguity in the estimation of $\boldsymbol{\Theta}$ is resolved by enforcing $\boldsymbol{\Theta}^\top \boldsymbol{\Theta} = \mathbf{I}_{k\times k}$. No assumptions are made about the distribution of the $n - n_1$ outlier points.

The carrier vectors are often nonlinear in the variables, thereby making the estimation problem *heteroscedastic*, i.e., each carrier vector has a different noise covariance matrix and in general can even have a different mean. Given the covariance matrices of the observed variables, $\mathbf{C}_{\mathbf{y}_i}$, and the *Jacobians* of the carrier vectors with respect to those variables, $\mathbf{J}_{\mathbf{x}_i|\mathbf{y}_i}$, the first-order approximation of the $m \times m$ carrier covariances can be computed using error propagation as

$$\mathbf{C}_{\mathbf{x}_i} = \mathbf{J}^\top_{\mathbf{x}_i|\mathbf{y}_i} \mathbf{C}_{\mathbf{y}_i} \mathbf{J}_{\mathbf{x}_i|\mathbf{y}_i}, \quad i = 1, \ldots, n \tag{6.13}$$

The covariance of the vector of variables $\mathbf{C}_{\mathbf{y}_i}$ is often assumed to be the same for all i, but in general can be different for each $\mathbf{y}_i$. Since some carriers are associated with outliers, their covariances are computed incorrectly.

For each projected point $\mathbf{z}_i = \boldsymbol{\Theta}^\top \mathbf{x}_i$, the $k \times k$ covariance matrix is computed as $\mathbf{H}_i = \boldsymbol{\Theta}^\top \mathbf{C}_{\mathbf{x}_i} \boldsymbol{\Theta}$. The $k \times k$ point-dependent bandwidth matrices $\mathbf{B}_i$ are computed as $\mathbf{B}_i = \mathbf{S}^\top \mathbf{H}_i \mathbf{S}$ using the $k \times k$ diagonal scale matrix $\mathbf{S}$, with the diagonal entries corresponding to the value of scale of inlier noise in each dimension of the null space. Therefore, in order to compute $\mathbf{B}_i$, we need to estimate the k-dimensional scale first. In gpbM, each inlier structure is estimated by using a three-step procedure:

- scale estimation,
- mean shift-based robust model estimation,
- inlier/outlier dichotomy.

In case of multiple structures, the set of inlier points associated with each detected structure is removed iteratively from the data and the three-step procedure is repeated until no more significant inlier structures are found.

The gpbM algorithm follows a hypothesize-then-test strategy such that an estimate of the parameter pair $[\boldsymbol{\Theta}, \boldsymbol{\alpha}]$ is computed from a randomly selected elemental subset, i.e., minimal set of points necessary to generate a subspace hypothesis. The hypothesis corresponding to the best model over the set of all randomly generated hypotheses is selected by maximizing the following heteroscedastic objective function

$$\left[\hat{\boldsymbol{\Theta}},\hat{\boldsymbol{\alpha}}\right]=\arg\max_{\boldsymbol{\Theta},\boldsymbol{\alpha}}\frac{1}{n}\sum_{i=1}^{n}\frac{K\left(\left((\boldsymbol{\Theta}^\top\mathbf{x}_i-\boldsymbol{\alpha})^\top\mathbf{B}_i^{-1}(\boldsymbol{\Theta}^\top\mathbf{x}_i-\boldsymbol{\alpha})\right)^{\frac{1}{2}}\right)}{\sqrt{\det\mathbf{B}_i}} \tag{6.14}$$

where $K(u)$ is the kernel function and is related to a redescending M-estimator loss function by $K(u) = 1-\rho(u)$. The loss function $\rho(u)$ is nonnegative, symmetric, nondecreasing with $|u|$ and has a unique minimum of $\rho(0)=0$ and a maximum of one for $|u|>1$.

Step 1: Heteroscedastic Scale Estimation The estimation of the scale of inlier noise is equivalently posed as the problem of estimating the approximate fraction of data points belonging to an inlier structure. For estimating the fraction, M elemental subset-based model hypotheses are generated. For each hypothesis $[\boldsymbol{\Theta},\boldsymbol{\alpha}]$, the value of fraction is varied between $(0,1]$ in Q steps, such that for $q=1,\ldots,Q$, the fraction is $\eta_q=q/Q=n_q/n$. A value of $Q=40$ was used for all applications shown in [12, 13]. The scale of the inlier noise is estimated by taking into account the heteroscedasticity of the carrier vector. In the projected space defined by $\boldsymbol{\Theta}$, the volume around the intercept $\boldsymbol{\alpha}$ containing n_q points is computed as

$$vol^q(\boldsymbol{\Theta},\boldsymbol{\alpha})=\sqrt{\sum_{l=1}^{n_q}(\mathbf{z}_l-\boldsymbol{\alpha})^\top\mathbf{H}_l^{-1}(\mathbf{z}_l-\boldsymbol{\alpha})} \tag{6.15}$$

where $\mathbf{z}_l, l=1,\ldots,n_q$ are Mahalanobis distance-based n_q nearest neighbors of $\boldsymbol{\alpha}$. Given a hypothesis $[\boldsymbol{\Theta},\boldsymbol{\alpha}]$, the density at each fraction η_q, $q=1,\ldots,Q$, is computed as $n_q/(vol^q(\boldsymbol{\Theta},\boldsymbol{\alpha})+\epsilon)$, where a small constant ϵ is added to suppress extremely high numerical values of the densities at low fractions. The computed density values are used to populate an $M\times Q$ matrix $\boldsymbol{\Psi}$.

From the matrix $\boldsymbol{\Psi}$, the peak density values along each of the M hypotheses (rows) and their corresponding fractions (columns) are retained. Typically, some columns are associated with several density peaks while other columns are not. At each column q, a fraction η_q of the highest peak density values are summed up. The largest sum of density peaks corresponds to $\eta_{\hat{q}}$, i.e., the estimate of the fraction of points comprising an inlier structure. This approach makes the fraction estimate more robust, especially when multiple inlier structures exist, each comprising very different number of points.

The hypothesis $[\boldsymbol{\Theta}^*,\boldsymbol{\alpha}^*]$ that gives the highest density at $\eta_{\hat{q}}$ is used to project the data points $\mathbf{x}_i$. The dimensions of the smallest k-dimensional rectangle enclosing the $n_{\hat{q}}$ nearest neighbors of $\boldsymbol{\alpha}^*$ provide the final estimate of the scale, which forms the diagonal of $\mathbf{S}$. The corresponding $n_{\hat{q}}$ points enclosed inside the rectangle form an initial estimate of the inliers.

Step 2: Model Estimation Model estimation is performed by generating N elemental subset-based model hypotheses. However, only the initial set of inliers returned by the scale estimation step is used for the selection of elemental subsets, making the model estimation very efficient. For a given hypothesis $[\boldsymbol{\Theta}, \boldsymbol{\alpha}]$, the problem (6.14) can be rewritten as that of estimating the kernel density of the data points projected to the k-dimensional null space. The *adaptive* kernel density function over the projections $\mathbf{z}_i = \boldsymbol{\Theta}^\top \mathbf{x}_i \in \mathbb{R}^k, i = 1, \ldots, n$ is defined as

$$\hat{f}_\kappa(\boldsymbol{\Theta}, \mathbf{z}) = \frac{1}{n} \sum_{i=1}^{n} \frac{\kappa\left((\mathbf{z} - \mathbf{z}_i)^\top \mathbf{B}_i^{-1} (\mathbf{z} - \mathbf{z}_i)\right)}{\sqrt{\det \mathbf{B}_i}} \tag{6.16}$$

where $\kappa(u^2) = K(u)$ is the profile of the kernel function $K(u)$. Differentiating (6.16) w.r.t. $\mathbf{z}$,

$$\frac{d\hat{f}_\kappa(\boldsymbol{\Theta}, \mathbf{z})}{d\mathbf{z}} = \frac{2}{n} \sum_{i=1}^{n} \mathbf{B}_i^{-1} (\mathbf{z} - \mathbf{z}_i) \frac{g\left((\mathbf{z} - \mathbf{z}_i)^\top \mathbf{B}_i^{-1} (\mathbf{z} - \mathbf{z}_i)\right)}{\sqrt{\det \mathbf{B}_i}} = 0 \tag{6.17}$$

where $g(u^2) = -d\left(\kappa(u^2)\right)/d(u^2)$. The Euclidean mean shift vector can be written as

$$\delta\mathbf{z} = \left[\sum_{i=1}^{n} \frac{\mathbf{B}_i^{-1} g(\ldots)}{\sqrt{\det \mathbf{B}_i}}\right]^{-1} \left[\sum_{i=1}^{n} \frac{\mathbf{B}_i^{-1} \mathbf{z}_i g(\ldots)}{\sqrt{\det \mathbf{B}_i}}\right] - \mathbf{z} \tag{6.18}$$

The mean shift procedure is initiated from $\mathbf{z}^{(0)}$, i.e., the projection of the elemental subset points on $\boldsymbol{\Theta}$. The update $\mathbf{z}^{(j+1)} = \delta\mathbf{z}^{(j)} + \mathbf{z}^{(j)}$ is a gradient ascent step converging to $\boldsymbol{\alpha}$, the closest mode of the KDE (6.16).

Step 2.1: Conjugate Gradient on the Grassmann Manifold Each $\boldsymbol{\Theta}$ is an $m \times k$ orthogonal matrix and can be represented as a point on the Grassmann manifold, $\mathbf{G}_{m,k}$. Continuous optimization techniques to maximize the objective function of (6.16) over $\mathbf{G}_{m,k}$ can therefore be employed.

The conjugate gradient algorithm is widely used for optimization of nonlinear objective functions defined over Euclidean spaces. This popularity is due to fast convergence rates achieved by iteratively moving along *linearly independent* directions in the solution space. Moreover, it avoids computing the Hessian, thus making each iteration less expensive than other alternatives like Newton's method. The optimization along a chosen direction is performed using *line search* methods and in this case Brent's method [14, pp. 402–405] is used. Edelman et al. [4] adapted the conjugate gradient algorithm to minimize a function $f : \mathbf{G}_{m,k} \to \mathbb{R}$ over the Grassmann manifold $\mathbf{G}_{m,k}$.

Conjugate gradient method originally being a function *minimization* algorithm, the function $f_\Diamond(\boldsymbol{\Theta}, \boldsymbol{\alpha}) = -\hat{f}_\kappa(\boldsymbol{\Theta}, \boldsymbol{\alpha})$ is optimized. The function $f_\Diamond(\boldsymbol{\Theta}, \boldsymbol{\alpha})$ is jointly minimized over its domain $\mathbf{G}_{m,k} \times \mathbb{R}^k$ with each iteration of conjugate gradient simultaneously updating both $\boldsymbol{\Theta} \in \mathbf{G}_{m,k}$ and $\boldsymbol{\alpha} \in \mathbb{R}^k$. Given an estimated pair $[\boldsymbol{\Theta}, \boldsymbol{\alpha}]$, the initial gradient of the objective function $f_\Diamond(\boldsymbol{\Theta}, \boldsymbol{\alpha})$ w.r.t. $\boldsymbol{\Theta}$ on $\mathbf{G}_{m,k}$ is computed using (6.7) as

$$\nabla \mathbf{f}_{\boldsymbol{\Theta}} = \partial \mathbf{f}_{\boldsymbol{\Theta}} - \boldsymbol{\Theta}\boldsymbol{\Theta}^{\top} \partial \mathbf{f}_{\boldsymbol{\Theta}} \tag{6.19}$$

where $\partial \mathbf{f}_{\boldsymbol{\Theta}}$ is the Jacobian of $f_{\Diamond}(\boldsymbol{\Theta}, \boldsymbol{\alpha})$ w.r.t. $\boldsymbol{\Theta}$. The corresponding gradient w.r.t. $\boldsymbol{\alpha}$ is given by

$$\nabla \mathbf{f}_{\boldsymbol{\alpha}} = \partial \mathbf{f}_{\boldsymbol{\alpha}} \tag{6.20}$$

where $\partial \mathbf{f}_{\boldsymbol{\alpha}}$ is the Jacobian of $f_{\Diamond}(\boldsymbol{\Theta}, \boldsymbol{\alpha})$ w.r.t. $\boldsymbol{\alpha}$.

The Jacobians $\partial \mathbf{f}_{\boldsymbol{\Theta}}$ and $\partial \mathbf{f}_{\boldsymbol{\alpha}}$ depend on the choice of the kernel function and are computed to a first-order approximation. This is equivalent to assuming an *explicit independence* among $\boldsymbol{\Theta}$, $\boldsymbol{\alpha}$, and the covariance matrices $\mathbf{H}_i, i = 1, \dots, n$, i.e., the Jacobian computation does not involve differentiating $\mathbf{H}_i$ w.r.t. $\boldsymbol{\Theta}$ and $\boldsymbol{\alpha}$. For the Epanechnikov kernel defined as

$$K(u) \simeq \begin{cases} 1 - u^2 & \text{if } |u| \leq 1 \\ 0 & \text{if } |u| > 1 \end{cases} \tag{6.21}$$

and $j = 1, \dots, k, \quad l = 1, \dots, m$, the entries of the $m \times k$ matrix $\partial \mathbf{f}_{\boldsymbol{\Theta}}$ are

$$\partial \mathbf{f}_{\boldsymbol{\Theta}}(l,j) = -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{\sqrt{\det \mathbf{B}_i}} \left(\frac{\partial \kappa(u^2)}{\partial \boldsymbol{\Theta}(l,j)} \right) = \frac{2}{n} \sum_{i=1}^{n} \mathbf{p}_i(j) \mathbf{x}_i(l) \tag{6.22}$$

where $u = \left((\boldsymbol{\Theta}^{\top}\mathbf{x}_i - \boldsymbol{\alpha})^{\top} \mathbf{B}_i^{-1} (\boldsymbol{\Theta}^{\top}\mathbf{x}_i - \boldsymbol{\alpha}) \right)^{\frac{1}{2}}$ and $\mathbf{p}_i = \frac{\mathbf{B}_i^{-1}(\boldsymbol{\Theta}^{\top}\mathbf{x}_i - \boldsymbol{\alpha})}{\sqrt{\det \mathbf{B}_i}}$. The entries of the k-dimensional vector $\partial \mathbf{f}_{\boldsymbol{\alpha}}$ are given as

$$\partial \mathbf{f}_{\boldsymbol{\alpha}}(j) = -\frac{1}{n} \sum_{i=1}^{n} \frac{1}{\sqrt{\det \mathbf{B}_i}} \left(\frac{\partial \kappa(u^2)}{\partial \boldsymbol{\alpha}(j)} \right) = -\frac{2}{n} \sum_{i=1}^{n} \mathbf{p}_i(j) \qquad j = 1, \dots, k \tag{6.23}$$

The conjugate gradient algorithm is initialized by setting the optimization variables to $\left[\boldsymbol{\Theta}^{(0)}, \boldsymbol{\alpha}^{(0)}\right]$ as estimated in the model estimation step. The initial search directions are taken to be the negative gradient direction, i.e., $\boldsymbol{\Lambda}^{(0)} = -\nabla \mathbf{f}_{\boldsymbol{\Theta}^{(0)}}$ and $\boldsymbol{\lambda}^{(0)} = -\nabla \mathbf{f}_{\boldsymbol{\alpha}^{(0)}}$, computed using (6.19) and (6.20). For each iteration j, Brent's method is applied for minimization of $f_{\Diamond}$ in directions along $\left[\boldsymbol{\Lambda}^{(j)}, \boldsymbol{\alpha}^{(j)}\right]$ and the variables $\left[\boldsymbol{\Theta}^{(j+1)}, \boldsymbol{\alpha}^{(j+1)}\right]$ are updated to this directional minimum. Both the search and the gradient directions on the Grassmann manifold are parallel-translated to the newly updated location $\boldsymbol{\Theta}^{(j+1)}$ using (6.2) and are denoted by $\boldsymbol{\Lambda}_{\tau}^{(j)}$ and $\nabla_{\tau} \mathbf{f}_{\boldsymbol{\Theta}^{(j)}}$, respectively. The equivalent operations for $\boldsymbol{\lambda}^{(j)}$ and $\mathbf{f}_{\boldsymbol{\alpha}^{(j)}}$ are simply the Euclidean translations in $\mathbb{R}^k$. The new gradient directions $\left[\nabla \mathbf{f}_{\boldsymbol{\Theta}^{(j+1)}}, \nabla \mathbf{f}_{\boldsymbol{\alpha}^{(j+1)}}\right]$ are computed at the updated points and the resulting conjugate directions are

$$\boldsymbol{\Lambda}^{(j+1)} = -\nabla \mathbf{f}_{\boldsymbol{\Theta}^{(j+1)}} + \omega^{(j)} \boldsymbol{\Lambda}_{\tau}^{(j)}$$

$$\boldsymbol{\lambda}^{(j+1)} = -\nabla \mathbf{f}_{\boldsymbol{\alpha}^{(j+1)}} + \omega^{(j)} \boldsymbol{\lambda}^{(j)} \tag{6.24}$$

where

$$\omega^{(j)} = \frac{\operatorname{trace}\left(\left[\nabla \mathbf{f}_{\boldsymbol{\Theta}^{(j+1)}} - \nabla_\tau \mathbf{f}_{\boldsymbol{\Theta}^{(j)}}\right]^\top \nabla \mathbf{f}_{\boldsymbol{\Theta}^{(j+1)}}\right) + \left[\nabla \mathbf{f}_{\boldsymbol{\alpha}^{(j+1)}} - \nabla \mathbf{f}_{\boldsymbol{\alpha}^{(j)}}\right]^\top \nabla \mathbf{f}_{\boldsymbol{\alpha}^{(j+1)}}}{\operatorname{trace}\left(\left(\nabla \mathbf{f}_{\boldsymbol{\Theta}^{(j)}}\right)^\top \nabla \mathbf{f}_{\boldsymbol{\Theta}^{(j)}}\right) + \left(\nabla \mathbf{f}_{\boldsymbol{\alpha}^{(j)}}\right)^\top \nabla \mathbf{f}_{\boldsymbol{\alpha}^{(j)}}}$$

While the covariance matrices $\mathbf{H}_i^{(j)} = \left(\boldsymbol{\Theta}^{(j)}\right)^\top \mathbf{C}_{\mathbf{x}_i} \boldsymbol{\Theta}^{(j)},\ i = 1, \ldots, n$, should ideally be recomputed in each iteration, it was shown in [12] that maintaining $\mathbf{H}_i^{(j)} = \mathbf{H}_i^{(0)},\ j = 1, 2, \ldots,\ i = 1, \ldots, n$ reduced the computational cost significantly without noticeable changes in the final estimates. After convergence of the conjugate gradient, the optimization variables $\left[\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{\alpha}}\right]$ give the final estimate of the parameter matrix and the intercept.

Step 3: Inlier/Outlier Dichotomy Given the estimated model $\left[\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{\alpha}}\right]$, the deviation of each point from the mode is normalized by its point-dependent covariance. For $\mathbf{z}_i = \hat{\boldsymbol{\Theta}}^\top \mathbf{x}_i, i = 1, \ldots, n$, the heteroscedastic projections around the mode are computed as

$$\tilde{\mathbf{z}}_i = \hat{\boldsymbol{\alpha}} + \frac{(\mathbf{z}_i - \hat{\boldsymbol{\alpha}})}{||\mathbf{z}_i - \hat{\boldsymbol{\alpha}}||_2} \sqrt{(\mathbf{z}_i - \hat{\boldsymbol{\alpha}})^\top \mathbf{H}_i^{-1} (\mathbf{z}_i - \hat{\boldsymbol{\alpha}})} \tag{6.25}$$

This reduces the inlier/outlier dichotomy to homoscedastic mean shift clustering problem since the bandwidth matrices $\tilde{\mathbf{B}}_i = \mathbf{S}^\top \mathbf{I}_{k\times k} \mathbf{S} = \mathbf{S}^\top \mathbf{S}$ also become constant for all points. The points for which the mean shift iterations converge to $\hat{\boldsymbol{\alpha}}$ (within a small tolerance) are considered inliers.

Using the estimated scale matrix $\mathbf{S}$, the strength of the detected inlier structure is computed as $\xi = f\left(\hat{\boldsymbol{\Theta}}, \hat{\boldsymbol{\alpha}}\right) / \|\mathbf{S}\|^2$. The algorithm stops if the strength drops by a factor of 20 compared to the maximum strength among the previously computed inlier structures, indicating that the remaining points comprise only gross outliers.

6.4 Applications

A result of the nonlinear mean shift for chromatic noise filtering is presented, followed by the application of generalized projection-based M-estimators (gpbM) algorithm applied to fundamental matrix and homography estimation. Finally, a quantitative comparison of both methods with related robust estimation techniques is presented using a subset of the Hopkins155 dataset.

6.4.1 Chromatic Noise Filtering

In general, pixels in images are represented by m-dimensional vectors, e.g., RGB values encoded as a three-vector. Chromatic image noise affects only a pixel's chromaticity, i.e., the direction of the color vector but not its intensity. Here, we restrict the filtering application to RGB images, and therefore the chromaticity can be represented by unit vectors in $\mathbb{R}^3$, which lies on the Grassmann manifold $\mathbf{G}_{3,1}$. The original mean shift has been used for the discontinuity preserving filtering of color images [2, 3]. This algorithm was extended to manifold-valued images in [18].

The image $\mathbf{I}$ is considered to be a mapping on a d-dimensional lattice which assigns a value to each lattice point. Visual images typically have $d = 2$, although 3D images with $d = 3$ are also used. In this case $d = 2$ and at each location $\mathbf{z}_i = [x_i, y_i]^\top$, the data values $\mathbf{I}(\mathbf{z}_i)$ are assumed to lie on $\mathbf{G}_{3,1}$. A pixel $\mathbf{I}(\mathbf{z}_i)$ along with its location $\mathbf{z}_i$ is considered as a single data point $\mathbf{x}_i = (\mathbf{z}_i, \mathbf{I}(\mathbf{z}_i))$, in the *joint domain* $\mathbb{R}^2 \times \mathbf{G}_{3,1}$.

The mean shift iterations are performed in this joint space to cluster the pixels. Consider an iteration starting at the point $\mathbf{x}_i = (\mathbf{z}_i, \mathbf{c}_i)$, where $\mathbf{c}_i = \mathbf{I}(\mathbf{z}_i)$, that converges to the mode $(\hat{\mathbf{z}}_i, \hat{\mathbf{c}}_i)$. In the filtered image $\hat{\mathbf{I}}$, all the pixel values converging to this mode are set to $\hat{\mathbf{c}}_i$. The profile in the joint domain is the product of a *spatial profile* defined on $\mathbb{R}^2$ and a *parameter profile* defined on the Grassmann manifold $\mathbf{G}_{3,1}$:

$$\kappa(\mathbf{x}, \mathbf{x}_i) = \kappa_s\left(\frac{\|\mathbf{z} - \mathbf{z}_i\|^2}{h_s^2}\right)\kappa_p\left(\frac{d^2(\mathbf{c}, \mathbf{c}_i)}{h_p^2}\right) \tag{6.26}$$

A truncated normal kernel was used for both κ_s and κ_p. The bandwidth in the joint domain consists of a one-dimensional spatial bandwidth h_s and a one-dimensional parameter bandwidth h_p. The bandwidths h_s and h_p can be varied by the user to achieve the desired quality in the output.

Subbarao and Meer [19] showed that chromatic filtering using Grassmann manifolds leads to remarkable improvements over the original mean shift which smooths both intensity and color. The filtering results for the 512×512 *peppers* image are shown in Fig. 6.2. The image is corrupted with chromatic noise by adding Gaussian noise with standard deviation $\sigma = 0.3$, along tangent directions followed by an exponential map onto $\mathbf{G}_{3,1}$. The original mean shift image filtering algorithm from EDISON was executed with carefully selected bandwidth parameters (spatial $h_s = 11.0$ and color $h_p = 10.5$) to obtain the middle image. Using larger h_p led to oversmoothing and using smaller values did not denoise the original noisy image sufficiently. The nonlinear mean shift was executed with the same $h_s = 11.0$ but with $h_p = 0.5$ to obtain the image on the right. The nonlinear chromatic filtering is clearly better than EDISON due to the smoothening of the correct noise model.

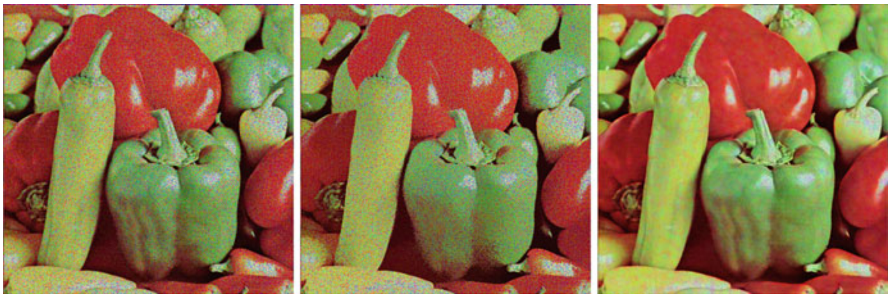


Fig. 6.2 *Chromatic Noise Filtering. Mean Shift over* $\mathbb{R}^2 \times \boldsymbol{G}_{3,1}$. The *peppers* image corrupted with chromatic noise is shown on the *left*. The results of using standard mean shift filtering with EDISON are in the *middle* and the result of nonlinear mean shift filtering is on the *right*

6.4.2 Fundamental Matrix Estimation

Reliable estimation of the fundamental matrix is often crucial for multi-view vision systems. Typically, in robust estimation formulations, the 3×3 fundamental matrix is represented by $\boldsymbol{\theta} \in \mathbb{R}^8$ while $\alpha \in \mathbb{R}$. Each data point is a vector of variables $\mathbf{y} = [x \;\; y \;\; x' \;\; y']^\top$ and lies in $\mathbb{R}^4$. Here, (x, y) and (x', y') are the coordinates of the corresponding points in the two images. Using the homogeneous image coordinates (without the points at infinity), the epipolar constraint can be written as

$$[x' \;\; y' \;\; 1]\mathbf{F}_{3\times3}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = 0 \tag{6.27}$$

The carrier vector is written as $\mathbf{x} = [x \;\; y \;\; x' \;\; y' \;\; xx' \;\; xy' \;\; yx' \;\; yy']^\top$ which lies in $\mathbb{R}^8$. Assuming the variables $\mathbf{y}$ have covariance $\sigma^2\mathbf{I}_{4\times4}$, the first-order approximation of the covariance matrix of $\mathbf{x}$ is computed from the Jacobian using error propagation

$$\mathbf{J}_{\mathbf{x}|\mathbf{y}} = \begin{bmatrix} 1 & 0 & 0 & 0 & x' & y' & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & x' & y' \\ 0 & 0 & 1 & 0 & x & 0 & y & 0 \\ 0 & 0 & 0 & 1 & 0 & x & 0 & y \end{bmatrix} = [\, \mathbf{I}_{4\times4} \;\; \mathbf{J}(\mathbf{y}) \,] \tag{6.28}$$

$$\mathbf{C}_{\mathbf{x}} = \sigma^2 \mathbf{J}_{\mathbf{x}|\mathbf{y}}^\top \, \mathbf{I}_{4\times4} \, \mathbf{J}_{\mathbf{x}|\mathbf{y}} = \sigma^2 \begin{bmatrix} \mathbf{I}_{4\times4} & \mathbf{J}(\mathbf{y}) \\ \mathbf{J}(\mathbf{y})^\top & \mathbf{J}(\mathbf{y})^\top\mathbf{J}(\mathbf{y}) \end{bmatrix} \tag{6.29}$$

Raglan Castle Images The gpbM algorithm was used to estimate the fundamental matrix between the *Raglan Castle* image pair shown in Fig. 6.3. Notice the large viewpoint change between the left and the right images. Using the SIFT algorithm

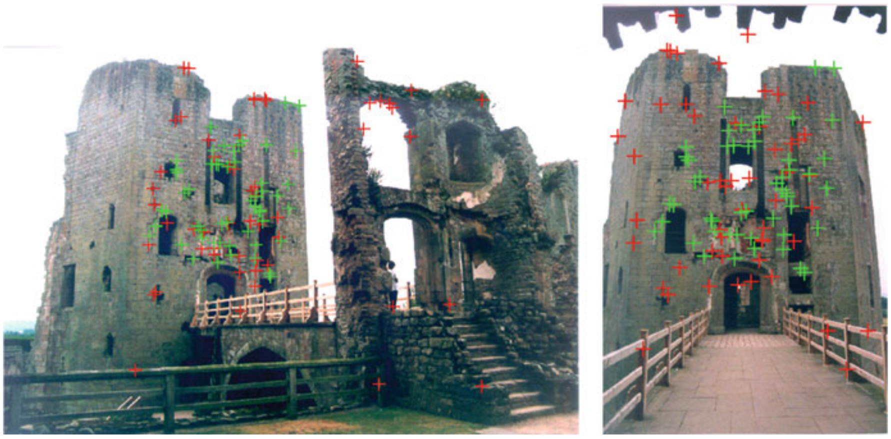


Fig. 6.3 Two images from the *Raglan Castle* sequence. The true inliers are marked with *green markers* while the outliers with *red markers*. The viewpoints of the two images are very different

[11], 109 point matches were obtained, out of which 54 were true inliers. With the values of $M = 400$ and $N = 200$, the performance of the gpbM algorithm was compared over 50 runs with and without the optimization on Grassmann manifold. On average, the gpbM algorithm misclassified 7.1 (out of 109) points, while only 5.6 points were classified wrongly after using the conjugate gradient algorithm. The average absolute residual error for the 54 true inlier points using gpbM algorithm was 1.86 pixels, while it was 1.77 pixels when optimization using conjugate gradient algorithm was also performed.

6.4.3 Planar Homography Estimation

A planar homography is a general 2D mapping between corresponding points on two projective planes. A pair of inlier homogeneous point correspondences $\mathbf{p}$ and $\mathbf{p}'$, represented in homogeneous coordinates, satisfy $\mathbf{p}' = \mathbf{H}\mathbf{p}$. Using the Direct Linear Transformation (DLT) [6, Alg. 7.1], the same equation can be rewritten as

$$\mathbf{A}_i\mathbf{h} = \begin{bmatrix} -\mathbf{p}_i^\top & \mathbf{0}_3^\top & x_i'\mathbf{p}_i^\top \\ \mathbf{0}_3^\top & -\mathbf{p}_i^\top & y_i'\mathbf{p}_i^\top \end{bmatrix} \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \mathbf{0}_2 \qquad i = 1, \ldots, n_1 \tag{6.30}$$

where $\mathbf{p}_i = [x_i \;\; y_i \;\; 1]^\top$ and $\mathbf{p}_i' = [x_i' \;\; y_i' \;\; 1]^\top$ are obtained from the image point correspondences of the n_1 inliers. The parameter vector $\boldsymbol{\theta}$ is obtained by rearranging the 3×3 homography matrix $\mathbf{H} = [\mathbf{h}_1 \; \mathbf{h}_2 \; \mathbf{h}_3]^\top$. The variables are $\mathbf{y}_i = [x_i \;\; y_i \;\; x_i' \;\; y_i']^\top$

and $\mathbf{A}_i$ is the 2×9 carrier matrix. The rows of $\mathbf{A}_i$ correspond to the *two* carrier vectors $\mathbf{x}_i^{[1]}, \mathbf{x}_i^{[2]} \in \mathbb{R}^9$ obtained from each point correspondence and are heteroscedastic due to multiplicative terms.

Given four point correspondences across two planes, the 8×9 data matrix is formed by stacking the corresponding four carrier matrices. For a point correspondence pair, the 4×4 covariance matrix of the variable vector $\mathbf{y}$ is given as $\mathbf{C}_\mathbf{y} = \sigma^2\mathbf{I}_{4\times4}$. The 4×9 Jacobians of the two carriers given by

$$\mathbf{J}_{\mathbf{x}_i^{[1]}|\mathbf{y}_i} = \begin{bmatrix} -\mathbf{I}_{2\times2} & & x'\mathbf{I}_{2\times2} & \mathbf{0}_2 \\ \mathbf{0}_2^\top & \mathbf{0}_{4\times4} & \mathbf{p}^\top & \\ \mathbf{0}_2^\top & & \mathbf{0}_2^\top & 0 \end{bmatrix}$$

$$\mathbf{J}_{\mathbf{x}_i^{[2]}|\mathbf{y}_i} = \begin{bmatrix} & -\mathbf{I}_{2\times2} & & y'\mathbf{I}_{2\times2} & \mathbf{0}_2 \\ \mathbf{0}_{4\times3} & \mathbf{0}_2^\top & \mathbf{0}_4 & \mathbf{0}_2^\top & 0 \\ & \mathbf{0}_2^\top & & \mathbf{p}^\top & \end{bmatrix} \quad (6.31)$$

are used to compute the 9×9 covariance matrices for each carrier vector $\mathbf{C}_i^{[c]} = \sigma^2\mathbf{J}^\top_{\mathbf{x}_i^{[c]}|\mathbf{y}_i}\mathbf{C}_{\mathbf{y}_i}\mathbf{J}_{\mathbf{x}_i^{[c]}|\mathbf{y}_i}$, $c = 1, 2$. The covariances correctly capture the point-dependent noise for inlier points only.

Graffiti Images Figure 6.4 shows two images from the *Graffiti* dataset used for planar homography estimation. The projective distortion between the two images is clearly visible. The SIFT algorithm [11] identified 61 point matches, out of which only 21 were true matches. Using $M = 800$ and $N = 200$, the performance of the gpbM algorithm was compared over 50 runs with and without performing the optimization on Grassmann manifolds. On average, the gpbM algorithm misclassified 8.68 (out of 61) points, while only 7.31 points were classified incorrectly after using the conjugate gradient algorithm. The relatively large number of misclassifications is due to the large projective distortion between

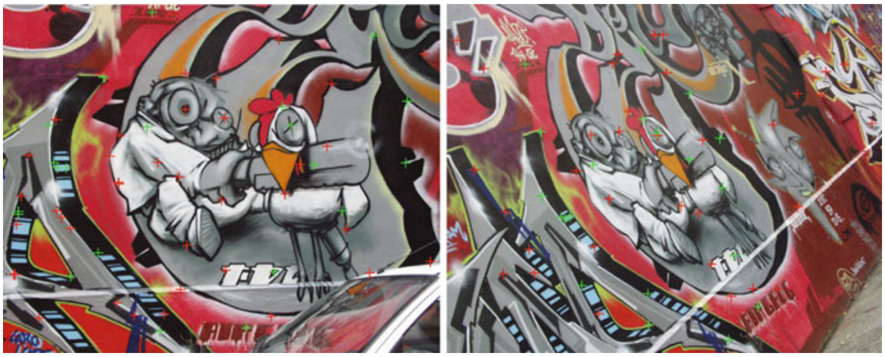


Fig. 6.4 Two images of the *Graffiti* dataset. The true inliers are shown with *green markers* while the outliers are in *red*

the two images. The average absolute residual error over the 21 true inlier points using gpbM algorithm was 1.132 pixels while it was 1.091 pixels after using the conjugate gradient algorithm too.

6.4.4 Affine Motion Factorization

When n_1 image points lying on a rigid object undergoing an affine motion are tracked over F frames, each trajectory can be represented as a $2F$-dimensional point. Due to the rigid body motion constraint, these points lie in a three-dimensional subspace of $\mathbb{R}^{2F}$. Please see [6, pp. 436–439] for more details about motion factorization. The gpbM algorithm was applied to projective motion factorization in [13] with extensive quantitative comparisons. Due to space constraints, only affine motion factorization is presented here to demonstrate the improvements achieved through the additional optimization on Grassmann manifolds.

The quantitative performance comparison of four different robust estimation algorithms is discussed. The nonlinear mean shift [19] is referred to as Ext-MS because the mean shift computation is performed in the tangent space *extrinsic* to the manifold. The remaining three algorithms are the intrinsic nonlinear mean shift (Int-MS) [1], generalized projection-based M-estimator (gpbM) [13], and gpbM with conjugate gradient on Grassmann manifolds (gpbM+CG) [12]. The input data are the point matches across the F frames and the performance is compared using percentage of misclassified points.

The algorithms are tested on ten video sequences containing multiple motions from *Hopkins155* dataset. This dataset does not contain outliers. For each sequence, $F = 5$ frames were used by picking every sixth or seventh frame from the sequence. For gpbM with and without using conjugate gradient on Grassmann manifolds, the values of $M = 1000$ and $N = 200$ were used. The Ext-MS algorithm used 1000 randomly generated elemental subset-based hypotheses. The corresponding parameter matrices were clustered on the Grassmann manifold $\mathbf{G}_{10,3}$ using the algorithm described in Sect. 6.3.1 with the bandwidth parameter h set to 0.1. The algorithm Int-MS used 1000 hypotheses generated as in the Ext-MS case.

Table 6.1 shows the comparative performance based on the percentage misclassification error. The results of the gpbM and gpbM+CG algorithms were averaged over 50 independent runs, while those of Ext-MS and the Int-MS were averaged over 20 independent runs. Neither of the methods assumed the knowledge of the true number of motions, but the nonlinear mean shift algorithms need the bandwidths as input from the user. It is clear that the optimization over Grassmann manifolds improves the estimates for every sequence.

Table 6.1 Average percentage of misclassified points

Sequence	Ext-MS (%) [19]	Int-MS (%) [1]	gpbM (%) [13]	gpbM+CG (%) [12]
Arm(2)	30.65	27.73	7.99	7.79
Articulated(3)	30.17	24.50	6.90	6.70
Cars1(2)	20.07	23.00	6.51	5.96
Cars2(2)	11.90	9.08	3.58	3.55
Cars4(2)	21.60	11.94	7.55	7.31
Cars5(3)	19.94	19.41	8.93	8.05
Cars6(2)	5.68	7.09	1.86	1.85
Cars8(2)	42.71	35.29	7.31	6.97
Truck1(2)	28.56	13.24	6.27	6.09
2RT3RC(3)	12.52	7.40	10.92	10.06
Overall	17.91	14.64	6.58	6.18

CG stands for conjugate gradient on Grassmann manifolds. The number in the parenthesis in the first column shows the true number of motions for each sequence. The results of gpbM and gpbM+CG were averaged over 50 runs while those of Ext-MS and Int-MS were averaged over 20 runs

6.5 Discussion

The image formation process imposes constraints on the imaged objects, e.g. the trajectory of points on a moving rigid body in a video sequence, human joint angle trajectories, shape of planar regions across multiple views, etc. Many computer vision techniques assume the pinhole camera model for image formation, which allows us to interpret the physical constraints as data points lying in an unknown linear subspace. The subspace parameters characterize physical world phenomena like camera rotation, object motion, human activity, etc. An orthonormal matrix $\mathbf{X}_{n\times k}$, lying on the Grassmann manifold can uniquely represent a linear subspace. Therefore, the geometry of Grassmann manifolds provides a natural mathematical framework to design subspace estimation or tracking algorithms for several computer vision applications.

The nonlinear mean shift on Grassmann manifolds takes a clustering-based approach to solve the robust subspace estimation problem. For the applications that we presented here, both the Ext-MS [19] and Int-MS [1] produced comparable results. However, these methods can only be used when the user provides a correct bandwidth parameter. A new research direction could be to make these algorithms fully automatic by employing purely data-driven techniques for estimating the bandwidth. In [19], many other Riemannian manifolds were analyzed—Lie groups, the space of symmetric positive definite matrices, and the essential manifold, which is a composite Lie group formed by the product space of two special orthogonal ($\mathbf{SO}(3)$) groups. Clustering-based approaches on these manifolds were used to solve a variety of computer vision problems.

We also discussed the generalized projection-based M-estimator (gpbM), which benefited from optimization over Grassmann manifolds. The first step of gpbM estimates an initial set of inliers and the scale of inlier noise using a discrete set of elemental subset-based hypotheses. The model estimation step optimizes a robust objective function over a smaller set of hypotheses generated only from the initial inlier set. To further refine the estimate, the optimal hypotheses are used to initialize the conjugate gradient algorithm over the Grassmann manifold, which is a continuous space of subspace parameters. It remains an open question if, after the scale estimation step, the Grassmann manifold can directly be used to estimate the model and the correct inliers. This would require a reformulation of the problem definition.

The gpbM algorithm has been used in many applications to recover multiple structures by iteratively removing inlier points associated with each structure [13]. Without any prior knowledge about the scale of inlier noise, it is difficult for the robust estimation algorithm to recover more than a few structures automatically, which seems to be an easy task for the human vision system. We believe that the capabilities of automatic estimation algorithms can be significantly improved if top-down information is effectively incorporated into existing data-driven techniques.

Data analysis on manifolds involves higher computation to account for the nonEuclidean geometry. These problems have been somewhat mitigated with availability of parallel processing hardware and development of efficient new algorithms. For example, the Grassmannian Robust Adaptive Subspace Tracking Algorithm (GRASTA) [9] uses stochastic gradient descent on Grassmann manifolds to move along the geodesic with efficient rank one updates. GRASTA was applied to foreground/background segmentation in videos achieving a frame rate of about 46 fps. Such algorithmic advances have bridged the gap between the theoretical research and practical applications to large-scale computer vision problems that are of interest to researchers and practitioners alike.

The computational complexity of solutions may be higher for problems that incorporate the underlying Riemannian geometry, but it is important to avoid heuristics and ad hoc approximations incurred due to solutions designed in Euclidean space. There is only a small amount of research work that has used Riemannian geometry to make significant impact in computer vision applications. Computer vision researchers could benefit from a course on Riemannian geometry with a special focus on its applications to computer vision problems. We hope that this book will serve as a catalyst to push forward computer vision research in this direction in order to understand and exploit the vast mathematical infrastructure of Riemannian geometry.

References

1. Cetingul H, Vidal R (2009) Intrinsic mean shift for clustering on Stiefel and Grassmann manifolds. In: CVPR, pp 1896–1902
2. Christoudias CM, Georgescu B, Meer P (2002) Synergism in low level vision. In: ICPR, pp 150–155
3. Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. IEEE Trans Pattern Anal Mach Intell 24(5):603–619

4. Edelman A, Arias TA, Smith ST (1999) The geometry of algorithms with orthogonality constraints. SIAM J Matrix Anal Appl 20(2):303–353
5. Fischler MA, Bolles RC (1981) Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395
6. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge
7. Hartley R, Trumpf J, Dai Y, Li H (2013) Rotation averaging. Int J Comput Vis 103(3):267–305
8. Hauberg S, Feragen A, Black MJ (2014) Grassmann averages for scalable robust PCA. In: CVPR, pp 3810–3817
9. He J, Balzano L, Szlam A (2012) Incremental gradient on the Grassmannian for online foreground and background separation in subsampled video. In: CVPR, pp 1568–1575
10. Jain S, Govindu V (2013) Efficient higher-order clustering on the Grassmann manifold. In: ICCV, pp 3511–3518
11. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
12. Mittal S, Meer P (2012) Conjugate gradient on Grassmann manifolds for robust subspace estimation. Image Vis Comput 30(6–7):417–427
13. Mittal S, Anand S, Meer P (2012) Generalized projection-based M-estimator. IEEE Trans Pattern Anal Mach Intell 34(12):2351–2364
14. Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical recipes in C. The art of scientific computing, 2nd edn. Cambridge University Press, Cambridge
15. Raguram R, Frahm JM, Pollefeys M (2008) A comparative analysis of RANSAC techniques leading to adaptive real-time random sample consensus. In: ECCV, pp 500–513
16. Shrivastava A, Shekhar S, Patel VM (2014) Unsupervised domain adaptation using parallel transport on Grassmann manifold. In: WACV, pp 277–284
17. Subbarao R, Meer P (2006) Beyond RANSAC: user independent robust regression. In: IEEE CVPR Workshop on 25 years of RANSAC, p 101
18. Subbarao R, Meer P (2007) Discontinuity preserving filtering over analytic manifolds. In: CVPR, pp 1–6
19. Subbarao R, Meer P (2009) Nonlinear mean shift over Riemannian manifolds. Int J Comput Vis 84(1):1–20
20. Turaga P, Veeraraghavan A, Srivastava A, Chellappa R (2011) Statistical computations on Grassmann and Stiefel manifolds for image and video-based recognition. IEEE Trans Pattern Anal Mach Intell 33(11):2273–2286
21. Xu J, Ithapu V, Mukherjee L, Rehg J, Singh V (2013) GOSUS: Grassmannian online subspace updates with structured-sparsity. In: ICCV, pp 3376–3383
22. Yang L (2010) Riemannian median and its estimation. LMS J Comput Math 13:461–479

Chapter 7
Motion Averaging in 3D Reconstruction Problems

Venu Madhav Govindu

Abstract We consider the problem of recovering individual motions for a set of cameras when we are given a number of relative motion estimates between camera pairs. Typically, the number of such relative motion pairs exceeds the number of unknown camera motions, resulting in an overdetermined set of relationships that needs to be averaged. This problem occurs in a variety of contexts and in this chapter we consider sensor localization in 3D reconstruction problems where the camera motions belong to specific finite-dimensional Lie groups or motion groups. The resulting problem has a rich geometric structure that leads to efficient and accurate algorithms that are also robust to the presence of outliers. We develop the motion averaging framework and demonstrate its utility in 3D motion estimation using images or depth scans. A number of real-world examples exemplify the motion averaging principle as well as elucidate its advantages over conventional approaches for 3D reconstruction.

7.1 Introduction

The availability of an accurate 3D digital model of a scene or an object allows us to interpret the 3D representation as well as manipulate it in a variety of ways. This is of use in a diverse set of applications such as human–computer interaction, modeling and archiving of cultural objects, industrial inspection, measurement of change, and shape and deformation analysis. Such 3D models may be acquired or estimated either from raw 3D depth measurements provided by depth scanners or from 2D images from conventional RGB cameras. Throughout this chapter, we shall use the term "sensor" to denote either the depth scanner or RGB camera as the case may be. In the case of 3D scanner data, the problem is one of using many individual scans taken from different viewpoints to build a single, unified, dense representation of a scene or an object. In contrast, when using 2D images, the most common approach is to use matched corner-like feature points to estimate a sparse 3D point

V.M. Govindu (✉)
Department of Electrical Engineering, Indian Institute of Science, Bengaluru 560012, India
e-mail: venu@ee.iisc.ernet.in

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_7

cloud (structure). In both cases, recovering 3D scene structure also involves solving for the 3D position and orientation parameters (motion) of the sensors involved. The motion estimation problem in building 3D scan models is referred to as *3D registration* whereas the approach using camera images is known as *structure-from-motion* (henceforth SfM). In some other cases, e.g., SLAM approaches to robot navigation, we are primarily interested in recovering the motion or path of the sensor. In this chapter we will develop an approach to sensor motion estimation that factors out the dependence on 3D depth of the scene. As a result, the motion estimation problem becomes easier to solve. In the case of SfM, once motion is either partially or fully recovered, solving for 3D structure is a significantly easier task.

If we denote a point in 3D (X, Y, Z) by its homogeneous form as $\mathbf{P} = [X, Y, Z, 1]^T$ and its projection on an image as $\mathbf{P} = [u, v, 1]^T$, then we have

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R} \mid \mathbf{T}\right]\mathbf{P} \tag{7.1}$$

where $\mathbf{K}$ is a 3×3 upper-triangular matrix denoting the intrinsic calibration parameters of the camera and $\mathbf{R}$ and $\mathbf{T}$ are the 3D rotation and translation of the camera with respect to a given frame of reference. Given enough image projections of the 3D points in different cameras [lhs of Eq. (7.1)], SfM methods recover the camera intrinsic calibration parameters, 3D motions and the 3D points, i.e., all unknowns in the rhs of Eq. (7.1). The well-known approach of bundle adjustment [25] performs a least squares minimization of the reprojection error, i.e., distance between observed image feature point locations (u, v) and their positions as estimated from the rhs of Eq. (7.1). A large number of advances in feature matching, efficient computation, and effective heuristics has made it possible to solve for the SfM problem at a large scale of many thousands of images covering areas as large as a city block [1, 24, 30].

In the case of 3D scans, we can directly observe the 3D locations of points. However, since each scan covers only a part of a scene or object, to build a unified 3D representation, we need to register them, i.e., place them in a common frame of reference. As each scan is a representation of a part of the scene defined in an independent frame of reference, we recognize that the relative motion between a pair of scans is a 3D rotation and translation, i.e., a rigid 3D Euclidean transformation. Therefore, for the same 3D point represented in two independent scans as homogeneous points $\mathbf{P}$ and $\mathbf{P}^{'}$, we have

$$\mathbf{P}^{'} = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{T} \\ \hline \mathbf{0} & 1 \end{array}\right]\mathbf{P} = \mathbf{M}\mathbf{P} \tag{7.2}$$

where $\mathbf{M}$ is a 4×4 matrix denoting a rigid 3D Euclidean transformation. Typically, we are given scans but not the correspondences, i.e., knowledge of the matching point pairs $(\mathbf{P}, \mathbf{P}^{'})$ is not available a priori and has to be estimated from the data themselves. The Iterative Closest Point (henceforth ICP) method alternately solves for the optimal 3D Euclidean transformation $\mathbf{M}$ given correspondences (motion step) and reestimates correspondences $(\mathbf{P}, \mathbf{P}^{'})$ given the motions (correspondence step). These steps are iterated till convergence. The ICP is the most well-known method for registration of a pair of 3D scans, and a variety of heuristics and improvements [23] make it feasible to use this method in practical situations.

Although the bundle adjustment approach to SfM and ICP is effective in solving their respective problems, both approaches suffer from some significant drawbacks. Bundle adjustment is a nonlinear minimization over a large number of unknowns and hence it needs a good initial guess for convergence. Moreover it is also computationally very expensive, often prohibitively so. As a result, all effective practical implementations need to incorporate a variety of heuristics that incrementally and robustly grow the solution by adding one camera at a time to the solution. Nevertheless, bundle adjustment-based approaches to SfM are computationally very demanding and may fail to work on occasion. In the case of scan registration, the problem lies in the fact that ICP is a greedy method that would fail to converge correctly if the two scans being registered have a large initial motion between them. This problem is further exacerbated in the case of currently popular depth cameras such as the Kinect which provide depth maps that are quite noisy.

In this chapter we develop an alternate framework based on the idea of *motion averaging* that addresses the above-mentioned drawbacks of bundle adjustment and the ICP algorithm. In both scenarios of 3D reconstruction using cameras or scans, we are interested in solving for rigid 3D rotations or Euclidean motions of the sensors involved.

The approach of motion averaging is based on the observation that we can estimate the global motion by considering the relative motions between pairs of sensors that can be estimated from the given data. Crucially, estimating the relative motion between a pair of sensors is often computationally inexpensive. Consider

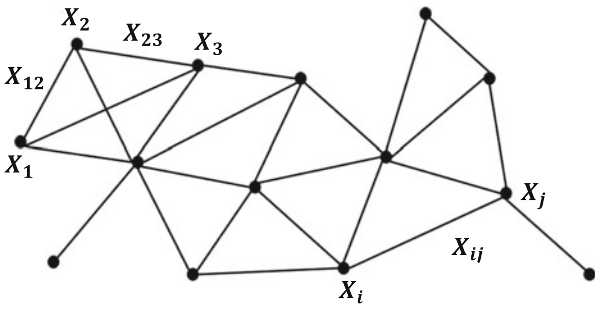


Fig. 7.1 A schematic graph representing the relationships available between different sensors. Each vertex in the graph denotes a sensor and the presence of an edge between two vertices implies that we can estimate the relative motion between the two sensors. The motion averaging problem is equivalent to determining the values of the vertices given the values on the edges, i.e., differences between vertices

an illustrative graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ shown in Fig. 7.1 where each vertex in $\mathcal{V}$ denotes a sensor and the presence of an edge between two vertices i and j denotes that we can solve for the relative motion between them. The problem on hand in our case is solving for the global motion (value at each vertex) given estimates of the relative motions between them (edge values). If we denote the motion of the ith sensor as $\mathbf{X}_i$ with respect to a given frame of reference, then we can denote our desired global solution as $\mathbf{X}_g = \{\mathbf{X}_1, \cdots, \mathbf{X}_N\}$ where N is the number of sensors involved. We note that in our case $\mathbf{X}$ may either denote the 3×3 rigid 3D rotation ($\mathbf{R}$) or the 4×4 3D Euclidean motion ($\mathbf{M}$) as specified in Eq. (7.2). However, the general principle holds true for all finite-dimensional Lie groups or matrix groups. Noting that the relative motion between the ith and jth vertex can be written in terms of the difference of motions between the global frame of reference and the two vertices, we have the fundamental relationship

$$\begin{aligned} \mathbf{X}_{ij} &= \mathbf{X}_j \mathbf{X}_i^{-1} \\ \Rightarrow \overbrace{\left[\cdots \mathbf{X}_{ij} \cdots -\mathbf{I} \cdots \right]}^{\text{Relative Motions}} & \underbrace{\begin{bmatrix} \vdots \\ \mathbf{X}_i \\ \vdots \\ \mathbf{X}_j \\ \vdots \end{bmatrix}}_{\mathbf{X}_g:\ \text{Global Motions}} = \mathbf{0} \end{aligned} \tag{7.3}$$

The motion averaging problem is one of estimating $\mathbf{X}_g$ given enough observations $\{\mathbf{X}_{ij} | \forall (i, j) \in \mathcal{E}\}$ of the form in Eq. (7.3) and was developed in [5, 9–12]. It will be immediately noted that we can arbitrarily choose the origin and a basis as our frame of reference to represent the observed relative motions. Typically, we remove this *gauge freedom* by fixing the origin and a basis to one of the vertices. It will also be noted that a solution for global motion $\mathbf{X}_g$ exists as long as we have a spanning tree present in the graph $\mathcal{G}$. For N vertices, while $\mathbf{X}_g$ can be specified by $(N - 1)$ motions, in a graph with $|\mathcal{V}| = N$ vertices we can have as many as ${}^N C_2 = \frac{N(N-1)}{2}$ relative motion observations on edges. In general we may have $|\mathcal{E}| > (N - 1)$ observations resulting in a redundant set of observations. Due to the presence of noise in our estimated $\mathbf{X}_{ij}$, these individual observations will not be consistent with each other, i.e., we may not have a solution $\mathbf{X}_g$ that exactly satisfies Eq. (7.3) for all $\mathbf{X}_{ij}$. In such a case, we seek the best possible solution that is most consistent with Eq. (7.3), i.e. we "average" the relative motion observations to solve for the global motion $\mathbf{X}_g$.

Such an approach of motion averaging has several advantages. First, by averaging out the noise or error in individual relative motion observations $\mathbf{X}_{ij}$ we arrive at a more accurate solution than is possible by considering only a minimal spanning tree. This is particularly true if we have loops in the viewgraph $\mathcal{G}$. In the minimal solution case, the error in $\mathbf{X}_i$ grows when we traverse the graph as the errors in individual edges add up. However, in the motion averaging scenario since we seek to be as consistent as possible with respect to all the relative motion observations, the errors will be evenly distributed over all edges resulting in greatly reduced drift of the solution. Additionally, as we shall see later in this chapter, motion averaging is highly efficient and can be effectively made robust to outliers in estimates $\mathbf{X}_{ij}$. This makes the approach of motion averaging attractive in both of our above-mentioned scenarios of SfM and 3D scan registration. We also remark here that while our motivating problems pertain to 3D reconstruction, the idea of averaging relative measurements to solve for global representations has a far wider range of applicability. It will be easily recognized that the problem of specifying values at vertices of a graph given edge measurements occurs in many other contexts including wireless and sensor localization problems, SLAM in robotics and the classic problem of multidimensional scaling.

By concatenating the relationships for each edge in the form of Eq. (7.3) we obtain a system of equations of the form $\mathbf{A}\mathbf{X}_g = \mathbf{0}$. If the motion matrices in $\mathbf{X}_g$ were elements of a vector space, then a solution can be obtained by solving the over-determined linear system of equations $\mathbf{A}\mathbf{X}_g \approx \mathbf{0}$. However, our motion models are not elements of a vector space representation, rather they are elements of nonlinear manifolds. Take for instance 3D rotation matrices $\mathbf{R}$. While $\mathbf{R}$ has nine entries, these individual entries need to satisfy six independent constraints specified by the orthonormality property of rotation matrices, i.e., $\mathbf{R}\mathbf{R}^T = \mathbf{I}$ where $\mathbf{I}$ is a 3×3 identity matrix. Therefore, while $\mathbf{R}$ is a 3×3 matrix it has only three degrees of freedom. Similarly, the 4×4 3D Euclidean motion model $\mathbf{M}$ has only six degrees of freedom. Consequently, in solving the motion averaging problem represented by the system of equations in Eq. (7.3), we need to ensure that the individual estimated motion matrices in $\mathbf{X}_g$ satisfy the underlying geometric properties of the corresponding motion model. Clearly, the linear solution of Eq. (7.3) will not result in a solution of $\mathbf{X}_g$ that will satisfy the requisite constraints. Both the 3D rotation and Euclidean motion models belong to a special class of nonlinear Riemannian manifolds, i.e., finite-dimensional Lie groups. To be able to develop our solution for averaging of relative motions on Lie groups, we now very briefly describe the relevant properties of Lie groups.

7.2 Lie Groups

In this section we provide an elementary introduction to Lie groups that will be limited to concepts of relevance to our problem. The literature on Lie groups is very extensive and an excellent treatment in the context of computer vision is provided in [16]. Readers may also consult [3, 28] for more general introductions. Throughout, we shall restrict ourselves to the class of square matrices that form groups. A group G is a set whose elements $(X, Y, Z \in G)$ satisfy the relations

$$X \circ Y \in G \text{ (closure)}$$

$$X \circ (Y \circ Z) = (X \circ Y) \circ Z \text{ (associativity)}$$
$$\exists E \in G \ni X \circ E = E \circ X = X \text{ (identity)}$$
$$\exists X^{-1} \in G \ni X \circ X^{-1} = X^{-1} \circ X = E \text{ (inverse)}$$

In addition to being a group, a Lie group is also a smooth, differentiable manifold. For Lie groups, the operations $X \times Y \mapsto XY$ and $X \mapsto X^{-1}$ are differentiable mappings. Viewed locally, a Lie group has the topological structure of vector space $\mathbb{R}^n$. This vector space is also equipped with a bilinear operation $[.,.] : \mathfrak{m} \times \mathfrak{m} \to \mathfrak{m}$ known as the Lie bracket. For matrix groups, the bracket is the commutator, i.e., $[\mathbf{x}, \mathbf{y}] = \mathbf{xy} - \mathbf{yx}$. It is easy to verify in this case that the bracket satisfies the relationships of $[\mathbf{x}, \mathbf{y}] = -[\mathbf{y}, \mathbf{x}]$ (antisymmetry) and $[\mathbf{x}, [\mathbf{y}, \mathbf{z}]] + [\mathbf{y}, [\mathbf{z}, \mathbf{x}]] + [\mathbf{z}, [\mathbf{x}, \mathbf{y}]] = \mathbf{0}$ (Jacobi identity). The tangent space about a point on G is known as the Lie algebra and is denoted as $\mathfrak{g}$. While we may locally describe a manifold by its tangent space, for Lie groups moving along a direction in $\mathfrak{g}$ is equivalent to moving along a one-dimensional subgroup of G (homomorphism from Lie algebra to Lie group).

The crucial relationship between a point on the Lie algebra and its mapping onto the Lie group is given by the exponential mapping. We can develop an intuition for this relationship by considering the simple case of rotations in the two-dimensional plane. Let a point $(\mathbf{x}, \mathbf{y})$ be rotated about the origin by an infinitesimal angle $\delta\theta$ to a new position $(\mathbf{x}', \mathbf{y}')$, then we have

$$\begin{bmatrix} \mathbf{x}' \\ \mathbf{y}' \end{bmatrix} = \begin{bmatrix} \cos\delta\theta & -\sin\delta\theta \\ \sin\delta\theta & \cos\delta\theta \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = (\mathbf{I} + \delta\theta \underbrace{\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}}_{\mathbf{B}}) \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \tag{7.4}$$

Now if we divide the rotation angle θ into a sequence of n rotations of size $\delta\theta = \frac{\theta}{n}$, then in the limit $n \to \infty$, the net transformation of $(I + \frac{\theta}{n}\mathbf{B})^n$ becomes $\mathbf{R} = \exp(\theta\mathbf{B})$ where $\mathbf{B}$ spans the one-dimensional Lie algebra for two-dimensional rotations. All two-dimensional rotations form a group denoted as $\mathbb{SO}(2)$, i.e., Special Orthogonal group of dimension 2 with a corresponding Lie algebra denoted as $\mathfrak{so}(2)$ which happens to be spanned by the matrix $\mathbf{B}$ as specified above. In the case of three-dimensional rotations $\mathbf{R} \in \mathbb{SO}(3)$ that is of special interest for us, we can see that since 3D rotations have 3 degrees of freedom, the corresponding Lie algebra $\mathfrak{so}(3)$ has to be three dimensional. If we use the conventional axis-angle representation of rotating by angle θ about an axis $\mathbf{n}$, denoting $\boldsymbol{\omega} = \theta\mathbf{n}$, the Lie algebra is given as $[\boldsymbol{\omega}]_\times \in \mathfrak{so}(3)$ where $[\mathbf{x}]_\times$ is the skew-symmetric form of vector $\mathbf{x}$. The corresponding 3D rotation is specified as $\mathbf{R} = \exp([\boldsymbol{\omega}]_\times) \in \mathbb{SO}(3)$. Analogously, the mapping from the Lie group to the Lie algebra is the logarithm operation, i.e., $[\boldsymbol{\omega}]_\times = \log(\mathbf{R})$. Similarly, all 3D rigid transformations form a group known as the Special Euclidean group denoted as $\mathbb{SE}(3)$ with a corresponding Lie algebra denoted as $\mathfrak{se}(3)$.

Unlike rotations in two dimensions, 3D rotations $\mathbf{R} \in \mathbb{SO}(3)$ form a noncommutative Lie group; therefore, in general $\mathbf{R}_1\mathbf{R}_2 \neq \mathbf{R}_2\mathbf{R}_1$, i.e., $e^{[\omega_1]_\times} e^{[\omega_2]_\times} \neq e^{[\omega_1+\omega_2]_\times}$. Instead, the equivalent mapping is specified by the BCH form $e^{\mathbf{x}}e^{\mathbf{y}} = e^{\mathrm{BCH}(x,y)}$, where BCH(.,.) is the Baker–Campbell–Hausdorff series [28] given by

$$\mathrm{BCH}(\mathbf{x},\mathbf{y}) = \mathbf{x} + \mathbf{y} + \frac{1}{2}[\mathbf{x},\mathbf{y}] + \frac{1}{12}[\mathbf{x}-\mathbf{y},[\mathbf{x},\mathbf{y}]] + \mathcal{O}(|(\mathbf{x},\mathbf{y})|^4) \tag{7.5}$$

Before proceeding further, we note a few aspects of $\mathbb{SO}(3)$ and $\mathbb{SE}(3)$ here. First, we remark that while we have only considered the representation of 3D rotations as elements of the Lie group $\mathbb{SO}(3)$ there are other representations such as quaternions that are commonly used. The reader is referred to [16] for a detailed discussion of various representations of 3D rotations and their equivalent relationships and [15] for a survey of averaging methods on $\mathbb{SO}(3)$. It should also be noted that for the $\mathbb{SO}(3)$ group, the BCH series of Eq. (7.5) has a closed-form expression. Second, in the case of 3D Euclidean transformations where we denote the elements of the Lie group and Lie algebra as $\mathbf{M} \in \mathbb{SE}(3)$ and $\mathfrak{m} \in \mathfrak{se}(3)$, respectively, the following relationships hold:

$$\mathbf{M} = \exp(\mathfrak{m}) = \sum_{k=0}^{\infty} \frac{\mathfrak{m}^k}{k!} = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \\ \hline 0 & 1 \end{array}\right] \text{ where } \mathfrak{m} = \left[\begin{array}{c|c} \boldsymbol{\Omega} & \mathbf{u} \\ \hline 0 & 0 \end{array}\right] \tag{7.6}$$

$$\text{i.e. } \mathbf{R} = \exp(\boldsymbol{\Omega}) \text{ and } \mathbf{t} = \mathbf{P}\mathbf{u}$$

$$\text{where } \mathbf{P} = \mathbf{I} + \frac{(1-\cos\theta)}{\theta^2}\boldsymbol{\Omega} + \frac{(\theta - \sin\theta)}{\theta^3}\boldsymbol{\Omega}^2 \text{ and } \theta = \sqrt{\frac{1}{2}tr(\boldsymbol{\Omega}^T\boldsymbol{\Omega})} \tag{7.7}$$

An important property of Lie groups that we will use in our subsequent discussions is the notion of a distance between elements of the group. We cannot develop this idea fully here but will only provide an elementary sketch of an argument. By appropriately integrating along paths, we can arrive at the length of any arbitrary path between two points on a Riemannian manifold. When defined, the shortest of all such paths is known as the *geodesic* distance and defines the *intrinsic* distance between any two points on a manifold. In general, measuring such geodesic distances can be a non-trivial problem. However, given its special structure, we can easily measure the intrinsic distance between two points on a matrix or Lie group. Let us consider two points $\mathbf{X}$ and $\mathbf{Y}$ that belong to a Lie group G. Further let the distance between them be given by $d(\mathbf{X},\mathbf{Y})$. The distance function $d(.,.)$ is left-invariant if $d(\mathbf{PX},\mathbf{PY}) = d(\mathbf{X},\mathbf{Y}), \forall \mathbf{P} \in G$. Similarly, $d(.,.)$ is right-invariant if $d(\mathbf{XP},\mathbf{YP}) = d(\mathbf{X},\mathbf{Y}), \forall \mathbf{P} \in G$ [32]. A metric is bi-invariant if it is both left- and right-invariant. While the metric on $\mathbb{SO}(3)$ is bi-invariant, the metric on $\mathbb{SE}(3)$ is left-invariant. Assuming $d(.,.)$ is left-invariant, we can see that

$d(\mathbf{X},\mathbf{Y}) = d(\mathbf{I},\mathbf{X}^{-1}\mathbf{Y})$ where $\mathbf{I}$ is the identity element of G. Therefore, we can now measure the requisite distance as the distance required to move from the identity element $\mathbf{I}$ along a one-dimensional subgroup to reach $\mathbf{X}^{-1}\mathbf{Y}$. Using the BCH form of Eq. (7.5), we can now recognize that

$$d(\mathbf{X},\mathbf{Y}) = d(\mathbf{I},\mathbf{X}^{-1}\mathbf{Y}) = ||\log(\mathbf{X}^{-1}\mathbf{Y})|| \tag{7.8}$$

7.3 Averaging on a Group

Given the above definition of distances on a Lie group, we can now derive a method to solve for the average of many elements of a Lie group. Consider a set of observations $\{\mathbf{X}_1,\cdots,\mathbf{X}_N\}$ and, for the moment, ignore their group structure and assume them to be members of a vector space $\mathbb{R}^n$. When we treat observations as being embedded in the *extrinsic* vector space $\mathbb{R}^n$, it is easy to see that their average is given by the arithmetic mean, i.e., $\overline{\mathbf{X}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{X}_i$. In general, this *extrinsic* average violates the geometric constraints of the underlying manifold, i.e., $\overline{\mathbf{X}} \notin G$. A simple remedy is to treat this violation as being due to a perturbation that moves the average away from the manifold. Hence a valid average is obtained by projecting the arithmetic average onto the closest point on the corresponding group G. Instances of this approach are the eight-point algorithm [13] and the SVD method to project a matrix to its closest point on $\mathbb{SO}(3)$ [17].

An estimate that implicitly satisfies the geometric constraints and lies on the manifold G is known as the *intrinsic* average. We can motivate an approach to derive the intrinsic average by interpreting the vector space average as the "variational minimizer," i.e., $\overline{\mathbf{X}}$ minimizes the cost function $\sum_{i=1}^{N} d^2(\mathbf{X}_i,\overline{\mathbf{X}}) = \sum_{i=1}^{N}||\mathbf{X}_i-\overline{\mathbf{X}}||^2$, i.e., sum of squared distances of points $\mathbf{X}_i$ from the mean $\overline{\mathbf{X}}$. Generalizing this notion we have the intrinsic average on Riemannian manifolds [18] given by

$$\arg\min_{\overline{\mathbf{X}}\in G}\sum_{k=1}^{N} d^2(\mathbf{X}_k,\overline{\mathbf{X}}) \tag{7.9}$$

Although the intrinsic average is preferable since it satisfies the underlying geometric constraints of G, it is often difficult to estimate for Riemannian manifolds. In the special case of finite-dimensional Lie groups or matrix groups, the intrinsic average can be computed efficiently. To motivate the solution, let us consider the vector space average that minimizes the cost function $C = \sum_{i=1}^{N}||\mathbf{X}_i-\overline{\mathbf{X}}||^2$. Suppose we want to solve for $\overline{\mathbf{X}}$ by considering C as a cost function to be minimized using gradient descent. Let the current estimate of the average be $\boldsymbol{\mu}$, then the gradient step is given by

$$\begin{aligned} C &= \sum_{i=1}^{N} ||\mathbf{X}_i - \boldsymbol{\mu}||^2 \\ \Rightarrow \frac{\partial C}{\partial \boldsymbol{\mu}} &= -2\sum_{i=1}^{N}(\mathbf{X}_i - \boldsymbol{\mu}) \\ \Rightarrow \boldsymbol{\mu} &\leftarrow \boldsymbol{\mu} + \lambda \sum_{i=1}^{N}(\mathbf{X}_i - \boldsymbol{\mu}) \text{ Update step} \end{aligned} \tag{7.10}$$

In Eq. (7.10) we can see that the contribution of each observation $\mathbf{X}_i$ towards the update step is in proportion to its distance from the current estimate. Obviously, since the vector space $\mathbb{R}^n$ is curvature free, we can make Eq. (7.10) converge in one step if we set $\lambda = \frac{1}{N}$. Now, recall that for a Lie group G, the distance is $d(\mathbf{X}, \mathbf{Y}) = ||\log(\mathbf{X}^{-1}\mathbf{Y})||$, which using the BCH form we can approximate as $d(\mathbf{X}, \mathbf{Y}) \approx ||\log(\mathbf{X}) - \log(\mathbf{Y})|| = ||\mathbf{x} - \mathbf{y}||$. Thus, we have replaced the true Riemannian distance by a “Euclidean” distance in the tangent space or Lie algebra $\mathfrak{g}$. Given the sample average in the Lie algebra $\mathfrak{g}$ we can see that the first-order approximation of distance suggests that we move in the tangent space by $\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$ and therefore the matrix update is equivalently $\Delta\boldsymbol{\mu} = \exp(\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i)$. Note that while we have approximated the true Riemannian distance on the group G, nevertheless our update matrix is geometrically valid, i.e., $\Delta\boldsymbol{\mu} \in G$. Now, as in the case of Eq. (7.10) for the vector space average, we can repeatedly update our estimate of the mean with one difference. Unlike the vector space, at every step the tangent space has to be redefined about the current estimate of $\boldsymbol{\mu}$, for the first-order approximation of the Riemannian distance to be valid. In other words, to account for the curved nature of our space G, we need to recenter our “origin” at every iteration and represent the observations $\mathbf{X}_i$ with respect to this new coordinate system. The resultant algorithm for averaging on a Lie group is given in Algorithm 1.

The approach in Algorithm 1 is similar in spirit to the gradient-descent step specified in Eq. (7.10) for vector space averaging. It has been shown in [20] that as long as the observations $\mathbf{X}_i$ lie within an open ball of a specified radius, this iterative

Algorithm 1 Lie algebraic averaging

```
Input: ${\mathbf{X}_1, \cdots, \mathbf{X}_N} \in G$ ($N$ observations)
Output: $\boldsymbol{\mu} \in G$ (Intrinsic Average)
Initialization: $\boldsymbol{\mu}$ to an initial guess and $\Delta\boldsymbol{\mu}$ to a large value

while $||\Delta\boldsymbol{\mu}|| > \epsilon$ do
    1. $\Delta X_i = \boldsymbol{\mu}^{-1}\mathbf{X}_i$
    2. $\Delta\mathbf{x}_i = \log(\Delta\mathbf{X}_i)$
    3. $\Delta\boldsymbol{\mu} = \exp(\frac{1}{N}\sum_{i=1}^{N}\Delta\mathbf{x}_i)$
    4. $\boldsymbol{\mu} = \boldsymbol{\mu}\Delta\boldsymbol{\mu}$
end while
```

approach is guaranteed to converge. In the case of $\mathbf{X}_i \in \mathbb{SO}(3)$, this radius which is related to the sectional curvature of the manifold is equal to $\frac{\pi}{2}$. We also note here that we could change our update step by considering higher-order terms in the BCH expression or by considering second-order methods [19] which may result in faster convergence. However, the choice of the order of approximation of distance in the Lie algebra does not change the condition for convergence. In practice, for problems in low-dimensional manifolds such as $\mathbb{SO}(3)$ and $\mathbb{SE}(3)$, Algorithm 1 which is a first-order gradient descent-type method converges quite fast.

7.4 Averaging of Relative Motions

Using the idea of averaging on a Lie group as detailed above in Sect. 7.3, we can now develop our approach for solving the problem specified by Eq. (7.3), i.e., averaging of relative motions. Analogous to the fitting cost defined in Eq. (7.9), we can define the problem of averaging relative motions as one of solving

$$\arg \min_{\mathbf{X}_g} \sum_{(i,j)\in\mathcal{E}} d^2(\mathbf{X}_{ij}, \mathbf{X}_j\mathbf{X}_i^{-1}) \tag{7.11}$$

In other words, we seek to find a global motion estimate $\mathbf{X}_g$ that is most consistent with the individual relative motion observations on all edges of the viewgraph $\mathcal{E}$. If we consider a single observation relationship $\mathbf{X}_{ij} = \mathbf{X}_j\mathbf{X}_i^{-1}$ representing a single edge, we can use a first-order approximation of the corresponding Lie algebraic relationship as $\mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i$. It will be noted that while the Lie algebraic representation $\mathbf{x}_{ij}$ is a matrix, for our purposes the equation $\mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ signifies a vectorial relationship. For example, in the case of $\mathbb{SO}(2)$, the Lie algebraic form is a 2×2 matrix, i.e., the relationship $\mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ is given $\theta_{ij}\mathbf{B} = \theta_j\mathbf{B} - \theta_i\mathbf{B}$ where $\mathbf{B}$ is as given in Eq. (7.4). However, since here $\mathbf{x}$ has only one degree of freedom, the equivalent relationship is a scalar one, i.e., $\theta_{ij} = \theta_j - \theta_i$. Similarly, for $\mathbb{SO}(3)$ and $\mathbb{SE}(3)$, the equivalent forms for $\mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ are relationships for vectors of size 3 and 6, respectively. In the following, for compactness of notation, we will denote this vector relationship as $\mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i$ where it is understood that the matrix form of the relationship in the Lie algebra is reduced to its equivalent vectorial representation.

If we denote the collection of all the Lie algebraic representations of the global motion model as $\mathbf{x}_g = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$, then the cost function in Eq. (7.11) translates to the system of equations

$$\mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i = \underbrace{\left[\cdots - \mathbf{I} \cdots \mathbf{I} \cdots\right]}_{\mathbf{A}_{ij}} \mathbf{x}_g \tag{7.12}$$

where in $\mathbf{A}_{ij}$, $\mathbf{I}$ and $-\mathbf{I}$ are placed as $d \times d$ blocks in the appropriate locations of j and i, respectively, and d is the number of degrees of freedom for the Lie group, e.g.,

Algorithm 2 Lie algebraic relative motion averaging

```
Input: $\{\mathbf{X}_{ij1}, \cdots, \mathbf{X}_{ijk}\}$ ($|\mathcal{E}|$ relative motions)
Output: $\mathbf{X}_g = \{\mathbf{X}_1, \cdots, \mathbf{X}_N\}$ ($|\mathcal{V}|$ absolute motions)
Initialization: $\mathbf{X}_g$ to an initial guess
    while $||\Delta\mathbf{x}_{rel}|| > \epsilon$ do
        1. $\Delta\mathbf{X}_{ij} = \mathbf{X}_j^{-1}\mathbf{X}_{ij}\mathbf{X}_i$
        2. $\Delta\mathbf{x}_{ij} = \log(\Delta\mathbf{X}_{ij})$
        3. Solve $\mathbf{A}\Delta\mathbf{x}_g = \Delta\mathbf{x}_{rel}$
        4. $\forall k \in [1, N], \mathbf{X}_k = \mathbf{X}_k \exp(\Delta\mathbf{x}_k)$
    end while
```

$d = 3$ for $\mathbb{SO}(3)$. Collecting all the relationships of the set of edges $\mathcal{E}$, we have

$$\mathbf{A}\mathbf{x}_g = \mathbf{x}_{\text{rel}} \tag{7.13}$$

where $\mathbf{A}$ represents the concatenation of all individual $\mathbf{A}_{ij}$, and $\mathbf{x}_{\text{rel}}$ is the vector concatenation of the Lie algebraic representations of the observed relative motions, i.e., $\mathbf{x}_{ij}$. Although in this chapter every relative motion relationship is treated equally, if we have reliable information of the relative significance of the relative motion observation on each edge, we can incorporate it in Eq. (7.13) as an appropriate weight. The result is an approach for averaging relative motions as detailed in Algorithm 2.

We can interpret Algorithm 2 in a manner analogous to that of Algorithm 1. Given a current estimate for $\mathbf{X}_g = \{\mathbf{X}_1, \cdots, \mathbf{X}_N\}$, we wish to develop the tangent space representation about each $\mathbf{X}_i$ in $\mathbf{X}_g$. Resultantly, the Lie algebraic representation of each observation developed about the current $\mathbf{X}_i$ is estimated in steps 1 and 2 of the algorithm. As a result, using a first-order approximation the residual or "unexplained" component $\Delta\mathbf{X}_{ij}$ can be seen to yield the system of equations $\mathbf{A}\Delta\mathbf{x}_g = \Delta\mathbf{x}_{\text{rel}}$, i.e., of the form of Eq. (7.13). Solving this system of equations in step 3 yields the requisite Lie algebraic update. In step 4 the update is carried out in the Lie group representation to yield the new estimate for $\mathbf{X}_g$. This process can be repeated till convergence.

It will be noted that as in the previous case, even though we are using a first-order approximation, since we are using the correct exponential mapping in step 4 in Algorithm 2, at all points the estimate for $\mathbf{X}_g$ is geometrically valid. In other words, ours is an *intrinsic* approach to averaging relative motions. It will also be noted that in step 3 of Algorithm 2, we have not given a specific method of solving for the system of equations. We have deliberately represented this step as a generic solution since, in essence, different variants of motion averaging differ only in the manner in which this step of averaging the residual relative motions in the tangent space is carried out.

7.5 Efficiency and Robustness of Averaging

In Algorithm 2, the specific solution for the system of equations $\mathbf{A}\Delta\mathbf{x}_g = \Delta\mathbf{x}_{\text{rel}}$ in step 3 plays a crucial role in determining the overall behavior. In [10], this system of equations was solved in a least-squares sense, resulting in step 3 of Algorithm 2 being given as the closed-form solution $\Delta\mathbf{x}_g = \mathbf{A}^{\dagger}\Delta\mathbf{x}_{\text{rel}}$ where $\mathbf{A}^{\dagger}$ is the pseudo-inverse of $\mathbf{A}$. It will be recognized that $\mathbf{A}$ is completely determined by the specific set of edges being solved for. As a result, it is fixed during the iterations of Algorithm 2 implying that for a given graph $\mathbf{A}^{\dagger}$ needs to be computed only once. For most SfM or 3D registration problems, the number of camera–camera pairs (equivalently scan–scan pairs) with common features tends to be small compared to the maximum possible value of ${}^{N}C_2 = \frac{N(N-1)}{2}$ pairs. Moreover, from Eq. (7.12), we can see that each row of $\mathbf{A}$ has only two nonzero entries of ± 1. This sparse structure of $\mathbf{A}$ with entries of ± 1 has significant computational implications for motion averaging, especially for large-sized problems. First, explicitly computing $\mathbf{A}^{\dagger}$ may be expensive for large problems and also result in a large dense matrix form. However, since $\mathbf{A}$ encodes a sparse graph, we can solve step 3 of Algorithm 2 efficiently by equivalently minimizing the cost function $||\mathbf{A}\Delta\mathbf{x}_g - \Delta\mathbf{x}_{\text{rel}}||^2$. Second, since $\mathbf{A}$ has nonzero entries of only ± 1, we note that during minimization any floating-point multiplications of the form $\mathbf{Ay}$ can be computed using only addition operations, thereby making the overall algorithm very efficient.

While solving the Lie algebraic averaging step of Eq. (7.13) as a nonlinear minimization is efficient, one drawback of this least-squares approach is its lack of robustness to outliers. It will be recalled the each individual relative motion $\mathbf{X}_{ij}$ which is an input to Algorithm 2 has to be estimated from either image or scan pairs. It is often the case that some of these estimated $\mathbf{X}_{ij}$ are erroneous. For instance in the case of SfM in urban areas, since many urban facades such as windows and doors are repetitive in nature, image feature matching methods may result in incorrect matches. This will in turn result in erroneous estimates of $\mathbf{X}_{ij}$ for the corresponding image pairs. If we solve for step 3 in Algorithm 2 using an ℓ_2 cost minimization as detailed above, the erroneous $\mathbf{X}_{ij}$ will corrupt the final estimate of $\mathbf{X}_g$. Hence, we need to solve for the system of equations in Eq. (7.13) in a manner that is both efficient and robust to the presence of outliers.

The approaches to robustness in averaging relative motions can be classified into two broad categories. In the first category, different methods are employed to identify erroneous or outlier edges and remove them from the set of observations. Subsequently, averaging is carried out in the ℓ_2 sense detailed above. In [11], a RANSAC approach [8] is adopted to identify the erroneous edges. From our discussion in Sect. 7.1, it will be recognized that ideally a loop of transformations should result in a net transformation equal to the identity element of the group. If a loop contains one or more outliers, then this constraint will be violated. By accumulating the statistics across many such loops in a graph, one can "disentangle" the outlier edges from the set of cleaner observations $\mathbf{X}_{ij}$. This problem of classifying the observations into inliers and outliers is solved in [21, 31] using loopy belief propagation. It may be noted that this approach is computationally expensive especially with increasing size of the viewgraph $\mathcal{G}$ and therefore not recommended.

The second approach to robustness is to solve the averaging problem using all the observations without requiring the explicit detection of outliers. These approaches rely on the intrinsic robustness of estimation methods used to suppress the effect of outlier edges. In [7], the problem of robust estimation is addressed by discretizing the solution space into a set of discrete labels. Subsequently, the solution for a robust version of averaging relative motions is obtained using discrete loopy belief propagation. This results in a discretized estimate for entries in $\mathbf{X}_g$ which are subsequently refined using nonlinear optimization, hence the name discrete-continuous optimization (DISCO) for the work in [7]. As in the case of [31], this approach is also cumbersome and computationally demanding. Another method in this category of robust estimation is [14] that utilizes the fact that the ℓ_1 norm is more robust to the presence of outliers compared to the ℓ_2 norm. Therefore, one may achieve a robust estimate by considering the ℓ_1 analogue of Eq. (7.11), i.e., $\sum_{(i,j)\in\mathcal{E}} d(\mathbf{X}_{ij}\mathbf{X}_i, \mathbf{X}_j)$. In [14], the authors utilize the fact that the Weiszfeld iteration can be used to find the ℓ_1 average of vector space observations. In their case, step 3 of Algorithm 2 is solved by a nested iteration. If we consider an individual vertex $\mathbf{X}_j$, then each edge connected to vertex j implies an estimate for it, i.e., $\mathbf{X}_j = \mathbf{X}_{ij}\mathbf{X}_i$. Hence, in the inner loop of [14], the ℓ_1 update for an individual vertex $\mathbf{X}_j$ is obtained as the Weiszfeld average of the set of individual edge estimates $\{\mathbf{X}_{ij}\mathbf{X}_i | \forall i \in \mathcal{N}(j)\}$ where $\mathcal{N}(j)$ is the set of vertices connected to vertex j. In the outer loop, this update is applied to all vertices in turn. In other words, the Weiszfeld method of [14] solves for the robust averaging of relative motions by repeatedly updating the estimates for individual vertices in turn. It can be seen that each vertex is updated based only on information provided by edges connected to it and the current estimate of its neighboring vertices, i.e., the Weiszfeld approach for motion averaging falls into the class of *distributed consensus* algorithms [26]. As the Weiszfeld approach of [14] is a distributed algorithm, its convergence rate is slow, especially for large graphs. Similarly, an interpretation of the work of [2] that pertains to 3D scan registration as a distributed consensus algorithm is provided in [12].

Instead of a distributed approach, in [5] a batch method is presented for robust and efficient averaging of relative motions. This approach is identical to that of Algorithm 2 where step 3 is specified by a robust approach to solving Eq. (7.13). The ℓ_1 equivalent here has the form $||\mathbf{A}\Delta\mathbf{x}_g - \Delta\mathbf{x}_{\text{rel}}||_{\ell_1}$. To account for the presence of outliers, we can model the corrupted observations as $\Delta\mathbf{x}_{\text{rel}} = \mathbf{A}\Delta\mathbf{x}_g + \mathbf{e}$ where $\mathbf{e}$ contains both the noise and outlier errors. Since $\mathbf{A}$ is known to be sparse, minimization of the ℓ_1 cost can be efficiently carried out using recently developed optimization methods for compressive sensing [4]. While the ℓ_1 optimization can be efficiently solved in the presence of outliers, it will be noted that the ℓ_1 metric is not necessarily the most desirable one for measuring the distance between the observation $\mathbf{X}_{ij}$ and the fitted value $\mathbf{X}_j\mathbf{X}_i^{-1}$. For instance for the case of the $\mathbb{SO}(3)$ group considered in [5, 14], the geodesic distance has a physical interpretation of the magnitude of 3D rotation angle. Hence it is desirable to refine the estimate provided by the robust ℓ_1 approach with an ℓ_2 distance metric. In [5], this refinement is carried out using the Iteratively Reweighted Least Squares (IRLS) approach. In this case, the use of ℓ_1 minimization allows for a robust estimation of $\mathbf{X}_g$ in the presence of outliers. Subsequently, this robust estimate is used as the initial guess for an IRLS-based ℓ_2 refinement. Since the ℓ_1 estimate is of good quality, it can be expected to fall within the basin of convergence of the IRLS step, thereby leading to a composite method that is both robust and efficient.

7.6 Applications

In this section we shall briefly demonstrate the application of the motion averaging principle for problems in SfM and 3D scan registration. As discussed in Sect. 7.1, the SfM problem is typically solved using bundle adjustment which is a nonlinear optimization method that estimates 3D points (structure) as well as 3D camera motions (motion) that best explain the observed image projections. While there have been many improvements in recent years [1, 30], for large-scale problems, bundle adjustment remains computationally very expensive. Another important consideration is the need for a good initial guess for the structure and motion parameters for bundle adjustment. In contrast, the relative geometry between two cameras specified by their epipolar geometry can be solved robustly and efficiently either using the five-point algorithm [22] in a RANSAC loop or by two-frame bundle adjustment [13, 24] which is far cheaper than bundle adjustment over all frames. In SfM the overall scale factor is arbitrary, i.e. we can only recover the 3D translation between two cameras up to an unknown scale factor. However, the 3D rotation between two cameras can be fully recovered from the estimated epipolar geometry. Therefore, instead of solving for the motion averaging problem for 3D Euclidean models, a common strategy is to average only the relative rotations on the $\mathbb{SO}(3)$ group [5, 7, 14]. In Table 7.1, we present a comparison of the computational

Table 7.1 Comparison of different methods for robust averaging of relative rotations for the Quad data set which has 5530 cameras and $222,044$ relative rotations

Median rotation error (°)			Computational time (s)		
DISCO [7]	Weiszfeld [14]	RRA [5]	DISCO [7]	Weiszfeld [14]	RRA [5]
5.02	6.95	1.97	2983	5707	801

This table is adapted from [5]. Our approach of Algorithm 2 (denoted as **RRA**) can be seen to be both substantially more accurate and efficient when compared with the DISCO [7] and Weiszfeld [14] methods. The median error and computational time for DISCO were obtained on a computer much more powerful than that used for the **RRA** and Weiszfeld methods. See [5] for details accuracy and time for these three methods on a large-scale real data set, i.e., Quad data set that contains $|\mathcal{V}| = 5530$ images and as many as $|\mathcal{E}| = 222044$ edges in the viewgraph. The computational time for the method of [7] is for a computer that is substantially more powerful than that used for the methods of [14] and [5]. It will be observed from Table 7.1 that the relative rotation averaging approach (denoted as **RRA**) of Algorithm 2 is both significantly more accurate and faster than the other methods. The method of [7] discretizes the representation space of $\mathbb{SO}(3)$ thereby ignoring the underlying geometric Lie group structure of the problem. Solving the resulting problem using loopy belief propagation is computationally expensive and also requires expensive hardware. In contrast, while [14] operates on the $\mathbb{SO}(3)$ manifold, being a distributed algorithm, it takes a long time to converge. This problem gets exacerbated for large-scale data sets such as the Quad data set considered here. For such data sets, the graph diameter can be quite large implying that distributed methods would become quite slow in convergence. In contrast, as can be seen from Table 7.1, the approach of jointly estimating the relative rotations update on the Lie algebra $\mathfrak{so}(3)$ is both efficient and accurate. Interested readers may consult [5] for details.

Once the global rotations are solved, we can also average the translation direction estimates to recover the 3D translation of the cameras, albeit up to one global scale factor. Briefly, we recognize that the translation direction $\mathbf{t}_{ij}$ between cameras i and j can be written as $\mathbf{t}_{ij} \propto (\mathbf{T}_j - \mathbf{R}_j\mathbf{R}_i^{-1}\mathbf{T}_i)$ where $\mathbf{T}_i$ is the 3D translation of camera i with respect to the global frame of reference. In [9], the unknown scale factor in the translation estimation is eliminated by taking a cross product with $\mathbf{t}_{ij}$ leading to a system of equations of the form $[\mathbf{t}_{ij}]_\times(\mathbf{T}_j - \mathbf{R}_j\mathbf{R}_i^{-1}\mathbf{T}_i) = 0$. This results in an iterative reweighted algorithm for estimating global translations from the relative translation directions. In [29], the problem is solved by considering the equivalent system of equations of the form $\mathbf{t}_{ij} = \frac{\mathbf{T}_j - \mathbf{R}_j\mathbf{R}_i^{-1}\mathbf{T}_i}{||\mathbf{T}_j - \mathbf{R}_j\mathbf{R}_i^{-1}\mathbf{T}_i||}$ which is also solved iteratively. Furthermore, to achieve robustness, [29] eliminates outliers from the set of relative translations before solving for the global translations. In general, the approach of eliminating unknowns using the cross product [9] is not preferable since the cross product is rank-deficient resulting in a cost function that does not correctly penalize deviations from the linear constraints in all directions. In this approach, the result is a global estimate for 3D rotation and translation, i.e., 3D Euclidean motion. Given these 3D camera motion estimates, we can use a variety of methods to solve for the 3D structure points using multi-view triangulation. An alternate strategy is based on the observation that once the 3D rotations are estimated, both the 3D translations and structure points can be considered as unknown in a system of projective equations. In [5], this observation is used to robustly solve for both 3D translation and structure.

In Fig. 7.2 we present the result of reconstructing the well-known Notre Dame cathedral data set (consisting of 715 images) using the approach of [5] where first 3D rotations are solved, followed by a joint estimation of 3D translation and structure. One may also use the results of 3D rotation and translation estimation to get an initial estimate for 3D structure. This SfM reconstruction using motion averaging can be subsequently used as an initial guess for a bundle adjustment pipeline. Since the motion-averaged estimate provides an accurate, global SfM

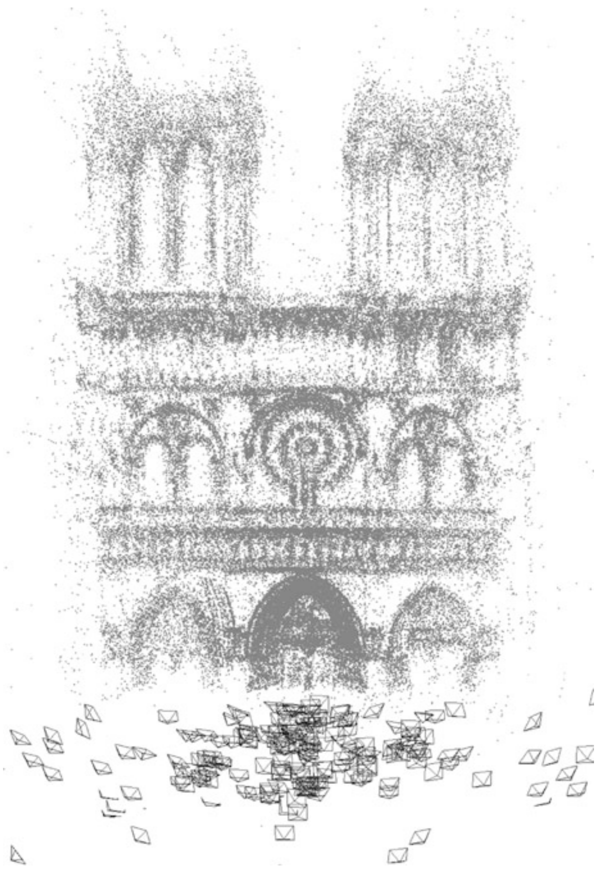


Fig. 7.2 Reconstruction of the Notre Dame data set (715 images) using a motion averaging approach. The recovered 3D point cloud and camera motions are shown here. Result adapted from [5]

solution at a low-computational cost, using this as an initial guess greatly speeds up the time to convergence for large-scale bundle adjustment approaches. The result is significant savings in computational time. The computational advantages and savings in computational time are extensively demonstrated in [7, 29] on a variety of large-scale data sets.

7.6.1 3D Registration

Unlike the case of SfM using images, we can estimate the full 3D Euclidean relative motion between two scans. Given enough corresponding 3D points from two scans, it is easy to solve for the best 3D rotation and translation that registers the points in a common frame of reference [27]. Since, in general, we are not given 3D correspondences a priori, the most commonly used ICP algorithm solves the correspondence and registration problem by greedy iterations that alternately solve for 3D correspondences and 3D motion while holding 3D motion and correspondences fixed, respectively [23]. Although ICP is effective, being a greedy algorithm it needs the initial 3D motion two scans to be small for convergence to the correct registration. Apart from this limitation, ICP suffers from the drawback of registering two scans at a time which often leads to poor quality results. Both of these problems can be significantly mitigated by jointly solving for the registration of all scans. When the 3D scans are systematically acquired so as to cover a whole object or scene, it is often the case that the last scan has significant overlap with the first one, or in general scans that are far apart in acquisition time can be viewing the same parts of the scene. Although, such "loop closures" provide additional constraints that the 3D registration estimates should satisfy, using pairwise ICP in a sequential manner fails to exploit these additional constraints. This problem can be remedied by incorporating motion averaging into the ICP iterations. We call the resultant solution *motion averaged ICP* or MAICP. We briefly describe MAICP here and the reader is referred to [12] for details.

Each iteration of MAICP consists of three steps: establishing correspondences, estimating relative motions, and motion averaging. The *correspondence step* is a straightforward extension from the standard two scan ICP approach. For all scan pairs i and j that have overlapping regions, i.e., $(i,j) \in \mathcal{E}$, we independently obtain correspondences. Typically, for each point (or a subset of points) in scan i we determine the closest point on scan j where distance is measured using the point-to-plane approach. Along with this distance metric, many heuristics are used to weed out potentially incorrect correspondences, see [12, 23] for details. Subsequently in the *motion step* we estimate the relative 3D Euclidean motion between all valid scan pairs i and j. Finally, the relative motion estimates obtained $\{\mathbf{M}_{ij} | \forall (i,j) \in \mathcal{E}\}$ are averaged using Algorithm 2 to obtain the global motion update $\mathbf{M}_g$ to be applied to all scans. This process is repeated till a convergence criterion is satisfied resulting in an accurate registration of 3D scans.

In Fig. 7.3 we present results (adapted from [12]) that compare the performance of sequential ICP and MAICP on two commonly used 3D registration data sets. As can be seen from the cross sections shown, sequential ICP fails to correctly register the scans. In contrast, MAICP is able to correctly register all of the scans as it uses all pairwise relationships available in a motion averaging framework thereby leading to correct 3D motion updates at each iteration. In Fig. 7.4 we present a result of using MAICP to build a 3D model of a metallic bust of the Indian scientist C.V. Raman. In this instance, the raw scans are obtained using the Kinect depth camera and Fig. 7.4 shows the final reconstructed 3D model as well as the estimated locations of individual scanners, i.e., $\mathbf{M}_g$ for the set of 21 scans used in this example.

7.7 Conclusion

In this chapter we have developed a method for averaging relative motions by utilizing the geometric properties of the underlying Lie group. The resulting intrinsic

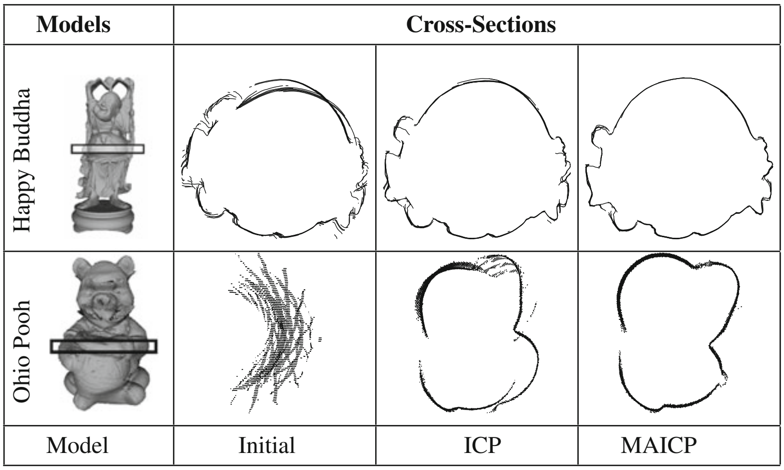


Fig. 7.3 Comparison of performance of ICP and motion-averaged ICP (MAICP) for two data sets. The cross-section regions are shown on the corresponding 3D models. It will be observed that sequential registration using pairwise ICP does poorly whereas MAICP correctly registers all scans to generate a single 3D model of the object scanned. Adapted from [12] where more results are presented

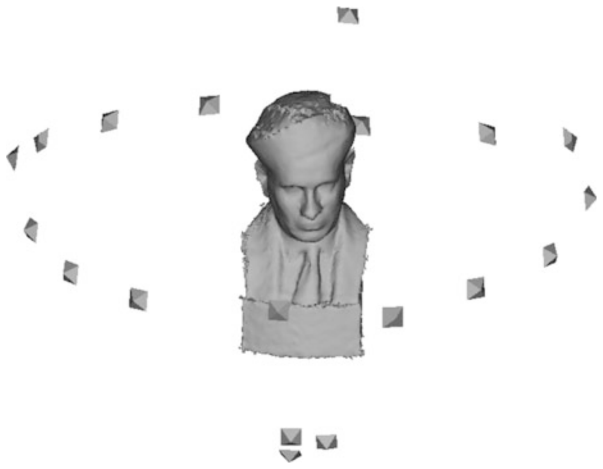


Fig. 7.4 3D reconstruction of a metallic bust using MAICP. This result uses 21 raw scans obtained using the Kinect depth camera and the estimated locations of individual scanners are also indicated. Adapted from [6]

estimate satisfies the underlying geometric constraints, provides an accurate solution, and can be efficiently solved. We also develop suitable modifications of the motion averaging method to incorporate robustness to the presence of outliers in the observed relative motions. The applicability of the motion averaging approach is demonstrated in the context of two 3D reconstruction problems, i.e., structure-from-motion using camera images and 3D reconstruction using multiple scans of an object or scene. Finally we note that while the exposition of this chapter is limited to the Lie groups corresponding to 3D rotations and Euclidean motions, the principle of motion averaging is general and can be applied in a wide range of scenarios.

Acknowledgements The author thanks an anonymous reviewer and Avishek Chatterjee for critical and useful comments.

References

1. Agarwal S, Snavely N, Simon I, Seitz S, Szeliski R (2009) Building rome in a day. In: Proceedings of the international conference on computer vision, pp 72–79
2. Benjemaa R, Schmitt F (1997) Fast global registration of 3d sampled surfaces using a multi-z-buffer technique. In: 3-D digital imaging and modeling, (3DIM), pp 113–120
3. Boothby WM (2003) An introduction to differentiable manifolds and riemannian geometry, revised 2nd edn. Academic, New York
4. Candes E, Tao T (2005) Decoding by linear programming. IEEE Trans Inf Theory 51(12):4203–4215
5. Chatterjee A, Govindu VM (2013) Efficient and robust large-scale rotation averaging. In: IEEE international conference on computer vision (ICCV)
6. Chatterjee A, Jain S, Govindu VM (2012) A pipeline for building 3d models using depth cameras. In: Proceedings of the eighth indian conference on computer vision, graphics and image processing
7. Crandall DJ, Owens A, Snavely N, Huttenlocher D (2011) Discrete-continuous optimization for large-scale structure from motion. In: CVPR
8. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24(6):381–395
9. Govindu VM (2001) Combining two-view constraints for motion estimation. In: CVPR
10. Govindu VM (2004) Lie-algebraic averaging for globally consistent motion estimation. In: CVPR
11. Govindu VM (2006) Robustness in motion averaging. In: Asian conference on computer vision (ACCV)
12. Govindu VM, Pooja A (2014) On averaging multiview relations for 3d scan registration. IEEE Trans Image Process 23(3):1289–1302
13. Hartley RI, Zisserman A (2004) Multiple view geometry in computer vision, 2nd edn. Cambridge University Press, Cambridge
14. Hartley RI, Aftab K, Trumpf J (2011) L1 rotation averaging using the weiszfeld algorithm. In: CVPR
15. Hartley RI, Trumpf J, Dai Y, Li H (2013) Rotation averaging. Int J Comput Vis 103(3):267–305
16. Kanatani K (1990) Group theoretic methods in image understanding. Springer, Heidelberg
17. Kanatani K (1993) Geometric computation for machine vision. Oxford University Press, Oxford
18. Karcher H (1997) Riemannian center of mass and mollifier smoothing. Commun Pure Appl Math 30(5):509–541
19. Krakowski K, Huper K, Manton J (2007) On the computation of the karcher mean on spheres and special orthogonal groups. In: Workshop on robotics and mathematics, RoboMat '07
20. Manton JH (2004) A globally convergent numerical algorithm for computing the centre of mass on compact lie groups. In: International conference on control, automation, robotics and vision, ICARCV 2004, pp 2211–2216
21. Moulon P, Monasse P, Marlet R (2013) Global fusion of relative motions for robust, accurate and scalable structure from motion. In: IEEE international conference on computer vision (ICCV)

22. Nister D (2004) An efficient solution to the five-point relative pose problem. IEEE Trans Pattern Anal Mach Intell 26(6):756–770
23. Rusinkiewicz S, Levoy M (2001) Efficient variants of the icp algorithm. In: 3DIM, pp 145–152
24. Snavely N, Seitz S, Szeliski R (2008) Modeling the world from internet photo collections. Int J Comput Vis 80(2):189–210
25. Triggs B, Mclauchlan P, Hartley R, Fitzgibbon A (2000) Bundle adjustment – a modern synthesis. In: Vision algorithms: theory and practice. LNCS, pp 298–372. Springer, Heidelberg
26. Tron R, Vidal R (2011) Distributed computer vision algorithms through distributed averaging. In: CVPR
27. Umeyama S (1991) Least-squares estimation of transformation parameters between two point patterns. IEEE Trans Pattern Anal Mach Intell 13(4):376–380
28. Varadarajan V (1984) Lie groups, lie algebras, and their representations. Springer, New York
29. Wilson K, Snavely N (2014) Robust global translations with 1dsfm. In: European conference on computer vision (ECCV), pp 61–75
30. Wu C (2013) Towards linear-time incremental structure from motion. In: Proceedings of the international conference on 3D vision, 3DV '13, pp 127–134
31. Zach C, Klopschitz M, Pollefeys M (2010) Disambiguating visual relations using loop constraints. In: CVPR
32. Zacur E, Bossa M, Olmos S (2014) Left-invariant riemannian geodesics on spatial transformation groups. SIAM J Imaging Sci 7(3):1503–1557

Chapter 8
Lie-Theoretic Multi-Robot Localization

Xiao Li and Gregory S. Chirikjian

Abstract This chapter presents a new distributed cooperative localization technique based on a second-order sensor fusion method developed for the special Euclidean group. Uncertainties in the robot pose, sensor measurements, and landmark positions (neighboring robots in this case) are modeled as Gaussian distributions in exponential coordinates. This proves to be a better fit for both the prior and posterior distributions resulting from the motion of nonholonomic kinematic systems with stochastic noise (as compared to standard Gaussians in Cartesian coordinates). We provide a recursive closed-form solution to the multi-sensor fusion problem that can be used to incorporate a large number of sensor measurements into the localization routine and can be implemented in real time. The technique can be used for nonlinear sensor models without the need for further simplifications given that the required relative pose and orientation information can be provided, and it is scalable in that the computational complexity does not increase with the size of the robot team and increases linearly with the number of measurements taken from nearby robots. The proposed approach is validated with simulation in Matlab.

8.1 Introduction

In recent years the field of robotics and automation has undergone a dramatic ascendency in terms of its significance in industrial and military applications as well as its growing importance in service applications. Multi-robot systems (also known as multi-agent systems) is a branch of robotics that deals with the collaboration among teams of robots (either homogenous or heterogenous) in

X. Li (✉)
Laboratory for Computational Sensing and Robotics (LCSR), Johns Hopkins University, Baltimore, MD, USA
e-mail: lixiao89@hotmail.com

G.S. Chirikjian
LCSR and Department of Mechanical Engineering, Johns Hopkins University, Baltimore, MD, USA
e-mail: gchirik1@jhu.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_8

accomplishing certain tasks. This section reviews some of the work done in the field of multi-robot localization followed by an overview of the remainder of this chapter.

8.1.1 Introduction to Multi-Robot Localization

The path to true autonomy starts with robots knowing where they are in a given workspace. Such a problem is known as robot localization. According to [15], the localization problem can be categorized into two subproblems: (1) position tracking (local localization) which aims to compensate for small dead reckoning errors using sensor feedback; this approach is local in that the initial pose is assumed known and (2) global localization in which the robot "figures out" its position given no knowledge of its initial pose. A tremendous amount of effort has been devoted to effectively and efficiently solving the localization problem and the field has seen major advancements in the establishment of highly practical and easy to implement algorithms with the EKF (extended Kalman filter)-based and PF (particle filter)-based approaches the most widely accepted solutions to the problem. However, the majority of existing approaches are tailored to localizing a single robot. The field of multi-robot localization remains relatively fresh and to be explored [6].

Performing the localization task with multiple robots possesses the advantage of information sharing. Robots within a team can exchange information with other members so to increase the accuracy and reduce uncertainty in their own estimates. This advantage is shown both in simulation and experimentally in [6] in which two robots explore an indoor environment executing their own single robot localization scheme when they are far away from each other. And the proposed collaborative fusion algorithm is used when the two robots come into each other's detection range. Results show that such an algorithmic reinforcement has the effect of significantly reducing the ambiguities existing in the original estimates. A collaborative architecture of this sort can effectively reduce the hardware cost of the entire team in that if at least one robot has a good knowledge of its location, then other team members can use this information along with relative measurements to infer their own position and reduce estimation errors.

8.1.2 Comparison of Existing Distributed Localization Methods

The problem of cooperative localization has been tackled with a wide variety of approaches over the years. And similar to single robot localization, many of the existing algorithms can be considered variations of two main categories. The first family of algorithms makes use of recursive Gaussian filters. Distributed versions of the Kalman filter are proposed in [1, 14] to solve the cooperative localization problem. The extended Kalman filter (EKF) is utilized in [12] while also providing analytical expressions for the upper bound of the estimate uncertainty. In [2] the EKF is also used, but the algorithm is reinforced with an entropic criterion to select optimal measurements that reduce global uncertainty. The advantage of using recursive Bayesian filters to fuse information lies in they are incremental in nature, which makes them applicable to real-time estimation. Closed-form expressions for state estimation and update also facilitate computational speed. However, these types of algorithms deal only with Gaussian noise which may not be the case for some real systems. And EKF linearizes the system dynamics around the state of estimate which is prone to failure when errors grow.

The second family of algorithms is built upon sampling-based nonparametric filters. Monte Carlo localization methods are used in [9] to estimate the pose of each member robot while using grid cells to describe the entire particle set. A global collaborative localization algorithm is presented in [6] that also builds upon sample-based Markov localization. In addition, [3, 8, 11] have all approached the problem with different variations of the particle filter and have also applied their algorithm in the SLAM (simultaneous localization and mapping) context. Further experimental validation is provided in [11] and [3]. Grid-based and sampling-based Markov localization techniques usually address the problem globally and can be improved via carefully designed resampling processes to counteract localization failures. They can also be used to accommodate non-Gaussian noise models. However, like all sampling-based approaches, a large number of grids/samples are usually needed to acquire reasonable outcomes, and the computational cost grows dramatically with the dimension of the problem. A table comparing the two families of methods is provided below.

These two main categories of localization techniques presented in Table 8.1 currently dominate the field. Both possess their own pros and cons and the choice of which depend heavily on the type of applications they are desired for. The two approaches can potentially be combined to yield superior outcomes. For more details regarding the extended Kalman filter (EKF) applied to multi-robot systems see [12, 14]. For details on collaborative Monte Carlo localization (MCL) see [6].

The following subsection explains how our approach differs.

8.1.3 Objectives, Contributions, and Outline

Existing approaches to the multi-robot localization problem usually consider only uncertainties in each robot's pose estimate and sensor measurement. The goal of this chapter is to explore cooperative localization in a more generalized setting where uncertainties in the sources of relative measurements (neighboring robots' pose estimates) are also considered. The distributed localization approach proposed in this chapter makes an effort to providing recursive closed-form expressions for real time cooperative sensor fusion used for pose updates of robots within a team. This work extends the method presented in [10], which considers cooperative

Table 8.1 Comparison between distributed EKF And MCL

	Distributed EKF	Multi-Robot MCL
Restrictions on error distribution	Requires Gaussian process and measurement error	Nonparametric particle representation of posterior belief, no assumptions on noise distribution
Global localization	No	Yes
State recovery	No	Possible given a well-designed resampling process
Localization accuracy	Accurate when error is small	Depends on the number of particles used
Computational cost	Small due to the closed-form propagation and update equations	Depends on the number of particles. Can increase dramatically with the dimension of the state space
Ease of implementation	Simple	Can be involved
Robustness	Prone to error due to linearization	Quite resistant to errors given multiple beliefs are maintained simultaneously
Process multiple detections simultaneously	No	No
Complexity relative to team size	Fully distributed. Complexity independent of team size	Complexity independent of team size

localization with only one exact noise-free measurement (relative to a neighboring robot), whereas the technique proposed here can take into account any number of relative measurements while also considering sensor noise. This method is developed under the framework of exponential coordinates for Lie groups which gives this exotic sounding methodology a down-to-earth benefit: Gaussian distribution in Cartesian coordinates, (x, y, θ)-planar coordinates and heading angle, possesses elliptical probability density contours for each fixed θ and for marginal densities in (x, y), whereas the banana-shaped distribution resulting from incremental motions of a stochastic differential-kinematic system (i.e., a probabilistic model of mobile robots with nonholonomic kinematic constraints) is better represented by a Gaussian in exponential coordinates which produce a more conformable density contour (see Fig. 8.2a). This underlying framework allows the proposed algorithm to tolerate higher errors without worrying about collapse of the normality assumption as uncertainty grows. Unlike most existing cooperative localization schemes that consider only uncertainty in the pose of the robot to be estimated and measurement noise, the presented method has also taken into account the uncertainty in the pose of nearby robots from which relative measurements are taken, making it a more realistic and dynamical localization technique. This approach is second order in its expansion of the Gaussians that describes the pose and measurement distributions using the *Baker–Campbell–Hausdorff (BCH) formula* [4], and no simplifications are made regarding the system kinematics, thus preserving the full nonlinear characteristics of the original system. Lastly, the form of sensor measurement in this method is kept generic without assuming the type of sensor or any underlying characteristics given the Gaussian-in-exponential-coordinate model can be applied.

The remainder of this chapter is outlined as follows. Section 8.2 introduces the mathematical foundation on which the proposed approach is based, namely the basics of matrix Lie groups and exponential mapping. Section 8.3 provides a detailed derivation of the proposed technique. Section 8.4 describes the experimental setup in simulation and provides a discussion of the results. Section 8.5 concludes this chapter.

8.2 Mathematical Background for the Group SE(n) and Exponential Mapping

8.2.1 The Special Euclidean Group and Exponential Coordinates

The Special Euclidean Motion Group

The proposed technique is largely based on the notion of Lie groups and their parameterizations. According to [4], a *group* is defined as a pair $(G, \circ)$ consisting of a set G and a binary operator $\circ$ such that $g_1 \circ g_2 \in G$ whenever $g_1, g_2 \in G$, the operator is associative in the sense that $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$ for all elements $g_1, g_2, g_3 \in G$, there exists an identity element $e \in G$ such that for all elements $g \in G$, $g \circ e = e \circ g = g$, and for each $g \in G$ there exists an inverse element $g^{-1} \in G$ such that $g \circ g^{-1} = g^{-1} \circ g = e$. For engineering applications, a group of great interest is the Special Euclidean Group, SE(n), that describes rigid-body motions in n-dimensional Euclidean space. The elements of SE(n) can be represented as $(n + 1) \times (n + 1)$ homogeneous transformation matrices of the form

$$\mathrm{SE}(n) = \left\{ \begin{bmatrix} R & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \middle| R \in \mathrm{SO}(n), \mathbf{t} \in \Re^n \right\}, \tag{8.1}$$

where SO(n) is the special orthogonal group consisting of $n \times n$ rotation matrices and $\Re^n$ is the n-dimensional vector space representing translations. The binary operation in this context is simply the matrix multiplication. The Special Euclidean Group is also a *matrix Lie group* since each element is a real-valued matrix, the whole set is a differentiable manifold, and both the operations of multiplication and inversion of homogeneous transformation matrices are smooth operations. We note that in most practical problems n takes only two values: $n = 2$ for planar motion and $n = 3$ for 3-dimensional space motion.

Now we introduce the concept of *Lie Algebra*. Again following [4], elements of a matrix Lie group can be written as $g = \exp(X)$ for $X \in \mathcal{G}$ where the set $\mathcal{G}$ is the matrix Lie algebra of G. The Lie Algebra for SE(2) [denoted as se(2)] can be represented by the linear combination of a set of basis

$$E_1^{\mathrm{se}(2)} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad E_2^{\mathrm{se}(2)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}, \qquad E_3^{\mathrm{se}(2)} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The exponential coordinates for an element of SE(2) can be defined as $\mathbf{x}^{\mathrm{se}(2)} = [v_1, v_2, \alpha]^T$ and under this definition an element of the Lie algebra se(2) can be written as a 3×3 matrix

$$X^{\mathrm{se}(2)} = \begin{bmatrix} 0 & -\alpha & v_1 \\ \alpha & 0 & v_2 \\ 0 & 0 & 0 \end{bmatrix} = \sum_{i=1}^{3} E_i^{\mathrm{se}(2)} x_i^{\mathrm{se}(2)}. \tag{8.2}$$

The "hat" and "vee" notation is convenient to identify an element of se(2) with a vector in $\Re^3$ as follows:

$$\hat{\mathbf{x}}^{\mathrm{se}(2)} = X^{\mathrm{se}(2)} \text{ and } X^{\mathrm{se}(2)\vee} = \mathbf{x}^{\mathrm{se}(2)}.$$

The exponential map $\exp : \mathrm{se}(2) \to \mathrm{SE}(2)$ is surjective, but is not injective since $\alpha = \pi$ and $-\pi$ correspond to the same group element of SE(2). But by removing from SE(2) the set of measure zero, M, corresponding to $\alpha = \pi$, it is possible to define an inverse map $\log : (\mathrm{SE}(2) - M) \to \mathrm{se}(2)$. Since the integrals of well-behaved functions over SE(2) and $\mathrm{SE}(2) - M$ are the same, we do not distinguish between SE(2) and $\mathrm{SE}(2) - M$ in the remainder of this chapter.

For SE(2), exponentiation gives

$$R = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \text{ and } \mathbf{t} = \begin{bmatrix} [v_2(\cos(\alpha) - 1) + v_1 \sin(\alpha)]/\alpha \\ [v_1(1 - \cos(\alpha)) + v_2 \sin(\alpha)]/\alpha \end{bmatrix}. \tag{8.3}$$

Adjoint Matrices

The *adjoint operators* $\mathrm{Ad}(g)$ and $\mathrm{ad}(X)$ are two important concepts in the derivations that follow, and so their definitions as well as relevant properties are introduced in this section. To define the adjoints, we need to first define the inner product and Lie bracket operations for Lie algebras. According to [4], an *inner product* between arbitrary elements of the Lie algebra $Y = \sum_i y_i E_i$ and $Z = \sum_i z_i E_i$ can be defined such that

$$(Y, Z) = \sum_{i}^{d} y_i z_i, \tag{8.4}$$

where d is the dimension of G. In particular, for $G = \mathrm{SE}(n)$ the dimension is $d = n(n+1)/2$. Choosing a basis $\{E_i\}$ and requiring that $(E_i, E_j) = \delta_{ij}$, where δ_{ij} is the Dirac delta function, defines an inner product for $\mathcal{G}$ and a metric for G.

The *Lie bracket* of $Y, Z \in \mathcal{G}$ is defined as

$$[Y, Z] \doteq YZ - ZY. \tag{8.5}$$

With the above definitions in place, and any $g \in G$, the adjoint operators are

$$\begin{aligned} \mathrm{Ad}(g)X &\doteq \frac{d}{dt}(g \circ \exp(tX) \circ g^{-1})|_{t=0} = \frac{d}{dt}\exp(tgXg^{-1})\Big|_{t=0} = g\,X\,g^{-1}, \\ \mathrm{ad}(X)Y &\doteq \frac{d}{dt}(\mathrm{Ad}(e^{tX})Y)\Bigg|_{t=0}. \end{aligned} \tag{8.6}$$

Since the adjoint operators are both linear operators, they can both be written as matrices that represent linear mapping. We call these matrices *adjoint matrices* and define them as (in component form) [4]

$$[\mathrm{Ad}(g)]_{ij} = (E_i, \mathrm{Ad}(g)E_j) = (E_i, g\,E_j\,g^{-1}), \quad [\mathrm{ad}(X)]_{ij} = (E_i, \mathrm{ad}(X)E_j) = (E_i, [X, E_j]). \tag{8.7}$$

Written in terms of columns, the matrices have the form

$$[\mathrm{Ad}(g)] = [(gE_1g^{-1})^\vee, \ldots, (gE_ng^{-1})^\vee], \quad [\mathrm{ad}(X)] = [[X, E_1]^\vee, \ldots, [X, E_n]^\vee]. \tag{8.8}$$

Some important properties of the adjoint matrices that are used in the following calculations are listed as follow:

1. $\mathrm{Ad}(\exp(X)) = \exp(\mathrm{ad}(X))$, $\mathrm{ad}(X)X^\vee = 0$, $\mathrm{ad}(X)Y = XY - YX = [X, Y]$
2. $\mathrm{ad}(X)Y^\vee = [X, Y]^\vee$, $\mathrm{ad}([X, Y]) = \mathrm{ad}(X)\mathrm{ad}(Y) - \mathrm{ad}(Y)\mathrm{ad}(X)$, $\mathrm{ad}(X)Y^\vee = -\mathrm{ad}(Y)X^\vee$
3. $\mathrm{Ad}(g_1)\mathrm{Ad}(g_2)X = g_1(g_2Xg_2^{-1})g_1^{-1} = (g_1g_2)X(g_1g_2)^{-1} = \mathrm{Ad}(g_1g_2)X$
4. $\log^\vee(g \circ e^X \circ g^{-1}) = \mathrm{Ad}(g)\log^\vee(e^X)$

For SE(2), the explicit form of the adjoint matrices are

$$\mathrm{Ad}(g) = \begin{bmatrix} R & M\mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \Re^{3\times3}, \quad \mathrm{ad}(g) = \begin{bmatrix} -\alpha M & M\mathbf{v} \\ \mathbf{0}^T & 0 \end{bmatrix} \in \Re^{3\times3} \tag{8.9}$$

where $M = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$ and R and $\mathbf{t}$ are defined by Eq. (8.1). $(\mathbf{v}, \alpha) = (v_1, v_2.\alpha)$ are the exponential coordinates of SE(2).

The Baker–Campbell–Hausdorff Formula

The *Baker–Campbell–Hausdorf (BCH) formula* [4] serves as the core of the second-order estimation of Gaussian convolutions (described in more detail in the next section). In essence, the BCH formula establishes a relationship between the Lie bracket [defined in Eq. (8.5)] and the matrix exponential. Let $X, Y \in \mathcal{G}$ and define $Z(X, Y) = \log(e^X e^Y)$, the BCH formula then takes the form

$$\begin{aligned} Z(X,Y) &= X + Y + \frac{1}{2}[X,Y] + \frac{1}{12}([X,[X,Y]] + [Y,[Y,X]]) \\ &\quad + \frac{1}{48}([Y,[X,[Y,X]]] + [X,[Y,[Y,X]]]) + \cdots . \end{aligned} \tag{8.10}$$

This can be verified by expanding e^X, e^Y using matrix exponential Taylor series $e^X = \sum_{k=0}^{\infty} \frac{X^k}{k!}$ and substitute into the Taylor series for matrix logarithm

$$\log(e^X e^Y) = \log(I + (e^X e^Y - I)) = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(e^X e^Y - I)^k}{k}. \tag{8.11}$$

Applying the $\vee$ operator to (8.10) results in

$$\begin{aligned} \mathbf{z} &= \mathbf{x} + \mathbf{y} + \frac{1}{2}\mathrm{ad}(X)\mathbf{y} + \frac{1}{12}(\mathrm{ad}(X)\mathrm{ad}(X)\mathbf{y} + \mathrm{ad}(Y)\mathrm{ad}(Y)\mathbf{x}) \\ &\quad + \frac{1}{48}(\mathrm{ad}(Y)\mathrm{ad}(X)\mathrm{ad}(Y)\mathbf{x} + \mathrm{ad}(X)\mathrm{ad}(Y)\mathrm{ad}(Y)\mathbf{x}) + \cdots . \end{aligned} \tag{8.12}$$

Equations (8.11) and (8.12) will be used extensively.

8.2.2 Gaussians on SE(n) and Second-Order Convolution Theory

A Gaussian on the SE(n) is defined as

$$f(g;\mu,\Sigma) \doteq \frac{1}{C(\Sigma)} \exp\left\{-\frac{1}{2}[\log^{\vee}(\mu^{-1}g)]^T \Sigma^{-1} [\log^{\vee}(\mu^{-1}g)]\right\}, \tag{8.13}$$

where $\mu, g \in \mathrm{SE}(n)$, $C(\Sigma) \approx (2\pi)^{\frac{d}{2}} \|\det(\Sigma)\|^{\frac{1}{2}}$ is the normalizing factor when $\|\Sigma\|$ is small.

For a domain of integration $G = \mathrm{SE}(n)$, the mean of the above Gaussian is defined by the value $\mu \in G$ for which

$$\int_G \log^{\vee}(\mu^{-1}g) f(g)\,\mathrm{d}g = \mathbf{0}, \tag{8.14}$$

and the covariance is given by

$$\Sigma \doteq \int_G [\log^{\vee}(\mu^{-1}g)][\log^{\vee}(\mu^{-1}g)]^T f(g)\,\mathrm{d}g. \tag{8.15}$$

For details on how to integrate on $\mathrm{SE}(n)$, please refer to [15, 16]. Given two Gaussians $f_1(g) = f(g;\mu_1,\Sigma_1)$ and $f_2(g) = f(g;\mu_2,\Sigma_2)$ in the form of (8.13), their convolution is defined as

$$(f_1 * f_2)(g) = \int_G f_1(h) f_2(h^{-1}g)\,\mathrm{d}h = \int_G \rho_1(\mu_1^{-1}h)\rho_2(\mu_2^{-1}h^{-1}g)\,\mathrm{d}h,$$

with $\rho_i(g) = f(g;e,\Sigma_i)$ denoting a Gaussian centered at the identity. It is proven (refer to [16]) that the convolution $(f_1 * f_2)(g)$ results (to the second order) in a Gaussian with mean and covariance

$$\mu_{1*2} = \mu_1\mu_2, \quad \Sigma_{1*2} = A + B + F(A,B) \tag{8.16}$$

with the terms A, B, and F defined by

$$A = \mathrm{Ad}(\mu_2^{-1})\Sigma_1\mathrm{Ad}(\mu_2^{-1})^T \text{ and } B = \Sigma_2, \tag{8.17}$$

where

$$F(A,B) = \frac{1}{4}\sum_{i,j=1}^{d} \mathrm{ad}(E_i)\; B\; \mathrm{ad}(E_j)^T A_{ij} + \frac{1}{12}\left\{[\sum_{i,j=1}^{d} A_{ij}'']B + B^T[\sum_{i,j=1}^{d} A_{ij}'']^T\right\}$$
$$+ \frac{1}{12}\left\{[\sum_{i,j=1}^{d} B_{ij}'']A + A^T[\sum_{i,j=1}^{d} B_{ij}'']^T\right\} \tag{8.18}$$

and

$$A_{ij}'' = \mathrm{ad}(E_i)\mathrm{ad}(E_j)A_{ij}, \quad B_{ij}'' = \mathrm{ad}(E_i)\mathrm{ad}(E_j)B_{ij}. \tag{8.19}$$

The above results will be used for SE(2) in the next section, where the basis elements E_i as well as *Ad* and *ad* matrices as defined in the previous section.

8.3 Derivation of Second-Order Bayesian Sensor Fusion on the *SE* Group

This section presents a detailed derivation of the proposed technique. Again the technique focuses on fusing the relative measurements of neighboring robots and their pose information to reduce the estimation uncertainty of the current robot. A probabilistic approach is adapted where uncertainties in the robot positions and sensor measurements are modeled by Gaussians (refer to [5] for a more generalized formulation of nonlinear measurement model approximation on Lie groups). In addition, since the motion of a stochastic system with differential constraints is modeled more accurately with Gaussians in exponential coordinates than that in Cartesian coordinates, the proposed technique is developed under exponential coordinates. The theory will first be developed for a system of two robots (which builds on [10] by taking sensor noise into consideration) and be extended to the multi-robot scenario.

8.3.1 Localization for a Robot Pair

The problem is given by two mobile robots i and j moving in the plane whose priors in position and orientation are Gaussians $f(a_i^{-1}g_i; \mu_i, \Sigma_i)$ and $f(a_j^{-1}g_j; \mu_j, \Sigma_j)$. Here $a_i, a_j \in \text{SE}(2)$ are the known initial positions of the robots relative to the world frame at $t = 0$. At time t, $\mu_i, \mu_j \in \text{SE}(2)$ and $\Sigma_i, \Sigma_j \in R^{3\times3}$ are the means (defined relative to the initial frames a_i, a_j), and covariances obtained from a previous prediction step which we will also assume to be known. In addition, a sensor measurement of robot j relative to i is also obtained at time t and is given by the homogeneous matrix $m_{ij} \in \text{SE}(2)$. Since we assume the sensor has Gaussian noise, its distribution is then characterized by a Gaussian of the form $M_{ij}(g_i, g_j) = f(g_j; g_i m_{ij}, \Sigma_m)$ which says that according to the sensor, the position of robot j with respect to robot i has a mean of m_{ij} and covariance of Σ_m.

The goal is then to calculate a posterior for the position of robot i using the sensor measurement to update its prior. Because the sensor provides a relative measurement, we first formulate the joint prior of robots i and j making the assumption that the priors are independent of each other, giving

$$p_{ij}(g_i, g_j) = f(a_i^{-1}g_i; \mu_i; \Sigma_i) f(a_j^{-1}g_j; \mu_j; \Sigma_j). \tag{8.20}$$

Then according to Bayes' Theorem, the joint posterior is given by

$$\bar{p}_{ij} = \eta_1 p_{ij} M_{ij}, \tag{8.21}$$

where η_1 is a constant normalizing factor. Similar normalizing factors result in all fusion processes that follow and will be denoted by η_i. To save space in the derivations, we will denote $\rho_i(\mu_i^{-1}g_i) = f(g_i;\mu_i;\Sigma_i)$ and the rest follows where $\rho_i(g)$ is a Gaussian with mean at the identity. The marginal distribution of the joint posterior for robot i is then

$$\begin{aligned}\overline{p_i}(g_i) &= f(g_i;\overline{\mu}_i,\overline{\Sigma}_i) = \eta_2 \int_G p_{ij}(g_i,g_j)M_{ij}(g_i,g_j)\,\mathrm{d}g_j \\ &= \eta_2\rho_i(\mu_i^{-1}a_i^{-1}g_i)\int_G \rho_j(\mu_j^{-1}a_j^{-1}g_j)\rho_m(m_{ij}^{-1}g_i^{-1}g_j)\,\mathrm{d}g_j.\end{aligned} \tag{8.22}$$

The goal is to find closed-form expressions for $\overline{\mu}_i$ and $\overline{\Sigma}_i$. Since ρ_m is symmetric around the mean, we have $\rho_m(m_{ij}^{-1}g_i^{-1}g_j) = \rho_m(g_j^{-1}g_im_{ij})$. Letting $g' = g_im_{ij}$, Eq. (8.22) becomes

$$\begin{aligned}\overline{p_i}(g_i) &= \eta_2\rho_i(\mu_i^{-1}a_i^{-1}g_i)\int_G \rho_j(\mu_j^{-1}a_j^{-1}g_j)\rho_m(g_j^{-1}g_im_{ij})\,\mathrm{d}g_j \\ &= \eta_2\rho_i(\mu_i^{-1}a_i^{-1}g_i)\int_G \rho_j(\mu_j^{-1}a_j^{-1}g_j)\rho_m(e^{-1}g_j^{-1}g')\,\mathrm{d}g_j,\end{aligned} \tag{8.23}$$

where $e \in \mathrm{SE}(2)$ is the identity element of SE(2). According to the definition of convolution in Sect. 8.2, the integral in Eq. (8.23) defines a convolution $(f_1 * f_2)(g')$ where $f_1(g') = f(g';a_j\mu_j,\Sigma_j)$ and $f_2(g') = f(g';e,\Sigma_m)$. Let $f_{1*2}(g';\mu_{1*2},\Sigma_{1*2}) = (f_1 * f_2)(g')$, then (8.16)–(8.19) can be used to calculate the closed-form expressions of μ_{1*2} (which equals to $a_j\mu_j$) and Σ_{1*2}. With the integral taken care of, (8.23) becomes

$$\overline{p_i}(g_i) = f(g_i;\overline{\mu}_i,\overline{\Sigma}_i) = \eta_2 f(\mu_i^{-1}a_i^{-1}g_i;e,\Sigma_i)f(g_im_{ij};a_j\mu_j,\Sigma_{1*2}). \tag{8.24}$$

For a posterior of robot i formulated in the form of (8.24), the fusion technique developed in [10] can be used to derive the closed-form expressions for $\overline{\mu}_i$ and $\overline{\Sigma}_i$.

8.3.2 Localization for Multi-Robot Team

Now we are ready to extend the technique to multi-robot localization. Similar to the previous subsection, the posterior of robot i is what we are trying to estimate, but instead of taking measurement from a single neighboring robot, multiple measurements are taken from however many neighboring robots that are in the sensing range (for derivation purposes we label the neighboring robots as $1,2,\ldots,n$). Following a similar approach we have the joint prior

$$\begin{aligned}p_{i,1,\ldots,n} &= f(a_i^{-1}g_i;\mu_i;\Sigma_i)f(a_1^{-1}g_1;\mu_1;\Sigma_1)\ldots f(a_n^{-1}g_n;\mu_n;\Sigma_n) \\ &= \rho_i(\mu_i^{-1}a_i^{-1}g_i)\rho_1(\mu_1^{-1}a_1^{-1}g_1)\ldots\rho_n(\mu_n^{-1}a_n^{-1}g_n).\end{aligned} \tag{8.25}$$

Let $M_{in} = f(g_n; g_i m_{in}, \Sigma_{in})$ be the distribution of the sensor measurement of robot n relative to robot i and assume independence among all the measurements. Then we have the joint measurement distribution

$$M_{i,1,\ldots,n} = M_{i1} M_{i2} \ldots M_{in}. \tag{8.26}$$

To further save space, we will write in short $\rho_i = \rho_i(\mu_i^{-1} a_i^{-1} g_i)$ as the position priors and $\rho_{in} = \rho_{in}(m_{in}^{-1} g_i^{-1} g_n) = M_{in}$ as the measurement distributions. We will further define $g'_{in} = g_i m_{in}$. The posterior for robot i is then

$$\begin{aligned}\overline{p_i}(g_i) &= f(g_i; \overline{\mu}_i, \overline{\Sigma}_i) = \eta_3 \int_G \int_G \ldots \int_G p_{i,1,\ldots,n} M_{i,1,\ldots,n} \, \mathrm{d}g_1 \, \mathrm{d}g_2 \ldots \mathrm{d}g_n \\ &= \eta_3 \rho_i \left(\int_G \rho_1 \rho_{i1} \, \mathrm{d}g_1 \right) \left(\int_G \rho_2 \rho_{i2} \, \mathrm{d}g_2 \right) \ldots \left(\int_G \rho_n \rho_{in} \, \mathrm{d}g_n \right).\end{aligned} \tag{8.27}$$

Let $f_n(g'_{in}) = f(g'_{in}; a_n \mu_n, \Sigma_n)$ and $f_{in}(g'_{in}) = f(g'_{in}; e, \Sigma_{in})$, then (8.27) becomes

$$\overline{p_i}(g_i) = f(g_i; \overline{\mu}_i, \overline{\Sigma}_i) = \eta_3 \rho_i(\mu_i^{-1} a_i^{-1} g_i)(f_1 * f_{i1})(g'_{i1})(f_2 * f_{i2})(g'_{i2}) \ldots (f_n * f_{in})(g'_{in}). \tag{8.28}$$

Calculating the convolutions using (8.16)–(8.19), we finally arrive at

$$\boxed{\overline{p_i}(g_i) = f(g_i; \overline{\mu}_i, \overline{\Sigma}_i) = \eta_3 f(\mu_i^{-1} a_i^{-1} g_i; e, \Sigma_i) f(g_i m_{i1}; a_1 \mu_1, \Sigma_{1*i1}) \times \ldots \times f(g_i m_{in}; a_n \mu_n, \Sigma_{n*in})} \tag{8.29}$$

An extension of the method provided by [10] (which fuses only one measurement) gives the equations to calculate $\overline{\mu}_i$ and $\overline{\Sigma}_i$ and is presented as follows:

For neighboring robots $1, \ldots, k, \ldots, n$

1. Define $q_k = m_{ik} \mu_k^{-1} a_k^{-1} a_i \mu_i$, $\exp(\hat{x}_k) = q_k$., $\Gamma_k = (I + \frac{1}{2}\mathrm{ad}(\hat{x}_k))$, $S_i = \Gamma_i^T \Sigma_i^{-1} \Gamma_i$
2. Define $S_k = \Gamma_m^T \mathrm{Ad}^{-T}(m_{ik}) \Sigma_{k*ik}^{-1} \mathrm{Ad}^{-1}(m_{ik}) \Gamma_k$
3. $\bar{S}' = S_i + \sum_{k=1}^{n} S_k$, $\bar{x}' = \bar{S}'^{-1} \sum_{k=1}^{n} S_k x_k$

With the above definitions, the posterior distribution for robot i can be calculated by

$$\boxed{\begin{aligned}\overline{\Sigma}_i &= \bar{\Gamma}' \bar{S}'^{-1} \bar{\Gamma}'^T \\ \overline{\mu}_i &= \mu_i \exp(-\hat{\bar{x}}')\end{aligned}} \tag{8.30}$$

with the operator $\wedge$ and $\vee$ defined in section 8.2.1.

8.3.3 A Complete Distributed Localization Algorithm Using Bayesian Filter in Exponential Coordinates

The fusion technique introduced above defines the state update step for the proposed localization method. However, like all Bayesian filters a complete recursive filter for state estimation consists of a state prediction step as well as a state update step. This section serves to provide the proposed algorithm in such a form.

Similar to the above setting, suppose at time t_k robot i is the robot to be localized, robots $1, \ldots, k, \ldots, n$ are its n neighbors. Their means are $\mu_i(t_k), \mu_k(t_k)$ and covariances $\Sigma_i(t_k)$, $\Sigma_k(t_k)$, respectively. Let the stochastic differential equation (SDE) governing the motion of the robots be of the form

$$(g^{-1}\dot{g})^{\vee} dt = \mathbf{h}dt + Hd\mathbf{w}, \tag{8.31}$$

where $\mathbf{h}$ is constant. When $g \approx e$ and for a sampling time Δt a constant command u is given to the system resulting in motion of the system from t_k to t_{k+1}, the distributed localization scheme that estimates the location of robot i at time t_{k+1} follows two steps (letting $\Delta t = t_{k+1} - t_k$). These are the prediction and update steps.

Prediction

$$\boxed{\begin{aligned}
\mu_i(\Delta t) &= \exp\left(\int_0^{\Delta t} \hat{\boldsymbol{h}}_i \, \mathrm{d}\tau\right) \\
\Sigma_i(\Delta t) &= \int_0^{\Delta t} \mathrm{Ad}(\mu_i^{-1}(t-\tau))H_i H_i^T \mathrm{Ad}^T(\mu_i^{-1}(t-\tau)) \, \mathrm{d}\tau \\
\mu_i(t_{k+1}^-) &= \mu_i(t_k) \circ \mu_i(\Delta t) \\
\Sigma_i(t_{k+1}^-) &= A_i(t_k) + B_i(t_k) + F(A_i(t_k), B_i(t_k))
\end{aligned}} \tag{8.32}$$

where

$$\begin{aligned}
A_i(t_k) &= \mathrm{Ad}(\mu_i(\Delta t)^{-1})\Sigma_i(t_k^+)\mathrm{Ad}(\mu_i(\Delta t)^{-1})^T, \quad B_i(t_k) = \Sigma_i(\Delta t) \\
A_i(t_k)_{ij}^{''} &= \mathrm{ad}(E_i)\mathrm{ad}(E_j)A_i(t_k)_{ij}, \quad B_i(t_k)_{ij}^{''} = \mathrm{ad}(E_i)\mathrm{ad}(E_j)B_i(t_k)_{ij}
\end{aligned} \tag{8.33}$$

$$
\begin{aligned}
F_i(A_i(t_k), B_i(t_k)) = \frac{1}{4}\sum_{i,j=1}^{d} \mathrm{ad}(E_i)B_i(t_k)\mathrm{ad}(E_j)^T A_i(t_k)_{ij} \\
+ \frac{1}{12}\left\{\left[\sum_{i,j=1}^{d} A_i(t_k)''_{ij}\right]B_i(t) + B_i(t)^T\left[\sum_{i,j=1}^{d} A_i(t)''_{ij}\right]^T\right\} \\
+ \frac{1}{12}\left\{\left[\sum_{i,j=1}^{d} B_i(t_k)''_{ij}\right]A_i(t_k) + A_i(t_k)^T\left[\sum_{i,j=1}^{d} B_i(t_k)''_{ij}\right]^T\right\}
\end{aligned} \tag{8.34}
$$

The second equation in (8.32) follows from equation (19) in [13] given by

$$
\Sigma_i(\Delta t) = \int_0^{\Delta t} Ad^{-1}[\mu_i^{-1}(\tau)\mu_i(t)]H_i H_i^T Ad^{-T}[\mu_i^{-1}(\tau)\mu_i(t)]\,\mathrm{d}\tau. \tag{8.35}
$$

Since in our context the mean takes the form of $\mu = \exp(Xt)$ where $X \in \mathcal{G}$ is a constant, it follows that $\mu_i^{-1}(t)\mu_i(\tau) = \mu_i(\tau)\mu_i^{-1}(t) = \mu_i^{-1}(t-\tau)$. Combined with the property of adjoint $Ad^{-1}(\mu) = \mathrm{Ad}(\mu^{-1})$ gives the final expression in (8.32).

Also in the above equations, $\mu_i(\Delta t)$ and $\Sigma_i(\Delta t)$ define the incremental distribution resulting solely from the input given at the Δt time frame with location given with respect to $\mu_i(t_k)$, not the fixed world frame. In order to take into account the uncertainties already present at time t given by $\Sigma_i(t_k)$, the distribution at time t_k is convolved with the incremental distribution resulting in the predicted distribution given by $\mu_i(t_{k+1}^-)$, $\Sigma_i(t_{k+1}^-)$.

Update

Now to incorporate the relative measurements, for each of the neighboring robots $1, \ldots, k, \ldots, n$, obtain the measurement distribution $m_{ik}(t)$, $\Sigma_{ik}(t)$, then

$$
\begin{aligned}
A_{ik}(t_k) &= \mathrm{Ad}(m_{ik}(t_k)^{-1})\Sigma_k(t_k)\mathrm{Ad}(m_{ik}(t_k)^{-1})^T, \quad B_{ik}(t_k) = \Sigma_{ik}(t_k) \\
A_{ik}(t_k)''_{ij} &= \mathrm{ad}(E_i)\mathrm{ad}(E_j)A_{ik}(t_k)_{ij}, \quad B_{ik}(t_k)''_{ij} = \mathrm{ad}(E_i)\mathrm{ad}(E_j)B_{ik}(t_k)_{ij}
\end{aligned} \tag{8.36}
$$

and using (8.18),

$$
\Sigma_{k*ik}(t(t_k)) = A_{ik}(t_k) + B_{ik}(t_k) + F(A_{ik}(t_k), B_{ik}(t_k)) \tag{8.37}
$$

1. Define $q_k(t_k) = m_{ik}(t_k)\mu_k(t_k)^{-1}a_k^{-1}a_i\mu_i(t_{k+1}^-)$, $\exp(\hat{x}_k(t_k)) = q_k(t_k)$
2. Define $\Gamma_k(t_k) = (I + \frac{1}{2}\mathrm{ad}(\hat{x}_k(t_k)))$, $S_i(t_k) = \Gamma_i(t_k)^T[\Sigma_i(t_{k+1}^-)]^{-1}\Gamma_i(t_k)$
3. Define $S_k(t_k) = \Gamma_m(t_k)^T \mathrm{Ad}^{-T}(m_{ik}(t_k))\Sigma_{k*ik}(t_k)^{-1}\mathrm{Ad}^{-1}(m_{ik}(t_k))\Gamma_k(t_k)$
4. $\bar{S}'(t_k) = S_i(t_k) + \sum_{k=1}^{n} S_k(t_k)$, $\bar{x}'(t_k) = \bar{S}'(t_k)^{-1}\sum_{k=1}^{n} S_k(t_k)x_k(t_k)$

Then

$$\boxed{\Sigma_i(t_{k+1}^{+}) = \bar{\Gamma}'(t_k)\bar{S}'(t_k)^{-1}\bar{\Gamma}'(t)^{t_k}, \quad \mu_i(t_{k+1}^{+}) = \mu_i(t_{k+1}^{-})\exp(-\hat{\bar{x}}'(t_k))} \tag{8.38}$$

8.4 Simulation and Discussions

This section provides verification for the proposed technique in a Matlab-simulated environment. A team of two-wheeled differential drive robots are moving in the field. The given inputs are such that all robots move along a straight line or a circular arc. However, due to the stochastic nature of the systems, errors accumulate over time such that odometry or dynamics alone is insufficient in estimating the robot poses. The results from the previous sections can therefore be used to update the robots' knowledge of their poses with the help of measuring their positions relative to neighboring robots.

Figure 8.1 depicts a simple model of the two-wheeled differential drive robot which is very useful in modeling segway-like mobile bases and various multi-robot experimental platforms (E-pucks, iRobot create, Khepera, etc.). According to [10], the kinematics of such a mobile robot can be characterized by the stochastic differential equation

$$(g^{-1}\dot{g})^{\vee}dt = \begin{bmatrix} \frac{r}{2}(\omega_1+\omega_2) \\ 0 \\ \frac{r}{2}(\omega_1-\omega_2) \end{bmatrix} dt + \sqrt{D}\begin{bmatrix} \frac{r}{2} & \frac{r}{2} \\ 0 & 0 \\ \frac{r}{l} & -\frac{r}{l} \end{bmatrix}\begin{bmatrix} dw_1 \\ dw_2 \end{bmatrix}, \tag{8.39}$$

where $g \in \mathrm{SE}(2)$ is the homogenous matrix representing the pose of the robot, r is the wheel radius, l is the axle length, ω_1, ω_2 are the wheel angular velocities, dw_i

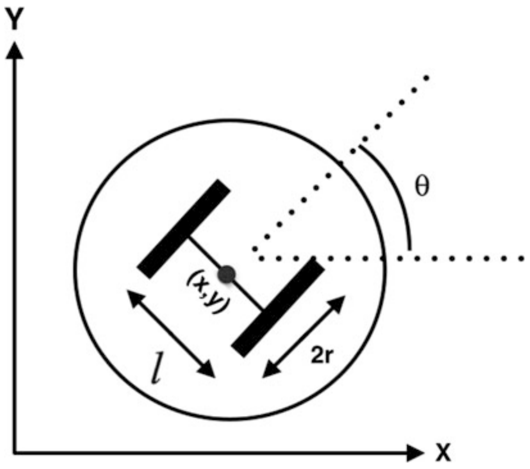


Fig. 8.1 Simple model for a two-wheeled differential drive mobile system

are unit strength Wiener processes, and D is a noise coefficient. This stochastic differential system can be simulated using Euler–Maruyama method described in [7]. Equation (8.39) can be written in short as

$$(g^{-1}\dot{g})^{\vee}dt = \mathbf{h}dt + \mathbf{H}d\mathbf{w} \tag{8.40}$$

when g is close to the identity, given an input pair $[\omega_1, \omega_2]^T$, the mean and covariance of system (8.39) can be estimated by

$$\begin{aligned} \mu(t) &= \exp\left(\int_0^t \hat{\mathbf{h}}\, \mathrm{d}\tau\right), \\ \Sigma(t) &= \int_0^t \mathrm{Ad}(\mu^{-1}(t-\tau))\mathbf{H}\mathbf{H}^T Ad^T(\mu^{-1}(t-\tau))\, \mathrm{d}\tau. \end{aligned} \tag{8.41}$$

For simple motions like straight-line motion when $\omega_1 = \omega_2$, (8.41) can be evaluated analytically as

$$\mu(t)_{\mathrm{st}} = \begin{bmatrix} 1 & 0 & r\omega t \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \Sigma(t)_{\mathrm{st}} = \begin{bmatrix} \frac{1}{2}Dr^2 t & 0 & 0 \\ 0 & \frac{2D\omega^2 r^4 t^3}{3l^2} & \frac{D\omega r^3 t^2}{l^2} \\ 0 & \frac{D\omega r^3 t^2}{l^2} & \frac{2Dr^2 t}{l^2} \end{bmatrix}. \tag{8.42}$$

The same can be done with circular motion of constant curvature

$$\mu(t)_{\mathrm{cir}} = \begin{bmatrix} \cos(\dot{\alpha}t) & -\sin(\dot{\alpha}t) & a\sin(\dot{\alpha}t) \\ \sin(\dot{\alpha}t) & \cos(\dot{\alpha}t) & a(1-\cos(\dot{\alpha}t)) \\ 0 & 0 & 1 \end{bmatrix}, \quad \Sigma(t)_{\mathrm{cir}} = \begin{bmatrix} \sigma_{11} & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22} & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{bmatrix}. \tag{8.43}$$

where

$$\begin{aligned} \sigma_{11} &= \frac{c}{8}[(4a^2 + l^2)(2\dot{\alpha}t + \sin(2\dot{\alpha}t)) + 16a^2(\dot{\alpha}t - \sin(2\dot{\alpha}t))], \\ \sigma_{12} = \sigma_{21} &= \frac{-c}{2}[4a^2(-1 + \cos(2\dot{\alpha}t)) + l^2]\sin(\dot{\alpha}t/2)^2, \\ \sigma_{13} = \sigma_{31} &= 2ca(\dot{\alpha}t - \sin(2\dot{\alpha}t)), \\ \sigma_{23} = \sigma_{32} &= -2ca(-1 + \cos(\dot{\alpha}t)), \\ \sigma_{33} &= 2c\dot{\alpha}, \\ c &= \frac{Dr^2}{l^2\dot{\alpha}}. \end{aligned} \tag{8.44}$$

With the pose priors calculated with (8.41)–(8.44), (8.36)–(8.38) are then applied with sensor measurements to update the priors. For arbitrary inputs (ω_1, ω_2) an approximation will be applied to (8.41) which will be discussed in the next section.

This example simulates localization of a robot team in straight and circular motion. In the setup of this simulation, the model based parameters are set as $r = 0.033, l = 0.2$. The simulation parameters for straight-line motion are $D = 5, v = 0.5, T = 1.3, \omega_1 = \omega_2 = \frac{v}{r}$, and $T = 2, \omega_1 = 26.166, \omega_2 = 21.408$ (for circular motion). The true robot motions are simulated 500 times using the Euler–Maruyama method [7] and the end position of each trial is plotted in the following figures. It can be observed that the posterior of such a stochastic differential system (SDE) results in a banana shaped distribution as is also discussed in [15].

In this simulation, all four robots are given the command to travel in a straight line for 1.3 % at 0.5 m/s or along an arc of constant curvature of 1 m at 45 deg/s for 1 s. The blue dashed lines in the figures represent the desired path of travel with the blue points at the two ends representing initial to final position. However, due to process noise each robot will eventually end up somewhere near the desired goal and our objective is to estimate its true position along with a quantification of our confidence of this estimate. Specifically for this example, the true pose of the middle robot (cyan colored) is what we are trying to estimate which we will call robot i, while the neighboring robots (yellow) are members of this team where relative measurements are obtained from. Among all the sampled end positions, one position for each robot is chosen as the true end pose (red point) and this is used to generate the mean of the measurement distribution m_{in}.

As the first step, the prior mean and covariance of robot i is calculated using (8.42) and (8.43), and plotted in Fig. 8.2a, b, the resultant prediction aligns perfectly with the desire path (blue dash line), and the error “ellipse” marginalized over the heading angle is also plotted from the calculated covariance (magenta loop). Since this error “ellipse” is a contour of the resultant distribution, it can be observed that a Gaussian distribution under exponential coordinates is a much better fit for characterizing the uncertainties in an SDE of this kind than that under Cartesian coordinates. It is obvious that this prediction gives the same resultant distribution regardless of the true position and is only effected by the system dynamics and input commands. Therefore the next step is to update this prediction with measurements relative to neighboring robots.

It is assumed that robot i can exchange information with its neighbor when it comes into its sensing range, which means when a relative measurement is taken of neighbor j relative to robot i, the belief (mean and covariance in this case) that j holds for its current position can be communicated to i so that i can make use of this information in its localization process (update step). In this example, this belief (μ_j, Σ_j) for each neighboring robot j is taken to be the pose prediction calculated from (8.42) or (8.43), but in reality this can very well be the posterior from its own localization results. The covariance of the measurement distribution is chosen to be

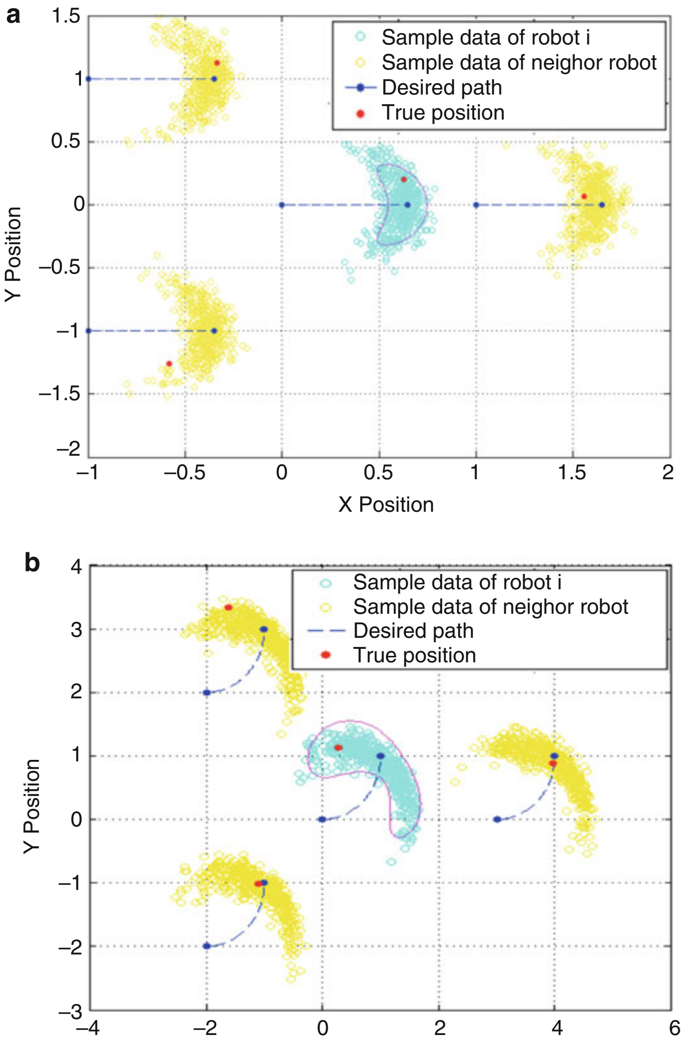
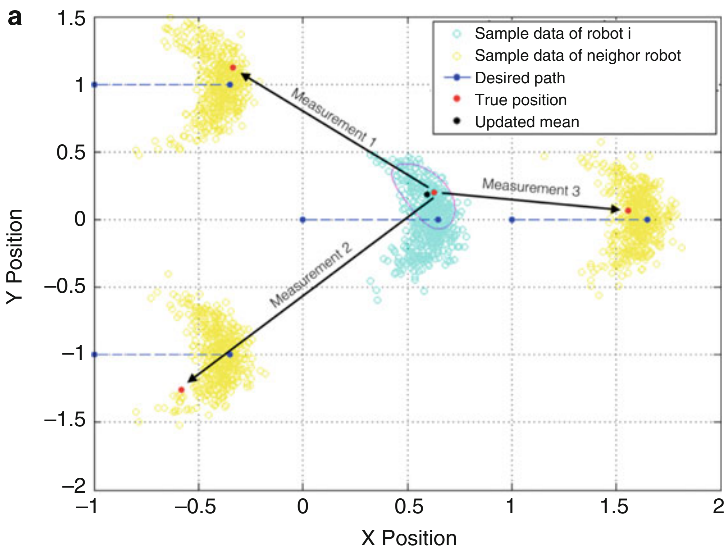


Fig. 8.2 Localization with only the prediction model. (**a**) Straight-line motion. (**b**) Circular motion

$$\Sigma_m = \begin{bmatrix} 0.01 & 0.02 & 0.001 \\ 0.02 & 0.25 & 0.015 \\ 0.002 & 0.0025 & 0.15 \end{bmatrix}.$$

Figures 8.3a, b show the updated posterior of robot i calculated from fusing the relative measurements taken from its three neighbors. The result indicates a more accurate position mean (black dot) and a shrinked error "ellipse" representing higher



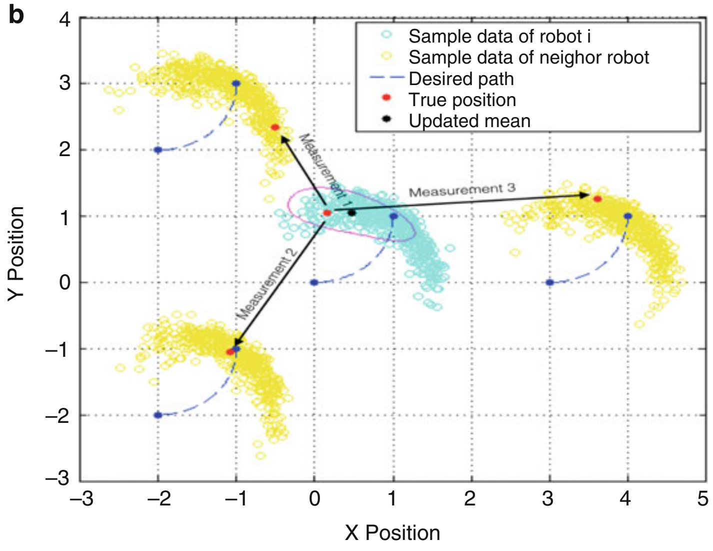


Fig. 8.3 Pose update after sensor measurement and fusion. (**a**) Straight-line motion. (**b**) Circular motion

confidence in the estimate. Since this is a distributed localization technique aimed to be implemented on the embedded processor of each individual robot, the procedure is demonstrated only for one robot and the same goes for all other.

8.5 Conclusion

This chapter proposed a distributed cooperative localization technique that can incorporate multiple sensor measurements to achieve higher estimation accuracy. Robots in a team can take measurements and exchange information among each other to update their knowledge of the current position. Simulation is used to validate the performance of the approach in Matlab. Results from the Matlab simulation show a good localization accuracy of the presented approach. The proposed multi-robot localization technique is distributed in that each robot can perform this localization process without the help of a centralized processor and is scalable for the computation time does not increase as the robot team enlarges and increases only linearly with the number of measurements taken. The generality of this scheme lies in the fact that uncertainties in the belief of the current robot, neighboring robots, and sensor measurements have all been considered which yields a more realistic estimate. Unlike sampling-based approaches, the proposed approach provides closed-form expressions which significantly increases computational efficiency. Most existing cooperative localization schemes possess a subset of the above attributes but rarely all. Lastly, this technique is of second order in its estimation of an updated posterior which is expected to be more accurate and reliable than first-order methods.

The limitation of this method is its dependency on Gaussian noises. Moreover, at present this is a local technique in that it depends on known initial poses and does not recover from localization failures (defined by [15]). In its current state, this approach does not possess the ability to serve as the sole scheme to localize a team of robots in that as errors accumulate in the beliefs of neighboring robots, erroneous information will be given to the current robot that leads to localization failures. However, this technique is local and prone to error accumulation only when none of the member robots have a reasonable estimate of their positions. As long as one robot possesses a good knowledge of its current pose (via more accurate sensors or sophisticated but computationally expensive algorithms) then this information can be used to drastically reduce the uncertainty of the entire team which introduces a level of robustness to the system and can also significantly reduce hardware and computational cost of the team. Table 8.2 shows a comparison of the proposed method with two of the most representative and accepted approaches.

The accuracy of the exponential localization method is expected to see great increase compared to results shown previously if the algorithm parameters (initial pose covariance, process and measurement noise covariances, etc.) can be fine-tuned. Establishing a systematic way of tuning these parameters can be a topic of its own. It is also incredibly beneficial if the proposed method can be combined with sampling-based approaches for their global localization and state recovery abilities. Lastly, experiments on hardware are required to fully establish the advantage of the proposed scheme. Overall this chapter has provided an alternative distributed cooperative localization technique in the domain of *Lie group* and *exponential*

Table 8.2 Final comparison

	Distributed EKF	Multi-Robot MCL	Exponential localization
Restrictions on error distribution	Requires Gaussian process and measurement error	Nonparametric particle representation of posterior belief, no assumptions on noise distribution	Gaussian
Global localization	No	Yes	No
State recovery	No	Possible given a well-designed resampling process	No
Localization accuracy	Accurate when error is small	Depends on the number of particles used	Accurate within an error range
Computational cost	Small due to the closed-form propagation and update equations	Depends on the number of particles. Can increase dramatically with the dimension of the state space	Small due to closed form equations
Ease of implementation	Simple	Can be involved	Simple
Robustness	Prone to error due to linearization	Quite resistant to errors given multiple beliefs are maintained simultaneously	Less susceptible to errors given the well conformity to the motion model
Process multiple detections	No	No	Yes
Complexity relative to team size	Fully distributed. Complexity independent of team size	Complexity independent of team size	Linear to the number of measurements processed

coordinates and has validated in simulation the potential of this technique as the next state of the art.

Acknowledgements This work was supported by NSF Grant RI-Medium: IIS-1162095.

References

1. Barooah P, Russell J, Hespanha J (2010) Approximate distributed Kalman filtering for cooperative multi-agent localization. In: Distributed computing in sensor systems, vol 6131. Springer, Heidelberg, pp 102–115
2. Caglioti V, Citterio A, Fossati A (2006) Cooperative, distributed localization in multi-robot systems: a minimum-entropy approach. In: Distributed intelligent systems: collective intelligence and its applications, pp 20–30
3. Carlone L, Kaouk M, Du J, Bona B, Indri M (2011) Simultaneous localization and mapping using Rao-Blackwellized particle filters in multi robot systems. J Intell Robot Syst 63:283–307
4. Chirikjian G (2012) Stochastic models, information theory, and lie groups, vol 2. Birkhäuser, Boston
5. Chirikjian G, Kobilarov M (2014) Gaussian approximation of non-linear measurement models on Lie groups. IEEE conference on decision and control, pp 6401–6406
6. Fox D, Burgard W, Kruppa H, Thrun S (2000) A probabilistic approach to collaborative multi-robot localization. Auton Robot 8(3):325–344
7. Higham H (2001) An algorithmic introduction to numerical simulation of stochastic differential equations. SIAM Rev 43:525–546

8. Howard A (2006) Multi-robot simultaneous localization and mapping using particle filters. Int J Robot Res 25(12):1243–1256
9. Liu J, Yuan K, Zou W, Yang Q (2005) Monte Carlo multi-robot localization based on grid cells and characteristic particles. In: International conference on advanced intelligent mechatronics, pp 510–515
10. Long A, Wolfe K, Mashner M, Chirikjian G (2012) The banana distribution is Gaussian: a localization study with exponential coordinates. In: Robotics: science and systems
11. Marjovi A, Marques L, Penders J (2009) Guardians robot swarm exploration and firefighter assistance. In: IEEE/RSJ international conference on intelligent robots and systems: workshop on network robot systems
12. Mourikis AI, Roumeliotis SI (2006) Performance analysis of multirobot cooperative localization. IEEE Trans Robot 22(4):666–681
13. Park W, LIu Y, Zhou Y, Moses M, Chirikjian G (2008) Kinematic state estimation and motion planning for stochastic nonholonomic systems using the exponential map. Robotica 26:419–434
14. Roumeliotis S, Bekey G (2002) Distributed multirobot localization. IEEE Trans Robot Autom 18(5):781–795
15. Thrun S, Burgard W, Fox D (2006) Probabilistic robotics. MIT Press, Cambridge
16. Wang Y, Chirikjian G (2008) Nonparametric second-order theory of error propagation on motion groups. Int J Robot Res 27(11–12):1258–1273

Part III
Shapes, Surfaces, and Trajectories

This part presents fundamental representations and algorithms for analysis of trajectories, shapes, surfaces in 2D and 3D data.

Chapter 9
Covariance Weighted Procrustes Analysis

Christopher J. Brignell, Ian L. Dryden, and William J. Browne

Abstract We revisit the popular Procrustes matching procedure of landmark shape analysis and consider the situation where the landmark coordinates have a completely general covariance matrix, extending previous approaches based on factored covariance structures. Procrustes matching is used to compute the Riemannian metric in shape space and is used more widely for carrying out inference such as estimation of mean shape and covariance structure. Rather than matching using the Euclidean distance we consider a general Mahalanobis distance. This approach allows us to consider different variances at each landmark, as well as covariance structure between the landmark coordinates, and more general covariance structures. Explicit expressions are given for the optimal translation and rotation in two dimensions and numerical procedures are used for higher dimensions. Simultaneous estimation of both mean shape and covariance structure is difficult due to the inherent non-identifiability. The method requires the specification of constraints to carry out inference, and we discuss some possible practical choices. We illustrate the methodology using data from fish silhouettes and mouse vertebra images.

9.1 Introduction

The need to describe and compare the shapes of objects has become very important in a wide variety of fields, including computer vision, biology, medicine and bioinformatics [3, 6, 14, 22]. The shape of an object is defined as the geometrical information that remains when location, rotational and scale effects are filtered out [6, 10, 11, 16, 18, 19]. However, the analysis of shapes is not straightforward as the non-linear constraints due to rotation and scale invariance lead to shape spaces having non-linear geometry. Riemannian manifolds are the natural home for developing suitable statistical methodology.

C.J. Brignell (✉) • I.L. Dryden
The University of Nottingham, University Park, Nottingham, NG7 2RD, UK
e-mail: chris.brignell@nottingham.ac.uk; ian.dryden@nottingham.ac.uk

W.J. Browne
University of Bristol, Bristol, BS8 1TX, UK
e-mail: william.browne@bristol.ac.uk

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_9

A key aim of statistical shape analysis is the estimation of mean shape and shape variability on Riemannian manifolds, which can then be incorporated into a prior model for Bayesian image analysis, or for carrying out investigations into the statistical significance of differences between populations. Commonly an object can be represented by a finite number of points, called landmarks, that are points of correspondence between and within populations. Probably the most widely used method of statistical shape analysis is Procrustes analysis [8, 9]. Procrustes analysis involves the matching of objects by minimising the Euclidean distance between the configurations, i.e. objects are translated, rotated and rescaled so that their landmarks match as closely as possible using a least squares criterion. The space in which the objects lie after removing translation, rotation and scale is a quotient space, and in particular is called Kendall's shape space, which is endowed with a Riemannian metric and positive curvature [12]. The quotienting out of the similarity transformations involves optimal estimation of translation, rotation and scale using Procrustes matching. Practical implementation of Procrustes methods is available in the `shapes` library in R, for example using the command `procGPA` [5]. However, Procrustes analysis is almost always carried out under the assumption of isotropic covariance structures, e.g. measurement error models, which is often rather dubious as a model for a general population of objects. In this chapter we consider extensions where completely general covariance structures are handled. The familiar Riemannian distance in shape space is replaced with a Mahalanobis type of distance.

9.2 Covariance Weighted OPA

9.2.1 Weighted Procrustes

Procrustes methods have been extended from the usual isotropic case by Goodall [8] by using weighted least squares (WLS) and iteratively reweighted least squares (IRLS). Within WLS the covariance structure is assumed known, but in IRLS is unknown and estimated. Generally [8] uses a factored covariance structure $\Sigma = \Sigma_m \otimes \Sigma_k$, where Σ_m is the covariance matrix at each landmark and Σ_k is the covariance matrix across landmarks. This structure means one can have non-identical variability at each landmark and non-identical variability between the dimensions, but this formulation is not completely general. The method when $\Sigma_m = I_m$ replaces configurations X by QX and μ by $Q\mu$, where Q^TQ is the Cholesky decomposition of Σ_k^{-1}. This accounts for the centring, and the rotation is estimated as with i.i.d. covariance. When $\Sigma_m \neq I_m$, from [8] there is no known explicit expression for the Procrustes rotation. In this case the iterative algorithm of [13] can be used, where they consider a diagonal weights matrix, D. Theobald and Wuttke [20] also consider the factored covariance model and use an empirical Bayes procedure for inference. However, our aim is to consider Procrustes methods for a completely general covariance matrix, although we do also consider factored covariance structures as special cases.

9.2.2 Partial Covariance Weighted OPA

Definition 9.1. The method of partial covariance weighted ordinary Procrustes analysis (partial CW OPA) involves the least squares matching of one configuration to another using rigid-body transformations. Estimation of the translation and rotation parameters, γ and Γ, is carried out by minimising the Mahalanobis norm

$$D^2_{\mathrm{pCWP}}(X, \mu; \Sigma) = \|\mu - X\Gamma - 1_k \gamma^T\|^2_{\Sigma}, \tag{9.1}$$

where Σ $(km \times km)$ is a symmetric positive definite matrix, γ is an $m \times 1$ location vector and Γ is an $m \times m$ special orthogonal rotation matrix, and

$$\|X\|^2_{\Sigma} = \mathrm{vec}(X)^T \Sigma^{-1} \mathrm{vec}(X).$$

The translation which minimises Eq. (9.1) is given by Result 9.2.1. In general, the minimising rotation is solved numerically; however, when $m = 2$ there is only one rotation angle and a solution is given by Result 9.2.2.

Result 9.2.1. *Given two configuration matrices, X and μ, and a symmetric positive definite matrix, Σ, the translation, as a function of rotation, which minimises the Mahalanobis norm, $D^2_{\mathrm{pCWP}}(X, \mu; \Sigma)$, is*

$$\hat{\gamma} = [(I_m \otimes 1_k)^T \Sigma^{-1} (I_m \otimes 1_k)]^{-1} (I_m \otimes 1_k)^T \Sigma^{-1} \mathrm{vec}(\mu - X\Gamma). \tag{9.2}$$

Result 9.2.2. *Consider $m = 2$, let $A = [(I_m \otimes 1_k)^T \Sigma^{-1} (I_m \otimes 1_k)]^{-1} (I_m \otimes 1_k)^T \Sigma^{-1}$ and denote the partitioned submatrices as*

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \qquad X = \begin{bmatrix} X_1 & X_2 \end{bmatrix}, \qquad \mu = \begin{bmatrix} \mu_1 & \mu_2 \end{bmatrix},$$

where the A_{ij} have dimension $(1 \times k)$ and X_i, μ_i have dimension $(k \times 1)$ for $i, j = 1, 2$, then given two configuration matrices, X and μ, and a symmetric positive definite matrix Σ, the rotation which minimises the Mahalanobis norm, $D^2_{\mathrm{pCWP}}(X, \mu; \Sigma)$, is given by

$$\cos\hat{\theta} = \frac{S(2\lambda - 2Q) + TR}{(2\lambda - 2P)(2\lambda - 2Q) - R^2},$$
$$\sin\hat{\theta} = \frac{T(2\lambda - 2P) + SR}{(2\lambda - 2P)(2\lambda - 2Q) - R^2}, \tag{9.3}$$

where

$$P = \begin{bmatrix} (X_1 + 1_k\delta_1) \\ (X_2 + 1_k\delta_2) \end{bmatrix}^T \Sigma^{-1} \begin{bmatrix} (X_1 + 1_k\delta_1) \\ (X_2 + 1_k\delta_2) \end{bmatrix},$$

$$\begin{aligned}
Q &= \begin{bmatrix} (X_2 - 1_k\zeta_1) \\ -(X_1 + 1_k\zeta_2) \end{bmatrix}^T \Sigma^{-1} \begin{bmatrix} (X_2 - 1_k\zeta_1) \\ -(X_1 + 1_k\zeta_2) \end{bmatrix}, \\
R &= -2\begin{bmatrix} (X_1 + 1_k\delta_1) \\ (X_2 + 1_k\delta_2) \end{bmatrix}^T \Sigma^{-1} \begin{bmatrix} (X_2 - 1_k\zeta_1) \\ -(X_1 + 1_k\zeta_2) \end{bmatrix}, \\
S &= -2\begin{bmatrix} (X_1 + 1_k\delta_1) \\ (X_2 + 1_k\delta_2) \end{bmatrix}^T \Sigma^{-1} \begin{bmatrix} (\mu_1 - 1_k\alpha_1) \\ (\mu_2 - 1_k\alpha_2) \end{bmatrix}, \qquad (9.4) \\
T &= 2\begin{bmatrix} (X_2 - 1_k\zeta_1) \\ -(X_1 + 1_k\zeta_2) \end{bmatrix}^T \Sigma^{-1} \begin{bmatrix} (\mu_1 - 1_k\alpha_1) \\ (\mu_2 - 1_k\alpha_2) \end{bmatrix}, \\
\alpha_i &= A_{i1}\mu_1 + A_{i2}\mu_2,\ \delta_i = -A_{i1}X_1 - A_{i2}X_2,\ \zeta_i = A_{i1}X_2 - A_{i2}X_1,
\end{aligned}$$

and λ is the real root less than $\frac{1}{2}\left(P + Q - \sqrt{(P-Q)^2 + R^2}\right)$ of the quartic equation:

$$\begin{aligned}
&16\lambda^4 - 32(P+Q)\lambda^3 + [16(P^2 + Q^2) + 64PQ - 4(S^2 + T^2) - 8R^2]\lambda^2 \\
&+[8R^2(P+Q) - 32PQ(P+Q) + 8(QS^2 + PT^2 - STR)]\lambda \qquad (9.5) \\
&+16P^2Q^2 + R^4 - R^2(S^2 + T^2) + 4RST(P+Q) \\
&-4P^2T^2 - 4Q^2S^2 - 8PQR^2 = 0.
\end{aligned}$$

Note that a unique solution of Eq. (9.5) that satisfies the constraint may not exist and it may be necessary to evaluate $D^2_{\text{pCWP}}(X, \mu; \Sigma)$ for several choices of λ or use numerical methods. In our experience, however, this is rarely required.

We will now demonstrate the use of CW Procrustes registration in practice. Figure 9.1 shows the partial ordinary covariance weighted Procrustes registration of one second thoracic mouse vertebra to another for three different weighting matrices, Σ. Note there are $k = 6$ landmarks in $m = 2$ dimensions taken at points of maximum curvature from a cross section. The algorithm for extracting the points from the original microscope images was described by Dryden [4, Chap. 5]. Starting at the far left and going clockwise, the landmarks are numbered: 4, 3, 2, 6, 1, 5. The weighting matrices used are

$$\Sigma_1 = I_{km}, \qquad \Sigma_2 = I_m \otimes \Sigma_k, \qquad \Sigma_3 = \Sigma_m \otimes \Sigma_k, \qquad (9.6)$$

where

$$\Sigma_m = \begin{bmatrix} 10 & 0 \\ 0 & 0.1 \end{bmatrix}, \qquad \Sigma_k = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10 & 0 & -9 \\ 0 & 0 & 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & -9 & 0 & 10 \end{bmatrix}.$$

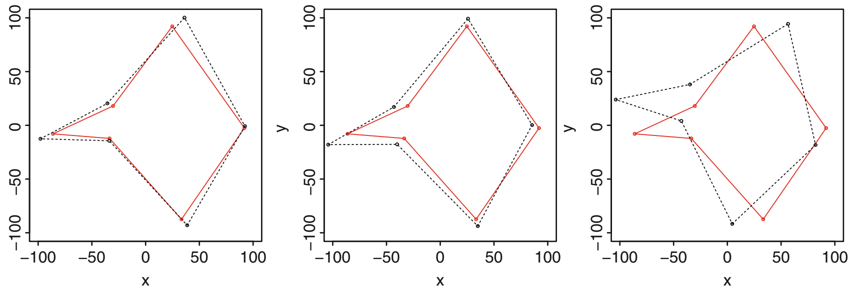


Fig. 9.1 The partial ordinary covariance weighted Procrustes registration of a mouse vertebra (*dotted*) to a template, using Σ_1 (*left*), Σ_2 (*middle*) and Σ_3 (*right*)

The first weighting matrix is the same as isotropic OPA, giving the same weighting to all coordinates and minimising the total distance between all pairs of landmarks. The second choice weights landmarks one and two more heavily than the others, and the third weighting matrix goes a stage further, and weights the y direction more heavily than the x direction.

9.2.3 Full Covariance Weighted OPA

Definition 9.2. The method of full covariance weighted ordinary Procrustes analysis (full CW OPA) involves the least squares matching of one configuration to another using similarity transformations. Estimation of the translation, rotation and scaling parameters, γ, Γ and β, is carried out by minimising the Mahalanobis norm

$$D^2_{\text{CWP}}(X, \mu; \Sigma) = \|\mu - \beta X \Gamma - 1_k \gamma^T\|^2_{\Sigma}, \tag{9.7}$$

where Σ ($km \times km$) is a symmetric positive definite matrix, γ is a $m \times 1$ location vector, Γ is an $m \times m$ special orthogonal rotation matrix and $\beta > 0$ is a scale parameter.

In general, the minimising rotation is solved numerically and the minimising translation and scaling are given by Result 9.2.3. However, when $m = 2$ all the similarity transformation parameters can be obtained by Result 9.2.4.

Result 9.2.3. *Given two configuration matrices, X and μ, and a symmetric positive definite matrix, Σ, the translation and scaling, as a function of rotation, which minimise the Mahalanobis norm, $D^2_{\text{CWP}}(X, \mu; \Sigma)$, are*

$$\begin{bmatrix} \hat{\gamma} \\ \hat{\beta} \end{bmatrix} = B^{-1} \begin{bmatrix} (I_m \otimes 1_k)^T \Sigma^{-1} \text{vec}(\mu) \\ \text{vec}(X\Gamma)^T \Sigma^{-1} \text{vec}(\mu) \end{bmatrix}, \tag{9.8}$$

where

$$B = \begin{bmatrix} (I_m \otimes 1_k)^T \Sigma^{-1}(I_m \otimes 1_k) & (I_m \otimes 1_k)^T \Sigma^{-1}\text{vec}(X\Gamma) \\ \text{vec}(X\Gamma)^T \Sigma^{-1}(I_m \otimes 1_k) & \text{vec}(X\Gamma)^T \Sigma^{-1}\text{vec}(X\Gamma) \end{bmatrix}.$$

Result 9.2.4. *Consider $m = 2$, let $A = [(I_m \otimes 1_k)^T \Sigma^{-1}(I_m \otimes 1_k)]^{-1}(I_m \otimes 1_k)^T \Sigma^{-1}$ and define P, Q, R, S and T as in Eq. (9.4), then given two configuration matrices, X and μ, and a symmetric positive definite matrix, Σ, the similarity transformation parameters which minimise the Mahalanobis norm, $D^2_{\text{CWP}}(X, \mu; \Sigma)$, are given by*

$$\hat{\gamma} = A\text{vec}(\mu - \hat{\beta}X\hat{\Gamma}), \qquad \hat{\beta} = {}^{+}\sqrt{\psi_1^2 + \psi_2^2},$$

$$\cos\hat{\theta} = \frac{\psi_1}{{}^{+}\sqrt{\psi_1^2 + \psi_2^2}}, \qquad \sin\hat{\theta} = \frac{\psi_2}{{}^{+}\sqrt{\psi_1^2 + \psi_2^2}},$$

where

$$\psi_1 = \frac{RT - 2QS}{4PQ - R^2}, \qquad \psi_2 = \frac{RS - 2PT}{4PQ - R^2}. \tag{9.9}$$

9.2.4 Special Case: $\Sigma = I_m \otimes \Sigma_k$

Note that if X and μ are replaced by $X_Q = QX$ and $\mu_Q = Q\mu$ respectively, where $\Sigma_k^{-1} = Q^T Q$ is the Cholesky decomposition, then the estimates of $\hat{\Gamma}$ and $\hat{\beta}$ for matching X to μ with $\Sigma = I_m \otimes \Sigma_k$ are equivalent to the estimates of $\hat{\Gamma}$ and $\hat{\beta}$ for matching X_Q to μ_Q with $\Sigma = I_{km}$, as claimed by Goodall [8]. The above results reduce to the same estimators, and for example we consider the $m = 2$ case explicitly in the next result.

Result 9.2.5. *If $m = 2$ and Σ is of the form $I_m \otimes \Sigma_k$, where $\Sigma_k(k \times k)$ is a symmetric positive definite matrix, and assume without loss of generality that X and μ are located such that $1_k^T \Sigma_k^{-1} X = 0 = 1_k^T \Sigma_k^{-1}\mu$, then*

$$\hat{\gamma} = 0, \qquad \hat{\beta} = \frac{(S^2 + T^2)^{1/2}}{2P},$$

$$\cos\hat{\theta} = \frac{-S}{(S^2 + T^2)^{1/2}}, \qquad \sin\hat{\theta} = \frac{-T}{(S^2 + T^2)^{1/2}},$$

where

$$P = X_1^T \Sigma_k^{-1} X_1 + X_2^T \Sigma_k^{-1} X_2,$$
$$S = -2(X_1^T \Sigma_k^{-1}\mu_1 + X_2^T \Sigma_k^{-1}\mu_2),$$
$$T = 2(X_2^T \Sigma_k^{-1}\mu_1 + X_1^T \Sigma_k^{-1}\mu_2).$$

9.3 Covariance Weighted GPA

9.3.1 Definition and Algorithm

In Sect. 9.2 covariance weighted ordinary Procrustes analysis was defined as obtaining the Procrustes estimators to minimise the Mahalanobis norm $D^2_{\mathrm{CWP}}(X, \mu; \Sigma) = \|\mu - X^P\|^2_{\Sigma}$, where one configuration, X, is translated, rotated and possibly scaled with respect to a reference configuration, μ. In this section CW OPA is extended to define covariance weighted generalised Procrustes analysis for a more general data set with $n \geq 2$ configurations, $X_1, X_2, \ldots, X_n$. This allows inferences to be made regarding the sample mean shape.

Definition 9.3. The method of full covariance weighted generalised Procrustes analysis (full CW GPA) involves the least squares matching of n configurations relative to each other using similarity transformations, and the procedure is appropriate under the model

$$X_i = \beta_i(\mu + E_i)\Gamma_i + 1_k\gamma_i^T,$$

where E_i are zero mean $k \times m$ independent random error matrices, μ is the $k \times m$ matrix of the mean configuration and γ_i, Γ_i and β_i are nuisance parameters for translation, rotation and scale. A quantity proportional to the sum of squared Mahalanobis norms of pairwise differences,

$$G_{\mathrm{CWP}}(X_1, \ldots, X_n; \Sigma) = \frac{1}{n}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \left\|(\beta_i X_i \Gamma_i + 1_k\gamma_i^T) - (\beta_j X_j \Gamma_j + 1_k\gamma_j^T)\right\|^2_{\Sigma}, \tag{9.10}$$

is minimised subject to the constraint on the size of the centred average shape, $\|\bar{X}\|^2 = 1$, where $\Gamma_i \in SO(m)$, $\beta_i > 0$ and

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n}(\beta_i X_i \Gamma_i + 1_k\gamma_i^T).$$

Partial covariance weighted generalised Procrustes analysis can be similarly defined using rigid-body transformations and without a constraint on the size of the mean shape. Minimising the sum of squared Mahalanobis norms of pairwise differences is equivalent to minimising the distance between each configuration and the mean of the configurations:

$$\begin{aligned} G_{\mathrm{CWP}}(X_1, \ldots, X_n; \Sigma) &= \inf_{\beta_i, \Gamma_i, \gamma_i} \frac{1}{n}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \left\|(\beta_i X_i \Gamma_i + 1_k\gamma_i^T) - (\beta_j X_j \Gamma_j + 1_k\gamma_j^T)\right\|^2_{\Sigma} \\ &= \inf_{\beta_i, \Gamma_i, \gamma_i} \sum_{i=1}^{n} \left\|(\beta_i X_i \Gamma_i + 1_k\gamma_i^T) - \bar{X}\right\|^2_{\Sigma}. \end{aligned}$$

An alternative is to minimise the distance from the ith shape to the mean of the rest of the sample:

$$
\begin{aligned}
G_{\text{CWP}}(X_1,\ldots,X_n;\Sigma) &= \inf_{\beta_i,\Gamma_i,\gamma_i} \sum_{i=1}^{n} \left\| (\beta_i X_i \Gamma_i + 1_k\gamma_i^T) - \frac{1}{n}\sum_{j=1}^{n}(\beta_j X_j \Gamma_j + 1_k \gamma_j^T) \right\|_{\Sigma}^{2} \\
&= \inf_{\beta_i,\Gamma_i,\gamma_i} \sum_{i=1}^{n} \left\| \frac{n-1}{n}(\beta_i X_i \Gamma_i + 1_k\gamma_i^T) - \frac{1}{n}\sum_{j=1,j\neq i}^{n}(\beta_j X_j \Gamma_j + 1_k \gamma_j^T) \right\|_{\Sigma}^{2} \\
&= \inf_{\beta_i,\Gamma_i,\gamma_i} \sum_{i=1}^{n} \frac{(n-1)^2}{n^2} \left\| (\beta_i X_i \Gamma_i + 1_k\gamma_i^T) - \bar{X}_{-i} \right\|_{\Sigma}^{2},
\end{aligned}
$$

where

$$
\bar{X}_{-i} = \frac{1}{n-1} \sum_{j=1,j\neq i}^{n} (\beta_j X_j \Gamma_j + 1_k \gamma_j^T). \tag{9.11}
$$

Once the transformation parameters that minimise Eq. (9.10) have been identified, it is possible to make the following definitions:

Definition 9.4. The full covariance weighted Procrustes fit of the each X_i is given by

$$
X_i^P = \hat{\beta}_i X_i \hat{\Gamma}_i + 1_k \hat{\gamma}_i^T, \tag{9.12}
$$

where $\hat{\gamma}_i$, $\hat{\Gamma}_i$ and $\hat{\beta}_i$ are the minimising parameters of Eq. (9.10), for $i = 1,\ldots,n$.

Definition 9.5. The full covariance weighted Procrustes estimate of the mean shape is given by

$$
\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} X_i^P, \tag{9.13}
$$

where X_i^P is the full covariance weighted Procrustes fit of X_i, $i = 1,\ldots,n$.

The partial covariance weighted Procrustes fit and the partial covariance weighted Procrustes estimate of the mean shape can be similarly defined by omitting the β parameter. The similarity transformation parameters which minimise Eq. (9.10) can be obtained by the following algorithm.

The algorithm is guaranteed to converge because $G_{\text{CWP}}(X_1,\ldots,X_n;\Sigma)$ is non-increasing at each step and bounded below. Step 2 of the algorithm translates and rotates all the shapes relative to the axes to speed up the rate of convergence. The location of the mean shape is arbitrary, but the orientation of the mean shape is not. If Σ has been defined with respect to some user-specified reference

Covariance Weighted GPA Algorithm

1. Initial registration: Given symmetric positive definite matrix, Σ, and n shapes, $X_1, X_2, \ldots, X_n$, calculate $\bar{X}$ as the mean shape resulting from the isotropic GPA algorithm.

2. Centre, scale and orientate mean shape: Centre and scale $\bar{X}$ such that $S(\bar{X}) = 1$, and apply the same translation and scaling to each X_i. Further, apply an identical rotation to $\bar{X}, X_1, \ldots, X_n$ to minimise $G_{\text{CWP}}(X_1, \ldots, X_n; \Sigma)$.

3. CW OPA: For $i = 1, \ldots, n$ register X_i to $\bar{X}_{-i}$ using covariance weighted OPA to minimise $D^2_{\text{CWP}}(X_i, \bar{X}_{-i}; \Sigma)$.

4. Repetition. Repeat steps (2) and (3) until $G_{\text{CWP}}(X_1, \ldots, X_n; \Sigma)$ cannot be reduced further.

configuration, μ_0, then minimising $D_{\text{OPA}}(\bar{X}, \mu_0)$ or $D^2_{\text{CWP}}(X_i, \mu_0; \Sigma)$ instead of $G_{\text{CWP}}(X_1, \ldots, X_n; \Sigma)$ at step 2 keeps the shapes in that frame of reference. Alternatively, if Σ has been chosen with respect to the axes, then the rotation in step 2 can be calculated numerically for $m \geq 3$, or by Result 9.3.1 for $m = 2$.

Result 9.3.1. *For $m = 2$, let the rotation matrix $\Gamma = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$, then*

$$G_{\text{CWP}}(X_1, \ldots, X_n; \Sigma) = \inf_{\beta_i, \Gamma_i, \gamma_i, \Gamma} \sum_{i=1}^{n} \|(\beta_i X_i \Gamma_i + 1_k \gamma_i^T - \bar{X})\Gamma\|^2_{\Sigma}$$

is minimised when θ is a solution of

$$\tan(2\theta) = \frac{2r}{p-q},$$

where

$$p = \sum_{i=1}^{n} \text{vec}(R_i)^T \Sigma^{-1} \text{vec}(R_i), \qquad q = \sum_{i=1}^{n} \text{vec}(R_i^{\perp})^T \Sigma^{-1} \text{vec}(R_i^{\perp}),$$

$$r = \sum_{i=1}^{n} \text{vec}(R_i)^T \Sigma^{-1} \text{vec}(R_i^{\perp}),$$

$$R_i = (\beta_i X_i \Gamma_i + 1_k \gamma_i^T - \bar{X}) = \begin{bmatrix} R_{i1} & R_{i2} \end{bmatrix}, \qquad R_i^{\perp} = \begin{bmatrix} -R_{i2} & R_{i1} \end{bmatrix},$$

where $R_i^{\perp}$ is used to denote that it is orthogonal to R_i, i.e. $R_i^T R_i^{\perp} = 0$.

We now return to our example of mouse vertebrae considered in Sect. 9.2.2. Figure 9.2 shows the partial generalised covariance weighted Procrustes registration of 30 thoracic mouse vertebrae for three different weighting matrices, Σ. Two of the vertebrae were considered earlier. The three weighting matrices used are given in Eq. (9.6). The first weighting matrix is the same as isotropic GPA, giving the same weighting to all coordinates and showing approximately the same variability at

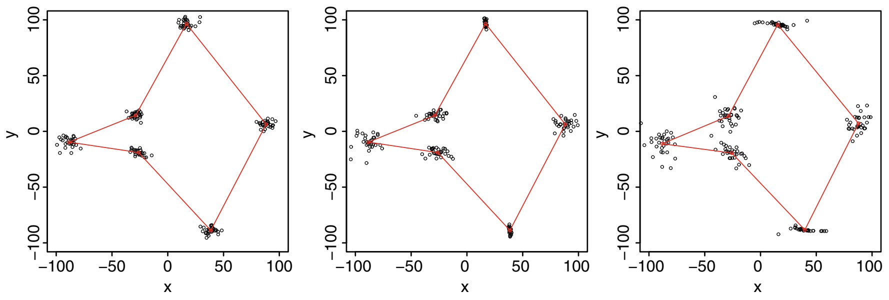


Fig. 9.2 The partial generalised covariance weighted Procrustes registration of 30 mice vertebrae, using Σ_1 (*left*), Σ_2 (*middle*) and Σ_3 (*right*). The mean shape is shown with a *solid line*

each landmark. The second choice weights landmarks 1 and 2 (bottom and top-most, respectively) more heavily than the others, and this is reflected in these landmarks forming lines pointing towards each other, and more variability being introduced elsewhere, particularly at landmark 4 (far left landmark). The third weighting matrix weights the y direction more heavily than the x direction. This forces large rotations to ensure landmarks 1 and 2 have similar y values across all configurations and gives large variability to the other landmarks.

9.3.2 Relating CW GPA to the Multivariate Normal Distribution

If the data set $X_1, X_2, \ldots, X_n$ comes from a multivariate normal distribution with unknown mean shape, μ, and known covariance matrix, Σ, that is $\mathrm{vec}(X_i) \sim N_{km}(\mathrm{vec}(\mu), \Sigma)$ then the CW GPA algorithm can be used to maximise the likelihood of the model and provide the maximum likelihood estimate (MLE) of the mean shape.

Result 9.3.2. *Maximising the likelihood of the multivariate normal model* $\mathrm{vec}(X_i) \sim N_{km}(\mathrm{vec}(\mu), \Sigma)$ *for a given* Σ *is equivalent to minimising* $G_{\mathrm{CWP}}(X_1, \ldots, X_n; \Sigma)$*, and the MLE of the mean shape is*

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} X_i^P, \tag{9.14}$$

where X_i^P *is the full covariance weighted Procrustes fit of* X_i*.*

9.4 CW GPA with Unknown Covariance Matrix

Previously, we have assumed the covariance matrix, Σ, is known or specified by the user. We now consider the more general case where Σ is unknown, and we use a maximum likelihood approach to estimating the transformation parameters, mean shape and covariance matrix for the multivariate normal model. This is subject to the constraint that Σ is positive definite, in order that the transformation parameters can be re-estimated.

Let $X_1, X_2, \ldots, X_n$ be shapes from the multivariate normal distribution $\text{vec}(X_i) \sim N_{km}(\text{vec}(\mu), \Sigma)$. The MLE of the mean shape was found in Sect. 9.3, and the MLE of Σ can be shown to be

$$\hat{\Sigma} = \frac{1}{n}\sum_{i=1}^{n} \text{vec}(X_i - \bar{X})\text{vec}(X_i - \bar{X})^T.$$

Often in applications, we wish to constrain Σ to be of the form $\Sigma_m \otimes \Sigma_k$, in which case the MLE is found by iteratively solving

$$\hat{\Sigma}_m = \frac{1}{nk}\sum_{i=1}^{n}(X_i - \bar{X})^T \hat{\Sigma}_k (X_i - \bar{X}), \qquad \hat{\Sigma}_k = \frac{1}{nm}\sum_{i=1}^{n}(X_i - \bar{X})\hat{\Sigma}_m (X_i - \bar{X})^T,$$

as shown by Dutilleul [7]. This estimate can only be solved up to a multiplicative constant, since $\Sigma_m \otimes \Sigma_k = q\Sigma_m \otimes (1/q)\Sigma_k$, for some constant $q > 0$. If Σ is singular, this parameterisation potentially removes some singularities. Other forms of constraints are possible but in general, for Σ to be positive definite, $c = r(2km - r + 1)/2$ individual entries of Σ need to specified. This means constraints can be written in the form $A\text{vech}(\Sigma) = b$ where the vech operator lists the $s = km(km + 1)/2$ distinct elements of Σ and A is a $c \times s$ matrix and b is a vector of length c.

One method of constraining Σ is to specify the amount of variability in r directions, where r is the number of transformation parameters to be estimated. Suppose $X = \beta\mu\Gamma + 1_k\gamma^T$, then after full Procrustes registration of the shape, variability due to translation, rotation and scaling has been removed. Let $m = 3$, then $r = 7$, and assuming the angles of rotations, θ_x, θ_y and θ_z are small, $\cos\theta$ and $\sin\theta$ can be approximated using the Taylor expansions to the first order, so

$$\Gamma = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & \sin\theta_x \\ 0 & -\sin\theta_x & \cos\theta_x \end{bmatrix} \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} \begin{bmatrix} \cos\theta_z & \sin\theta_z & 0 \\ -\sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & \theta_x \\ 0 & -\theta_x & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & \theta_y \\ 0 & 1 & 0 \\ -\theta_y & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & \theta_z & 0 \\ -\theta_z & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + O(\theta_x^2) + O(\theta_y^2) + O(\theta_z^2).$$

We write $O(\theta^2) = O(\theta_x^2) + O(\theta_y^2) + O(\theta_z^2)$, and we have

$$\begin{aligned}
\mathrm{vec}(X) &= \beta \mathrm{vec}\left(\left[\, \mu_x\ \mu_y\ \mu_z \,\right] \Gamma\right) + \mathrm{vec}\left(\left[\, 1_k\gamma_x\ 1_k\gamma_y\ 1_k\gamma_z \,\right]\right) \\
&= \beta \begin{bmatrix} \mu_x - \mu_y\theta_z - \mu_z\theta_y + O(\theta^2) \\ \mu_x\theta_z + \mu_y - \mu_z\theta_x + O(\theta^2) \\ \mu_x\theta_y + \mu_y\theta_x + \mu_z + O(\theta^2) \end{bmatrix} + \begin{bmatrix} 1_k\gamma_x \\ 1_k\gamma_y \\ 1_k\gamma_z \end{bmatrix} \\
&= \beta \begin{bmatrix} \mu_x \\ \mu_y \\ \mu_z \end{bmatrix} + \beta \begin{bmatrix} 0_k & -\mu_z & -\mu_y \\ -\mu_z & 0_k & \mu_x \\ \mu_y & \mu_x & 0_k \end{bmatrix} \begin{bmatrix} \theta_x \\ \theta_y \\ \theta_z \end{bmatrix} + \begin{bmatrix} 1_k & 0_k & 0_k \\ 0_k & 1_k & 0_k \\ 0_k & 0_k & 1_k \end{bmatrix} \begin{bmatrix} \gamma_x \\ \gamma_y \\ \gamma_z \end{bmatrix} + O(\theta^2) \\
&= \beta v_1 + \beta\theta_x v_2 + \beta\theta_y v_3 + \beta\theta_z v_4 + \gamma_x v_5 + \gamma_y v_6 + \gamma_z v_7 + O(\theta^2), \qquad (9.15)
\end{aligned}$$

where

$$v_1 = \begin{bmatrix} \mu_x \\ \mu_y \\ \mu_z \end{bmatrix}, \quad v_2 = \begin{bmatrix} 0_k \\ -\mu_z \\ \mu_y \end{bmatrix}, \quad v_3 = \begin{bmatrix} -\mu_z \\ 0_k \\ \mu_x \end{bmatrix}, \quad v_4 = \begin{bmatrix} -\mu_y \\ \mu_x \\ 0_k \end{bmatrix},$$

$$v_5 = \begin{bmatrix} 1_k \\ 0_k \\ 0_k \end{bmatrix}, \quad v_6 = \begin{bmatrix} 0_k \\ 1_k \\ 0_k \end{bmatrix}, \quad v_7 = \begin{bmatrix} 0_k \\ 0_k \\ 1_k \end{bmatrix}.$$

Therefore, there is approximately no variability in the direction of v_j for $j = 1, \ldots, r$ following isotropic registration. These vectors are linearly independent and, using the Gram–Schmidt process, can be transformed into r orthonormal vectors, u_j, which is simpler if the mean shape is centred. Specifying a non-zero amount of variability, σ_j, for the direction u_j, for $j = 1, \ldots, r$, will be sufficient to ensure Σ is positive definite. Note, that if $m = 2$ then $r = 4$ in the case of full Procrustes analysis, and Eq. (9.15) simplifies to $\mathrm{vec}(X) = \beta u_1 + \beta\theta u_2 + \gamma_x u_3 + \gamma_y u_4 + O(\theta^2)$ where

$$u_1 = \frac{1}{\|\mu\|} \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \quad u_2 = \frac{1}{\|\mu\|} \begin{bmatrix} -\mu_y \\ \mu_x \end{bmatrix}, \quad u_3 = \frac{1}{\sqrt{k}} \begin{bmatrix} 1_k \\ 0_k \end{bmatrix}, \quad u_4 = \frac{1}{\sqrt{k}} \begin{bmatrix} 0_k \\ 1_k \end{bmatrix}.$$

Constraining Σ to have eigenvectors aligned to the axes simplifies the estimation of the covariance weighted Procrustes estimate of the translation, as demonstrated by the following result.

Result 9.4.1. *Let $\Sigma = \sum_{j=1}^{r} \sigma_j u_j u_j^T + \sum_{j=1}^{km-r} \lambda_j v_j v_j^T$ where u_j are orthonormal vectors on which variability has been constrained to be σ_j, and λ_j and v_j are the remaining eigenvalues and eigenvectors orthogonal to u_j. Then, assuming m of the vectors u_j are the columns of $(I_m \otimes 1_k)/\sqrt{k}$, X_i are centred and $\mathrm{vec}(X_i) \sim N_{km}(\mathrm{vec}(\mu), \Sigma)$, the likelihood is maximised with translation parameter $\gamma = 0$.*

This leads to the following algorithm, which combines CW Procrustes estimates of the transformation parameters and maximum likelihood estimates of the mean shape and covariance matrix to maximise the likelihood in the space orthogonal to the constraint vectors, for the model $\mathrm{vec}(X_i) \sim N_{km}(\mathrm{vec}(\mu), \Sigma)$. Different starting points for the data may be tried to see if the likelihood can be further improved. Possible starting points may include the output of isotropic GPA or Bookstein registrations of a baseline, see [1].

1. **Centre shapes:** Centre $X_1, X_2, \ldots, X_n$.
2. **Evaluate mean shape and covariance matrix:** Calculate the maximum likelihood estimates of the mean shape, $\hat{\mu}$, and the covariance matrix, $\hat{\Sigma}$. Evaluate the projection matrix, $U = I_{km} - \sum_{j=1}^{r} u_j u_j^T$.
3. **Evaluate log-likelihood:** Calculate the variability in $\hat{\Sigma}$ orthogonal to the constraint vectors, $\tilde{\Sigma} = U\hat{\Sigma}U^T$. Replace the last r eigenvalues of $\tilde{\Sigma}$ with σ_j for $j = 1, \ldots, r$ and label it $\hat{\Sigma}^*$. Evaluate the log-likelihood as

$$\log L = -\frac{n}{2}\sum_{j=1}^{km}\log(2\pi\hat{\lambda}_j) - \frac{1}{2}\sum_{i=1}^{n}\left(\mathrm{vec}(\beta_i X_i \Gamma_i - \hat{\mu})^T(\hat{\Sigma}^*)^{-1}\mathrm{vec}(\beta_i X_i \Gamma_i - \hat{\mu})\right),$$

   where $\hat{\lambda}_j$ is the jth eigenvalue of $\hat{\Sigma}^*$.
4. **CW OPA:** Estimate β_i and Γ_i using CW OPA to register X_i to $\bar{X}$, using $\hat{\Sigma}^*$ as the covariance matrix.
5. **Repetition:** Repeat steps (2)–(4) until $\log L$ cannot be increased further.

We will now illustrate the method with an example from computer vision. Figure 9.3 shows the registration of 65 fish outlines obtained from the Kimia database of [17]. The data used here are five Gaussian perturbations of $n = 13$ fish silhouettes, where in each image $k = 20$ landmarks have been placed by hand on the outlines. Constructing shape models in this manner is useful as a prior model in computer vision [3]. The data have been registered using the isotropic GPA algorithm and the covariance weighted Procrustes algorithm. Note that if σ_j is small then the

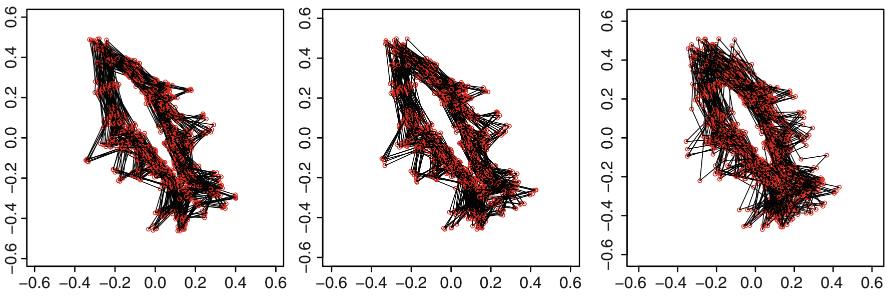


Fig. 9.3 The partial covariance weighted Procrustes registration of 65 fish outlines, using isotropic GPA (*left*) and CW Procrustes under the constraints $\sigma_j = 0.003$ (*middle*) and $\sigma_j = 0.01$ (*right*)

registration following CW Procrustes is similar to that obtained by isotropic GPA. However, if σ_j is allowed to increase then variability is allowed to remain in the directions of rotation and translation.

9.5 Simulation Study

In order to highlight the benefits compared to isotropic Procrustes analysis, CW Procrustes analysis was carried out on simulated data sets from two multivariate normal distributions, $\text{vec}(X_i) \sim N_{km}(\text{vec}(\mu), \Sigma)$. The examples were first suggested by Lele [15]:

Example 1: $\text{vec}(X) \sim N_{km}(\text{vec}(\mu_A), \Sigma_A = I_m \otimes \Sigma_k)$

Example 2: $\text{vec}(X) \sim N_{km}(\text{vec}(\mu_B), \Sigma_B = \Sigma_m \otimes \Sigma_k)$

where $\Sigma_m = \text{diag}(0.001, 1)$ and

$$\mu_A = \mu_B = \begin{bmatrix} 0 & 5 \\ 40 & 0 \\ 0 & -5 \\ -40 & 0 \end{bmatrix}, \qquad \Sigma_k = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 10 & 0 & -9.999 \\ 0 & 0 & 0.01 & 0 \\ 0 & -9.999 & 0 & 10 \end{bmatrix}.$$

A total of 130 configurations were sampled from each distribution. In Example 1 we use $\sigma_j = 25$ and in Example 2, $\sigma_j = 10$. To assess the difference between isotropic GPA and covariance weighted Procrustes, $N = 1000$ Monte Carlo samples of each example were generated. Following registration, the data were rotated to minimise the Euclidean distance between $\hat{\mu}$, the estimated mean shape, and μ, the true mean shape of the model, and $\hat{\Sigma}$ and $\hat{\Sigma}^*$ could then be calculated. This step removes the arbitrary rotation of the mean shape, and $\hat{\mu}$ was also translated to give the partial Procrustes fit onto μ. The bias of $\hat{\mu}$ and the root mean square error of $\hat{\mu}$, $\hat{\Sigma}$ and $\hat{\Sigma}^*$ were calculated using

$$\text{Bias}(\hat{A}) = \frac{1}{N}\sum_{i=1}^{N} \hat{A} - A, \qquad \text{RMSE}(\hat{A}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|\hat{A} - A\|^2},$$

where $\hat{A}$ is an estimator of the parameters in A (here equal to either μ or Σ). The results for each of the examples are given in the tables.

In Example 1, the estimates of the mean shape from both algorithms are broadly similar, largely due to the even variability in the x and y directions at each landmark. CW Procrustes, however, preserves some of the variability at the outer landmarks in the y direction, where isotropic GPA does not. The first two principal component scores from the CW Procrustes algorithm are approximately even and orthogonal to each other, as we would expect. The isotropic GPA principal components imply most of the variability is in the scaling, which is a false assessment.

Example 1	Isotropic GPA	CW Procrustes
Bias($\hat{\mu}_A$)	$\begin{bmatrix} 0 & -0.015 \\ 0.119 & 0 \\ 0 & 0.016 \\ -0.119 & 0 \end{bmatrix}$	$\begin{bmatrix} 00 & -0.033 \\ 0.018 & 0 \\ 0 & 0.033 \\ -0.018 & 0 \end{bmatrix}$
RMSE($\hat{\mu}_A$)	0.588	0.596
RMSE($\hat{\Sigma}_A$)	4.489	3.849
RMSE($\hat{\Sigma}_A^*$)	1.412	1.416

Example 2	Isotropic GPA	CW Procrustes
Bias($\hat{\mu}_B$)	$\begin{bmatrix} 0 & -0.015 \\ 0.124 & 0 \\ 0 & 0.015 \\ -0.124 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -0.01 \\ -0.026 & 0 \\ 0 & 0.011 \\ 0.026 & 0 \end{bmatrix}$
RMSE($\hat{\mu}_B$)	0.428	0.261
RMSE($\hat{\Sigma}_B$)	4.472	2.309
RMSE($\hat{\Sigma}_B^*$)	0.285	0.228

In Example 2, isotropic GPA has removed nearly all variability at the outer landmarks, because the primary source of variability is in the direction of a rotation, which isotropic GPA successfully filters out. In turn this pushes the outer landmarks further away and introduces a sizeable bias into the estimate of the mean shape. However, CW Procrustes maintains a large proportion of the variability, because it lies in the direction of a rotation, which has been constrained to a non-zero value. As a consequence, the estimate of the covariance matrix has much less error than the isotropic estimate. In summary, covariance weighted Procrustes gives much better estimates of the true covariance matrix.

9.6 Discussion

As well as the maximum likelihood-based procedure with constraints considered here, an alternative is to use a Bayesian procedure where the constraints are replaced by prior modelling assumptions. Theobald and Wuttke [20, 21] have provided a method for non-isotropic Procrustes based on an empirical Bayes procedure with factored covariance models. Their algorithm, called THESEUS, employs an inverse gamma prior model for the covariance parameters, and the parameters of the prior are estimated from the data. The advantage of using an empirical Bayes prior is that one does not need to state specific values for σ_j. However, it is worth highlighting that there is nothing in the data that gives information about what constraints should be used, so whether one chooses to fix constraints on σ_j as we have done or uses an empirical Bayes prior is completely up to the user. Alternative approaches for Bayesian covariance weighted shape analysis were given by Brignell [2, Chap. 3] using Markov chain Monte Carlo algorithms to simulate from the posterior distribution with a Wishart prior for inverse of the covariance matrix.

In conclusion, in this chapter we have revisited the popular Procrustes matching procedure of landmark shape analysis and developed the methodology to the situation where the landmark coordinates have a completely general covariance matrix. This work extends previous approaches that were based on factored covariance structures, and we have demonstrated the methodology in several examples and applications.

Acknowledgements We acknowledge the support of a Royal Society Wolfson Research Merit Award and EPSRC grant EP/K022547/1.

Appendix

Proof of Result 2.1. Let $v = \text{vec}(\mu - X\Gamma)$, then

$$\begin{aligned} D^2_{\text{pCWP}}(X, \mu; \Sigma) &= (v - (I_m \otimes 1_k)\gamma)^T \Sigma^{-1}(v - (I_m \otimes 1_k)\gamma) \\ &= v^T \Sigma^{-1} v - 2v^T \Sigma^{-1}(I_m \otimes 1_k)\gamma \\ &\quad + \gamma^T (I_m \otimes 1_k)^T \Sigma^{-1}(I_m \otimes 1_k)\gamma. \end{aligned}$$

The minimising translation is found by setting the first derivative equal to zero:

$$\frac{dD^2_{\text{pCWP}}}{d\gamma} = -2(I_m \otimes 1_k)^T \Sigma^{-1} v + 2(I_m \otimes 1_k)^T \Sigma^{-1}(I_m \otimes 1_k)\gamma = 0.$$

The second derivative is clearly positive because Σ^{-1} is positive definite. Therefore, D^2_{pCWP} is minimised when $\gamma = [(I_m \otimes 1_k)^T \Sigma^{-1}(I_m \otimes 1_k)]^{-1}(I_m \otimes 1_k)\Sigma^{-1} v$. □

Proof of Result 2.2. From Eq. (9.2) the minimising translation is $\hat{\gamma} = A\text{vec}(\mu - X\Gamma)$, so for $m = 2$,

$$\begin{bmatrix} \hat{\gamma}_1 \\ \hat{\gamma}_2 \end{bmatrix} = \begin{bmatrix} \alpha_1 + \delta_1 \cos\theta + \zeta_1 \sin\theta \\ \alpha_2 + \delta_2 \cos\theta + \zeta_2 \sin\theta \end{bmatrix}, \text{ because, } \Gamma = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}.$$

Therefore,

$$\begin{aligned} &\text{vec}(\mu - X\Gamma - 1_k\gamma^T) \\ &= \begin{bmatrix} (\mu_1 - 1_k\alpha_1) - (X_1 + 1_k\delta_1)\cos\theta + (X_2 - 1_k\zeta_1)\sin\theta \\ (\mu_2 - 1_k\alpha_2) - (X_2 + 1_k\delta_2)\cos\theta - (X_1 + 1_k\zeta_2)\sin\theta \end{bmatrix}, \end{aligned}$$

and $D^2_{\text{pCWP}}(X, \mu; \Sigma) = C + P\cos^2\theta + Q\sin^2\theta + R\cos\theta\sin\theta + S\cos\theta + T\sin\theta$ where

$$C = \begin{bmatrix} (\mu_1 - 1_k\alpha_1) \\ (\mu_2 + 1_k\alpha_2) \end{bmatrix}^T \Sigma^{-1} \begin{bmatrix} (\mu_1 - 1_k\alpha_1) \\ (\mu_2 + 1_k\alpha_2) \end{bmatrix}.$$

Let λ be the real Lagrangian multiplier to enforce the constraint $\cos^2\theta + \sin^2\theta = 1$ and let $L = D^2_{\text{pCWP}}(X, \mu; \Sigma) + \lambda(1 - \cos^2\theta - \sin^2\theta)$. Then,

$$\begin{aligned} \frac{\partial L}{\partial(\cos\theta)} &= 2(P - \lambda)\cos\theta + R\sin\theta + S = 0, \\ \frac{\partial L}{\partial(\sin\theta)} &= 2(Q - \lambda)\sin\theta + R\cos\theta + T = 0, \\ \frac{\partial L}{\partial\lambda} &= 1 - \cos^2\theta - \sin^2\theta = 0. \end{aligned}$$

Solving the first two equations simultaneously and substituting the solutions in the third gives the expressions for $\cos\theta$, $\sin\theta$ and the quartic equation, respectively. To show this is a minimum of D^2_{pCWP}, consider the matrix of second derivatives,

$$S^* = \begin{bmatrix} \frac{\partial^2 L}{\partial(\cos^2\theta)} & \frac{\partial^2 L}{\partial(\cos\theta)\partial(\sin\theta)} \\ \frac{\partial^2 L}{\partial(\cos\theta)\partial(\sin\theta)} & \frac{\partial^2 L}{\partial(\sin^2\theta)} \end{bmatrix} = \begin{bmatrix} 2(P-\lambda) & R \\ R & 2(Q-\lambda) \end{bmatrix}.$$

Let $\xi_1 \geq \xi_2$ be the eigenvalues of S^*. Then, $|S^* - \xi_i I| = (\xi_i + 2\lambda)^2 - 2(P + Q)(\xi_i + 2\lambda) + 4PQ - R^2$, so $(\xi_i + 2\lambda) = P + Q \pm \sqrt{(P-Q)^2 + R^2}$. Given Σ^{-1} is positive definite, $P > 0$ and $Q > 0$, then ξ_2 is strictly positive if $P + Q - 2\lambda - \sqrt{(P-Q)^2 + R^2} > 0$ which is true if the constraint on λ is satisfied. □

Proof of Result 2.3. Let $v = \text{vec}(\mu)$ and $\xi = \text{vec}(X\Gamma)$, then

$$\begin{aligned} D^2_{\text{CWP}}(X, \mu; \Sigma) &= (v - \beta\xi - (I_m \otimes 1_k)\gamma)^T \Sigma^{-1} (v - \beta\xi - (I_m \otimes 1_k)\gamma) \\ &= v^T\Sigma^{-1}v - 2\beta\xi^T\Sigma^{-1}v - 2v^T\Sigma^{-1}(I_m \otimes 1_k)\gamma + \beta^2\xi^T\Sigma^{-1}\xi \\ &\quad + 2\beta\xi^T\Sigma^{-1}(I_m \otimes 1_k)\gamma + \gamma^T(I_m \otimes 1_k)^T\Sigma^{-1}(I_m \otimes 1_k)\gamma. \end{aligned}$$

This implies

$$\begin{aligned} \frac{dD^2_{\text{CWP}}}{d\gamma} &= -2(I_m \otimes 1_k)^T\Sigma^{-1}v + 2\beta(I_m \otimes 1_k)^T\Sigma^{-1}\xi \\ &\quad + 2(I_m \otimes 1_k)^T\Sigma^{-1}(I_m \otimes 1_k)\gamma, \\ \frac{dD^2_{\text{CWP}}}{d\beta} &= -2\xi^T\Sigma^{-1}v + 2\beta\xi^T\Sigma^{-1}\xi + 2\xi^T\Sigma^{-1}(I_m \otimes 1_k)\gamma. \end{aligned}$$

Therefore, the minimum is at the solution of

$$B \begin{bmatrix} \gamma \\ \beta \end{bmatrix} = \begin{bmatrix} (I_m \otimes 1_k)^T \Sigma^{-1} \text{vec}(\mu) \\ \text{vec}(X\Gamma)^T \Sigma^{-1} \text{vec}(\mu) \end{bmatrix}.$$

The matrix of second derivatives is clearly positive because Σ^{-1} is positive definite. □

Proof of Result 2.4. Replacing $\cos\theta$ with $\beta\cos\theta$ and $\sin\theta$ with $\beta\sin\theta$ in the proof of Result 9.2.2 gives $D^2_{\text{CWP}}(X, \mu; \Sigma) = C + P\beta^2\cos^2\theta + Q\beta^2\sin^2\theta + R\beta^2\cos\theta\sin\theta + S\beta\cos\theta + T\beta\sin\theta$. Let $\psi_1 = \beta\cos\theta$ and $\psi_2 = \beta\sin\theta$, then

$$\frac{dD^2_{\text{CWP}}}{d\psi_1} = 2P\psi_1 + R\psi_2 + S, \qquad \frac{dD^2_{\text{CWP}}}{d\psi_2} = 2Q\psi_2 + R\psi_1 + T.$$

Setting these expressions equal to zero and solving them simultaneously gives the required expressions for ψ_1 and ψ_2. Solving $\psi_1 = \beta\cos\theta$ and $\psi_2 = \beta\sin\theta$ subject to the constraint that $\cos^2\theta + \sin^2\theta = 1$ gives the rotation and scale parameters. Given these, the translation is obtained by letting $v = \text{vec}(\mu - \beta X\Gamma)$ in the proof of Result 9.2.1. □

Proof of Result 2.5. If $\Sigma = I_m \otimes \Sigma_k$, then the similarity transformation estimates of Result 9.2.4 can be simplified. For the translation,

$$\begin{aligned} \hat{\gamma} &= [(I_m \otimes 1_k)^T (I_m \otimes \Sigma_k)^{-1} \\ &\quad \times (I_m \otimes 1_k)]^{-1} (I_m \otimes 1_k)^T (I_m \otimes \Sigma_k)^{-1} \text{vec}(\mu - \beta X\Gamma) \\ &= [I_m \otimes (1_k^T \Sigma_k^{-1} 1_k)]^{-1} [I_m \otimes (1_k^T \Sigma_k^{-1})] \text{vec}(\mu - \beta X\Gamma) \\ &= [I_m \otimes (1_k^T \Sigma_k^{-1} 1_k)^{-1} (1_k^T \Sigma_k^{-1})] \text{vec}(\mu - \beta X\Gamma). \end{aligned}$$

Therefore, $\hat{\gamma}^T = (1_k^T \Sigma_k^{-1} 1_k)^{-1} 1_k^T \Sigma_k^{-1} (\mu - \beta X\Gamma)$, which is zero given $1_k^T \Sigma_k^{-1} X = 0 = 1_k^T \Sigma_k^{-1} \mu$. Referring to the notation of Result 9.2.2, if $\Sigma = I_m \otimes \Sigma_k$ then $A_{11} = A_{22} = (1_k^T \Sigma_k^{-1} 1_k)^{-1} 1_k^T \Sigma_k^{-1}$ and $A_{12} = A_{21} = 0_k^T$. Then, from Eq. (9.4), if X and μ are located such that $1_k^T \Sigma_k^{-1} X = 0 = 1_k^T \Sigma_k^{-1} \mu$, then $\alpha_i = \delta_i = \zeta_i = 0$, for $i = 1, 2$, and P, Q, R, S and T simplify to

$$\begin{aligned} P = Q &= X_1^T \Sigma_k^{-1} X_1 + X_2^T \Sigma_k^{-1} X_2, \\ R &= -2(X_1^T \Sigma_k^{-1} X_2 - X_2^T \Sigma_k^{-1} X_1) = 0, \\ S &= -2(X_1^T \Sigma_k^{-1} \mu_1 + X_2^T \Sigma_k^{-1} \mu_2), \\ T &= 2(X_2^T \Sigma_k^{-1} \mu_1 - X_1^T \Sigma_k^{-1} \mu_2). \end{aligned}$$

The minimising rotation and scaling can then be obtained and have been derived by Brignell [2]. □

Proof of Result 3.1.

$$\begin{aligned}
G_{\text{CWP}}(X_1,\ldots,X_n;\varSigma) &= \sum_{i=1}^{n} \operatorname{vec}(R_i\varGamma)^T \varSigma^{-1} \operatorname{vec}(R_i\varGamma),\\
&= \sum_{i=1}^{n} \begin{bmatrix} R_{i1}\cos\theta - R_{i2}\sin\theta \\ R_{i2}\cos\theta + R_{i1}\sin\theta \end{bmatrix}^T \varSigma^{-1} \begin{bmatrix} R_{i1}\cos\theta - R_{i2}\sin\theta \\ R_{i2}\cos\theta + R_{i1}\sin\theta \end{bmatrix}\\
&= p\cos^2\theta + q\sin^2\theta + 2r\cos\theta\sin\theta\\
\frac{dG_{\text{CWP}}}{d\theta} &= (q-p)\sin 2\theta + 2r\cos 2\theta.
\end{aligned}$$

Therefore, the minimum of G_{CWP} is when θ is a solution of this last equation. □

Proof of Result 3.2. The log-likelihood of the multivariate normal model, $\operatorname{vec}(X_i) \sim N_{km}(\operatorname{vec}(\mu), \varSigma)$, where X_i are shapes invariant under Euclidean similarity transformations, is

$$\log L(X_1,\ldots,X_n;\mu,\varSigma) = -\frac{n}{2}\log|2\pi\varSigma|$$
$$-\frac{1}{2}\sum_{i=1}^{n} \operatorname{vec}(\beta_i X_i \varGamma_i + 1_k\gamma_i^T - \mu)^T \varSigma^{-1} \operatorname{vec}(\beta_i X_i \varGamma_i + 1_k\gamma_i^T - \mu).$$

Therefore, the MLE of the mean shape is the solution of

$$\frac{d\log L}{d\mu} = \sum_{i=1}^{n} \varSigma^{-1}\operatorname{vec}(\beta_i X_i\varGamma_i + 1_k\gamma_i^T) - n\varSigma^{-1}\mu = 0.$$

Hence, $\hat{\mu} = \bar{X} = \frac{1}{n}\sum_{i=1}^{n}(\beta_i X_i \varGamma_i + 1_k\gamma_i^T)$ and

$$\begin{aligned}
\log L &= -\frac{n}{2}\log|2\pi\varSigma| - \frac{1}{2}\inf_{\beta_i,\varGamma_i,\gamma_i}\sum_{i=1}^{n} \|\beta_i X_i\varGamma_i + 1_k\gamma_i^T - \bar{X}\|_{\varSigma}^2\\
&= -\frac{n}{2}\log|2\pi\varSigma| - \frac{1}{2}G_{\text{CWP}}(X_1,\ldots,X_n;\varSigma).
\end{aligned}$$

Therefore, minimising G_{CWP} is equivalent to maximising $L(X_1,\ldots,X_n;\mu,\varSigma)$. □

Proof of Result 4.1. Let the m columns of $(I_m \otimes 1_k)$ be 1_j for $j = 1,\ldots,m$ and let γ_{ij} be the jth element of the translation vector for shape X_i, then the log of the likelihood, L, for the multivariate normal model can be written:

$$\log L = -\frac{n}{2}\log|2\pi\varSigma| - \frac{1}{2}\sum_{i=1}^{n}\left(\operatorname{vec}(\beta_i X_i\varGamma_i - \mu)^T\varSigma^{-1}\operatorname{vec}(\beta_i X_i\varGamma_i - \mu)\right)$$
$$-\frac{1}{2}\sum_{i=1}^{n}\left(-2\sum_{j=1}^{m}\gamma_{ij}1_j^T\varSigma^{-1}\operatorname{vec}(\beta_i X_i\varGamma_i - \mu) + \sum_{j=1}^{m}\gamma_{ij}^2 1_j^T\varSigma^{-1}1_j\right).$$

Now, $1_j^T \Sigma^{-1} = \frac{\sigma_j^{-1}}{\sqrt{k}} 1_j^T 1_j 1_j^T$ as all the eigenvectors of Σ are orthogonal to 1_j except the one proportional to 1_j. Therefore,

$$\log L = -\frac{n}{2}\log|2\pi\Sigma| - \frac{1}{2}\sum_{i=1}^{n}\left(\operatorname{vec}(\beta_i X_i \Gamma_i - \mu)^T \Sigma^{-1}\operatorname{vec}(\beta_i X_i \Gamma_i - \mu)\right)$$
$$-\frac{1}{2}\sum_{i=1}^{n}\left(-2\sum_{j=1}^{m}\frac{\gamma_{ij}}{\sigma_j\sqrt{k}}1_j^T 1_j 1_j^T \operatorname{vec}(\beta_i X_i \Gamma_i - \mu) + \sum_{j=1}^{m}\frac{\gamma_{ij}^2}{\sigma_j\sqrt{k}}1_j^T 1_j 1_j^T 1_j\right).$$

Given X_i and μ are all centred, $1_j^T\operatorname{vec}(\beta_i X_i \Gamma_i - \mu) = 0$, and the maximizing translation is clearly $\gamma_{ij} = 0$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$. □

References

1. Bookstein FL (1986) Size and shape spaces for landmark data in two dimensions (with discussion). Stat Sci 1:181–242
2. Brignell CJ (2007) Shape analysis and statistical modelling in brain imaging. Ph.D. thesis, University of Nottingham
3. Davies RH, Twining CJ, Taylor CJ (2008) Statistical models of shape: optimisation and evaluation. Springer, Heidelberg. http://www.springer.com/computer/computer+imaging/book/978-1-84800-137-4
4. Dryden IL (1989) The statistical analysis of shape data. Ph.D. thesis, University of Leeds
5. Dryden IL (2014) Shapes: statistical shape analysis. R package version 1.1–10. http://CRAN.R-project.org/package=shapes.
6. Dryden IL, Mardia KV (1998) Statistical shape analysis. Wiley, Chichester
7. Dutilleul P (1999) The MLE algorithm for the matrix normal distribution. J Stat Comput Simul 64:105–123
8. Goodall CR (1991) Procrustes methods in the statistical analysis of shape (with discussion). J R Stat Soc Ser B 53:285–339
9. Gower JC (1975) Generalized Procrustes analysis. Psychometrika 40:33–50
10. Kendall DG (1984) Shape manifolds, Procrustean metrics and complex projective spaces. Bull Lond Math Soc 16:81–121
11. Kendall DG (1989) A survey of the statistical theory of shape (with discussion). Stat Sci 4:87–120
12. Kendall DG, Barden D, Carne TK, Le H (1999) Shape and shape theory. Wiley, Chichester
13. Koschat M, Swayne D (1991) A weighted procrustes criterion. Psychometrika 56(2):229–239. doi:10.1007/BF02294460. http://dx.doi.org/10.1007/BF02294460
14. Krim H, Yezzi AJ (2006) Statistics and analysis of shapes. Springer, Berlin
15. Lele S (1993) Euclidean distance matrix analysis (EDMA): estimation of mean form and mean form difference. Math Geol 25(5):573–602. DOI 10.1007/BF00890247. http://dx.doi.org/10.1007/BF00890247
16. Mardia KV, Dryden IL (1989) The statistical analysis of shape data. Biometrika 76:271–282
17. Sharvit D, Chan J, Tek H, Kimia BB (1998) Symmetry-based indexing of image databases. J Vis Commun Image Represent 9(4):366–380

18. Srivastava A, Klassen E, Joshi SH, Jermyn IH (2011) Shape analysis of elastic curves in Euclidean spaces. IEEE Trans Pattern Anal Mach Intell 33(7):1415–1428. http://doi.ieeecomputersociety.org/10.1109/TPAMI.2010.184
19. Srivastava A, Turaga PK, Kurtek S (2012) On advances in differential-geometric approaches for 2D and 3D shape analyses and activity recognition. Image Vis Comput 30(6–7):398–416
20. Theobald DL, Wuttke DS (2006) Empirical Bayes hierarchical models for regularizing maximum likelihood estimation in the matrix gaussian procrustes problem. Proc Nat Acad Sci 103(49):18521–18527. doi:10.1073/pnas.0508445103. http://www.pnas.org/content/103/49/18521.abstract
21. Theobald DL, Wuttke DS (2008) Accurate structural correlations from maximum likelihood superpositions. PLoS Comput Biol 4(2):e43
22. Younes L (2010) Shapes and diffeomorphisms. Applied mathematical sciences, vol 171. Springer, Berlin. doi:10.1007/978-3-642-12055-8. http://dx.doi.org/10.1007/978-3-642-12055-8

Chapter 10
Elastic Shape Analysis of Functions, Curves and Trajectories

Shantanu H. Joshi, Jingyong Su, Zhengwu Zhang, and Boulbaba Ben Amor

Abstract We present a Riemannian framework for geometric shape analysis of curves, functions, and trajectories on nonlinear manifolds. Since scalar functions and trajectories can also have important geometric features, we use shape as an all-encompassing term for the descriptors of curves, scalar functions and trajectories. Our framework relies on functional representation and analysis of curves and scalar functions, by square-root velocity fields (SRVF) under the Fisher–Rao metric, and of trajectories by transported square-root vector fields (TSRVF). SRVFs are general functional representations that jointly capture both the shape (geometry) and the reparameterization (sampling speed) of curves, whereas TSRVFs also capture temporal reparameterizations of time-indexed shapes. The space of SRVFs for shapes of curves becomes a subset of a spherical Riemannian manifold under certain special constraints. A fundamental tool in shape analysis is the construction and implementation of geodesic paths between shapes. This is used to accomplish a variety of tasks, including the definition of a metric to compare shapes, the computation of intrinsic statistics for a set of shapes, and the definition of probability models on shape spaces. We demonstrate our approach using several applications from computer vision and medical imaging including the analysis of (1) curves, (2) human growth, (3) bird migration patterns, and (4) human actions from video surveillance images and skeletons from depth images.

S.H. Joshi (✉)
University of California, Los Angeles, CA, USA
e-mail: s.joshi@g.ucla.edu

J. Su
Texas Tech University, Lubbock, TX, USA
e-mail: jingyong.su@ttu.edu

Z. Zhang
Florida State University, Tallahassee, FL, USA
e-mail: zhengwu@stat.fsu.edu

B. Ben Amor
Institut Mines-Télécom/Télécom Lille, CRIStAL (UMR CNRS 9189), Lille, France
e-mail: boulbaba.benamor@telecom-lille.fr

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_10

10.1 Introduction and Background

This chapter describes several geometric ideas for the analysis of scalar functions, curve shapes from boundaries, and trajectories of shapes from data recordings, images, and video sequences. These different modalities are finding novel uses in statistical pattern recognition, machine vision, medical imaging, and intelligent informatics-based applications. Examples of functional data include growth curves, mass spectrometry data, bio-signals, and human activity data [5, 14, 20, 21, 23, 30, 37]. Curve shapes are geometric descriptions of the underlying morphological information of objects from images. Applications in computer vision and medical imaging are often interested in analyzing well-defined continuous landmarks or boundary characterizations of objects that give rise to contour (both open or closed)-based representations of shapes [18, 19, 29, 35]. Trajectories of shapes can arise from several different applications. For example, human activity recognition has attracted tremendous interest in recent years because of its potential in applications such as surveillance, security, and human body animation. The shape sequences have also been called *trajectories* or curves on shape spaces [13, 15]. Figure 10.1 shows examples of data arising from functions, curves, and trajectories.

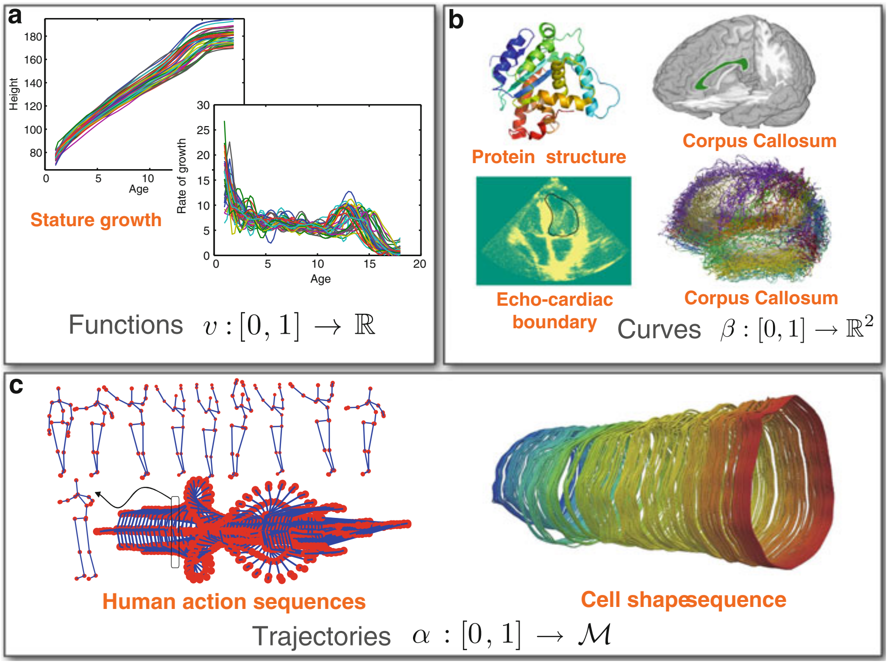


Fig. 10.1 Examples of (**a**) functions from growth data [5], (**b**) curves from biomedical images [6, 9, 10] and protein structure [17], and (**c**) trajectories from human action sequences [2] and a cell video sequence [1]

There are important reasons for studying such diverse descriptions of data using a common underlying approach. These data can be efficiently represented by continuous scalar-, vector- or tensor-valued functions. While curves have obvious geometric features associated with them, it is observed that continuous representations of functional data and trajectories also have local characteristic patterns such as ridges and valleys that correspond to their shapes. Essential tasks in performing curve, function, or trajectory analyses involve matching their representations to obtain correspondences between them, quantifying the mismatch by distances between them, and estimating statistical quantities such as means and covariances of a collection of objects. Further, these objects can undergo certain invariant transformations that preserve their shapes. For example, curves can be rotated, translated, scaled, or reparameterized without changing their shapes. The speed of trajectories can change without changing the form of the actions. Functions can be uniformly scaled without changing their shapes. Finally, these objects can also have additional local measures (for example, gene expression values along with shape or thickness of corpus callosum curve boundaries) associated with them. What is needed then is a methodology that will allow the representation, matching, and analysis of functions, curves, and trajectories by using their shapes. Throughout this chapter we will use the term *shape* to denote the invariant feature-rich geometric information in functions, curves, and trajectories.

This motivates the need for a metric-based geometric approach where one can have a geometric representation of shapes, construct a space of such representations, and define an appropriate metric on the space. The spaces of such representations turn out to have a nonlinear geometry because of various constraints (such as rotation or scale invariance) on the definitions of shapes. Riemannian approaches are particularly well suited for the formulation and analysis of continuous-valued representations arising from such complex applications, since they can efficiently exploit the intrinsic nonlinearity of the representations and the geometry of the underlying spaces of those representations. We also observe that this metric-based approach is fundamentally different from the discriminative approach where one is only interested in matching objects, without necessarily defining the underlying space. A main advantage of the discriminative approach over the metric-based approach is that it tends to be considerably faster, since it is mostly concerned with finding distances between shapes only. This is helpful in problems such as pattern recognition and classification for massive data sets. However, discriminative approaches have several limitations. They generally lack tools to allow continuous deformations between shapes, and thus there is no notion of correspondences between shapes. This makes it difficult to understand local shape differences, especially in biological applications. Importantly, they do not provide tools for computing statistical quantities such as shape averages and covariances. Thus they do not allow generative modeling of shapes, where one may be interested in sampling from distributions imposed on shapes. As a result, they do not aid in the goal of shape understanding, which is not only concerned with representing and matching shapes, but is also broadly involved in performing statistical inferences on shapes. The ground work for the metric-based approach was laid in Darcy Thompson's [31] work where he imposed coordinate systems on geometric objects and suggested that instead of comparing them directly, one could compare the transformations used to warp them to each other. Kendall [12] and Dryden and Mardia [3] proposed a geometric approach for analyzing landmark-based configurations. A new approach that analyzed differential geometric representations of images in computer vision and medical imaging was proposed by Grenander and Miller [4]. This approach directly modeled transformations between objects as elements of differential groups or manifolds and presented a method for computing distances between them.

In this chapter, we will follow the metric-based Riemannian approach, for the analysis of the functional, curve, and trajectory data, where (1) we will define a suitable continuous-valued representation, (2) construct a shape space of such representations along with a set of invariant transformations that preserve the shape, (3) define a suitable metric on the shape space, (4) find shortest paths (geodesics) between shapes that enable matching as well as deformations between them, and finally (5) introduce tools that allow statistical analysis of shapes. The underlying representation for functions, curves, and trajectories makes use of the square-root velocity formulation [7, 8, 25] that simplifies the representation as well as the metric and leads to efficient shape comparisons between objects. Fundamental to this approach is the notion of the Riemannian metric which turns out to be the Fisher–Rao metric in special cases [7, 22, 25, 26]. The reader will also appreciate the common features in the Riemannian framework that is used to analyze different elements such as curves, functions, and trajectories.

10.2 Elastic Shape Analysis of Functions

Functional data analysis (FDA) is widely applicable to different problems in both computer vision and statistics [14, 20, 21, 23, 30, 37]. Here, the objects of interest are functions on a certain domain $D, f : D \to \mathbb{R}$, and one is interested in using them to perform modeling, prediction, and regression for a variety of problems. These observations are typically treated as square-integrable functions, with the resulting set $\mathbb{L}^2$ forming an infinite-dimensional Hilbert space. The standard $\mathbb{L}^2$ inner product, $\langle f_1, f_2 \rangle = \int_D f_1(t) f_2(t) dt$, provides the Hilbert structure for comparing and analyzing functions. For example, one can perform functional principal component analysis (FPCA) of a given set $\{f_i\}$ using this Hilbert structure. Similarly, a variety of applications, such as functional linear regressions and partial least squares, have been proposed for working with functional data.

A difficulty arises when the observed functions exhibit variability in their arguments. In other words, instead of observing a function $f(t)$ on an interval, say $[0, 1]$, one observes a "time-warped" function $f(\gamma(t))$ where γ is a time-warping function. The time-warping functions are defined as the set of orientation-preserving diffeomorphisms of a certain interval, say $[0, 1]$. We use Γ to denote all such time-warping functions, where $\Gamma = \{\gamma : [0, 1] \to [0, 1], \gamma(0) = 0, \gamma(1) = 1, \gamma$ is a diffeomorphism$\}$. This extraneous effect of γ, also termed *phase* *variability*, has the potential to add artificial variance in the observed data and needs to be accounted for in statistical analysis. Let $\{f_i\}$ be a set of observations of a functional variable f. Then, for any time t, the observations $\{f_i(t)\}$ have some inherent variability, termed *amplitude variability*. However, if we observe $\{f_i \circ \gamma_i\}$ instead, for random warpings γ_is, then the resulting variability in $\{f_i(\gamma_i(t))\}$ has been enhanced due to random γ_is. The problem of removing the randomness in γ_is is called functional registration or *phase–amplitude separation* [26, 33].

A natural idea of performing registration for a pair of functions f_1,f_2 is using the criterion $\inf_{\gamma\in\Gamma} \|f_1 - f_2 \circ \gamma\|$, where $\|\cdot\|$ denotes the standard $\mathbb{L}^2$ norm. But it turns out to be problematic because of several issues. The first issue is that it is not symmetric. The optimal alignment of f_1 to f_2 is different from the alignment of f_2 to f_1, in general, i.e., $\inf_{\gamma\in\Gamma} \|f_1 \circ \gamma - f_2\| \neq \inf_{\gamma\in\Gamma} \|f_1 - f_2 \circ \gamma\|$. The second issue is that it allows degeneracy, that is, one can reduce the cost arbitrarily close to zero even when the two functions may be quite different. This is commonly referred to as the pinching problem in the literature [21]. Pinching implies that a severely distorted γ is used to eliminate those parts of one function (f_1) that do not match the other (f_2). The pinching problem can happen even when f_1 and f_2 are very different. To avoid this problem, people may impose a roughness penalty on γ such that the criterion becomes $\inf_\gamma(\|f_1 - f_2 \circ \gamma\| + \lambda\mathcal{R}(\gamma))$, where $\mathcal{R}$ is a smoothness penalty on the γ to keep it close to $\gamma_{id} = t$. But important issues such as how to choose the parameter λ or how to align multiple functions still remain. The functional registration problem suffers from these issues because $\|f_1 - f_2\| \neq \|f_1 \circ \gamma - f_2 \circ \gamma\|$. We can understand this point by the following way: if $f_1(t)$ matches $f_2(t)$ at the very beginning, then by warping with γ, now $f_1(\gamma(t))$ matches $f_2(\gamma(t))$. Each pointwise registration remains unchanged but its $\mathbb{L}^2$ norm changes. Hence, the $\mathbb{L}^2$ norm is not a proper metric to solve the registration problem. We suggest that a solution could be achieved by deriving an elastic metric that is better suited for registration of functions.

Representation of Functions Let f be a real-valued function on the domain $[0, 1]$: $f : [0, 1] \to \mathbb{R}$ such that f is absolutely continuous. Let $\mathcal{F}$ denote the set of all such functions. For any function $f \in \mathcal{F}$, we define a new function called the *square-root velocity field* (SRVF) according to

$$v : [0, 1] \to \mathbb{R}, v(t) = \frac{\dot{f}(t)}{\sqrt{|\dot{f}(t)|}} . \tag{10.1}$$

If the original f is absolutely continuous, then the resulting v is square integrable. Thus, we define the set of all SRVFs as $\mathbb{L}^2([0, 1], \mathbb{R})$, or simple $\mathbb{L}^2$. For every $v \in \mathbb{L}^2$, one can precisely recover the function f using the equation: $f(t) = f(0) + \int_0^t v(s)|v(s)|ds$. Given f(0), the mapping $f \Leftrightarrow v$ given by Eq. (10.1) is invertible. We use $v \cdot \gamma$ to denote the SRVF of $f \circ \gamma$, which can be calculated as

$$v \cdot \gamma = \frac{(\dot{f} \circ \gamma)\dot{\gamma}}{\sqrt{|(\dot{f} \circ \gamma)\dot{\gamma}|}} = \frac{(\dot{f} \circ \gamma)}{\sqrt{|(\dot{f} \circ \gamma)|}}\sqrt{\dot{\gamma}} = (v \circ \gamma)\sqrt{\dot{\gamma}}. \tag{10.2}$$

Elastic Riemannian Metric The motivation of using SRVF representation for FDA is that an elastic Riemannian metric, which is invariant to the domain warping, becomes the $\mathbb{L}^2$ metric under the SRVF representation [26]. That is, under the Riemannian metric, we have $d_r(f_1,f_2) = d_r(f_1 \circ \gamma, f_2 \circ \gamma)$, where d_r represents the distance resulting from this Riemannian metric. But, this metric is hard to analyze by the nature of its definition [26]. However, the SRVF transformation provides a simple solution: under the SRVF representation, this Riemannian metric becomes the standard $\mathbb{L}^2$ metric: $d_r(f_1,f_2) = \|v_1 - v_2\|$. Therefore, for any two SRVFs given by $v_1, v_2 \in \mathbb{L}^2$ and $\gamma \in \Gamma$, we have $\|v_1 \cdot \gamma - v_2 \cdot \gamma\| = \|v_1 - v_2\|$. This property allows us to solve the problem of registration of functions in a simple manner.

Pairwise and Multiple Functions Alignment With the help of the elastic Riemannian metric, the registration (*phase–amplitude separation*) between functions can be solved as follows. For two functions f_1, f_2 that need to be registered and their SRVFs v_1, v_2, we minimize the following objective function:

$$\inf_{\gamma \in \Gamma} \|v_1 - (v_2 \circ \gamma)\sqrt{\dot{\gamma}}\| = \inf_{\gamma \in \Gamma} \|v_2 - (v_1 \circ \gamma)\sqrt{\dot{\gamma}}\|. \tag{10.3}$$

The optimization can be performed using the well-known numerical procedure called the *dynamic programming* algorithm. Here, not only does the optimal γ help to register the functions f_2 and f_1 but the infimum value of the objective function also becomes a proper metric (i.e., it satisfies nonnegativity, symmetry, and triangle inequality) for comparing two functional objects. Thus it defines a distance between two functions, which we call the amplitude difference. This metric enables statistical analysis of functions, for example, one can calculate the mean function, perform functional principal component analysis, and develop statistical models for capturing observed functional variations and performing hypothesis tests. Figure 10.2 shows an example of this alignment between two Gaussian density functions. After optimization, the two functions are nicely aligned, as shown in the middle panel, and the resulting optimal warping γ^* is shown in the right panel. In case we have multiple functions that need to be aligned, we can extend the previous

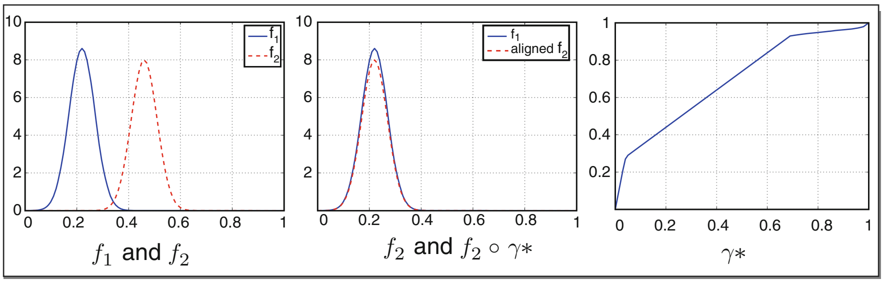


Fig. 10.2 *Left*: unmatched functions, *middle*: aligned functions, *right*: optimal alignment function γ^*

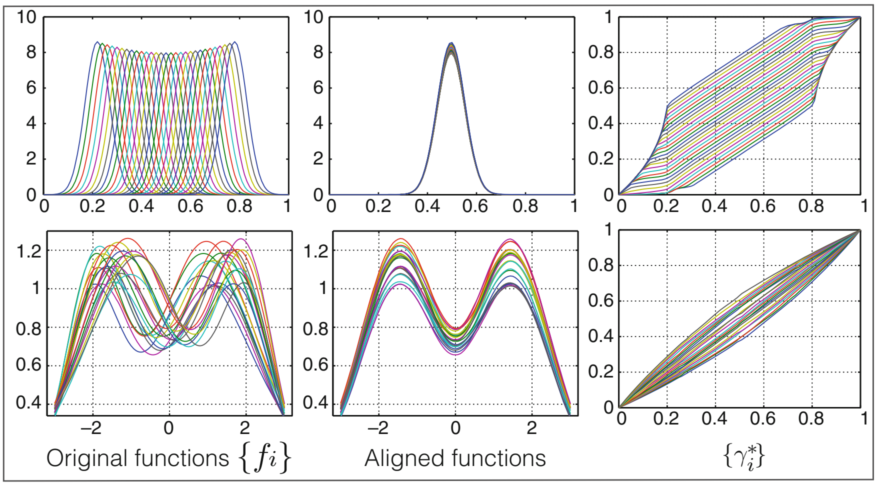


Fig. 10.3 Two different examples of *left*: unmatched functions, *middle*: aligned functions, *right*: optimal alignment functions γ_i^*s

pairwise alignment as follows. Equation (10.3) defines a proper metric in a certain quotient space ($\mathbb{L}^2/\Gamma$), and we can use this metric to define a mean function, e.g., Karcher mean [11]. This mean function serves as a template for aligning other functions, i.e., each function is aligned to this mean function. Given a collection of functions $f_1, f_2, \ldots, f_n$, let $v_1, v_2, \ldots, v_n$ denote their SRVFs, respectively. The problem of multiple alignment of these functions and Karcher mean computation are formulated and solved jointly using an iterative procedure: initialize the mean function μ and iteratively solve for

$$\gamma_i = \arg\inf_{\gamma\in\Gamma} \|\mu - (v_i \circ \gamma)\sqrt{\dot{\gamma}}\|, i = 1, 2, \cdots, n, \text{and}$$

$$\mu = \frac{1}{n}\sum_{i=1}^{n}(v_i \circ \gamma_i)\sqrt{\dot{\gamma}_i}. \tag{10.4}$$

Two synthetic examples of multiple function alignment are shown in Fig. 10.3. The leftmost panel shows the original functions whose heights and locations of peaks are different. The aligned functions are shown in the middle panel, and the optimal warping functions γ_i^*s are shown in the right panel.

In the next experiment, we show an example (Fig. 10.4) of the multiple functions alignment for real functional data, the Berkeley growth data set, which contains 54 female and 39 male subjects. To better illustrate, we analyze the first derivatives of the growth curves. The top row of Fig. 10.4 shows the alignment for 54 female subjects and the bottom row shows the alignment for 39 male subjects. The left column shows the original data given by the first derivative of the growth curves.

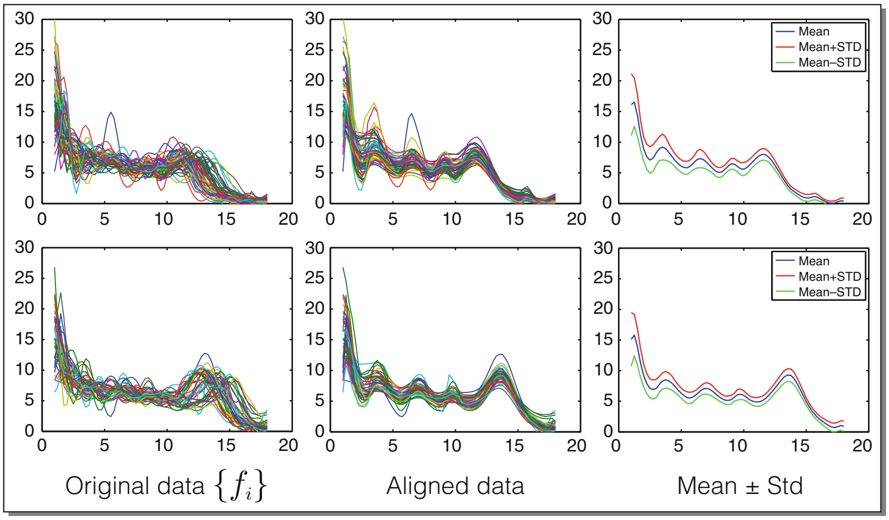


Fig. 10.4 Alignment of growth data. The *top row* shows the growth data for 54 females and the *bottom row* shows the growth data for 39 males

The middle column shows the aligned data, and the right column shows the mean $\pm$ (cross-sectional) standard deviation after alignment. The functional alignment results reveal that while the growth spurts for different individuals occur at slightly different times, several important growth spurts occur between the ages 3 and 5, 6 and 8, 9 and 11, and 13 and 14 years for males.

10.3 Elastic Shape Analysis of Curves

In Sect. 10.2, we studied the problem of registration of functions, where we tried to find a time-warping function γ that achieves an optimal matching between f_1 and $f_2 \circ \gamma$. In this section, the objects of interest are shapes of curves, which are continuous functions from $[0, 1]$ to $\mathbb{R}^n$. The warping function γ in this case performs the role of reparameterization of curves, which is a shape-preserving transformation, and needs to be explicitly modeled in the task of comparing shapes in an elastic manner. Similar to Sect. 10.2, we will represent shapes of curves using SRVFs and use the fully invariant elastic Riemannian metric to enable matching between them.

Representation and Shape Space of Curves The shape of an n-dimensional open parameterized curve β, such that $\beta(s) : [0, 1] \to \mathbb{R}^n, \forall s$ is defined by a function [7, 8, 25] $q : [0, 1] \to \mathbb{R}^n$ as

$$q(s) = \frac{\dot{\beta}(s)}{\sqrt{||\dot{\beta}(s)||_{\mathbb{R}^n}}} \in \mathbb{R}^n, \tag{10.5}$$

where $n = 2, 3$ for two- and three-dimensional curves. In this section we will assume $n = 2$ without loss of generality. This vector-valued function q is the tangent vector normalized by the square root of the instantaneous speed along the curve and is a local descriptor of the geometry of the curve. It is observed that in the absence of the square-root sign, the function q is a unit tangent vector to the curve. The original curve β can be reconstructed up to a translation using $\beta(s) = \int_0^s ||q(t)||\ q(t)\ dt$. The scale invariant shape representation is given by normalizing the function q by its magnitude as $\frac{q}{\sqrt{\int_0^1 (q(s),q(s))_{\mathbb{R}^2} ds}}$. The norm in the denominator is a Euclidean norm, and $(\cdot, \cdot)_{\mathbb{R}^2}$ is the standard Euclidean inner product in $\mathbb{R}^2$. Throughout this section, the function q refers to this scale-invariant form unless indicated otherwise.

Due to this unit-scaling constraint, the space of all translation and scale-invariant shapes becomes a Hilbert sphere denoted by $\mathcal{Q}$. Formally, the space $\mathcal{Q}$ is defined as

$$\mathcal{Q} \equiv \left\{ q \in \mathbb{L}^2 | \int_0^1 (q(s), q(s))_{\mathbb{R}^2} ds = 1, q(s) : [0, 1] \to \mathbb{R}^2 \right\}. \tag{10.6}$$

Ultimately, we are interested in analyzing curves in a fully invariant manner, i.e., we would like to consider an invariant space of shapes given by the quotient space modulo "shape preserving" transformations including rigid rotations and reparameterizations. Rotations are modeled by multiplication of a rotation matrix $O \cdot q(s) = Oq(s), \forall s$, where $O \in SO(2)$. However, the invariance to reparameterization is the most interesting shape-preserving transformation and facilitates elastic shape analysis of curves. Reparameterization gives rise to a change in speed of the curve without changing its shape. It is represented by a nonlinear differentiable map (with a differentiable inverse) also referred to as a diffeomorphism, which is defined as $\gamma \in \Gamma$ in Sect. 10.2. Analogous to Eq. (10.2), a reparameterization of a shape q by γ is given by $q \cdot \gamma = (\ q \circ \gamma) \sqrt{\dot{\gamma}}$. Thus the elastic shape space of open curves is defined as the quotient space,

$$\mathcal{S}_\text{O} = \mathcal{Q}/(\text{SO}(2) \times \Gamma). \tag{10.7}$$

This framework also allows us to represent closed boundaries ($\beta(s) : [0, 2\pi] \to \mathbb{R}^2, \forall s$) of objects by imposing an additional closure constraint on the curves. This closure constraint is written as $\int_0^{2\pi} \dot{\beta}(s) ds = 0$, in terms of the coordinate function, and is specified as $\int_0^{2\pi} q(s)||q(s)||ds = 0$, in terms of the shape function. One can then define the set of such translation, scale-invariant, and closed representations as the pre-shape space of the curve shapes and denote it by $\mathcal{C} \equiv \{q | q \in \mathcal{Q}, \int q(s)||q(s)||ds = 0\}$. This pre-shape space $\mathcal{C}$ is actually a subset of an infinite-dimensional unit sphere as a result of the scale-invariant constraint and represents all closed elastic curves invariant to translation and uniform scaling.

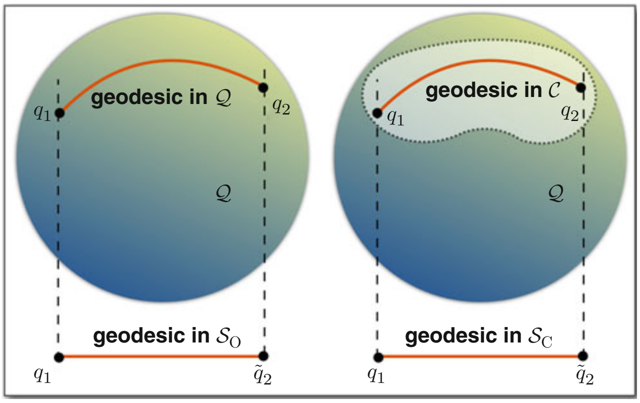

geodesic in $\mathcal{Q}$
q_1
q_2
$\mathcal{Q}$
geodesic in $\mathcal{S}_O$
q_1
$\tilde{q}_2$
geodesic in $\mathcal{C}$
q_1
q_2
$\mathcal{Q}$
geodesic in $\mathcal{S}_C$
q_1
$\tilde{q}_2$

Fig. 10.5 Illustration of the shape space of open curves (*left*) and closed curves (*right*). Also shown are the illustrations for quotient spaces and the geodesics for *open* and *closed curves*, respectively

The elastic shape space for closed curves is then given by the quotient space $\mathcal{S}_C = \mathcal{C}/(\mathbb{S}^1 \times SO(2) \times \Gamma)$. Figure 10.5 shows a schematic of the shape space of open and closed curves.

For simplicity, we will describe the theory for open curves in this chapter. The reader is referred to [7, 8, 25] for the theory and implementations for closed curves in detail.

Parameterization-Invariant Elastic Metric An important geometrical construct for the statistical analysis of the shapes is the definition of a tangent space. The tangent space allows for local linearization of the shape space and enables the use of Euclidean shape statistics for a population. The tangent space of $\mathcal{Q}$ is given by

$$T_q(\mathcal{Q}) \equiv \left\{ v \in \mathbb{L}^2 | v \perp \mathcal{Q} \right\}. \tag{10.8}$$

We equip the tangent spaces of $\mathcal{Q}$ with a smoothly varying Riemannian metric that measures infinitesimal lengths on the pre-shape space. This inner product is first defined generally on $\mathbb{L}^2$ and then induced on the tangent space of $\mathcal{Q}$. Given a pair of tangent vectors $u, v \in T_q(\mathcal{Q})$ the metric is defined as

$$\langle u, v \rangle = \int_0^1 (u(s), v(s))_{\mathbb{R}^2} ds. \tag{10.9}$$

This metric is fully invariant to rigid motions as well as reparameterizations. Owing to the reparameterization due to invariance, this metric is an elastic metric on the shape space of curves.

Elastic Shape Matching Similar to Sect. 10.2, we use geodesics between two points on the shape space for comparing shapes. The length of the geodesic determines an elastic quantitative distance between two shapes, whereas the full geodesic path achieves a continuous elastic deformation between them. The geodesic is computed under the Riemannian metric defined in Eq. (10.9). Since the space $\mathcal{Q}$ is a Hilbert sphere, the geodesic between two points (shapes) q_1 and q_2 can be expressed analytically as

$$\chi_t(q_1;f) = \cos\left(t\cos^{-1}\langle q_1, q_2\rangle\right) q_1 + \sin\left(t\cos^{-1}\langle q_1, q_2\rangle\right) f, \tag{10.10}$$

where $t \in [0, 1]$ and the initial tangent vector $f \in T_{q_1}(\mathcal{Q})$ is given by $f = q_2 - \langle q_1, q_2\rangle q_1$. Then, the geodesic distance between the two shapes q_1 and q_2 in $\mathcal{Q}$ is given by

$$d(q_1, q_2) = \int_0^1 \sqrt{\langle \dot{\chi}_t, \dot{\chi}_t\rangle} dt \tag{10.11}$$

The quantity $\dot{\chi}_t$ is also referred to as the velocity vector along the geodesic path χ_t. It is also noted that $\chi_0(q_1) = q_1$ and $\chi_1(q_1) = q_2$. The geodesic is computed using a path-straightening method [7, 8, 25] that initially connects the two points q_1 and q_2 using an arbitrary path in $\mathcal{Q}$ and then iteratively straightens it to form the shortest path. The elastic geodesic distance between shapes of open curves is then defined as

$$d_{\mathcal{S}_O}(q_1, q_2) = \min_{O \in SO(2), \gamma \in \Gamma} d(q_1, (Oq_2) \cdot \gamma). \tag{10.12}$$

The above geodesic is also computed iteratively using the path-straightening method. At each step, the optimal rotation O^* is found by carrying out a singular value decomposition $O^* = \int_0^1 q_1(s) q_2(s)^T ds$. The optimal reparameterization γ^* at each step can be efficiently found as the minimizer

$$\gamma^* = \underset{\gamma}{\text{argmin}} \left(\int_0^1 \left[||q_1 - \gamma \cdot \widetilde{q_2}||^2 \right] ds \right), \tag{10.13}$$

where $\widetilde{q_2} = O^* q_2$. Again similar to Sect. 10.2, Eq. (10.13) can be solved numerically using dynamic programming. Figure 10.6 shows examples of geodesics between both open and closed curves using nonelastic ($\gamma^*(s) = s$) and elastic matching. It is observed that elastic geodesics preserve local geometric features when deforming curves thus leading to better shape matching.

10.4 Elastic Shape Analysis of Trajectories on Manifolds

In Sects. 10.2 and 10.3, we modeled functional data and shape boundaries of objects in images. There the focus was on function/shape representation, matching, and analysis. In this section, our aim is to model a temporal sequence of such objects

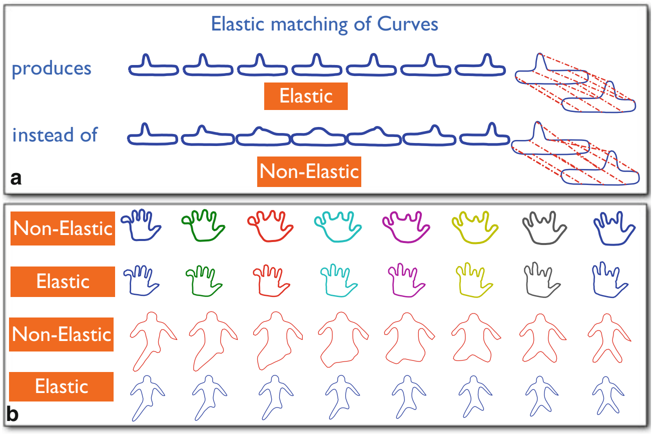


Fig. 10.6 Examples of geodesics between shapes of curves for nonelastic ($\gamma^*(s) = s$) and elastic matching. (**a**) An example of matching toy curves along with the correspondence between them. (**b**) Examples of hand and human shapes

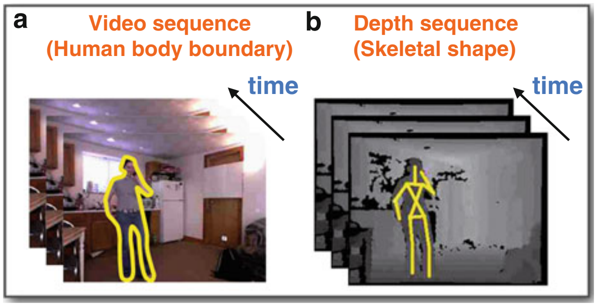


Fig. 10.7 Color (**a**) and depth (**b**) video streams filmed by a Kinect-like camera from the Cornell activity data set. The body boundaries are extracted to yield close *curve* shape representation or the body joints are estimated to obtain a skeletal shape representation from videos

instead of the objects themselves. Specifically we address the problem of registration, modeling and comparison, and statistical analysis of shapes of trajectories on Riemannian manifolds. Shown in Fig. 10.7 are two examples of temporal shape representations by trajectories (animations) on a Riemannian manifold. The figure illustrates video streams of color and depth images collected by a depth-consumer camera. When the camera detects a person, an important problem in computer vision is to recognize his/her action or activity. To tackle this problem the following two temporal representations can be considered; (1) the body contours detected from the flow of color images and (2) the body joints (3D skeleton) estimated from the depth images [24]. In the first case, one can use static shape representation given in Sect. 10.3 and extend it to trajectories (animation of curve shapes) to model the temporal evolution. In the second case, trajectories on Kendall's shape space

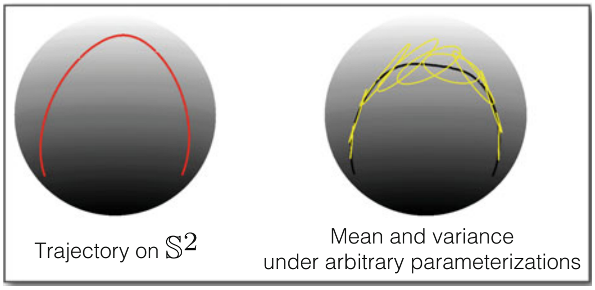

Trajectory on $\mathbb{S}^2$
Mean and variance under arbitrary parameterizations

Fig. 10.8 *Left*: a trajectory on $\mathbb{S}^2$, *Right*: mean trajectory (shown in *black*) and point-wise variance of this trajectory (shown by *yellow ellipses*) observed under arbitrary parameterizations

(of 3D landmark configurations) [3, 12] provide a natural representation suitable for temporal shape analysis, as will be described later in this section.

An important issue here is that trajectories are often observed under arbitrary temporal evolutions. If this temporal variability is not accounted for in the analysis, then the resulting statistical summaries will not be precise. The mean trajectory may not be representative of individual trajectories and the cross-sectional variance will be artificially inflated. This, in turn, will greatly reduce the effectiveness of any subsequent modeling or analysis based on the estimated mean and covariance. As a simple example consider the trajectory on $\mathbb{S}^2$ shown in the left of Fig. 10.8. We simulate a set of random, discrete observation times and generate observations of this trajectory at these random times. These simulated trajectories are identical in terms of the points traversed but their evolutions or parameterizations are quite different. The results for the cross-sectional mean and variance are shown on the right. We draw the sample mean trajectory in black and the sample variance at discrete times using tangential ellipses. Not only is the mean fairly different from the original curve, the variance is purely due to randomness in observation times and is somewhat artificial. This problem does not exist if we observe the trajectory at fixed, synchronized times instead.

Another issue is that shape observations are typically not in Euclidean spaces. Instead, the underlying spaces are Riemannian manifolds. This fact presents a formidable challenge in developing a comprehensive framework for analyzing such data. Although there has been progress in the removal of temporal variability in Euclidean spaces [14, 32, 33], there has not been any treatment of trajectories on Riemannian manifolds. Since activities can be performed at different execution rates, their corresponding shape curves will exhibit distinct evolution rates. Veeraraghavan et al. [34] accounted for the time-warping variability but their method has some fundamental problems. The cost function is not a proper distance. In fact, it is not even symmetric. We describe this problem in mathematical terms as follows. Let α denote a smooth trajectory on a Riemannian manifold of interest M, where M is endowed with a Riemannian metric $\langle \cdot, \cdot \rangle$. Let $\mathcal{M}$ denote the set of all such trajectories: $\mathcal{M} = \{\alpha : [0,1] \to M | \alpha \text{ is smooth}\}$. Let Γ be a diffeomorphism as defined in Sect. 10.2. If α is a trajectory on M, then $\alpha \circ \gamma$ is a trajectory that follows the same sequence of points as α but at the evolution rate governed by γ. More technically, the group Γ acts on $\mathcal{M}$, $\mathcal{M} \times \Gamma \to \mathcal{M}$, according to $(\alpha, \gamma) = \alpha \circ \gamma$.

Given two smooth trajectories $\alpha_1,\ \alpha_2 \in \mathcal{M}$, we want to register points along the trajectories and compute a time-warping invariant distance between them. For performing comparison of trajectories, we need a metric and, at first, we consider a more conventional solution. Since M is a Riemannian manifold, we have a natural geodesic distance d_m between points on M. Using d_m, one can compare any two trajectories: $\alpha_1, \alpha_2 : [0,1] \to M$, as $d_x(\alpha_1, \alpha_2) = \int_0^1 d_m(\alpha_1(t), \alpha_2(t))dt$. Although this quantity represents a natural extension of d_m from M to $\mathcal{M}$, it suffers from the problem that $d_x(\alpha_1, \alpha_2) \neq d_x(\alpha_1 \circ \gamma_1, \alpha_2 \circ \gamma_2)$ in general. It is not preserved even when the same γ is applied to both the trajectories, i.e., $d_x(\alpha_1, \alpha_2) \neq d_x(\alpha_1 \circ \gamma, \alpha_2 \circ \gamma)$ generally. If we have an equality in the last case, for all γs, then one can develop a fully invariant distance and use it to register trajectories properly, as described later. So, the failure to have this equality is in fact a key issue that forces us to look for other solutions in situations where trajectories are observed at random temporal evolutions.

We introduce a quantity that provides both a cost function for temporal registration and a proper distance for comparison of trajectories [27, 28]. This distance is used to define statistical summaries, such as sample means and covariances, of synchronized trajectories and "Gaussian-type" models to capture their variability at discrete times. It is invariant to identical time warpings (or temporal reparameterizations) of trajectories. This is based on a novel mathematical representation of trajectories, termed transported square-root vector field (TSRVF), and the $\mathbb{L}^2$ norm on the space of TSRVFs.

Representation and Space of Trajectories Let c be a point in M that we will designate as a reference point. For any smooth trajectory $\alpha \in \mathcal{M}$, the transported square-root vector field (TSRVF) is a parallel transport of a scaled velocity vector field of α to a reference point $c \in M$ according to

$$h_\alpha(t) = \frac{\dot{\alpha}(t)_{\alpha(t) \to c}}{\sqrt{|\dot{\alpha}(t)|}} \in T_c(M) \ ,$$

where $|\cdot|$ is defined by the Riemannian metric on M and the tangent space at c is denoted by $T_c(M)$. Since α is smooth, so is the vector field h_α. Let $\mathcal{H} = \{h_\alpha | \alpha \in \mathcal{M}\}$ be the set of smooth curves in $T_c(M)$ obtained as TSRVFs of trajectories in M. If $M = \mathbb{R}^n$ with the Euclidean metric then h is exactly the square-root velocity function defined in [7, 8, 25] and in Sect. 10.3 of the present chapter.

Rate-Invariant Metric for Comparison and Registration Since a TSRVF is a path in $T_c(M)$, one can use the $\mathbb{L}^2$ norm to compare such trajectories. Let α_1 and α_2 be two smooth trajectories on M and let h_{α_1} and h_{α_2} be the corresponding TSRVFs. The distance between them is $d_h(h_{\alpha_1}, h_{\alpha_2}) = \left(\int_0^1 |h_{\alpha_1}(t) - h_{\alpha_2}(t)|^2 dt\right)^{\frac{1}{2}}$. The main motivation of this setup—the TSRVF representation and $\mathbb{L}^2$ norm—comes from the following fact. For any $\alpha_1, \alpha_2 \in \mathcal{M}$ and $\gamma \in \Gamma$, the distance d_h satisfies $d_h(h_{\alpha_1 \circ \gamma}, h_{\alpha_2 \circ \gamma)}) = d_h(h_{\alpha_1}, h_{\alpha_2})$. In geometric terms, this implies that the action of Γ on $\mathcal{H}$ under the $\mathbb{L}^2$ metric is by isometries. Now we are ready to define the quantity that will serve as both the cost function for registration and the distance for comparison. This quantity is essentially the shortest distance d_h between parameterized trajectories, defined as

$$d_s(h_{\alpha_1}, h_{\alpha_2}) = \inf_{\gamma_1,\gamma_2 \in \Gamma} d_h(h_{\alpha_1 \circ \gamma_1}, h_{\alpha_2 \circ \gamma_2}) \ . \tag{10.14}$$

It can be shown that d_s is a proper distance and invariant to arbitrary time warpings. The optimal correspondence between α_1 and α_2 can be solved according to

$$\gamma^* = \operatorname*{argmin}_{\gamma \in \Gamma} \left(\int_0^1 |h_{\alpha_1}(t) - h_{\alpha_2}(\gamma(t))\sqrt{\dot{\gamma}(t)}|^2 dt \right)^{\frac{1}{2}} . \tag{10.15}$$

Statistical Summaries of Trajectories An additional advantage of this framework is that one can compute an average of several trajectories and use it as a template for future classification. Furthermore, this template can be used for registering multiple trajectories. We use the notion of the Karcher mean [11] to define and compute average trajectories. Given a set of sample trajectories $\alpha_1, \ldots, \alpha_n$ on M, the Karcher mean is defined as $\mu = \mathrm{argmin}_{\alpha \in \mathcal{M}} \sum_{i=1}^n d_s(h_\alpha, h_{\alpha_i})^2$. For computing and analyzing the second and higher moments of a sample trajectory, the tangent space $T_{\mu(t)}(M)$, for $t \in [0, 1]$, is used. This is convenient because it is a vector space and one can apply more traditional methods here. Let $\hat{K}(t)$ be the sample covariance matrix, with the trace $\hat{\rho}(t) = \mathrm{trace}(\hat{K}(t))$. This $\hat{\rho}(t)$ represents a quantification of the cross-sectional variance, as a function of t, and can be used to study the level of alignment of trajectories. Also, for capturing the essential variability in the data, one can perform principal component analysis (PCA) on the tangent spaces. An important use of means and covariances of trajectories is in devising probability models for capturing the observed statistical variability and for using these models in evaluating p-values of future observations. One can use the tangent space to impose a probability model since this is a vector space. Then, the p-value is defined as the proportion of random trajectories that will have lower probability density under a given model when compared to the test trajectory.

As a simple example, we apply this framework to bird migration data first, where the observations are on unit spheres. This data set has 35 migration trajectories of Swainson's Hawk, measured from 1995 to 1997, each having geographic coordinates measured at some random times, where the underlying space is $M = \mathbb{S}^2$. Several sample paths are shown at the top row in Fig. 10.9a. In the bottom panel of Fig. 10.9a, we show the optimal warping functions $\{\gamma_i^*\}$ used in aligning them and this clearly highlights a significant temporal variation present in the data. In Fig. 10.9b,c, we show the Karcher mean μ and the cross-sectional variance $\hat{\rho}$ without and with registration, respectively. In the top row, μ is displayed using colors, where red areas correspond to higher variability. In the bottom row, the principal modes of variation are displayed by ellipses on tangent spaces. We use the first and second principal tangential directions as the major and minor axes of

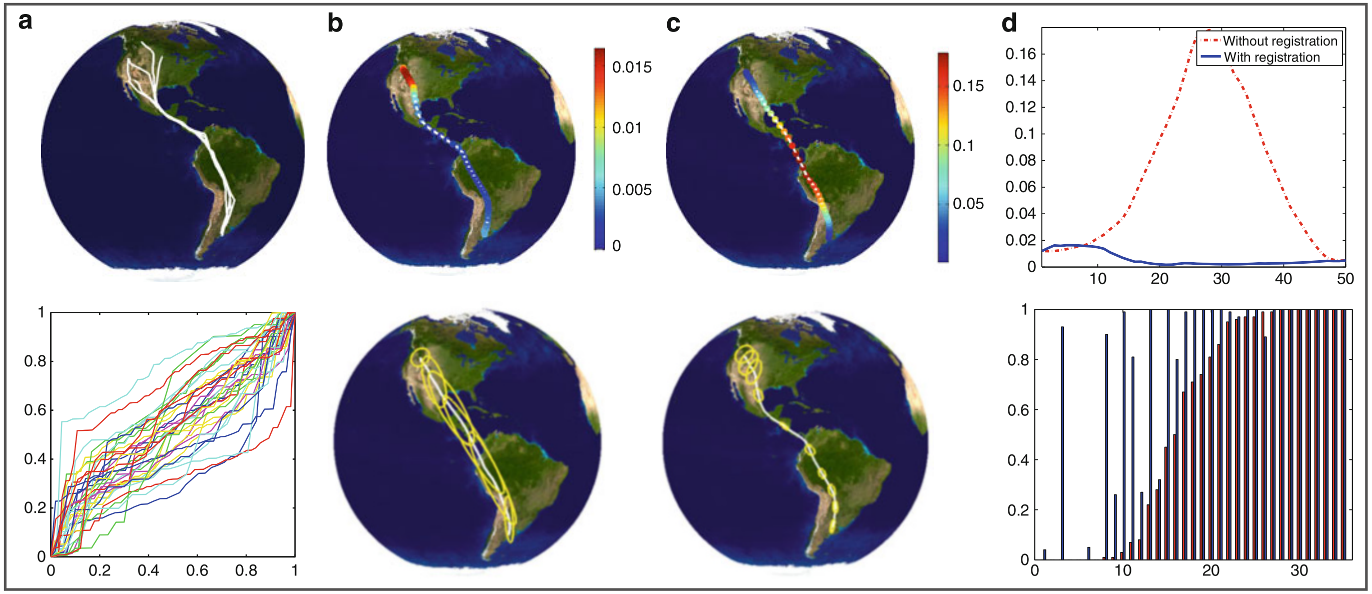


Fig. 10.9 Swainson's Hawk migration: (**a**) $\{\alpha_i\}$ (*top*) and $\{\gamma_i^*\}$ (*bottom*); (**b**) μ and $\hat{\rho}$ without registration; (**c**) μ and $\hat{\rho}$ with registration; (**d**) $\hat{\rho}$ (*top*) and p-values (*bottom*). *Red*: without registration, *Blue*: after registration

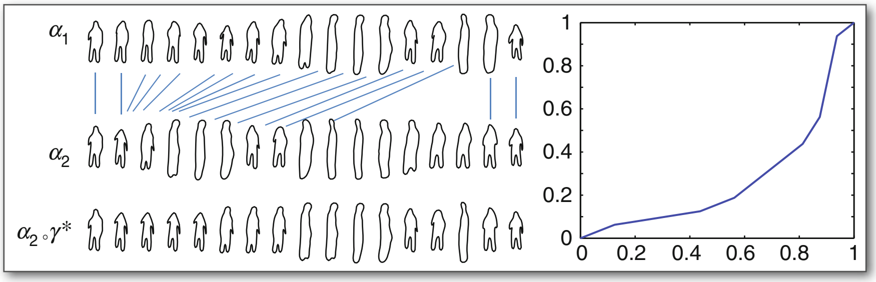


Fig. 10.10 Registration of two trajectories on the shape space of planar contours

ellipses and the corresponding singular values as their lengths. We observe that (1) the mean after registration better preserves the shapes of trajectories and (2) the variance ellipses before registration have their major axes along the trajectory while the ellipses after registration exhibit the actual variability in the data. Most of the variability after registration is limited to the top end where the original trajectories indeed have differences. Next we construct a "Gaussian-type" model for these trajectories using estimated summaries for two cases (with and without temporal registration), as described previously, and compute p-values of individual trajectories using Monte Carlo simulation. The results are shown in Fig. 10.9d, where we note a general increase in the p-values for the original trajectories after the alignment. This is attributed to a reduced variance in the model due to temporal alignment and the resulting movement of individual samples closer to the mean values.

Motivated by the problem of analyzing human activities using video data, we are interested in alignment, comparison, and averaging of trajectories on the shape space of planar, closed curves. There are several mathematical representations available for this analysis, and we use the representation of [25]. The benefits of using this representation over other methods are discussed in Sect. 10.3. So here M is the elastic shape space of planar closed curves. An example of registering two trajectories of planar closed curves from the same class is shown in Fig. 10.10. The distance d_h between the two trajectories decreases from 4.27 to 3.26. The optimal γ^* for this registration is shown in the right panel. We give an example of averaging and registration of multiple trajectories in Fig. 10.11. The aligned sample trajectories within the same class are much closer to each other than before temporal alignment. For this activity data set we computed the full pairwise distance matrix for trajectories, using d_h (without registration) and d_s (with registration). The leave-one-out nearest neighbor classification rate (1-NN as described earlier) for d_s is 95 % as compared to only 87.5 % when using d_h.

The recent growth in cheap and mobile range imaging sensors [36] has pushed forth a new research direction where one explores an additional channel of information called *depth*. An important advantage of these data is that it is relatively easy to remove background and to isolate and track human body. More specifically, depth

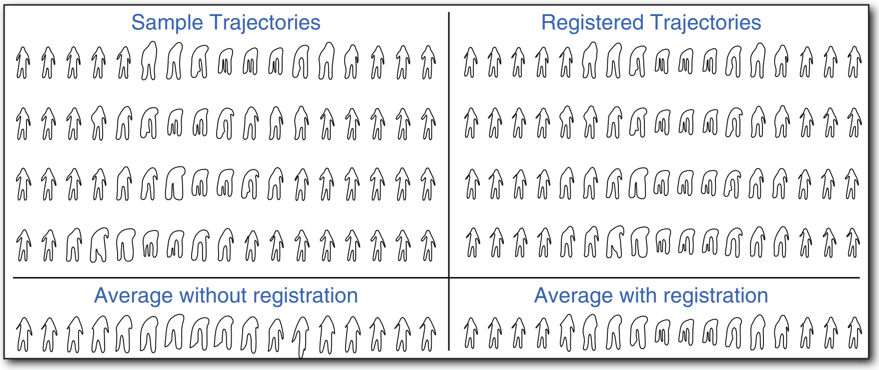


Fig. 10.11 Registration and summary of multiple trajectories

sensors such as Microsoft Kinect$^{\text{TM}}$ can provide reliable range data that can be used to estimate parameters (joint locations and orientations) describing human skeletons at each observed time [24]. These skeletons, or rather their temporal variations, provide efficient representations of actions being performed by human subjects. Since human skeletons are characterized by sets of registered points (or landmarks), it is natural to use Kendall's approach [3, 12] to perform shape analysis here. The specification of a shape manifold M and the corresponding metric enables us to compare arbitrary skeletons in terms of their shapes, using geodesic lengths. Additionally, the development of shape tools, such as computation of sample mean and sample covariance statistics, and the transfer of deformations using parallel transports also becomes straightforward. Then, it is also convenient to define an action as a sequence of skeletal shapes or, more precisely, a parameterized trajectory on this manifold of skeleton shapes.

Again, let us consider α the observation of an action over the time interval $[0, 1]$. For each time $t \in [0, 1]$, the skeleton at time t has a shape denoted by $\alpha(t) \in M$ (here, M denotes the shape space of skeletons as we will describe later). We are focusing only on the shape of the skeleton at any time, ignoring its scale, position, and pose [2]. To this end, we consider the set of all centered-scaled configurations of n landmarks in $\mathbb{R}^3$. Because our goal is also to filter out the rotations (elements of the rotation group $SO(3)$), we consider first the *pre-shape space* of all centered-scaled configurations, termed M_0, then define the *shape space* M as a quotient space of the original space given by $M = M_0/SO(3)$. To remove the rotation (pose) variability, we define for each $X \in M_0$ equivalence class that represents all rotations of a configuration X. Actually, the original space M_0 is a unit sphere in $\mathbb{R}^{3(n-1)}$; thus, its geometry is well known and the definition of tangent spaces, geodesics, the exponential map and its inverse, and the parallel translation are known and can be easily adapted to M. Then, the space of trajectories can be generally written as $\mathcal{M} = M^{[0,1]}$. Note that the parameterization of a trajectory denotes the rate at which the corresponding action was performed. As described previously, our goal is to perform action recognition in a time-invariant fashion. We will again

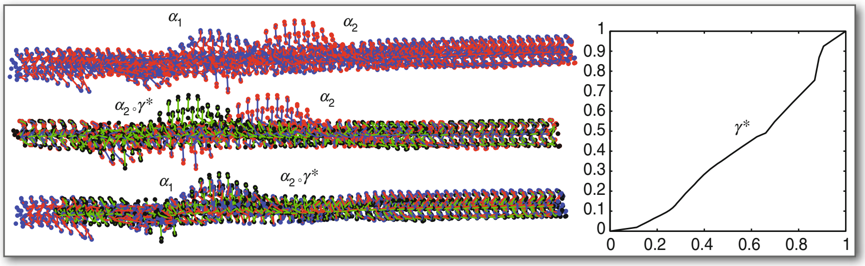


Fig. 10.12 Registration of two trajectories on the shape space of skeletons M. On the *top left*: the trajectories α_1 and α_2 to be aligned; on the *bottom left*: the trajectories after temporal alignment α_1 and $\alpha_2 \circ \gamma^*$, on the *middle left:* are shown the trajectories α_2 and $\alpha_2 \circ \gamma^*$. On the *right* is reported the best reparameterization function γ^* to align α_1 and α_2

adopt the solution proposed in [27] and described previously to temporally align the skeletal trajectories in action classification. Figure 10.12 illustrates an example of non-synchronized trajectories (of skeletons), α_1 and α_2, as well as the result after performing the alignment $\alpha_2 \circ \gamma^*$.

In order to perform action classification, we can take one of two approaches—metric based or model based. In a model-based approach one defines a statistical model, using generative or discriminative models, and uses it to classify future actions. In the metric-based approach, the main ingredient is the choice of a distance, with appropriate properties, that can be combined with a classifier, such as the nearest-neighbor classifier or SVM, for performing classification. Taking a metric approach, we seek a metric on the space $\mathcal{M}$ that can be used to distinguish between different activities. From the perspective of classification an action is invariant to its execution rate and, therefore, we need a distance between trajectories that is invariant to their arbitrary reparameterizations. Two ideas have been tested on the MSR Action 3D data set [16] which consists of a total of 20 types of segmented actions collected by Kinect. Each action starts and ends with a neutral pose and is performed two or three times by each of the 10 subjects. Following the cross-subject experimental setting described in [16] and used later in other papers, we use the five first subjects for training and the last five for testing. The two classifiers mentioned previously—NN classifier, and SVM classifier—are used for classification purposes. With NN classifier using d_s (defined in Eq. (10.14)) our approach achieves 69 % average recognition rate (compared to 42 % with d_h). This result highlights the importance of aligning trajectories when performing the action classification. The SVM classifier is applied differently as following, the TSRVFs are time derivatives of α computed using shooting vectors between successive points along trajectories, i.e., $h_\alpha(t) = v(t)_{\alpha \to [C]}/\sqrt{|v(t)|}$, where $v(t) = \exp^{-1}_{\alpha(t)}(\alpha(t+\delta))$ where $\delta = 2, 3, 4 \ldots$. The resulting TSRVF, denoted by $h_\alpha(t)$, is still a curve in the tangent space $T_{[c]}(M)$ and can be treated as a Euclidean representation of a trajectory α. This makes sense since finite difference with a larger time step is akin to approximating derivative after smoothing α. This approach achieved higher performance compared to the previous one, the average recognition rate is 89 % on the Action 3D dataset, when performing the alignment, compared to 60 % without the temporal alignment.

10.5 Summary

We present a Riemannian approach for the shape analysis of functions, curves, and trajectories. The SRVF and the TSRVF formulations not only admit compact representations of shapes but also simplify the Riemannian metric, while enabling efficient matching via geodesics. This approach studies shapes as intrinsic elements of a quotient space, and thus naturally allows calculation of statistical quantities on the shape space. This is different from the conventional two-step registration-based analysis approach where one first establishes correspondences between shapes (usually under an $\mathbb{L}^2$ metric) and then carries out the statistical analysis subsequently in the ambient space. In other words, our metric-based shape analysis framework not only achieves correspondence-based registration but also integrates tools for intrinsic statistical analysis into an elastic matching framework.

References

1. An X, Liu Z, Shi Y, Li N, Wang Y, Joshi SH (2012) Modeling dynamic cellular morphology in images. In: Medical image computing and computer-assisted intervention–MICCAI 2012. Springer, Berlin, pp 340–347
2. Amor BB, Su J, Srivastava A (2014) Action recognition using rate-invariant analysis of skeletal shape trajectories. IEEE Trans Pattern Anal Mach Intell 1(99):1, 10.1109/TPAMI.2015.2439257
3. Dryden IL, Mardia K (1998) Statistical shape analysis. Wiley series in probability and statistics: probability and statistics. Wiley, Chichester
4. Grenander U, Miller MI (1998) Computational anatomy: an emerging discipline. Q Appl Math 56(4): 617–694
5. Jones HE, Bayley N (1941) The Berkeley growth study. Child Dev 12(2):167–173
6. Joshi SH, Srivastava A (2009) Intrinsic Bayesian active contours for extraction of object boundaries in images. Int J Comput Vis 81(3):331–355
7. Joshi SH, Klassen E, Srivastava A, Jermyn I (2007) A novel representation for Riemannian analysis of elastic curves in R^n. In: IEEE conference on computer vision and pattern recognition (CVPR). IEEE, New York, pp 1–7
8. Joshi SH, Klassen E, Srivastava A, Jermyn I (2007) Removing shape-preserving transformations in square-root elastic (SRE) framework for shape analysis of curves. In: Energy minimization methods in computer vision and pattern recognition (EMMCVPR), pp 387–398
9. Joshi SH, Cabeen RP, Joshi AA, Sun B, Dinov I, Narr KL, Toga AW, Woods RP (2012) Diffeomorphic sulcal shape analysis on the cortex. IEEE Trans Med Imaging 31(6):1195–1212
10. Joshi SH, Narr KL, Philips OR, Nuechterlein KH, Asarnow RF, Toga AW, Woods RP (2013) Statistical shape analysis of the corpus callosum in schizophrenia. Neuroimage 64:547–559
11. Karcher H (1977) Riemannian center of mass and mollifier smoothing. Commun Pure Appl Math 30:509–541

12. Kendall DG (1984) Shape manifolds, procrustean metrics, and complex projective spaces. Bull Lond Math Soc 16(2):81–121
13. Kenobi K, Dryden IL, Le H (2010) Shape curves and geodesic modeling. Biometrika 97(3):567–584
14. Kneip A, Ramsay JO Combining registration and fitting for functional models. J Am Stat Assoc 103(483):1155–1165 (2008)
15. Le H (2003) Unrolling shape curves. J Lond Math Soc 68(2):511–526
16. Li W, Zhang Z, Liu Z (2010) Action recognition based on a bag of 3D points. In: IEEE international workshop on CVPR for human communicative behavior analysis, pp 9–14
17. Liu W, Srivastava A, Zhang J (2011) A mathematical framework for protein structure comparison. PLoS Comput Biol 7(2):e1001,075
18. Michor PW, Mumford D (2006) Riemannian geometries on spaces of plane curves. J Eur Math Soc 8:1–48
19. Mio W, Srivastava A, Joshi SH (2007) On shape of plane elastic curves. Int J Comput Vis 73(3):307–324
20. Ramsay JO, Li X (1998) Curve registration. J R Stat Soc Ser B (Stat Methodol) 60(2):351–363
21. Ramsay JO, Silverman BW (2005) Functional data analysis. Springer series in statistics, 2nd edn. Springer, New York
22. Rao RC (1945) Information and the accuracy attainable in the estimation of statistical parameters. Bull Calcutta Math Soc 37:81–91
23. Rossi F, Villa N (2006) Support vector machine for functional data classification. Neurocomputing 69(7–9):730–742; New issues in Neurocomputing: 13th European symposium on artificial neural networks 13th European symposium on artificial neural networks, 2005
24. Shotton J, Sharp T, Kipman A, Fitzgibbon A, Finocchio M, Blake A, Cook M, Moore R (2013) Real-time human pose recognition in parts from single depth images. Commun ACM 56(1):116–124
25. Srivastava A, Klassen E, Joshi SH, Jermyn IH (2011) Shape analysis of elastic curves in Euclidean spaces. IEEE Trans Pattern Anal Mach Intell 33:1415–1428
26. Srivastava A, Wu W, Kurtek S, Klassen E, Marron JS (2011) Registration of functional data using Fisher-Rao metric. arXiv:1103.3817v2
27. Su J, Kurtek S, Klassen E, Srivastava A (2014) Statistical analysis of trajectories on Riemannian manifolds: bird migration, hurricane tracking, and video surveillance. Ann Appl Stat 8(1):530–552
28. Su J, Srivastava A, de Souza FD, Sarkar S (2014) Rate-invariant analysis of trajectories on Riemannian manifolds with application in visual speech recognition. In: IEEE conference on computer vision and pattern recognition (CVPR). IEEE, New York, pp 620–627
29. Sundaramoorthi G, Mennucci A, Soatto S, Yezzi A (2011) A new geometric metric in the space of curves, and applications to tracking deforming objects by prediction and filtering. SIAM J Imaging Sci 4(1):109–145
30. Tang R, Muller HG (2008) Pairwise curve synchronization for functional data. Biometrika 95(4):875–889
31. Thompson DW (1943) On growth and form. Cambridge University Press, Cambridge
32. Trouve A, Younes L (2000) On a class of diffeomorphic matching problems in one dimension. SIAM J Control Optim 39(4):1112–1135
33. Tucker JD, Wu W, Srivastava A (2013) Generative models for functional data using phase and amplitude separation. Comput Stat Data Anal 61:50–66
34. Veeraraghavan A, Srivastava A, Roy-Chowdhury AK, Chellappa R (2009) Rate-invariant recognition of humans and their activities. IEEE Trans Image Process 8(6):1326–1339
35. Younes L (1998) Computable elastic distance between shapes. SIAM J Appl Math 58(2):565–586
36. Zhang Z (2012) Microsoft kinect sensor and its effect. IEEE Multimedia 19(2):4–10
37. Zhang Z, Klassen E, Srivastava A, Turaga PK, Chellappa R (2011) Blurring-invariant Riemannian metrics for comparing signals and images. In: Proceedings of ICCV, pp 1770–1775

Chapter 11
Why Use Sobolev Metrics on the Space of Curves

Martin Bauer, Martins Bruveris, and Peter W. Michor

Abstract In this chapter we study reparametrization invariant Sobolev metrics on spaces of regular curves. We discuss their completeness properties and the resulting usability for applications in shape analysis. In particular, we will argue that the development of efficient numerical methods for higher-order Sobolev-type metrics is an extremely desirable goal.

11.1 Introduction

Over the past decade Riemannian geometry on the infinite-dimensional spaces of parametrized and unparametrized curves has developed into an active research area. The interest has been fueled by the important role of these spaces in the areas of shape analysis and computer vision. Contributions in these fields include applications to medical image diagnostics [18], target and activity recognition in image analysis [38], plant leaf classification [23], and protein structure analysis [26] or human motion analysis in computer graphics [17]. In these research areas one is interested in studying the variability within a certain class of shapes. As a consequence an important goal is the development of statistical tools for these spaces.

Riemannian metrics provide the additional structure that is needed to capture the nonlinearity of the space and at the same time linearize it sufficiently to enable computations. In this chapter we want to acquaint the reader with some of the metrics, that can be defined on the space of curves and discuss their properties with a view towards applications in shape analysis. We will concentrate particularly

M. Bauer (✉) • P.W. Michor
Faculty for Mathematics, University of Vienna,
Oskar-Morgenstern-Platz 1, 1090 Vienna, Austria
e-mail: bauer.martin@univie.ac.at; peter.michor@univie.ac.at

M. Bruveris
Department of Mathematics, College of Engineering Design and Physical Sciences,
Brunel University London, John Crank Building, Uxbridge UB8 3PH, UK
e-mail: martins.bruveris@brunel.ac.uk

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_11

on completeness properties: do geodesics exist, when do they stop existing, and how does it depend on the metric. For a more wide-ranging overview of Riemannian metrics on spaces of functions, see [13].

Parametrized Curves In this chapter we will discuss Riemannian metrics on two different spaces: first, the space of smooth, regular, closed curves in $\mathbb{R}^n$

$$\operatorname{Imm}(S^1, \mathbb{R}^d) = \left\{ c \in C^\infty(S^1, \mathbb{R}^d) : c'(\theta) \neq 0,\ \forall \theta \in S^1 \right\}; \tag{11.1}$$

here Imm stands for *immersion*. This is an open set in the Fréchet space $C^\infty(S^1, \mathbb{R}^d)$ of all smooth functions and as such it is itself a Fréchet manifold. As an open subset of a vector space, its tangent space at any curve is the vector space itself, $T \operatorname{Imm}(S^1, \mathbb{R}^d) \cong \operatorname{Imm}(S^1, \mathbb{R}^d) \times C^\infty(S^1, \mathbb{R}^d)$. A Riemannian metric on $\operatorname{Imm}(S^1, \mathbb{R}^d)$ is a smooth map

$$G : \operatorname{Imm}(S^1, \mathbb{R}^d) \times C^\infty(S^1, \mathbb{R}^d) \times C^\infty(S^1, \mathbb{R}^d) \to \mathbb{R},$$

such that $G_c(\cdot, \cdot)$ is a symmetric, positive definite bilinear form for all curves c. An example of a Riemannian metric is the L^2-metric $G_c(h, k) = \int_{S^1} \langle h, k \rangle |c'| \, \mathrm{d}\theta$, which we will look at more closely in Sect. 11.2. When studying particular Riemannian metrics, it will be useful to consider larger spaces of less regular curves, but $\operatorname{Imm}(S^1, \mathbb{R}^d)$ will always be the common core.

Unparametrized Curves The other space, which we will consider, is the space of unparametrized curves, sometimes also denoted *shape space*. There are several closely related but slightly differing ways to define this space mathematically. We will consider an *unparametrized curve* or *shape* to be an equivalence class of parametrized curves that differ only by a reparametrization. In other words, c_1 and c_2 represent the same shape, if $c_1 = c_2 \circ \varphi$ for some reparametrization $\varphi \in \operatorname{Diff}(S^1)$; mathematically $\operatorname{Diff}(S^1)$ is the diffeomorphism group of the circle, that is, the set of all smooth invertible maps $\varphi : S^1 \to S^1$. With this definition the space of unparametrized curves is the quotient

$$B(S^1, \mathbb{R}^d) = \operatorname{Imm}(S^1, \mathbb{R}^d) / \operatorname{Diff}(S^1).$$

Apart from isolated singular points, the space $B(S^1, \mathbb{R}^d)$ is also an infinite-dimensional manifold and the projection $p : c \to [c]$ assigning each curve its equivalence class is a submersion.[1]

[1]In applications one often wants to consider curves and shapes modulo Euclidean motions, leading to the spaces $\operatorname{Imm}(S^1, \mathbb{R}^d) / \operatorname{Mot}$ and $B(S^1, \mathbb{R}^d) / \operatorname{Mot}$, where $\operatorname{Mot} = SO(d) \ltimes \mathbb{R}^d$ denotes the Euclidean motion group. All metrics discussed in this chapter are invariant under the motion group and therefore induce a Riemannian metric on the quotients $\operatorname{Imm}(S^1, \mathbb{R}^d) / \operatorname{Mot}$ and $B(S^1, \mathbb{R}^d) / \operatorname{Mot}$. In Sect. 11.3 we will encounter metrics that live naturally on the space $\operatorname{Imm}(S^1, \mathbb{R}^d) / \operatorname{Tra}$ of curves modulo translations.

Reparametrization Invariant Metrics To define a Riemannian metric on the space $B(S^1, \mathbb{R}^d)$ we will start with a Riemannian metric G on $\operatorname{Imm}(S^1, \mathbb{R}^d)$, that is invariant under the action of $\operatorname{Diff}(S^1)$; such metrics are called *reparametrization invariant*. This means G has to satisfy

$$G_{c\circ\varphi}(h \circ \varphi, k \circ \varphi) = G_c(h, k),$$

for all curves c, tangent vectors h, k, and reparametrizations φ. Then we can use the formula

$$G_{[c]}(X, X) = \inf \{G_c(h, h) \ : \ T_c p.h = X\}$$

to define a Riemannian metric on shape space $B(S^1, \mathbb{R}^d)$ such that the projection p is a Riemannian submersion.

Geodesic Distance An important concept in shape analysis is the notion of distance between two curves or shapes. A Riemannian metric leads to a natural distance function, the *induced geodesic distance*. The distance measures the length of the shortest path between two curves. If $c_0, c_1 \in \operatorname{Imm}(S^1, \mathbb{R}^d)$ are two parametrized curves, the distance between them is defined as

$$\operatorname{dist}_I(c_0, c_1) = \inf_{\substack{\gamma(0)=c_0 \\ \gamma(1)=c_1}} \int_0^1 \sqrt{G_{\gamma(t)}(\gamma_t(t), \gamma_t(t))}\, \mathrm{d}t,$$

where the infimum is taken over all smooth paths γ that connect the curves c_0 and c_1. Whether there exists a path, realizing this infimum, is an interesting and non-trivial question in Riemannian geometry.

If we start with a reparametrization invariant metric G and it induces a Riemannian metric on shape space $B(S^1, \mathbb{R}^d)$, then we will be interested in computing the geodesic distance on $B(S^1, \mathbb{R}^d)$. The geodesic distances for parametrized and unparametrized curves are related by

$$\operatorname{dist}_B([c_0], [c_1]) = \inf_{\varphi \in \operatorname{Diff}(S^1)} \operatorname{dist}_I(c_0, c_1 \circ \varphi). \tag{11.2}$$

For applications in shape analysis it is important to find a stable and fast method to numerically compute this quantity for arbitrary shapes $[c_0]$ and $[c_1]$. We will comment in the later sections, for which metrics a reparametrization φ realizing the above infimum, exists, and what the obstructions to its existence are in other cases.

Organization We will look at three families of metrics: first, the L^2-metric in Sect. 11.2, which is among the simplest reparametrization invariant metrics, but unfortunately unsuitable for shape analysis; then, first-order Sobolev metrics in Sect. 11.3, which are very well suited for numerical computations and therefore among the most widely used Riemannian metrics in applications; finally, we will look at higher-order Sobolev metrics in Sect. 11.4 and argue why their theoretical properties make them good candidates for use in shape analysis. At the end we will explain how these metrics can be generalized to spaces of parametrized and unparametrized surfaces.

11.2 The L^2-Metric

The arguably simplest Riemannian metric on the space of smooth, regular curves that is invariant under reparametrizations is the L^2-metric

$$G_c(h,k) = \int_{S^1} \langle h, k\rangle \, \mathrm{d}s,$$

where we use $\mathrm{d}s = |c'|\,\mathrm{d}\theta$ to denote arc length integration. It is integration with respect to $\mathrm{d}s$ rather than $\mathrm{d}\theta$ that makes this metric reparametrization invariant, as can be seen from the following calculation:

$$G_{c\circ\varphi}(h\circ\varphi, k\circ\varphi) = \int_{S^1} \langle h\circ\varphi, k\circ\varphi\rangle \,(|c'|\circ\varphi)\,\varphi'\,\mathrm{d}\theta = G_c(h,k).$$

Similarly if we wanted to include derivatives of h, k in the metric and keep the metric reparametrization invariant, we would need to use the arc length differentiation $D_s h = \frac{1}{|c'|}h'$ rather that $h' = \partial_\theta h$.

Geodesic Equation The geodesic equation of the L^2-metric is a nonlinear, second-order PDE for the path $c(t,\theta)$. It has the form

$$(|c_\theta| c_t)_t = -\frac{1}{2}\left(\frac{|c_t|^2}{|c_\theta|}\,c_\theta\right)_\theta \tag{11.3}$$

where $c_\theta = \partial_\theta c = c'$ and $c_t = \partial_t c$ denote the partial derivatives. While the equation is as simple as one can hope for—the geodesic equations for higher-order metrics have many more terms—there are currently no existence results available for it.

Open Question Given a pair of an initial curve and an initial velocity $(c_0, u_0) \in T\,\mathrm{Imm}(S^1, \mathbb{R}^d)$, does the geodesic equation admit short-time solutions with the given initial conditions?

We know that we cannot hope for long time existence, since it is possible to shrink a circle along a geodesic path down to a point in finite time. Numerical evidence in [29, Sect. 5.3] suggests that geodesics should exist as long as the curvature of the curve remains bounded.

Geodesic Distance The lack of existence results for the geodesic equation is not the biggest problem of the L^2-metric. The crucial property that makes it unsuitable for applications in shape analysis is that the induced geodesic distance vanishes.

The *geodesic distance* between two curves $c_0, c_1 \in \operatorname{Imm}(S^1, \mathbb{R}^d)$ is defined as the infimum over the lengths of all paths γ, connecting the two curves, i.e.,

$$\operatorname{dist}_I(c_0, c_1) = \inf_{\substack{\gamma(0)=c_0 \\ \gamma(1)=c_1}} \int_0^1 \sqrt{G_{\gamma(t)}(\gamma_t(t), \gamma_t(t))}\, \mathrm{d}t.$$

It was found [8, 28, 29] that for the L^2-metric the geodesic distance between any two curves is zero.[2] What does this mean? If γ is a smooth, nonconstant path, then $\partial_t \gamma(t)$ cannot be identically zero and so the length $\int_0^1 \sqrt{G_\gamma(\gamma_t, \gamma_t)}\, \mathrm{d}t$ will be strictly positive. The meaning of $\operatorname{dist}_I(c_0, c_1) = 0$ is that we can find arbitrary short paths connecting c_0 and c_1. No path will have zero length, but given any $\varepsilon > 0$, we can find a path with length $< \varepsilon$. How do these paths look like? They are easier to visualize for the geodesic distance on the space of unparametrized curves, which we will describe next.

For the L^2-metric the geodesic distance between the unparametrized curves $[c_0], [c_1] \in B(S^1, \mathbb{R}^d)$, represented by c_0, c_1, can be computed as the following infimum:

$$\operatorname{dist}_B([c_0], [c_1]) = \inf_\gamma \int_0^1 \sqrt{G_\gamma(\gamma_t^\perp, \gamma_t^\perp)}\, \mathrm{d}t;$$

here $\gamma(t)$ is a path starting at c_0 and ending at any curve in the equivalence class $[c_1]$, that is $\gamma(1) = c_1 \circ \varphi$ for some $\varphi \in \operatorname{Diff}(S^1)$. We denote by $\gamma_t^\perp = \gamma_t - \langle \gamma_t, v \rangle v$, with $v = D_s \gamma$, the projection of the vector $\gamma_t(t, \theta) \in \mathbb{R}^d$ to the subspace orthogonal to the curve $\gamma(t)$ at θ.

A short path connecting two concentric circles can be seen in Fig. 11.1. The key observation is that the sawtooth-shaped curves have a large tangential velocity, but only a small normal velocity. Since for the geodesic distance on $B(S^1, \mathbb{R}^d)$ we only measure the normal part of the velocity vector, these paths have a short length. The more teeth we use, the smaller the normal component of the velocity and the smaller the length of these paths.

The vanishing of the geodesic distance started the search for stronger metrics that would be more useful to shape analysis.

Almost Local Metrics One class of metrics, designed to have nonvanishing distance while being as simple as possible, is the class of almost local metrics.

[2] We encounter the vanishing of the geodesic distance for L^2-metrics on several spaces: on the space $\operatorname{Imm}(M, N)$ of immersions between two manifolds M, N of arbitrary dimension, M compact; on the Virasoro–Bott group [8]; and even for Sobolev metrics on the diffeomorphism group of a compact manifold, provided the order of the metric is $< \frac{1}{2}$ [9, 10].

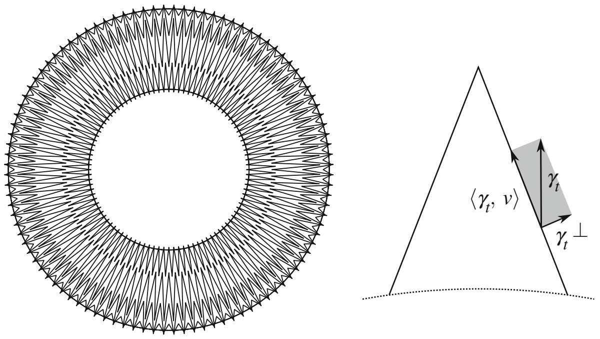


Fig. 11.1 *Left side*: a short curve with respect to the L^2-metric in the space $B(S^1, \mathbb{R}^2)$ of unparametrized curves connecting two concentric circles. We see that the intermediate curves are sawtooth shaped. *Right side*: Along a sawtooth the tangential component $\langle \gamma_t, v\rangle v$ is large, while the normal component $\gamma^{\perp}$ becomes small, the steeper the slope of the sawtooth

The motivating idea behind almost local metrics was the observation that for paths with short length in the L^2-metric, the intermediate curves are long and have large curvature. Thus one hopes that by adding weights that depend on length and curvature to the metric, these paths will be sufficiently penalized and the geodesic distance will become nonzero. *Almost local metrics*[3] are metrics of the form

$$G_c(h,k) = \int_{S^1} \Phi(\ell_c, \kappa)\langle h, k\rangle \,\mathrm{d}s, \tag{11.4}$$

with Φ some function of the two variables $\ell_c = \int_{S^1} \mathrm{d}s$ (length) and κ (curvature). If Φ depends only on ℓ_c, the resulting metric $G_c(h,k) = \Phi(\ell_c)\int_{S^1}\langle h,k\rangle \,\mathrm{d}s$ is a conformal rescaling of the L^2-metric [34, 41]. Other choices for Φ include $\Phi(\kappa) = 1 + A\kappa^2$ with A a positive constant [29] or the scale invariant metric $\Phi(\ell_c, \kappa) = \frac{1}{\ell_c^3} + \kappa^2\ell_c$ [30].

For all these metrics it has been shown that they induce a point-separating distance function[4] on the space $B(S^1, \mathbb{R}^d)$ of unparametrized curves. However similarly to the L^2-metric, little is known about solutions of the geodesic equation and while the geodesic distance is point separating on the space $B(S^1, \mathbb{R}^d)$, it is not point separating on the space $\mathrm{Imm}(S^1, \mathbb{R}^d)$ of parametrized curves. In the next two sections we will discuss a different strategy to strengthen the L^2-metric, leading to the class of Sobolev metrics.

[3]These metrics are not local, because the length ℓ_c is not a local quantity; however, it is only a mild non-locality, and hence the name "almost local" metrics.

[4]A distance function $d(\cdot,\cdot)$ is *point separating*, if $d(x,y) > 0$ whenever $x \neq y$. This is stronger than nonvanishing, which would only require two points x, y with $d(x,y) \neq 0$.

11.3 First-Order Metrics and the Square Root Velocity Transform

One way to deal with the degeneracy of the L^2-metric is by adding terms that involve first derivatives of the tangent vectors. Such metrics are called *first order Sobolev metrics* or, short, H^1-metrics. An example is the metric

$$G_c(h,k) = \int_{S^1} \langle h,k\rangle + \langle D_s h, D_s k\rangle \,\mathrm{d}s,$$

with $D_s h = \frac{1}{|c'|}h'$ denoting the arc length derivative and $\mathrm{d}s = |c'|\,\mathrm{d}\theta$. If we omit the L^2-term, we arrive at $G_c(h,k) = \int_{S^1}\langle D_s h, D_s k\rangle\,\mathrm{d}s$, which is a metric on the space $\mathrm{Imm}(S^1,\mathbb{R}^d)/\,\mathrm{Tra}$ of regular curves modulo translations. The scale-invariant version of this metric has been studied in [42, 43] and it has the remarkable property that one can find explicit formulas for minimizing geodesics between any two curves.

We will concentrate in this section on a related metric, obtained by using different weights for the tangential and normal components of $D_s h$,

$$G_c(h,k) = \int_{S^1} \langle D_s h^\perp, D_s k^\perp\rangle + \frac{1}{4}\langle D_s h, v\rangle\langle D_s k, v\rangle\,\mathrm{d}s; \tag{11.5}$$

here $v = D_s c = \frac{1}{|c'|}c'$ is the unit length tangent vector along c and $D_s h^\perp = D_s h - \langle D_s h, v\rangle v$ is the projection of $D_s h$ to the subspace $\{v\}^\perp$ orthogonal to the curve. This is a Riemannian metric on $\mathrm{Imm}(S^1,\mathbb{R}^d)/\,\mathrm{Tra}$ and it is the metric used in the *square root velocity (SRV)* framework [37]. The reason for singling out this metric is that the SRV framework has been used successfully in applications [23, 38, 40] and the SRV transform has a simple and accessible form. We will comment on other H^1-metrics at the end of the section.

The Square Root Velocity Transform The *square root velocity transform (SRVT)* is the map

$$R : \mathrm{Imm}(S^1,\mathbb{R}^d) \to C^\infty(S^1,\mathbb{R}^d)\,, \quad c \mapsto \frac{1}{\sqrt{|c'|}}c',$$

assigning each curve c a function $q = R(c)$.

Every vector space $(V, \langle\cdot,\cdot\rangle)$ with an inner product can be regarded as a Riemannian manifold: the Riemannian metric g at each point $x \in V$ is simply the inner product, $g_x(\cdot,\cdot) = \langle\cdot,\cdot\rangle$. The SRVT is an isometry between the Riemannian manifold $\big(\mathrm{Imm}(S^1,\mathbb{R}^d)/\,\mathrm{Tra}, G\big)$, where G is the Riemannian metric (11.5), and the space $C^\infty(S^1,\mathbb{R}^d)$ with the L^2-inner product $\langle u,v\rangle_{L^2} = \int_{S^1}\langle u,v\rangle\,\mathrm{d}\theta$.

The SRVT for Open Curves Things are simple on the space of open curves $\mathrm{Imm}([0,2\pi],\mathbb{R}^d)/\,\mathrm{Tra}$. The SRVT is a one-to-one mapping between the space $\mathrm{Imm}([0,2\pi],\mathbb{R}^d)/\,\mathrm{Tra}$ and the set $C^\infty(S^1,\mathbb{R}^d\setminus\{0\})$ of functions that do not pass through the origin in $\mathbb{R}^d$. This is an open subset of all functions and thus geodesics with respect to the metric (11.5) correspond to straight lines under the SRVT: the path $c(t) = R^{-1}(q_0 + th)$ is a geodesic in $\operatorname{Imm}([0, 2\pi], \mathbb{R}^2)/\operatorname{Tra}$, and given two curves c_0, c_1, the geodesic connecting them is

$$c(t) = R^{-1}((1-t)q_0 + tq_1) \,,$$

with $q_i = R(c_i)$.

The SRVT for Closed Curves Things are slightly more complicated for closed curves. The inverse of the SRVT is given by the formula

$$R^{-1}(q)(\theta) = \int_0^\theta q|q| \,\mathrm{d}\sigma,$$

and we see that if we want the curve $c = R^{-1}(q)$ to be closed, i.e., $c(0) = c(2\pi)$, then we need $\int_{S^1} q|q| \,\mathrm{d}\sigma = 0$. Indeed the image of $\operatorname{Imm}([0, 2\pi], \mathbb{R}^d)/\operatorname{Tra}$ under the SRVT is the set

$$\operatorname{Im}(R) = \left\{ q \in C^\infty(S^1, \mathbb{R}^2) : \; q(\theta) \neq 0 \text{ and } \int_{S^1} |q|q \,\mathrm{d}\theta = 0 \right\} .$$

We have the condition $q'(\theta) \neq 0$ as before to ensure that $c'(\theta) \neq 0$ and an additional constraint so that the curves $c = R^{-1}(q)$ are closed. Even though we do not have a closed expression for the geodesics, it is still possible to compute the geodesics numerically without much difficulty.

Minimizing Geodesics for Parametrized Curves Let us first look at open curves. In the SRV representation, $q_i = R(c_i)$, the minimizing path between q_0 and q_1 is the straight line $q(t) = (1-t)q_0 + tq_1$. In particular the minimizing path always exists. It can happen, however, that the straight line between $q_0(\theta)$ and $q_1(\theta)$ passes through the origin; at points (t, θ), where this happens the derivative of the curve $c(t, \theta) = R^{-1}(q(t))(\theta)$ vanishes, i.e., $c'(t, \theta) = 0$ and thus the curve is not regular at those points. Apart from that, any two curves can be joined by a unique minimizing geodesic, which can be computed via an explicit formula, and we know when the intermediate curves will fail to be regular.

For closed curves the situation is less explicit, because now we also have to satisfy the nonlinear constraint $\int_{S^1} q|q| \,\mathrm{d}\theta = 0$. This is a d-dimensional constraint on an otherwise infinite-dimensional space and furthermore the function $q \mapsto \int_{S^1} q|q| \,\mathrm{d}\theta$ is continuous with respect to the L^2-topology. Numerical evidence suggests that minimizing geodesics continue to exist between any two curves. In particular, computing minimizing geodesics between parametrized curves is a fast and stable operation; an example of a geodesic can be seen in Fig. 11.2.

Smoothness of the minimizing geodesics is another issue. The natural target space for the SRVT is the space $L^2(S^1, \mathbb{R}^d)$ of square-integrable functions. If the SRVT of a curve lies in $L^2(S^1, \mathbb{R}^d)$, the curve itself is only absolutely continuous.

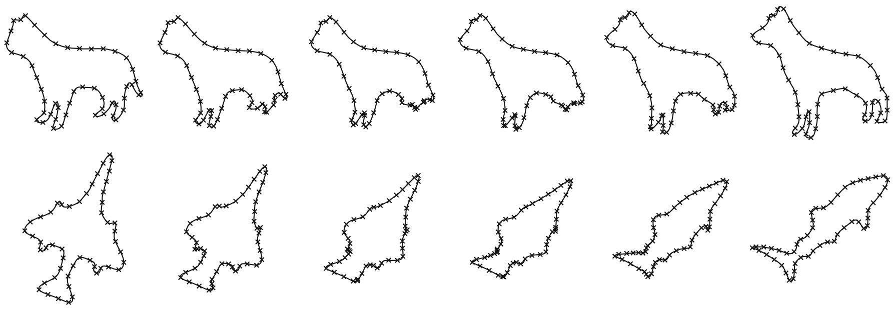


Fig. 11.2 Minimal geodesics between two pairs of parametrized curves. Images taken from [11]

Unfortunately the Riemannian metric $G_c(h,k)$, given by (11.5), does not have to be finite for absolutely continuous curves c and tangent vectors h,k; the term $D_s h = \frac{1}{|c'|}h'$ may well become infinite. We are approaching the frontier of the Riemannian framework now: any two (open) curves can be joined by a minimizing path; however, the space, where the path lives—the completion of the space of smooth curves, if one wants to use the term—is not a Riemannian manifold any more.

Minimizing Geodesics for Unparametrized Curves If we want to find minimizing geodesics between two unparametrized curves $C_0, C_1 \in B(S^1,\mathbb{R}^d)/\operatorname{Tra}$, represented by the curves $c_0, c_1 \in \operatorname{Imm}(S^1,\mathbb{R}^d)/\operatorname{Tra}$, one way to do this is to minimize $\operatorname{dist}_I(c_0, c_1\circ\varphi)$ over $\varphi \in \operatorname{Diff}(S^1)$ or equivalently over all parametrized curves $c_1\circ\varphi$ representing the shape C_1; indeed, the geodesic distance on $B(S^1,\mathbb{R}^d)/\operatorname{Tra}$ is given by

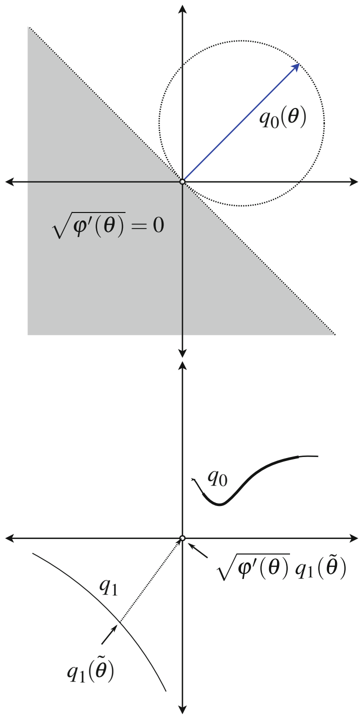
$$\operatorname{dist}_B(C_0,C_1) = \inf_{\varphi\in\operatorname{Diff}(S^1)} \operatorname{dist}_I(c_0, c_1\circ\varphi). \tag{11.6}$$

If the infimum is attained for $\psi \in \operatorname{Diff}(S^1)$ and if we denote by $c(t)$ the minimizing geodesic between c_0 and $c_1\circ\psi$, then the curve $[c(t)]$ in $B(S^1,\mathbb{R}^d)/\operatorname{Tra}$ is the minimizing geodesic between C_0 and C_1. Thus we are interested whether the infimum (11.6) is attained and if it is, in what space.

Let us look at $\operatorname{dist}_I(c_0, c_1\circ\varphi)$, first for open curves in the SRV representation. Let $q_i = R(c_i)$. We have $R(c_1\circ\varphi) = \sqrt{\varphi'}\,q_1\circ\varphi$ and

$$\operatorname{dist}_I(c_0, c_1\circ\varphi)^2 = \int_0^{2\pi} |q_0(\theta) - \sqrt{\varphi'(\theta)}q_1(\varphi(\theta))|^2 \,\mathrm{d}\theta.$$

Assume that ψ minimizes this expression, fix $\theta \in S^1$ and set $\tilde\theta = \psi(\theta)$. Even though finding ψ has to be done over the whole interval $[0,2\pi]$ simultaneously, it is very instructive to look at just one point at a time. Consider the infimum



$q_0(\theta)$
$\sqrt{\varphi'(\theta)} = 0$
q_0
$\sqrt{\varphi'(\theta)}\, q_1(\tilde{\theta})$
q_1
$q_1(\tilde{\theta})$

Fig. 11.3 *Left side*: solution to the finite-dimensional minimization problem. In the halfplane below the *dotted line*, the solution is given by $\sqrt{\varphi'(\theta)} = 0$. On the halfplane above the *dotted line* the solution is given by the unique value $\sqrt{\varphi'(\theta)}$ such that $\sqrt{\varphi'(\theta)}q_1(\tilde{\theta})$ lies on the *dotted circle*. *Right side*: effect of the reparametrization action on the space of SRVTs

$$\inf_{\sqrt{\varphi'(\theta)}\geq 0} |q_0(\theta) - \sqrt{\varphi'(\theta)}q_1(\tilde{\theta})|^2.$$

This is a d-dimensional minimization problem that can be solved explicitly; its solution is visualized in Fig. 11.3. Denote by α the angle between $q_0(\theta)$ and $q_1(\tilde{\theta})$. If $\frac{\pi}{2} \leq \alpha \leq \frac{3\pi}{2}$, then the infimum is attained for $\sqrt{\varphi'(\theta)} = 0$. In other words, for $q_1(\tilde{\theta})$ lying in the half-plane "opposite" $q_0(\theta)$, the optimal reparametrization would scale it to 0. Next we look at close by points. If the optimal scaling at θ is $\sqrt{\varphi'(\theta)} = 0$ and $\theta + \Delta\theta$ is close enough, then the angle between $q_0(\theta + \Delta\theta)$ and $q_1(\tilde{\theta})$ will also lie inside $[\frac{\pi}{2}, \frac{3\pi}{2}]$ and so $\sqrt{\varphi'(\theta + \Delta\theta)} = 0$ as well. But this would lead to φ being constant on a whole subinterval of $[0, 2\pi]$.

The true situation is more complicated than that, in particular, for closed curves, where we additionally have the nonlocal constraint $\int_{S^1} q|q|\, \mathrm{d}\theta = 0$ to satisfy. But we do observe the scaling-to-zero behavior in numerical computations; see for example Fig. 11.4.

Incompleteness The key conclusion is this: we should expect the solution ψ of the minimization problem $\inf_\varphi \operatorname{dist}_I(c_1, c_2 \circ \varphi)$ to have intervals, where it is a constant—that is true, even if we solve the problem on a finite-dimensional approximation space. If I is such an interval and $\psi|_I = \theta_I \in S^1$, then this means that the whole segment $c_1(I)$ of the first curve corresponds to the point $c_2(\theta_I)$ on the second curve.

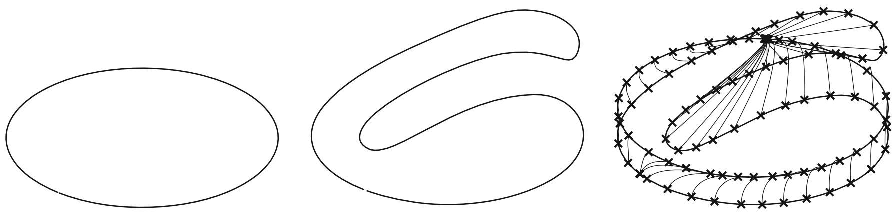


Fig. 11.4 *Left side*: initial curve. *Middle figure*: target curve. *Right figure*: minimal geodesic on shape $\mathcal{S}(S^1, \mathbb{R}^2)$ between an ellipse and an ellipse with a large fold. One can see that the fold grows out of a singular point. Image taken from [11]

Now we can switch c_1 and c_2. Then the optimal reparametrization is ψ^{-1}. However since ψ is constant on the interval I, its inverse ψ^{-1} will have a jump at the point θ_I. What does this mean for minimizing geodesics? If $c(t)$ is a length-minimizing path between c_2 and $c_1 \circ \psi^{-1}$, then the point $c_2(\theta_I)$ will "open up" to the whole segment $c_1(I)$.

A geodesic is supposed to encode the differences between the shapes represented by c_1 and c_2 in its initial velocity $\partial_t \gamma(0)$. However the geodesic starting at c_2 sees only the parametrized curve $c_1 \circ \psi^{-1}$ and since ψ^{-1} has a jump at θ_I, jumping over the interval I, this interval is missing from the curve $c_1 \circ \psi^{-1}$. How then can the geodesic encode the shape differences, if it does not "see" them?

We understand that this is not a rigorous proof.[5] However there is numerical evidence pointing in the same direction. In Fig. 11.4 we see an attempt to numerically compute the minimizing geodesic between an ellipse and an ellipse with a large fold, both considered as unparametrized curves. The picture on the right shows the point-to-point correspondences after we have minimized over the reparametrization group. We can see that indeed one point on the ellipse wants to correspond to the part of the fold, where the tangent vector points in the opposite direction to the tangent vector of the ellipse.

In the example of Fig. 11.4 we could have cleverly selected the shape with the fold as the initial curve and compute our geodesic starting from there. Then there would be no problem of one point wanting to become a whole segment. However for two general curves c_1, c_2, we would expect to encounter mixed behavior: some segments of c_1 would collapse to points on c_2 and other points on c_1 would expand to segments of c_2.

This effect is not caused by the global "curvedness" of the manifold of shapes; it is rather a manifestation of the incompleteness. We expect to see this as soon as we match two curves whose tangent vectors point in opposite directions. Since in the SRV representation distances are measured in the L^2-norm, this behavior can occur for a pair of curves with arbitrary small (geodesic) distance. The fold in Fig. 11.4 can be arbitrary small, but the behavior will be the same.

[5]See [43, Sect. 4.2] for a rigorous proof that this behavior indeed occurs for the metric $G_c(h,k) = \frac{1}{\ell_c}\int_0^{2\pi} \langle D_s h, D_s k\rangle \,\mathrm{d}s$ on the space of unparametrized open curves modulo translations.

Let us fix a representative curve c for the shape $[c]$ and let us look at the space of unparametrized curves through the lens of the exponential map, while standing at the curve c. We look at a shape $[c_1]$, represented by a curve c_1, by finding a reparametrization ψ_1, s.t. $\operatorname{dist}_I(c, c_1 \circ \psi)$ is minimal, and then we compute $v_1 = \exp_c^{-1}(c_1 \circ \psi_1)$. This vector v_1 is what we see, when we look at the shape $[c_1]$. Now, if the reparametrization ψ has jumps, then the curve $c_1 \circ \psi_1$ will miss parts of the shape $[c_1]$; furthermore, there will be several shapes $[c_2]$, distinct from $[c_1]$ only in the part that is missing from $c_1 \circ \psi_1$, such that the corresponding optimal representing curves $c_2 \circ \psi_2$ coincide with $c_1 \circ \psi_1$. This implies that $v_2 = \exp_c^{-1}(c_2 \circ \psi_2)$ coincides with v_1, while the shapes $[c_2]$ and $[c_1]$ differ. In other words, we look at different shapes, but see the same thing. In fact there are many regions in shape space that cannot be distinguished using the exponential map, and these regions start arbitrary close to the starting shape $[c]$.

Joint Reparametrizations It is possible that searching for *one* reparametrization is the wrong problem. Mathematically an equivalent way to define the geodesic distance on $B(S^1, \mathbb{R}^d)/\operatorname{Tra}$ is via

$$\operatorname{dist}_B(C_1, C_2) = \inf_{\varphi_1, \varphi_2 \in \operatorname{Diff}(S^1)} \operatorname{dist}_I(c_1 \circ \varphi_1, c_2 \circ \varphi_2).$$

Using the invariance of dist_I under reparametrizations we can recover (11.6). Now we are looking for reparametrizations of both curves such that the infimum is attained. The advantage of this approach is that we can avoid jumps that would necessarily appear in the one-reparametrization strategy, by instead setting the other reparametrization to be constant on the corresponding interval.

The underlying behavior does not change: points on one curve can be matched to intervals on the other and vice versa. If ψ_1, ψ_2 represent a pair of optimal reparametrizations for two curves c_1, c_2 and ψ_1 is constant on the interval I with $\psi_1|_I = \theta_I$, then the point $c_1(\theta_I)$ will correspond to the interval $c_2(\psi_2(I))$. Instead of jumping over the interval $\psi_2(I)$, we now reparametrize c_1 and the reparametrized curve $c_1 \circ \psi_1$ waits at θ_I until ψ_2 has moved past I.

The strategy of joint reparametrizations was proposed in [24, 33], where the authors consider only curves without the periodicity constraint. Even for open curves, it is not known whether for any two absolutely continuous curves there exists a pair of reparametrizations realizing the geodesic distance; in [24] this is shown only under the additional assumption that one of the curves is piecewise linear.

Other H^1-Metrics There are many different H^1-metrics to choose from. For a start, the constants 1 and $\frac{1}{4}$ are rather arbitrary and we could look at the full family of metrics of the form

$$G_c(h, k) = \int_{S^1} a^2 \langle D_s h^\perp, D_s k^\perp \rangle + b^2 \langle D_s h, v \rangle \langle D_s k, v \rangle \, \mathrm{d}s,$$

with $a, b > 0$. This family has been given the name *elastic metrics* and has been studied for plane curves in [11, 31]. All metrics in this family are uniformly equivalent, i.e., if G and H are two elastic metrics with possibly different constants a, b, there exists a constant C such that

$$C^{-1} H_c(h, h) \leq G_c(h, h) \leq C H_c(h, h)$$

holds for all curves c and all tangent vectors h.

All H^1-metrics can be made invariant with respect to scalings by multiplying them with an appropriate power of the length $\ell(c)$, for example the following metric is scale-invariant:

$$G_c(h, k) = \int_{S^1} \ell_c^{-3} \langle h, k \rangle + \ell_c^{-1} \langle D_s h, D_s k \rangle \, \mathrm{d}s.$$

A modified H^1-metric was introduced in [39] with the property that scalings, translations, and general deformations of the curve are all orthogonal to each other; this metric was then applied to tracking moving objects.

We have looked at completeness properties only for the H^1-metric corresponding to the SRVT; a similar, more rigorous discussion can be found for the scale-invariant version of elastic metric corresponding to the choice $a = b$ in [43] and we conjecture that all H^1-metrics share the same qualitative behavior. We will see in the next section, what happens, if we additionally penalize second and higher derivatives of the tangent vectors.

11.4 Higher-Order Sobolev Metrics

Riemannian metrics involving first derivatives of the tangent vectors can lead to very efficient computations, but some of their mathematical properties are less convenient. Now we will make the metric dependent on higher derivatives. It is not easy to give a definition of a general Sobolev-type Riemannian metric that is both general enough to encompass all known examples and concrete enough so that we can work easily with it. We will approach this class by looking at families of examples instead, noting common features as well as differences.

A very useful family is that of *Sobolev metrics with constant coefficients*. These are metrics of the form

$$G_c(h, k) = \int_{S^1} a_0 \langle h, k \rangle + a_1 \langle D_s h, D_s k \rangle + \cdots + a_n \langle D_s^n h, D_s^n k \rangle \, \mathrm{d}s, \tag{11.7}$$

with constants $a_0, \ldots, a_n$. The largest n such that $a_n \neq 0$ is called the *order* of the metric. We require $a_j \geq 0$ for the metric to be positive semi-definite, $a_n > 0$ for it to be a metric of order n, and $a_0 > 0$ for it to be nondegenerate. If $a_0 = 0$, then constant tangent vectors are in the kernel of G and, provided there is at least one nonzero coefficient, G defines a nondegenerate metric on the quotient space $\operatorname{Imm}(S^1, \mathbb{R}^d)/\operatorname{Tra}$ of regular curves modulo translations. Most of the metrics encountered in Sect. 11.3 were of this type.

Using integration by parts we can rewrite (11.7) to obtain

$$G_c(h,k) = \int_{S^1} a_0 \langle h,k\rangle + a_1\langle -D_s^2 h, k\rangle + \cdots + a_n \langle (-1)^n D_s^{2n} h, k\rangle \,\mathrm{d}s, \tag{11.8}$$

enabling us to write the metric in the form $G_c(h,k) = \int_{S^1} \langle L_c h, k\rangle \,\mathrm{d}s$, with $L_c = \sum_{j=0}^n (-1)^j a_j D_s^{2j}$, a differential operator of order $2n$.

Metrics with Nonconstant Coefficients We could loosen our restrictions on the coefficients a_j and permit them to be functions that depend on the curve c and quantities derived from it, e.g., $a_j = a_j(\ell_c, \kappa, D_s\kappa, \ldots)$. In Sect. 11.2 we have considered such metrics of order zero, the almost local metrics. Several examples of higher order metrics with nonconstant coefficients have been investigated in the literature. The completeness properties of first- and second-order metrics with coefficients depending on the length are studied in [27]. The idea in [4, 35] is to decompose a tangent vector $h = h^{\parallel} + h^{\perp}$ into a part tangent and a part normal to the curve and consider derivatives of these quantities. Some special examples of second-order metrics can be found in [14].

Geodesic Equation The geodesic equation of a Sobolev metric with constant coefficients is a nonlinear PDE, second order in time and of order $2n$ in the space variable. It is given by

$$\begin{aligned}\partial_t \left(\sum_{j=0}^n (-1)^j a_j \,|c'|\, D_s^{2j} c_t \right) = &- \frac{a_0}{2}\,|c'|\, D_s\left(\langle c_t, c_t\rangle D_s c\right) \\ &+ \sum_{k=1}^n \sum_{j=1}^{2k-1} (-1)^{k+j} \frac{a_k}{2}\,|c'|\, D_s\left(\langle D_s^{2k-j} c_t, D_s^j c_t\rangle D_s c\right).\end{aligned}$$

We can see that if $a_j = 0$ for $j \geq 1$, then this equation reduces to the geodesic equation (11.3) of the L^2-metric. The left-hand side of the geodesic equation is the time derivative of the momentum, $L_c c_t\,|c'|$. For metrics of order $n \geq 1$ the geodesic equation is locally well posed [30].

Now we come to the main difference between Sobolev metrics of order one and metrics of higher order. In a nutshell, first-order Sobolev metrics are only *weak* Riemannian metrics, while Sobolev metrics of higher order, if extended to a suitable, larger space, are *strong* Riemannian metrics.

Weak Sobolev Metrics Let G be a Sobolev metric of order one,

$$G_c(h,k) = \int_{S^1} \langle h,k\rangle + \langle D_s h, D_s k\rangle \,\mathrm{d}s = \int_{S^1} \langle h,k\rangle |c'| + \langle h', k'\rangle |c'|^{-1} \,\mathrm{d}\theta.$$

Fix the curve c and look at the inner product $G_c(\cdot,\cdot)$. The natural space to define $G_c(\cdot,\cdot)$ is the Sobolev space $H^1(S^1,\mathbb{R}^d)$ of functions with square-integrable derivatives, together with the inner product

$$\langle h,k\rangle_{H^1} = \int_{S^1} \langle h,k\rangle + \langle h',k'\rangle \,\mathrm{d}\theta.$$

If c is smooth enough, say $c \in C^1$, then we see that $G_c(\cdot,\cdot)$ defines an inner product on H^1, which is equivalent to the standard inner product. Unfortunately we cannot allow c itself to be an H^1-function. We need uniform control on the derivative c' to guarantee that the integral $\int_{S^1}\langle h',k'\rangle|c'|^{-1}\,\mathrm{d}\theta$ is finite, but the H^1-norm does not provide that. The best we can do is to extend G to the space

$$G : C^1\mathrm{Imm}(S^1,\mathbb{R}^d) \times H^1(S^1,\mathbb{R}^d) \times H^1(S^1,\mathbb{R}^d) \to \mathbb{R}.$$

In this sense G is a *weak Riemannian metric*[6]; the topology induced by the inner product $G_c(\cdot,\cdot)$, in this case the H^1-topology, is weaker than the manifold topology, here the C^∞- or C^1-topology.

Strong Sobolev Metrics The situation is different for Sobolev metrics with constant coefficients of order 2 or higher. Let us look at the example

$$G_c(h,k) = \int_{S^1} \langle h,k\rangle + \langle D_s^2 h, D_s^2 k\rangle \,\mathrm{d}s,$$

with $D_s h = \frac{1}{|c'|}h'$ and $D_s^2 h = \frac{1}{|c'|}h'' - \frac{\langle c',c''\rangle}{|c'|}h'$. Again the natural space for $G_c(\cdot,\cdot)$ is the Sobolev space $H^2(S^1,\mathbb{R}^d)$ and it would appear that we need $c \in C^2$ for the inner product to be well defined. However a careful application of Sobolev embedding and multiplier theorems—see [16, Sect. 3.2]—shows that we can extend G to a smooth inner product on the space

$$G : \mathcal{I}^2(S^1,\mathbb{R}^d) \times H^2(S^1,\mathbb{R}^d) \times H^2(S^1,\mathbb{R}^d) \to \mathbb{R};$$

here we denote by $\mathcal{I}^2(S^1,\mathbb{R}^d) = \{c \in H^2 \,:\, c'(\theta) \neq 0 \ \forall\theta \in S^1\}$ the space of H^2-curves with nonvanishing tangent vectors. The crucial fact is the Sobolev embedding $H^2 \hookrightarrow C^1$, implying that the H^2-norm controls first derivatives uniformly. This also implies that $\mathcal{I}^2$ is an open set in H^2. Thus G becomes a *strong Riemannian metric* on $\mathcal{I}^2(S^1,\mathbb{R}^d)$; the topology induced by each inner product $G_c(\cdot,\cdot)$ coincides with the manifold topology.

[6]An infinite-dimensional Riemannian manifold (M,g) is called *strong*, if g induces the natural topology on each tangent space, or equivalently, if the map $g : TM \to (TM)'$ is an isomorphism. If g is merely a smoothly varying nondegenerate bilinear form on TM we call (M,g) a *weak* Riemannian manifold, indicating that the topology induced by g can be weaker than the natural topology on TM or equivalently $g : TM \to (TM)'$ is only injective.

Similarly Sobolev metrics of order n with constant coefficients induce strong metrics on the space $\mathcal{I}^n(S^1, \mathbb{R}^d)$ of regular Sobolev curves, provided $n \geq 2$.

Note however that Sobolev metrics of order 2 and higher are strong metrics only when considered on the larger space $\mathcal{I}^n(S^1, \mathbb{R}^d)$, not on the space $\mathrm{Imm}(S^1, \mathbb{R}^d)$ of smooth curves. On $\mathrm{Imm}(S^1, \mathbb{R}^d)$ the metric is still a weak metric. That said the difference between metrics of order 1 and those of higher order is that for higher-order metrics we are able to pass to the larger space $\mathcal{I}^n(S^1, \mathbb{R}^d)$—one could say we are "completing" the space of smooth curves—on which it becomes a strong metric, while for first-order metrics such a completion does not exist.

Properties of Strong Metrics The following is a list of properties we get "for free" simply by working with a smooth, strong Riemannian metric as opposed to a weak one:

- The Levi-Civita covariant derivative exists and the geodesic equation has local solutions, which depend smoothly on the initial conditions.
- The exponential map exists and is a local diffeomorphism.
- The induced geodesic distance is point separating and generates the same topology as the underlying manifold.

The theory of strong, infinite-dimensional Riemannian manifolds is described in [20, 25]. For weak Riemannian manifolds all these properties have to be established by hand and there are examples where they fail to hold. The geodesic distance for the L^2-metric discussed in Sect. 11.2 on the space of curves vanishes identically [28, 29]; it is not known whether the geodesic equation for the L^2-metric has solutions and [12] presents a weak Riemannian manifold that has no Levi-Civita covariant derivative.

We would like to note that the distinction between weak and strong Riemannian metrics arises only in infinite dimensions. Every Riemannian metric on a finite dimensional manifold is strong. Therefore phenomena, like the vanishing of the geodesic distance or the failure of geodesics to exist, can only arise in infinite dimensions. For better or for worse, this is the setting, where the joy and pain of shape analysis occurs.

Completeness Properties What are the operations on the manifold of curves that are used in shape analysis?

1. We want to use the Riemannian exponential map $\exp_c : T_cM \to M$ to pass between the tangent space at one point and the manifold itself. Its inverse $\log_c = \exp_c^{-1}$ allows us to represent the nonlinear manifold or at least a part thereof in a vector space.
2. Given two curves, we want to compute the geodesic distance between them, that is, the length of the shortest path joining them. Often we are also interested in the shortest path itself; it can be used to transfer information between two curves and the midpoint can serve as the average of its endpoints.
3. Given a finite set $\{c_1, \dots, c_n\}$ of curves, we are interested in the average of this set. On a Riemannian manifold this is usually the Fréchet or Karcher mean, i.e., a curve c^*, minimizing the sum of squared distances, $c^* = \operatorname{argmin}_c \sum_i d(c, c_i)^2$, where d is the geodesic distance.

This is by no means an exhaustive list, but describes rather the basic operations, one wants to perform. The ability to do so places conditions on the manifold of curves and the Riemannian metric. Let us look at these conditions:

1. On a general Riemannian manifold the exponential map $\exp_c : U \to M$ is defined only on an open neighborhood $U \subseteq T_cM$ of 0 and it is rarely known exactly how U looks like. For us to be able to freely map tangent vectors to curves, we want a globally defined exponential map. Since the exponential map is defined as $\exp_c(h) = \gamma(1)$, where γ is the geodesic with initial conditions (c, h), and using the property $\exp_c(th) = \gamma(t)$, we can see that a globally defined exponential map is equivalent to requiring that geodesics exist for all time. This property is called *geodesic completeness*.

(1') Asking for the exponential map to be invertible is more difficult. On a strong Riemannian manifold this is always the case locally. Imposing it globally is a very restrictive condition. The Weil–Peterson metric [36] comes closest; it is a metric with negative sectional curvature, meaning that the derivative of the exponential map is everywhere invertible.

2. Here we want to know whether any two curves can be joined by a minimizing geodesic, i.e., a geodesic, whose length realizes the geodesic distance. On a finite-dimensional manifold geodesic completeness would imply this property; this is not the case in infinite dimensions [1]. This is not to say that we cannot hope for minimizing geodesics to exist, but rather that it will have to be proven independently of (1).
3. Ensuring that the Fréchet mean exists for all finite collections of curves is difficult. But there is a theorem [3] stating that the mean exists and is unique on a dense subset of the n-fold product $M \times \cdots \times M$, provided the manifold is *metrically complete*. This means that the manifold (M, d) together with the induced geodesic distance is complete as a metric space.

Properties (1), (2), and (3) for Riemannian manifolds are called completeness properties. In finite dimensions the theorem of Hopf–Rinow states that (1) and (3) are equivalent and either one implies (2). For infinite-dimensional strong Riemannian manifolds the only remaining implication is that metric completeness implies geodesic completeness.[7]

Completeness for Sobolev Metrics Let us look at the situation for Sobolev metrics on the space of parametrized curves. We have argued in Sect. 11.3 that we should not expect H^1-metrics to be geodesically or metrically complete. Things look better

[7] A counterexample, showing that in infinite dimensions metric and geodesic completeness together do not imply existence of minimizing geodesics, can be found in [1]; similarly, that geodesic completeness does not imply metric completeness is shown in [2].

for Sobolev metrics of higher order. In fact it is shown in [15, 16] that these metrics satisfy all the above-mentioned completeness properties.[8]

Theorem 11.1. *Let $n \geq 2$ and let G be a Sobolev metric of order n with constant coefficients. Then*

1. *$(\mathcal{I}^n(S^1, \mathbb{R}^d), G)$ is geodesically complete;*
2. *Any two elements in the same connected component of $\mathcal{I}^n(S^1, \mathbb{R}^d)$ can be joined by a minimizing geodesic;*
3. *$(\mathcal{I}^n(S^1, \mathbb{R}^d), \operatorname{dist}_{\mathcal{I}})$ is a complete metric space.*

The geodesic equation for Sobolev metrics has a smoothness preserving property. If the initial conditions are smoother that H^n—let us say the initial curve and initial velocity are C^∞—then the whole geodesic will be as smooth as the initial conditions. Therefore the space $(\operatorname{Imm}(S^1, \mathbb{R}^d), G)$ of smooth immersions with a Sobolev metric of order n is also geodesically complete.

Completeness for Unparametrized Curves Similar completeness properties hold for unparametrized curves. The correct space, where to look for completeness, is the quotient

$$\mathcal{B}^n(S^1, \mathbb{R}^d) = \mathcal{I}^n(S^1, \mathbb{R}^d)/\mathcal{D}^n(S^1)$$

of regular curves of Sobolev class H^n modulo reparametrizations of the same regularity; the group $\mathcal{D}^n(S^1) = \{\varphi \in H^n(S^1, S^1) : \varphi'(\theta) > 0\}$ is the group of H^n-diffeomorphisms. We have the following theorem [15]:

Theorem 11.2. *Let $n \geq 2$ and let G be a Sobolev metric of order n with constant coefficients. Then*

1. *$(\mathcal{B}^n(S^1, \mathbb{R}^d), \operatorname{dist}_{\mathcal{B}})$ with the quotient metric induced by the geodesic distance on $(\mathcal{I}^n, G)$ is a complete metric space;*
2. *Given $C_1, C_2 \in \mathcal{B}^n$ in the same connected component, there exist $c_1, c_2 \in \mathcal{I}^n$ with $c_1 \in \pi^{-1}(C_1)$ and $c_2 \in \pi^{-1}(C_2)$, such that*

$$\operatorname{dist}_{\mathcal{B}}(C_1, C_2) = \operatorname{dist}_{\mathcal{I}}(c_1, c_2);$$

equivalently, the infimum in

$$\operatorname{dist}_{\mathcal{B}}(\pi(c_1), \pi(c_2)) = \inf_{\varphi \in \mathcal{D}^n(S^1)} \operatorname{dist}_{\mathcal{I}}(c_1, c_2 \circ \varphi)$$

is attained;

[8]A related result holds for the space of curves of bounded second variation, together with a Finsler BV^2-metric. It is shown in [32] that any two curves in the same connected component of the space $\mathrm{BV}^2(S^1, \mathbb{R}^2)$ can be joined by a length-minimizing path.

3. *Any two shapes in the same connected component of* $\mathcal{B}^n(S^1, \mathbb{R}^d)$ *can be joined by a minimizing geodesic.*

The only drawback is that the space $\mathcal{B}^n(S^1, \mathbb{R}^d)$ of Sobolev shapes is not a manifold any more.[9] It is however a topological space and equipped with the geodesic distance function a metric space. We have to understand a minimizing geodesic in the sense of metric spaces, i.e., a curve $\gamma : I \to \mathcal{B}^n$ is a minimizing geodesic, if

$$\operatorname{dist}_{\mathcal{B}}(\gamma(t), \gamma(s)) = \lambda |t - s|$$

holds for some $\lambda > 0$ and all $t, s \in I$.

We would like to point out that part (2) of Theorem 11.2 is the counterpart of the incompleteness discussion in Sect. 11.3. This theorem states that given two shapes, represented by two parametrized curves, we can find an optimal reparametrization φ of the second curve. The fact that $\varphi \in \mathcal{D}^n(S^1)$ guarantees that φ is at least a C^1-diffeomorphism of the circle; thus no intervals are collapsed to single points and neither φ nor φ^{-1} has any jumps.

Numerical Computations We have argued that Sobolev metrics of sufficiently high order have nice mathematical properties, which are relevant to applications in shape analysis. What we have not done is present convincing applications, showcasing their superior performance. This is because the numerical computation of minimizing geodesics between parametrized or unparametrized curves is still an open problem. First attempts at discretizing special cases of H^2-metrics can be found in [4, 32]. While first-order metrics have nice representations in terms of the SRVT or the basic mapping [43], which greatly simplifies the numerics, there is no such analogon for higher-order metrics.[10] Finding a robust and stable discretization of metrics of order 2 and higher remains a challenge.

11.5 Riemannian Metrics on the Space of Surfaces

In the previous sections we have presented several reparametrization invariant metrics on the space of curves. We want to conclude the exposition with a short excursion to the space of regularly parametrized surfaces, i.e., $\operatorname{Imm}(M, \mathbb{R}^d) = \{f \in$

[9]This has to do with smoothness properties of the composition map in Sobolev spaces. While the smooth reparametrization group $\operatorname{Diff}(S^1)$ acts smoothly on the space $\operatorname{Imm}(S^1, \mathbb{R}^d)$ of smooth curves [21], the group of Sobolev reparametrizations $\mathcal{D}^n(S^1)$ acts only continuously on Sobolev curves $\mathcal{I}^n(S^1, \mathbb{R}^d)$. Moreover, the smooth shape space $B(S^1, \mathbb{R}^d)$ is (apart from isolated singularities) a manifold, but the Sobolev shape space $\mathcal{B}^n(S^1, \mathbb{R}^d)$ is only a topological space. This is the price we have to pay for completeness. See [15, Sect. 6] for details.

[10]In [14] a representation of second-order metrics, similar to the SRVT, was developed. However image of the resulting transformations has infinite co-dimension, which, compared to the SRVT, complicates the situation.

$C^\infty(M,\mathbb{R}^d)\ :\ T_xf$ injective $\forall x \in M\}$ with M a compact two-dimensional manifold without boundary. Typical choices for M are the sphere S^2 and the torus $S^1 \times S^1$. In particular we will describe how the previously described metrics can be generalized from $\operatorname{Imm}(S^1,\mathbb{R}^2)$ to $\operatorname{Imm}(M,\mathbb{R}^d)$. We will follow the presentation of [5, 6].

Sobolev Metrics To generalize Sobolev metrics (11.8) from the space of curves to the space of surfaces, we need the right replacements for the arc length derivative D_s and integration $\mathrm{d}s$. For an immersion $f \in \operatorname{Imm}(M,\mathbb{R}^d)$, we denote by $g = g^f = f^*\langle\cdot,\cdot\rangle$ the pullback of the Euclidean metric to M. This makes (M,g) into a Riemannian manifold with a Laplace operator $\Delta^g = -\operatorname{div}^g \circ \operatorname{grad}^g$ and volume form vol^g. We will use Δ^g and vol^g as replacements for $-D_s^2$ and $\mathrm{d}s$. A Sobolev metric of order n on $\operatorname{Imm}(M,\mathbb{R}^d)$ is given by

$$G_f(h,k) = \int_M a_0\langle h,k\rangle + a_1\langle \Delta^g h,k\rangle + \cdots + a_n\langle(\Delta^g)^n h,k\rangle \operatorname{vol}^g; \tag{11.9}$$

here the tangent vectors h,k are seen as maps $h,k : M \to \mathbb{R}^d$ and the Laplace operator acts on each coordinate separately.[11]

The associated operator L_f, allowing us to write $G_f(h,k) = \int_{S^1}\langle L_f h,k\rangle\,\mathrm{d}s$, is given by $L_f = \sum_{j=0}^n a_j\,(\Delta^g)^j \otimes \operatorname{vol}^g$. Every metric in this family is invariant under the action of the diffeomorphism group $\operatorname{Diff}(M)$ and induces a Riemannian metric on the quotient space of unparametrized surfaces $B(M,\mathbb{R}^3) = \operatorname{Imm}(M,\mathbb{R}^3)/\operatorname{Diff}(M)$.

Similarly as in the previous section one can also allow the coefficients a_j to depend on the surface f. Coefficients depending on the total volume, the mean curvature, and the Gauß curvature have been considered in [7]. The class of almost local metrics on surfaces has also been studied [6].

Geodesic Distance For the geodesic distance we obtain similar results as for curves: the geodesic distance vanishes for the L^2-metric on $\operatorname{Imm}(M,\mathbb{R}^d)$. Both higher-order Sobolev metrics and almost local metrics depending on mean curvature or total volume induce point-separating distances on the space of unparametrized surfaces. Whether higher-order Sobolev metrics induce a point-separating distance function on $\operatorname{Imm}(M,\mathbb{R}^d)$ itself is not known.

Geodesic Equation The formulas for the geodesic equations for Sobolev metrics (11.9) become very quickly very technical; see [5]. As an example we present

[11] On $\operatorname{Imm}(M,\mathbb{R}^d)$ there is no canonical way to define Sobolev metrics. We could have also used the definition

$$G_f(h,k) = \int_M a_0\langle h,k\rangle + \cdots + a_n\langle(\nabla^g)^n h,(\nabla^g)^n k\rangle \operatorname{vol}^g,$$

where ∇^g is the covariant derivative on M. Then the differential operator associated to this metric is $L_f = \sum_{j=0}^n a_j\,((\nabla^g)^*)^n\,(\nabla^g)^n \otimes \operatorname{vol}^g$, with $(\nabla^g)^* = \operatorname{Tr}(g^{-1}\nabla^g)$ denoting the adjoint of ∇^g. When $n \geq 2$, the operator $((\nabla^g)^*)^n\,(\nabla^g)^n$ differs from $(\Delta^g)^n$ in the lower-order terms—they are connected via the Weitzenböck formulas.

the geodesic equation of the H^1-metric with coefficients $a_0 = a_1 = 1$; this is the metric induced by the operator field $L_f = 1 + \Delta^g$:

$$\begin{aligned}\partial_t \left(L_f f_t \otimes \operatorname{vol}^g\right) = \Bigg(& \operatorname{Tr}\left(g^{-1} S^f g^{-1} \langle \nabla^g f_t, \nabla^g f_t \rangle\right) - \frac{1}{2} \operatorname{Tr}\left(g^{-1} \nabla^g \langle \nabla^g f_t, f_t \rangle\right) . H^f \\ & - \frac{1}{2} \langle L_f f_t, f_t \rangle . H^f - Tf . \langle L_f f_t, \nabla^g f_t \rangle^\sharp \Bigg) \otimes \operatorname{vol}^g;\end{aligned}$$

here S^f denotes the second fundamental form, $H^f = \operatorname{Tr}(g^{-1} S^f)$ is the vector valued mean curvature, and ∇^g is the covariant derivative of the surface f.

Outlook On spaces of surfaces the theory of Sobolev metrics is significantly less developed than on the space of curves. We conjecture that Sobolev metric of order $n \geq 3$ will again be strong Riemannian metrics on the space $\mathcal{I}^n(M, \mathbb{R}^d)$ of Sobolev surfaces. Nothing is known about completeness properties of these metrics.

An analogue of the SRVT transform has been developed for surfaces in [19, 22]. However questions regarding the invertibility of the transform and the characterization of its image remain open. So far no numerical experiments for higher-order Sobolev metrics on the space of surfaces have been conducted.

Acknowledgements This work was supported by the Erwin Schrödinger Institute. Martin Bauer was supported by the 'Fonds zur Förderung der wissenschaftlichen Forschung, Projekt P 24625.'

References

1. Atkin CJ (1975) The Hopf-Rinow theorem is false in infinite dimensions. Bull Lond Math Soc 7(3):261–266
2. Atkin CJ (1997) Geodesic and metric completeness in infinite dimensions. Hokkaido Math J 26(1):1–61
3. Azagra D, Ferrera J (2005) Proximal calculus on Riemannian manifolds. Mediterr J Math 2(4):437–450
4. Bauer M, Harms P (2015) Metrics on spaces of surfaces where horizontality equals normality, with P. Harms, Differential Geometry and its Applications 39, pp 166–183
5. Bauer M, Harms P, Michor PW (2011) Sobolev metrics on shape space of surfaces. J Geom Mech 3(4):389–438
6. Bauer M, Harms P, Michor PW (2012) Almost local metrics on shape space of hypersurfaces in n-space. SIAM J Imaging Sci 5(1):244–310
7. Bauer M, Harms P, Michor PW (2012) Sobolev metrics on shape space, II: weighted Sobolev metrics and almost local metrics. J Geom Mech 4(4):365–383
8. Bauer M, Bruveris M, Harms P, Michor PW (2012) Vanishing geodesic distance for the Riemannian metric with geodesic equation the KdV-equation. Ann Global Anal Geom 41(4):461–472
9. Bauer M, Bruveris M, Harms P, Michor PW (2013) Geodesic distance for right invariant Sobolev metrics of fractional order on the diffeomorphism group. Ann Global Anal Geom 44(1):5–21

10. Bauer M, Bruveris M, Michor PW (2013) Geodesic distance for right invariant Sobolev metrics of fractional order on the diffeomorphism group. II. Ann Global Anal Geom 44(4):361–368
11. Bauer M, Bruveris M, Marsland S, Michor PW (2014) Constructing reparameterization invariant metrics on spaces of plane curves. Differ Geom Appl 34:139–165
12. Bauer M, Bruveris M, Michor PW (2014) Homogeneous Sobolev metric of order one on diffeomorphism groups on real line. J Nonlinear Sci, 24(5):769–808
13. Bauer M, Bruveris M, Michor PW (2014) Overview of the geometries of shape spaces and diffeomorphism groups. J Math Imaging Vision 50:60–97
14. Bauer M, Bruveris M, Michor PW (2014) R-transforms for Sobolev H^2-metrics on spaces of plane curves. Geom Imaging Comput 1(1):1–56
15. Bruveris M (2015) Completeness properties of Sobolev metrics on the space of curves. J Geom Mech 7(2):125–150
16. Bruveris M, Michor PW, Mumford D (2014) Geodesic completeness for Sobolev metrics on the space of immersed plane curves. Forum Math Sigma 2:e19 (38 p.)
17. Eslitzbichler M (2015) Modelling character motions on infinite-dimensional manifolds. Vis Comput 31(9):1179–1190
18. Glaunès J, Qiu A, Miller MI, Younes L (2008) Large deformation diffeomorphic metric curve mapping. Int J Comput Vis 80(3):317–336
19. Jermyn IH, Kurtek S, Klassen E, Srivastava A (2012) Elastic shape matching of parameterized surfaces using square root normal fields. In: Proceedings of the 12th European conference on computer vision - volume part V, ECCV'12. Springer, Berlin, Heidelberg, pp 804–817
20. Klingenberg WPA (1995) Riemannian geometry, 2nd edn. de Gruyter studies in mathematics, vol 1. Walter de Gruyter & Co., Berlin
21. Kriegl A, Michor PW (1997) The convenient setting for global analysis. Surveys and monographs, vol 53. AMS, Providence
22. Kurtek S, Klassen E, Ding Z, Srivastava A (2010) A novel riemannian framework for shape analysis of 3D objects. In: IEEE computer society conference on computer vision and pattern recognition, pp 1625–1632
23. Laga H, Kurtek S, Srivastava A, Miklavcic SJ (2014) Landmark-free statistical analysis of the shape of plant leaves. J Theor Biol 363:41–52
24. Lahiri S, Robinson D, Klassen E (2015) Precise matching of PL curves in R^N in the square root velocity framework. www.arxiv.org/abs/1501.00577
25. Lang S (1999) Fundamentals of differential geometry. Graduate texts in mathematics, vol 191. Springer, New York
26. Liu W, Srivastava A, Zhang J (2011) A mathematical framework for protein structure comparison. PLoS Comput Biol 7(2):e1001075
27. Mennucci A, Yezzi A., Sundaramoorthi G (2008) Properties of Sobolev-type metrics in the space of curves. Interfaces Free Bound 10(4):423–445
28. Michor PW, Mumford D (2005) Vanishing geodesic distance on spaces of submanifolds and diffeomorphisms. Doc Math 10:217–245 (electronic)
29. Michor PW, Mumford D (2006) Riemannian geometries on spaces of plane curves. J Eur Math Soc (JEMS) 8:1–48
30. Michor PW, Mumford D (2007) An overview of the Riemannian metrics on spaces of curves using the Hamiltonian approach. Appl Comput Harmon Anal 23(1):74–113
31. Mio W, Srivastava A, Joshi S (2007) On shape of plane elastic curves. Int J Comput Vis 73(3):307–324
32. Nardi G, Peyré G, Vialard F-X (2014) Geodesics on shape spaces with bounded variation and Sobolev metrics. http://www.arxiv.org/abs/1402.6504
33. Robinson DT (2012) Functional data analysis and partial shape matching in the square root velocity framework. Ph.D. thesis, Florida State University
34. Shah J (2008) H^0-type Riemannian metrics on the space of planar curves. Q Appl Math 66(1):123–137
35. Shah J (2013) An H^2 Riemannian metric on the space of planar curves modulo similitudes. Adv Appl Math 51(4):483–506

36. Sharon E, Mumford D (2006S) 2D-shape analysis using conformal mapping. Int J Comput Vis 70:55–75
37. Srivastava A, Klassen E, Joshi SH, Jermyn IH (2011) Shape analysis of elastic curves in Euclidean spaces. IEEE Trans Pattern Anal 33(7):1415–1428
38. Su J, Kurtek S, Klassen E, Srivastava A (2014) Statistical analysis of trajectories on Riemannian manifolds: bird migration, hurricane tracking and video surveillance. Ann Appl Stat 8(1):530–552
39. Sundaramoorthi G, Mennucci A, Soatto S, Yezzi A (2011) A new geometric metric in the space of curves, and applications to tracking deforming objects by prediction and filtering. SIAM J Imaging Sci 4(1):109–145
40. Xie Q, Kurtek S, Srivastava A (2014) Analysis of AneuRisk65 data: elastic shape registration of curves. Electron J Stat 8:1920–1929
41. Yezzi A, Mennucci A (2005) Conformal metrics and true “gradient flows” for curves. In: Proceedings of the 10th IEEE international conference on computer vision, vol 1, pp 913–919. IEEE Computer Society, Washington, DC.
42. Younes L (1998) Computable elastic distances between shapes. SIAM J Appl Math 58(2):565–586 (electronic)
43. Younes L, Michor PW, Shah J, Mumford D (2008) A metric on shape space with explicit geodesics. Atti Accad Naz Lincei Cl Sci Fis Mat Natur Rend Lincei (9) Mat Appl 19(1):25–57

Chapter 12
Elastic Shape Analysis of Surfaces and Images

Sebastian Kurtek, Ian H. Jermyn, Qian Xie, Eric Klassen, and Hamid Laga

Abstract We describe two Riemannian frameworks for statistical shape analysis of parameterized surfaces. These methods provide tools for registration, comparison, deformation, averaging, statistical modeling, and random sampling of surface shapes. A crucial property of both of these frameworks is that they are invariant to reparameterizations of surfaces. Thus, they result in natural shape comparisons and statistics. The first method we describe is based on a special representation of surfaces termed square-root functions (SRFs). The pullback of the $\mathbb{L}^2$ metric from the SRF space results in the Riemannian metric on the space of surfaces. The second method is based on the elastic surface metric. We show that a restriction of this metric, which we call the partial elastic metric, becomes the standard $\mathbb{L}^2$ metric under the square-root normal field (SRNF) representation. We show the advantages of these methods by computing geodesic paths between highly articulated surfaces and shape statistics of manually generated surfaces. We also describe applications of this framework to image registration and medical diagnosis.

S. Kurtek (✉)
Department of Statistics, The Ohio State University, Columbus, OH, USA
e-mail: kurtek.1@stat.osu.edu

I.H. Jermyn
Department of Mathematical Sciences, Durham University, Durham, UK
e-mail: i.h.jermyn@durham.ac.uk

Q. Xie
Department of Statistics, Florida State University, Tallahassee, FL, USA
e-mail: qxie@stat.fsu.edu

E. Klassen
Department of Mathematics, Florida State University, Tallahassee, FL, USA
e-mail: klassen@math.fsu.edu

H. Laga
Phenomics and Bioinformatics Research Centre, University of South Australia, Adelaide, SA, Australia
e-mail: Hamid.Laga@unisa.edu.au

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_12

12.1 Introduction

Shape is an important physical property of 3D objects that helps characterize their appearance. As a result, statistical shape analysis, which is concerned with quantifying shape as a random object and developing tools for generating shape registrations, comparisons, deformations, averages, probability models, hypothesis tests, Bayesian estimates, and other statistical procedures on shape spaces, plays an important role in many applications including medical imaging, biometrics, bioinformatics, 3D printing, and computer graphics. Medical imaging is a primary example of an application where shape statistics can play a very important role. Advances in noninvasive imaging technology, such as magnetic resonance imaging (MRI), have enabled researchers to study biological variations of anatomical structures. Studying shapes of 3D anatomies is of particular interest because many complex diseases can potentially be linked to alterations of these shapes. Thus, statistical shape analysis can be a central tool in disease diagnosis and design of novel treatment strategies. The two methods described in this chapter have been successfully applied to classification of attention deficit hyperactivity disorder (ADHD) [31, 32, 47] and mathematics deficiency [35] using shapes of subcortical structures, and statistical modeling of endometrial tissues [41] in the presence of endometriosis.

In this chapter, we focus on shape analysis of parameterized surfaces. In particular, we describe two recent Riemannian frameworks that allow comparison, matching, deformation, averaging, and statistical modeling of observed shapes. This work was motivated by the widespread success of elastic Riemannian shape analysis of curves [27, 37, 40, 42, 51]. The main benefit of Riemannian shape analysis is in the breadth of mathematical tools at our disposal, resulting in a *principled* statistical shape analysis framework. We note that there are currently very few Riemannian approaches to shape analysis of 3D objects. Similar to curves, researchers have proposed many different representations of surfaces. Many groups study shapes of surfaces by embedding them in volumes and deforming these volumes under the large deformation diffeomorphic metric mapping (LDDMM) framework [11, 12, 20, 26, 46]. While these methods are both prominent and pioneering in medical image analysis, they are typically computationally expensive since they try to match not only the objects of interest but also some background space containing them. An important benefit of the LDDMM framework is that it utilizes the Riemannian geometry of the reparameterization group to compute shape comparisons and deformations. A closely related approach utilizes inner metrics to describe shape deformations, which are prescribed directly on the surface [3]. Others study 3D shapes using manually generated landmarks under Kendall's shape theory [15], level sets [39], curvature flows [22], point clouds [2], or medial axis representations [5, 19].

However, the most natural representation for studying the shape of a 3D object would seem to be a parameterized surface. In this case, there is an additional difficulty in handling the parameterization variability. Specifically, a reparameterization of a surface (achieved using an appropriate function $\gamma \in \Gamma$ made precise later) does not change the shape of the object. Thus, a main goal in shape analysis is to define Riemannian metrics and subsequent statistical analyses, which are invariant to the introduction of an arbitrary parameterization in shape representations. Methods such as SPHARM [6, 29] or SPHARM-PDM [17, 44] tackle this problem by choosing a fixed parameterization that is analogous to the arc-length parameterization on curves. Kilian et al. [30] presented a technique for computing geodesics between triangulated meshes (discretized surfaces) but at their given parameterizations, thus requiring the registration problem to be solved manually or using some other available method. In fact, a large set of papers in the literature treat surface registration as a preprocessing step [28]. In such methods, points across surfaces are first registered using some predefined energy functions such as the entropy [8] or the minimum description length [13]. Once the surfaces are registered, they are compared using standard procedures. There are several fundamental problems with such approaches; first, the energy used for registration does not lead to a proper distance on the shape space of surfaces. Second, the registration procedure is typically completely unrelated to the rest of the analysis. In other words, the two tasks are performed under different metrics. Figure 12.1 displays the various representations of surfaces used for shape analysis.

The remainder of this chapter describes two Riemannian frameworks for statistical shape analysis of parameterized surfaces that overcome the above presented difficulties. In particular, the defined Riemannian metrics are invariant to reparameterizations of surfaces and allow shape comparisons via geodesic paths in the shape space. Geodesics can in turn be used to define statistics on shape spaces including the Karcher mean and covariance. Tools for other statistical procedures are also presented including principal component analysis and random sampling from

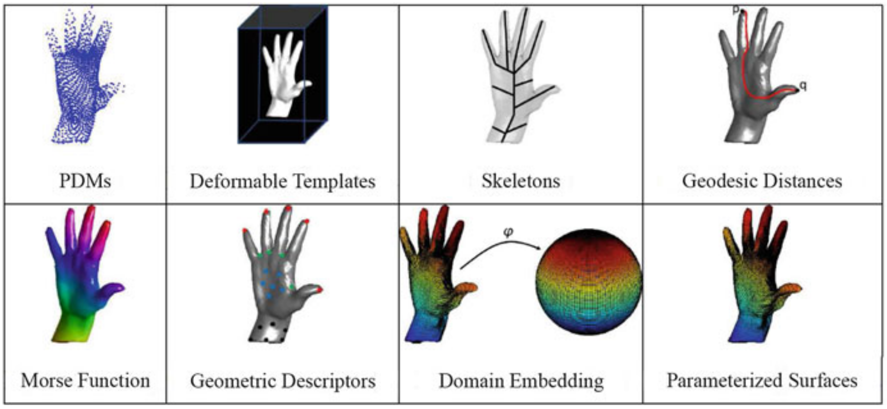


Fig. 12.1 Different representations of surfaces for shape analysis. (courtesy of Srivastava et al. [43])

Gaussian-type shape models. Finally, an application to classification of attention deficit hyperactivity disorder is presented, where the benefits of these methods are seen through a superior classification rate.

12.2 Surface Representations and Riemannian Metrics

Let $\mathcal{F}$ be the space of all smooth embeddings $f : D \to \mathbb{R}^3$, where D represents the surface domain. Depending on the application of interest D can be the sphere for closed surfaces, disk for hemispherical surfaces, or square for quadrilateral surfaces. Each such embedding defines a surface $f(D) \subset \mathbb{R}^3$. Let Γ be the set of all diffeomorphisms of D. Γ will act as the reparameterization group for surfaces. It acts naturally on $\mathcal{F}$ by composition: $(f, \gamma) \mapsto f \circ \gamma$. Thus, the space of surfaces can be thought of as the quotient space $\mathcal{F}/\Gamma$, i.e. the space of equivalence classes under the action of Γ endowed with appropriate structures.

Since $\mathcal{F}$ is a vector space, the tangent space at any point $f \in \mathcal{F}$ denoted by $T_f(\mathcal{F})$ can be identified with $\mathcal{F}$. Given the Lebesgue measure ds on D, one can define a Riemannian structure on $\mathcal{F}$ as follows. For $\delta f_1, \delta f_2 \in T_f(\mathcal{F})$, define the $\mathbb{L}^2$ Riemannian metric as $\langle \delta f_1, \delta f_2 \rangle = \int_D \langle \delta f_1(s), \delta f_2(s) \rangle \, ds$, where the inner product inside the integral is the Euclidean inner product in $\mathbb{R}^3$. The resulting squared $\mathbb{L}^2$ distance between two surfaces $f_1, f_2 \in \mathcal{F}$ is $\int_D |f_1(s) - f_2(s)|^2 ds$. In this expression, $| \cdot |$ denotes the standard two-norm of a vector in $\mathbb{R}^3$. While simple, this metric has a critical defect: just as in the case of curves, the action of Γ does not preserve distances. In other words, the group Γ does not act on the space $\mathcal{F}$ by isometries under the $\mathbb{L}^2$ metric. This is easily seen through the following expression: $d(f_1 \circ \gamma, f_2 \circ \gamma)^2 = \int_D |f_1(\gamma(s)) - f_2(\gamma(s))|^2 ds = \int_D |f_1(\tilde{s}) - f_2(\tilde{s})|^2 J_\gamma(s)^{-1} d\tilde{s} \neq d(f_1, f_2)^2$, where J_γ is the determinant of the Jacobian of γ. In this equation, we have used the substitution $\tilde{s} = \gamma(s)$ and $J_\gamma(s) ds = d\tilde{s}$. The inequality comes from the fact that, in general, γ is not area preserving and thus the determinant of the Jacobian is not one at all points. The lack of isometry means that the shape space $\mathcal{F}/\Gamma$ does not inherit the structure of a Riemannian manifold from $\mathcal{F}$, thereby making this metric difficult to use for analyzing shapes of parameterized surfaces. One solution is to restrict attention to area-preserving diffeomorphisms [21], but this restriction proves to be very limiting in practice. Another solution is to develop a new representation of surfaces such that the action of Γ preserves $\mathbb{L}^2$ distances. Then, one can use the pullback of the $\mathbb{L}^2$ metric from the new representation space to define a Riemannian structure on the space $\mathcal{F}$. We take this approach and present two different representations of surfaces that satisfy these conditions: the square-root function (SRF) and the square-root normal field (SRNF).

12.2.1 Square-Root Function Representation and Pullback Metric

Let $(x, y) : D \to \mathbb{R}^2$ be coordinates on D; then, $f_x(s) = \frac{\partial f}{\partial x}(s)$ and $f_y(s) = \frac{\partial f}{\partial y}(s)$. To endow $\mathcal{F}$ with a Riemannian metric, we begin by defining a new representation of surfaces called square-root functions or SRFs [32, 33, 38]:

Definition 1. Define the mapping $Q : \mathcal{F} \to \mathbb{L}^2$ as $Q(f)(s) = q(s) = \sqrt{|n(s)|}f(s)$, where $n(s) = f_x(s) \times f_y(s)$ is the normal vector to the surface f at point s.

The factor $|n(s)|$ can be interpreted as the ratio of infinitesimal areas of the surface at $f(s)$ and the domain at s, the "area multiplication factor." For any $f \in \mathcal{F}$, we will refer to $q(s) = Q(f)(s)$ as the SRF of f. Since we defined $\mathcal{F}$ as the space of smooth surfaces, $Q(\mathcal{F})$ is a subset of $\mathbb{L}^2(D, \mathbb{R}^3)$, henceforth denoted $\mathbb{L}^2$. If a surface f is reparameterized by γ, then its SRF changes to $(q, \gamma) = (q \circ \gamma)\sqrt{J_\gamma}$. This can be extended to a right action of Γ on $\mathbb{L}^2$.

We choose the natural $\mathbb{L}^2$ metric on the space of SRFs: the inner product of any two elements $\delta q_1, \delta q_2 \in T_q(\mathbb{L}^2)$ is $\langle \delta q_1, \delta q_2 \rangle = \int_D \langle \delta q_1(s), \delta q_2(s) \rangle \, ds$. This metric has the key property that Γ acts by isometries on $\mathbb{L}^2$. As a result, if we pullback this metric to $\mathcal{F}$, the resulting Riemannian metric is also preserved by the action of Γ, unlike the plain $\mathbb{L}^2$ metric on $\mathcal{F}$ mentioned at the beginning of this section. To obtain the pullback metric, we must first derive the differential of the mapping Q at any surface f, denoted by $Q_{*,f}$. This is a linear mapping between tangent spaces $T_f(\mathcal{F})$ and $T_{Q(f)}(\mathbb{L}^2)$. For a tangent vector $\delta f \in T_f(\mathcal{F})$, the mapping $Q_{*,f} : T_f(\mathcal{F}) \to T_{Q(f)}(\mathbb{L}^2)$ is given by

$$Q_{*,f}(\delta f) = \frac{1}{2|n|^{\frac{3}{2}}}(n \cdot n_{\delta f})f + \sqrt{|n|}\, \delta f. \tag{12.1}$$

The quantity $n_{\delta f}$ depends on both f and δf and is defined as $f_x \times \delta f_y + \delta f_x \times f_y$. The pullback metric on $\mathcal{F}$ is then defined as usual.

Definition 2. For any $f \in \mathcal{F}$ and any $\delta f_1, \delta f_2 \in T_f(\mathcal{F})$, define the inner product

$$\langle\langle \delta f_1, \delta f_2 \rangle\rangle_f = \left\langle Q_{*,f}(\delta f_1), Q_{*,f}(\delta f_2) \right\rangle , \tag{12.2}$$

where the inner product on the right side is the standard inner product in $\mathbb{L}^2$.

To write the metric in Definition 2 in full detail, we use the expression for $Q_{*,f}(\delta f)$ given in Eq. (12.1):

$$\begin{aligned}\langle\langle \delta f_1, \delta f_2 \rangle\rangle_f &= \langle \frac{1}{2|n|^{\frac{3}{2}}}(n \cdot n_{\delta f_1})f + \sqrt{|n|}\, \delta f_1, \frac{1}{2|n|^{\frac{3}{2}}}(n \cdot n_{\delta f_2})f + \sqrt{|n|}\, \delta f_2 \rangle \\ &= \left\langle \frac{1}{4|n|^3}(n \cdot n_{\delta f_1})f, (n \cdot n_{\delta f_2})f \right\rangle + \left\langle \frac{1}{2|n|}[(n \cdot n_{\delta f_2})\delta f_1 + (n \cdot n_{\delta f_1})\delta f_2], f \right\rangle + \langle |n| \delta f_1, \delta f_2 \rangle .\end{aligned}$$

As stated, because of the structure of Q, the action of Γ on $\mathcal{F}$ is by isometries under this metric. That is, for any surface $f \in \mathcal{F}$, a $\gamma \in \Gamma$, and two tangent vectors $\delta f_1, \delta f_2 \in T_f(\mathcal{F})$, we have $\langle\langle \delta f_1 \circ \gamma, \delta f_2 \circ \gamma \rangle\rangle_{f\circ\gamma} = \langle\langle \delta f_1, \delta f_2 \rangle\rangle_f$.

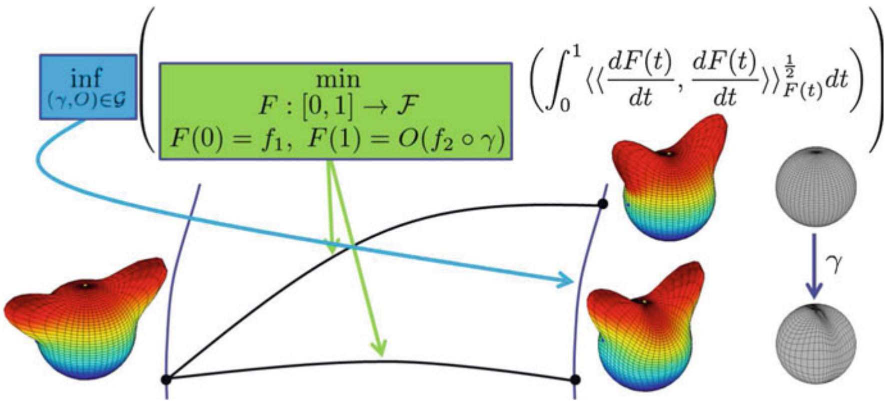
It is frequently important for shape analysis to be invariant to changes in the positions, orientations, and sizes of the surfaces being compared. In other words, the metric should be invariant to the action of the (direct) similarity group of translations, rotations, and scalings. One way to achieve such invariance is by normalizing, i.e. by picking a distinguished representative of each equivalence class under the group action, and then computing distances only between these distinguished elements. Thus, translations may be removed by centering, $f_{\text{centered}}(s) = f(s) - \frac{\int_D f(s)|n(s)|ds}{\int_D |n(s)|ds}$, while scalings may be removed by rescaling all surfaces to have unit area, $f_{\text{scaled}}(s) = \frac{f(s)}{\sqrt{\int_D |n(s)|ds}}$. Slightly abusing notation, we denote the space of such normalized surfaces also by $\mathcal{F}$. Then, $\mathcal{F}$ forms the "preshape space" in our analysis. Paired with the Riemannian metric $\langle\langle \cdot, \cdot \rangle\rangle$, it becomes a Riemannian manifold.

Another, often equivalent, way to achieve invariance is by constructing the quotient space under the group action, inducing a metric on the quotient space from the covering space and then computing distances between points in the quotient space (i.e. between equivalence classes). This is how we will deal with the actions of the rotation group SO(3) and Γ. The rotation group acts on $\mathcal{F}$ according to $(O, f) = Of$, for $O \in \text{SO}(3)$ and $f \in \mathcal{F}$. It is easy to check that this action is by isometries. We have already seen that the action of Γ is by isometries too. Furthermore, the actions of Γ and SO(3) on $\mathcal{F}$ commute, allowing us to define an action of the product group. The equivalence class or orbit of a surface f is given by $[f] = \{O(f \circ \gamma) | O \in \text{SO}(3), \gamma \in \Gamma\}$, and the set of all $[f]$ is by definition the quotient space $\mathcal{S} = \mathcal{F}/\Gamma = \{[f] | f \in \mathcal{F}\}$. This quotient space is called the "shape space."

The next step is to define geodesic paths and distances in the shape space $\mathcal{S}$. This is accomplished using the following joint optimization problem. Let f_1 and f_2 denote two surfaces and let $\langle\langle \cdot, \cdot \rangle\rangle$ be the Riemannian metric on $\mathcal{F}$. Then, the geodesic distance between shapes of f_1 and f_2 is given by

$$d([f_1],[f_2]) = \min_{(O,\gamma)\in \text{SO}(3)\times\Gamma} \left(\min_{\substack{F:[0,1]\to\mathcal{F} \\ F(0)=f_1,\ F(1)=O(f_2\circ\gamma)}} \left(\int_0^1 \langle\langle \frac{dF(t)}{dt}, \frac{dF(t)}{dt} \rangle\rangle^{(1/2)}\, dt \right) \right). \tag{12.3}$$

In this equation, $F(t)$ is a path in $\mathcal{F}$ indexed by t. The quantity $L(F) = \int_0^1 \langle\langle \frac{dF(t)}{dt}, \frac{dF(t)}{dt} \rangle\rangle^{(1/2)}\, dt$ provides the length of the path F. The minimization inside the brackets represents the problem of finding a geodesic path between the surfaces f_1 and $O(f_2 \circ \gamma)$, where O and γ stand for an arbitrary rotation and reparameterization of f_2, respectively. This is computed using a path-straightening technique. We omit the details of this method here and refer the interested reader to [34] for



$\inf_{(\gamma,O)\in\mathcal{G}}$
$\min_{F:[0,1]\to\mathcal{F},\ F(0)=f_1,\ F(1)=O(f_2\circ\gamma)}$
$\left(\int_0^1 \langle\langle \frac{dF(t)}{dt}, \frac{dF(t)}{dt}\rangle\rangle^{\frac{1}{2}}_{F(t)} dt\right)$
γ

Fig. 12.2 Pictorial description of the process of computing geodesics in the shape space of surfaces

details. The minimization outside the brackets seeks the optimal rotation and reparameterization of the second surface so as to best match it with the first surface. This optimization is performed iteratively using Procrustes analysis to solve for optimal rotations and a gradient descent algorithm [32, 33] to solve for the optimal reparameterization. A few details of this registration procedure are presented in the next section. In simple words, the outside optimization solves the registration problem while the inside optimization solves for both an optimal deformation (geodesic path, F^*) and a formal geodesic distance between shapes. Figure 12.2 displays the joint optimization problem defined in Eq. (12.3). Figure 12.3 displays several examples of geodesic comparisons for complex surfaces with many articulated parts. We note the clear benefit of finding optimal reparameterizations during the geodesic computation. The geodesics in the shape space are much more natural than those in the pre-shape space. Furthermore, the decrease in the distance due to optimization over the reparameterization group is significant in all of the presented examples. These properties will additionally lead to improved shape statistics and more parsimonious shape models.

12.2.2 Application to Image Registration

The problem of image registration is common across multiple application areas. Given a set of observed images, the goal is to establish point correspondence across the domains of these images. Although the registration problem has been studied for almost two decades, there continue to be some fundamental limitations in the popular solutions that make them suboptimal, difficult to evaluate, and limited in scope. To explain these limitations, let f_1 and f_2 represent two $\mathbb{R}^n$-valued images. A pairwise registration between these images is defined as finding a diffeomorphic

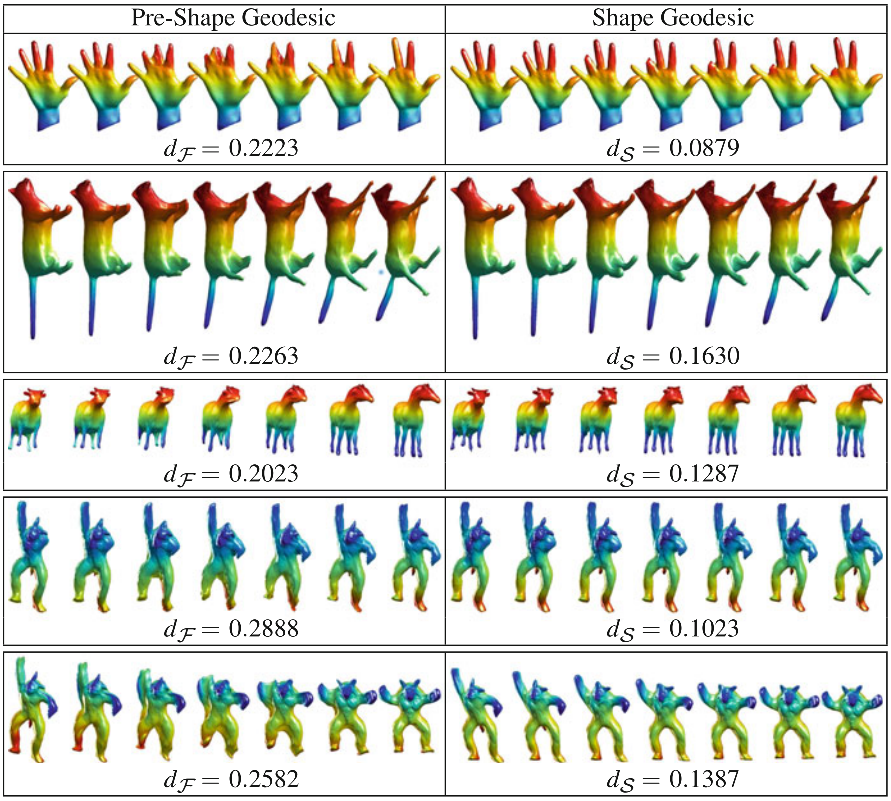

Pre-Shape Geodesic
Shape Geodesic
$d_{\mathcal{F}} = 0.2223$
$d_{\mathcal{S}} = 0.0879$
$d_{\mathcal{F}} = 0.2263$
$d_{\mathcal{S}} = 0.1630$
$d_{\mathcal{F}} = 0.2023$
$d_{\mathcal{S}} = 0.1287$
$d_{\mathcal{F}} = 0.2888$
$d_{\mathcal{S}} = 0.1023$
$d_{\mathcal{F}} = 0.2582$
$d_{\mathcal{S}} = 0.1387$

Fig. 12.3 Comparison of geodesics computed under $\langle\langle\cdot,\cdot\rangle\rangle$ in the pre-shape space and shape space

mapping γ of the image domain D to itself, such that pixels $f_1(s)$ and $f_2(\gamma(s))$ are optimally matched to each other for all $s \in D$. To develop an algorithm for registration one needs an objective function for formalizing the notion of optimality. A common type of objective function used in registration frameworks is $\mathcal{L}(f_1, f_2 \circ \gamma) = \int_D |f_1(s) - f_2(\gamma(s))|^2 ds + \lambda R(\gamma)$, where R is a regularization penalty on γ and λ is a positive constant. This objective function has many shortcomings including lack of symmetry (registration of f_1 to f_2 is different from registration of f_2 to f_1). Next, we outline several important properties for image registration and show that the framework for shape analysis of surfaces can be applied to the image registration problem. Furthermore, this framework overcomes many of the shortcomings of current registration methods.

In the case of images we extend the definition of $\mathcal{F}$ to all images on some domain D that take value in $\mathbb{R}^n$, $n > 1$. (When dealing with grayscale images, we add the gradient of the image f to create an image in $\mathbb{R}^3$.) We let Γ denote the reparameterization group, also called the image-warping group. Let $\mathcal{L}(f_1, (f_2, \gamma))$ denote the objective function for matching f_1 and f_2 by optimizing over Γ (here γ is assumed to be applied to f_2 resulting in $(f_2, \gamma) \in \mathcal{F}$). Then, the desired properties of $\mathcal{L}$ are (for any $f_1, f_2 \in \mathcal{F}$ and $\gamma \in \Gamma$): (1) symmetry; (2) positive definiteness; (3) lack of bias: if f_1, f_2 are constant functions then $\mathcal{L}(f_1, f_2) = \mathcal{L}(f_1, (f_2, \gamma))$; (4) invariance to identical warping: $\mathcal{L}(f_1, f_2) = \mathcal{L}((f_1, \gamma), (f_2, \gamma))$; (5) triangle inequality; and (6) Γ is a group with composition as the group action. These properties have been discussed previously in [9, 45, 49].

Next, we define a representation of images, similar to the SRF, which allows invariance to Γ under the $\mathbb{L}^2$ metric [48, 49].

Definition 3. Define the mapping $\tilde{Q} : \mathcal{F} \to \mathbb{L}^2(D, \mathbb{R}^n), n > 1$, as $\tilde{Q}(f)(s) = \tilde{q}(s) = \sqrt{|a(s)|}f(s)$, where $|a(s)| = |f_x(s) \wedge f_y(s)|$, where $\wedge$ is the wedge product.

We will refer to this representation as the extended SRF. The extended SRF is simply a generalization of the SRF used for surfaces to functions taking values in $\mathbb{R}^n$, $n > 1$. Assuming the original set of images to be smooth, the set of all extended SRFs is a subset of $\mathbb{L}^2$. One can show that the action of Γ on $\mathbb{L}^2$ is exactly the same, *mutatis mutandis*, as that in the previous section. This implies that this group acts on $\mathbb{L}^2$ by isometries, satisfying property (4). This leads to the following definition of the objective function.

Definition 4. Define an objective function for registration of any two images f_1 and f_2, represented by their extended SRFs $\tilde{q}_1$ and $\tilde{q}_2$, as $\mathcal{L}(f_1, (f_2, \gamma)) = \|\tilde{q}_1 - (\tilde{q}_2, \gamma)\|$.

The registration is then achieved by minimizing this objective function: $\gamma^* = \text{arginf}_{\gamma \in \Gamma} \mathcal{L}(f_1, (f_2, \gamma)) = \text{arginf}_{\gamma \in \Gamma} \|\tilde{q}_1 - (\tilde{q}_2, \gamma)\|$. The objective function $\mathcal{L}$ given in Definition 4 satisfies all of the properties listed earlier. The $\mathbb{L}^2$ norm between extended SRFs of images becomes a proper measure of registration between images since it remains the same if the pixel correspondence is unchanged. This leads to a quantity that serves as both the registration objective function and an extrinsic distance between registered images ($\|\tilde{q}_1 - (\tilde{q}_2, \gamma^*)\|$). It is important to note that the proposed objective function has only one term (similarity term) and the regularity term appears to be missing. However, the similarity term has built-in regularity, since it includes the determinant of the Jacobian of the transformation γ. Additional regularity can also be introduced as in the LDDMM framework [4].

Gradient Descent Method for Optimization Over Γ The optimization problem over Γ is a major component of this registration framework and we use a gradient descent method to solve it. Since Γ is a group, we use the gradient to solve for the incremental warping γ, on top of the previous cumulative warping γ_o, as follows. First, define a cost function with respect to γ as $E(\gamma) = \|\tilde{q}_1 - \phi_{\tilde{q}_2^o}(\gamma)\|^2$, where $\phi_{\tilde{q}} : \Gamma \to [\tilde{q}]$ is defined to be $\phi_{\tilde{q}}(\gamma) = (\tilde{q}, \gamma)$ and $\tilde{q}_2^o = (\tilde{q}_2, \gamma_o)$ with γ_o being the current registration function. Given a set of orthonormal basis elements, say $\mathbf{B}$, of $T_{\gamma_{id}}(\Gamma)$, the gradient at γ_{id} takes the form $\nabla E(\gamma_{id}) = \sum_{b \in \mathbf{B}} \langle \tilde{q}_1 - \phi_{\tilde{q}_2^o}(\gamma_{id}), \phi_{\tilde{q}_2^o,*}(b)$. In this equation, $\phi_{\tilde{q}_2^o,*}(b)$ denotes the differential of $\phi_{\tilde{q}}$ at γ_{id} in the direction of b and brackets denote the $\mathbb{L}^2$ inner product. We omit the derivation of $\phi_{\tilde{q},*}$ and note that it is the same as presented in [33]. We note that this gradient-based solution can also be employed in the search for optimal γ in the case of parameterized surfaces.

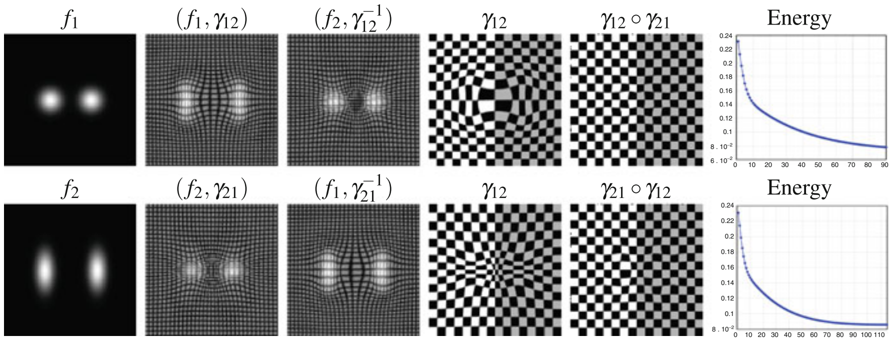


Fig. 12.4 Registration of two synthetic grayscale images. $\gamma_{12} = \mathrm{argmin}_{\gamma\in\Gamma} \mathcal{L}(f_2,(f_1,\gamma))$ and $\gamma_{21} = \mathrm{argmin}_{\gamma\in\Gamma} \mathcal{L}(f_1,(f_2,\gamma))$. $\|q_1-q_2\| = 0.2312$, $\|\tilde{q}_1-(\tilde{q}_2,\gamma_{21})\| = 0.0728$, and $\|(\tilde{q}_1,\gamma_{12})-\tilde{q}_2\| = 0.0859$. (Courtesy of Xie et al. [49])

There, one has to perform an additional search over the rotation group SO(3), and then solve for the geodesic using path straightening.

In Fig. 12.4 we display an example of registering two smooth grayscale images using this framework. The correspondence appears to be good, and more importantly, the resulting distance between the registered images is approximately symmetric and the registration is inverse consistent. That is, the compositions $\gamma_{21}\circ\gamma_{12}$ and $\gamma_{12}\circ\gamma_{21}$ result in the identity mapping. This provides empirical support for the claim that this method satisfies all properties outlined earlier.

12.2.3 Elastic Riemannian Metric

While the SRF representation has its advantages, it has two important drawbacks. First, there is no intuitive interpretation or justification for the use of the metric, unlike the elastic Riemannian metric used in the case of curves; rather, it was solely devised for the convenience of being able to compare the shapes of parameterized surfaces using the $\mathbb{L}^2$ metric in the SRF space. Second, the associated metric is not translation invariant; translating two shapes equally does not preserve the distance between them, which can cause some issues during statistical analysis. To overcome these two drawbacks, while preserving the advantages of the SRF representation, a different representation of surfaces was introduced in [25]: the square-root normal field, or SRNF, the details of which we present next.

We first recall the elastic metric in the case of curves [42]. Let $\beta : D \to \mathbb{R}^2$ be a curve, where D temporarily denotes either the circle S^1 or the interval. Let $r\,ds = |\dot{\beta}|\,ds$ be the induced metric measure, where ds is Lebesgue measure on D, and let $\tilde{n}$ be the normalized normal vector to the curve. We can represent the curve uniquely up to translations by the pair $(r,\tilde{n})$. Then, the family of elastic metrics takes the form

$$\langle\langle\langle(\delta r_1, \delta\tilde{n}_1), (\delta r_2, \delta\tilde{n}_2)\rangle\rangle\rangle_{(r,\tilde{n})} = 2\lambda \int_D \frac{\delta r_1(s)\delta r_2(s)}{r(s)} ds + c \int_D \langle \delta\tilde{n}_1(s), \delta\tilde{n}_2(s)\rangle \, r(s) \, ds \,, \tag{12.4}$$

where $(\delta r_i, \delta\tilde{n}_i)$, $i \in \{1, 2\}$ are tangent vectors at the curve $(r, \tilde{n})$, and $\lambda, c \in \mathbb{R}_+$.

Although this metric was defined for curves, it can immediately be applied to surfaces, with $r = |n|$ denoting the induced metric measure and $\tilde{n} = \frac{n}{r}$ denoting the unit normal vector. We thereby define a new Riemannian metric on the space of parameterized surfaces known, for reasons that will become clear, as the "partial elastic metric." To understand this metric, consider that a small change in f on an infinitesimal patch in the surface around the point $f(s)$ can be decomposed into a change in the normal direction of the patch ("bending") and a change in its geometry ("stretching"). Since a change in $\tilde{n}(s)$ corresponds to a change in normal direction and a change in r corresponds to a change in the area of the patch, we see that this metric has an interpretation directly analogous to its interpretation in the case of curves.

However, when D is two-dimensional, the change in area does not completely characterize a change in the geometry of a patch. A change in geometry can be decomposed into a change in area and an area-preserving change of shape [10, 14]. The partial elastic metric measures the first type of change, but does not measure changes in f that change the shape of a patch while preserving its area and normal direction. This limitation came about because the correspondence that we used between the quantity r in the case of curves and in the case of surfaces was in fact incomplete. We interpreted r for curves as the induced metric measure, but r^2 is also the full induced metric; in one dimension there is no difference. This suggests that instead of using $(r, \tilde{n})$ to represent a surface, we should use $(g, \tilde{n})$, where $g = f^*h$ is the full pullback metric (with h the metric on $\mathbb{R}^3$). The metric g contains more information than r, because r is just $|g|^{\frac{1}{2}}$, where $|\cdot|$ indicates the determinant.

This in turn suggests that Eq. (12.4) is merely a special case of a more general elastic metric for surfaces, and indeed this is the case. This "full elastic metric" is defined, up to an overall scale, by

$$\begin{aligned}&\langle(\delta g_1, \delta\tilde{n}_1), (\delta g_2, \delta\tilde{n}_2)\rangle_{(g,\tilde{n})} \\ &= \int_D \left[\operatorname{tr}(g^{-1}\delta g_1 g^{-1}\delta g_2) + \frac{\lambda}{2}\operatorname{tr}(g^{-1}\delta g_1)\operatorname{tr}(g^{-1}\delta g_2) + c\,\langle\delta\tilde{n}_1, \delta\tilde{n}_2\rangle\right]|g|^{\frac{1}{2}} ds,\end{aligned} \tag{12.5}$$

where, for positivity, $\lambda > \frac{-2}{3}$ and $c \in \mathbb{R}_+$.[1]

To see that the partial metric is indeed a special case of this full metric, note that the term multiplied by λ can be written as

[1]It is interesting to note that the first two terms in the full elastic metric form the unique family of ultralocal metrics on the space of Riemannian metrics on which diffeomorphisms act by isometries [14]. The $\lambda = 0$ case of this metric on Riemannian metrics has been studied in e.g., [10, 16, 18].

$$\int_D ds|g|^{\frac{1}{2}}\mathrm{tr}(g^{-1}\delta g_1)\mathrm{tr}(g^{-1}\delta g_2) = 4\int_D ds|g|^{-\frac{1}{2}}\delta_1(|g|^{\frac{1}{2}})\delta_2(|g|^{\frac{1}{2}}). \tag{12.6}$$

Since $r = |g|^{\frac{1}{2}}$, this is the same as the first term in Eq. (12.4), while the last term is the same in each case. Unlike the partial metric, however, the first two terms in Eq. (12.5) measure not only changes in local area but also any "stretching" of the surface, i.e., changes in both the area and the shape of local patches. The third term continues to measure changes in the normal direction, that is, "bending." The intuitive interpretation of the metric thus remains unchanged, but now all types of change in f are measured. Indeed, the map from $\mathcal{F}$ to the $(g, \tilde{n})$ representation is bijective, up to translations (although it is not surjective) [1].

Having defined this metric for surfaces, i.e., codimension-1 embedded submanifolds of $\mathbb{R}^3$, it is easy to see that it applies in fact to codimension-1 embedded submanifolds in any number of dimensions, and, with a simple generalization of the third term, to embedded submanifolds of any codimension in any number of dimensions. In particular, when D is one-dimensional, and thus g is a scalar, the first two terms become the same, so that the full metric becomes identical to the partial metric (which is thus no longer partial), which is, in turn, just the elastic metric for curves. Equation (12.5) is thus the most general form of the elastic metric and deserves further study. However, we defer further analysis of its properties to another place. We now focus on the partial metric.

12.2.4 Square-Root Normal Field Representation of Surfaces

An important and remarkable property of the partial elastic metric in Eq. (12.4) is that, despite appearances, for a particular choice of the ratio of λ and c, it is Euclidean: i.e., we can find a transformation of the representation such that this metric takes on the simple $\mathbb{L}^2$ form. This is strictly analogous to the case of the elastic metric for curves, which gave rise to the square-root velocity function (SRVF) [42]. Just as in this case, and in the case of the SRF, the existence of a representation that transforms the metric to $\mathbb{L}^2$ form means that parts of the shape analysis framework can be dramatically simplified.

This new, convenient representation of surfaces is called the square-root normal field (SRNF) and is defined as follows [25]:

Definition 5. Define the mapping $H : \mathcal{F} \to \mathbb{L}^2$ as $H(f)(s) = h(s) = \sqrt{r(s)}\tilde{n}(s) = \frac{n(s)}{\sqrt{r(s)}} = \frac{n(s)}{\sqrt{|n(s)|}}$.

This is strictly analogous to the SRVF defined for curves, except that there the tangent vector is used instead of the normal vector/form. Since $|h|^2 = r$, the $\mathbb{L}^2$ norm of h is just the area of the surface (again, *cf.* the case of curves, where the $\mathbb{L}^2$ norm of the SRVF gives the length of the curve). Thus, just as in the case of SRFs, the space of SRNFs is also a subset of $\mathbb{L}^2(D, \mathbb{R}^3)$ or simply $\mathbb{L}^2$.

We now show that the $\mathbb{L}^2$ metric in the SRNF space is the same as that in Eq. (12.4). The derivative map between the tangent spaces at $(r, \tilde{n})$ and $h \in \mathbb{L}^2$ is given by

$$\delta h(s) = \frac{1}{2\sqrt{r(s)}} \tilde{n}(s) \delta r(s) + \sqrt{r(s)} \delta \tilde{n}(s). \tag{12.7}$$

Taking the $\mathbb{L}^2$ inner product between two such vectors, we obtain

$$\langle \delta h_1, \delta h_2 \rangle_h = \frac{1}{4} \int_D \frac{\delta r_1(s) \delta r_2(s)}{r(s)} ds + \int_D r(s) \langle \delta \tilde{n}_1(s), \delta \tilde{n}_2(s) \rangle \, ds \tag{12.8}$$

since $\langle \tilde{n}(s), \delta \tilde{n}_i(s) \rangle = 0$. This is just the partial elastic metric for $\lambda = 1/8$ and $c = 1$. We thus find that if $c/\lambda = 8$, the SRNF representation acts as "Euclidean coordinates" for the partial elastic metric, bringing it to $\mathbb{L}^2$ form.

As in the case of SRFs, we must also remove all shape-preserving transformations in order to generate shape comparisons. A major advantage of the SRNF is that it (and consequently the partial and full elastic metrics) is automatically invariant to translations, simply because it depends only on derivatives of f. As previously, we can scale all surfaces to have unit area. It is easily checked that rotating a surface $f \mapsto Of$ sends $h \mapsto Oh$ and reparameterizing a surface $f \mapsto f \circ \gamma$ sends $h \mapsto (h, \gamma) = (q \circ \gamma)\sqrt{J_\gamma}$. It is also easy to verify that just as in the SRF case, the action of Γ on the space of surfaces under the partial elastic metric is by isometries. This allows the use of this metric for parameterization-invariant shape analysis.

In order to generate comparisons of shapes using geodesic paths and distances, we take a similar approach to that presented in Sect. 12.2.1 for the SRF representation. Because the action of Γ on the SRNF space is the same as that on the SRF space, the gradient descent algorithm to compute optimal reparameterizations remains unchanged. To compute geodesics, we also use the path-straightening approach [41]. In Fig. 12.5, we again provide a comparison of geodesics computed in the pre-shape space and shape space under the partial elastic metric. We use the same examples as in Fig. 12.3. As previously, the geodesics in the shape space have much lower distances due to the additional optimization over Γ. They are also much more natural due to the improved correspondence of geometric features across surfaces. When comparing the shape space results in this figure to those in Fig. 12.3, we notice that the partial elastic metric provides better registrations than the pullback metric from SRF space. This is expected due to the nice stretching and bending interpretation of this metric.

12.3 Shape Statistics

In this section, we present tools for computing two fundamental shape statistics, the mean, and the covariance. We then utilize these quantities to estimate a generative

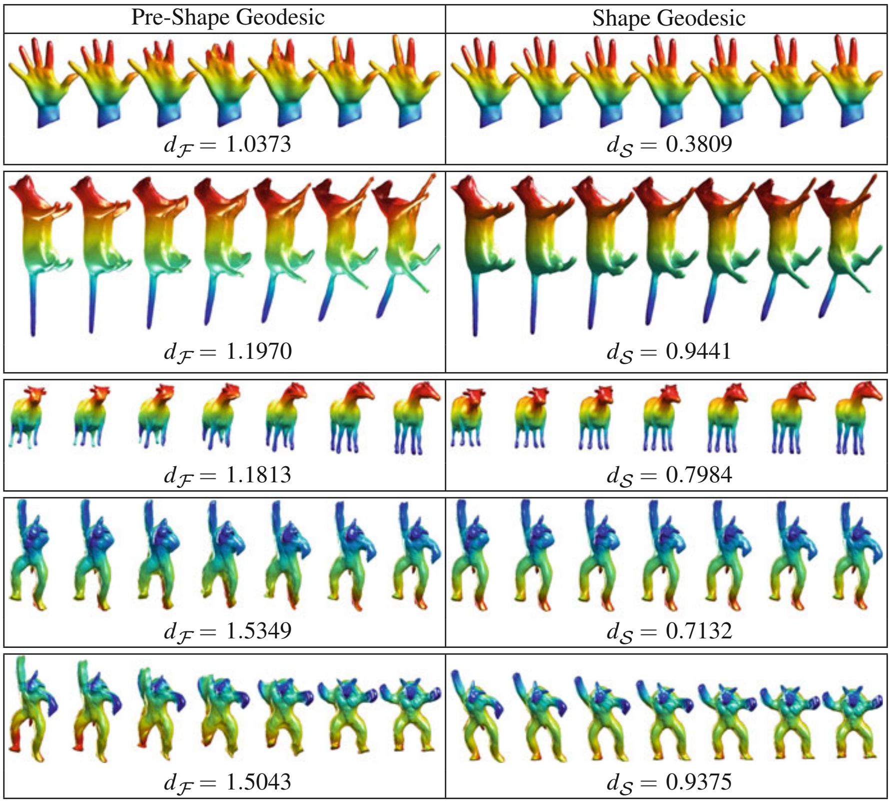

Pre-Shape Geodesic
Shape Geodesic
$d_{\mathcal{F}} = 1.0373$
$d_{\mathcal{S}} = 0.3809$
$d_{\mathcal{F}} = 1.1970$
$d_{\mathcal{S}} = 0.9441$
$d_{\mathcal{F}} = 1.1813$
$d_{\mathcal{S}} = 0.7984$
$d_{\mathcal{F}} = 1.5349$
$d_{\mathcal{S}} = 0.7132$
$d_{\mathcal{F}} = 1.5043$
$d_{\mathcal{S}} = 0.9375$

Fig. 12.5 Comparison of geodesics computed under $\langle\langle\langle\cdot,\cdot\rangle\rangle\rangle$ in the pre-shape space and shape space

Gaussian model and draw random samples from this model. These methods have been previously presented for SRFs and SRNFs in [31, 36].

First, we define an intrinsic shape mean, called the Karcher mean. Let $\{f_1, f_2, \ldots, f_n\} \in \mathcal{F}$ denote a sample of surfaces. Also, let F_i^* denote a geodesic path (in the shape space) between the shape of some surface f and the shape of the ith surface in the sample, f_i. Then, the sample Karcher mean shape is given by $[\bar{f}] = \operatorname{argmin}_{[f]\in\mathcal{S}} \sum_{i=1}^{n} L(F_i^*)^2$. A gradient-based approach for finding the Karcher mean is given in, for example, [15] and is omitted here for brevity. Note that the resulting Karcher mean shape is an entire equivalence class of surfaces.

Once the sample Karcher mean has been computed, the evaluation of the Karcher covariance is performed as follows. First, find the shooting vectors from the estimated Karcher mean $\bar{f} \in [\bar{f}]$ to each of the surface shapes in the sample, $\nu_i = \frac{dF_i^*}{dt}|_{t=0}$, where $i = 1, 2, \ldots, n$ and F^* denotes a geodesic path in $\mathcal{S}$. This is accomplished using the inverse exponential map, which is used to map points from the representation space to the tangent space. Then, perform principal component analysis by applying the Gram–Schmidt procedure (under the chosen metric $\langle\langle\cdot,\cdot\rangle\rangle$) to generate an orthonormal basis $\{B_j|j = 1,\ldots,k\}$, $k \leq n$, of the observed $\{v_i\}$. Project each of the vectors v_i onto this orthonormal basis using $v_i \approx \sum_{j=1}^{k} c_{i,j}B_j$, where $c_{i,j} = \langle\langle v_i, B_j\rangle\rangle$. Now, each original shape can simply be represented using the coefficient vector $c_i = \{c_{i,j}\}$. Then, the sample covariance matrix can be computed in the coefficient space as $K = (1/(n-1))\sum_{i=1}^{n} c_i c_i^T \in \mathbb{R}^{k\times k}$. One can use the singular value decomposition of K to determine the principal directions of variation in the given data. For example, if $u \in \mathbb{R}^k$ corresponds to a principal singular vector of K, then the corresponding tangent vector in $T_{\bar{f}}(\mathcal{F})$ is given by $v = \sum_{j=1}^{k} u_j B_j$. One can then map this vector to a surface f using the exponential map, which is used to map points from the tangent space to the representation space. The exponential map must be computed under one of the nonstandard metrics introduced earlier, which is not a simple task. This can be accomplished using a tool called parallel transport, which was derived for the SRNF representation of surfaces by Xie et al. [50]. For brevity, we do not provide details here but rather refer the interested reader to that paper. We also note that when computing the following results we approximated the exponential map using a linear mapping.

Given the mean and covariance, we can impose a Gaussian distribution in the tangent space at the mean shape. This will provide a model, which can be used to generate random shapes. A random tangent vector $v \in T_{\bar{f}}(\mathcal{F})$ can be sampled from the Gaussian distribution using $v = \sum_{j=1}^{k} z_j \sqrt{S_{jj}} u_j B_j$, where $z_j \overset{iid}{\sim} N(0,1)$, S_{jj} is the variance of the jth principal component, u_j is the corresponding principal singular vector, and B_j is a basis element as defined previously. One can then map this element of the tangent space to a surface shape using the exponential map to obtain a random shape from the Gaussian distribution.

We present an example of computing shape statistics for a toy data set in Figs. 12.6 and 12.7. The data in this example was simulated in such a way that one of the peaks on each surface was already matched perfectly while the position of the second peak was slightly perturbed. All surfaces in this data set are displayed in the top panel of Fig. 12.6. We computed three averages in this example: simple average in the pre-shape space (bottom left of Fig. 12.6), Karcher mean under the SRF pullback metric $\langle\langle\cdot,\cdot\rangle\rangle$ (bottom center of Fig. 12.6), and Karcher mean under the partial elastic metric $\langle\langle\langle\cdot,\cdot\rangle\rangle\rangle$ (bottom right of Fig. 12.6). The pre-shape mean is not a very good summary of the given data. One of the peaks is sharp while the other is averaged out due to misalignment. The Karcher mean under the SRF pullback metric is much better, although it also shows slight averaging out of the second peak. The best representative shape is given by the Karcher mean computed under the partial elastic metric where the two peaks are sharp as in the original data. In Fig. 12.7, we display the two principal directions of variation in the given data computed by those three methods. The result computed in the pre-shape space does not reflect the true variability in the given data. In fact, as one goes in the positive second principal direction, the resulting shapes have three peaks. This result is again

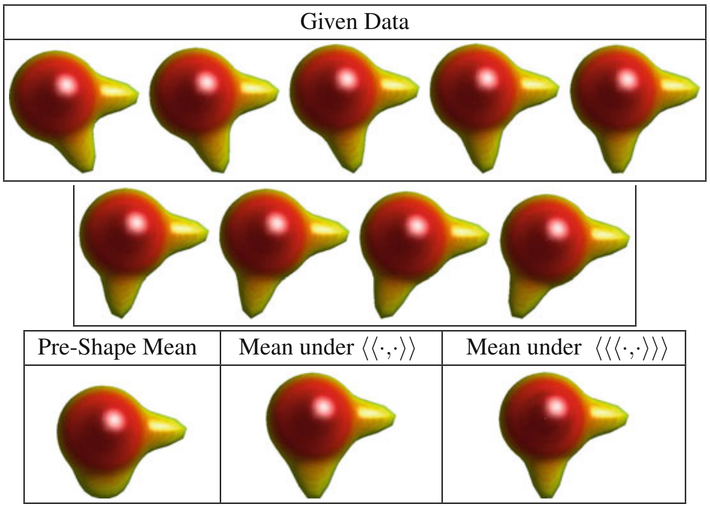

Given Data
Pre-Shape Mean
Mean under $\langle\langle\cdot,\cdot\rangle\rangle$
Mean under $\langle\langle\langle\cdot,\cdot\rangle\rangle\rangle$

Fig. 12.6 *Top*: given data. *Bottom*: shape means computed (1) in the pre-shape space, (2) under the SRF pullback metric, and (3) under the partial elastic metric

improved under the SRF pullback metric. But, there is still some misalignment, which can be seen in the second principal direction where a wide peak evolves into a thin peak. The best result is observed in the case of the partial elastic metric. Here, all of the variability is contained in the first principal direction of variation where the peak naturally moves without any distortion. Based on the given data, this is the most intuitive summary of variability.

In Fig. 12.8, we show two random samples from the Gaussian distribution defined (1) in the pre-shape space, (2) in the shape space under the SRF pullback metric, and (3) in the shape space under the partial elastic metric. The two random samples in the pre-shape space do not resemble the given data as they both have three peaks. While method (2) produced random samples with two peaks, one can see that in both cases one of the peaks has a strange shape (either too thin or too wide). Method (3) produces the best random samples, which clearly resemble the structure in the given data.

12.4 Classification of Attention Deficit Hyperactivity Disorder

In this last section, we present an application of the described methods to medical imaging: the diagnosis of attention deficit hyperactivity disorder (ADHD) using structural MRI. The results presented here have been previously published in [31, 32, 47]. The surfaces of six left and right brain structures were extracted from

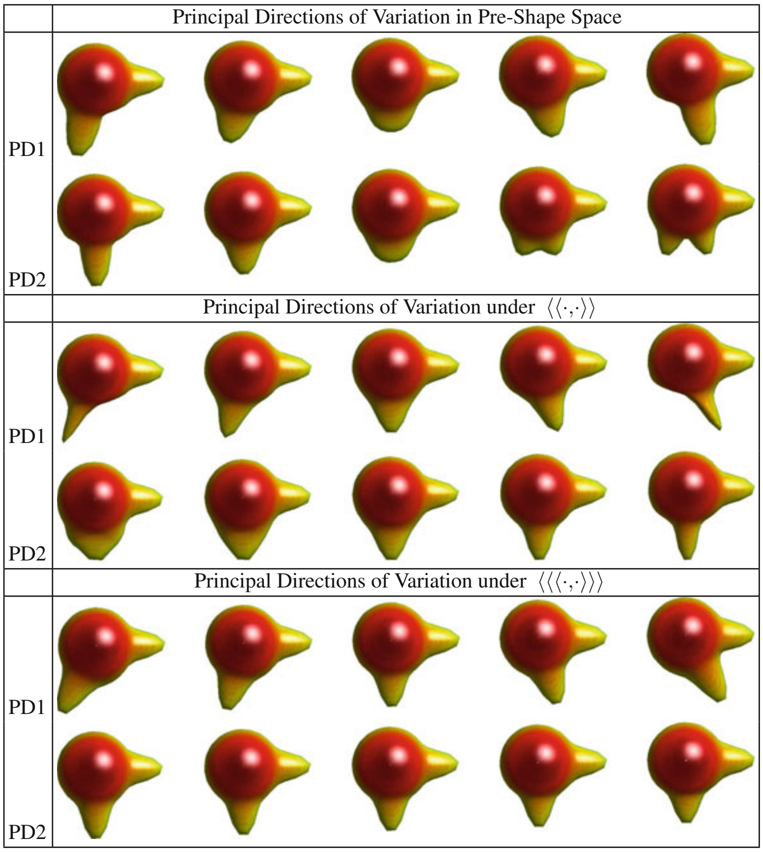

Principal Directions of Variation in Pre-Shape Space
PD1
PD2
Principal Directions of Variation under $\langle\langle\cdot,\cdot\rangle\rangle$
PD1
PD2
Principal Directions of Variation under $\langle\langle\langle\cdot,\cdot\rangle\rangle\rangle$
PD1
PD2

Fig. 12.7 The two principal directions of variation (from -2σ to $+2\sigma$) for (1) pre-shape mean and covariance without optimization over Γ, (2) pullback metric under SRF representation, and (3) partial elastic metric

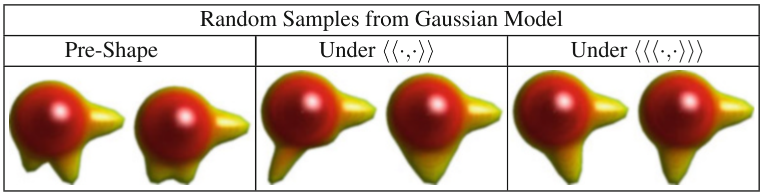

Random Samples from Gaussian Model
Pre-Shape
Under $\langle\langle\cdot,\cdot\rangle\rangle$
Under $\langle\langle\langle\cdot,\cdot\rangle\rangle\rangle$

Fig. 12.8 Random samples from Gaussian model computed (1) in the pre-shape space, (2) under the pullback metric under SRF representation, and (3) under the partial elastic metric

T1-weighted brain magnetic resonance images of young adults aged between 18 and 21. These subjects were recruited from the Detroit Fetal Alcohol and Drug Exposure Cohort [7, 23, 24]. Among the 34 subjects studied, 19 were diagnosed

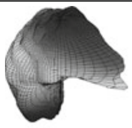
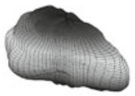
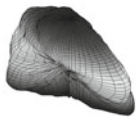
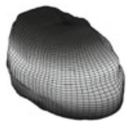
Caudate	Pallidus	Putamen	Thalamus
			

Fig. 12.9 Left subcortical structures used for ADHD classification. (Courtesy of Kurtek et al. [32])

Table 12.1 Classification accuracy (%) for five different techniques

Method structure (%)	SRNF Gauss [47]	SRF Gauss [31]	SRF NN [32]	Harmonic	ICP [2]	SPHARM-PDM [44]
L. Caudate	67.7	–	41.2	64.7	32.4	61.8
L. Pallidus	85.3	82.4	76.5	79.4	67.7	44.1
L. Putamen	94.1	88.2	82.4	70.6	61.8	50.0
L. Thalamus	67.7	–	58.8	67.7	35.5	52.9
R. Caudate	55.9	–	50.0	44.1	50.0	70.6
R. Pallidus	76.5	67.6	61.8	67.7	55.9	52.9
R. Putamen	67.7	82.4	67.7	55.9	47.2	55.9
R. Thalamus	67.7	–	58.8	52.9	64.7	64.7

with ADHD and the remaining 15 were controls (non-ADHD). Some examples of the left structures used for classification are displayed in Fig. 12.9. In order to distinguish between ADHD and controls, we utilize several methods for comparison: SRNF Gaussian, SRF Gaussian, SRF distance nearest neighbor (NN), iterative closest point (ICP) algorithm, $\mathbb{L}^2$ distance between fixed surface parameterizations (Harmonic), and SPHARM-PDM. The classification performance for all methods is reported in Table 12.1. The results suggest that the Riemannian approaches based on parameterization-invariant metrics perform best in this setting due to improved matching across surfaces. Furthermore, the use of Gaussian probability models outperforms standard distance-based classification.

12.5 Summary

This chapter describes two Riemannian frameworks for statistical shape analysis of 3D objects. The important feature of these methods is that they define reparameterization-invariant Riemannian metrics on the space of parameterized surfaces. The first framework develops the metric by pulling back the $\mathbb{L}^2$ metric from the space of square-root function representations of surfaces. The main drawbacks of this method are the lack of interpretability of the resulting Riemannian metric and the lack of invariance to translations. The second framework starts with a full elastic metric on the space of parameterized surfaces. This metric is then restricted, resulting in a partial elastic metric, which becomes the simple $\mathbb{L}^2$ metric under the square-root normal field representation of surfaces. This metric has a nice interpretation in terms of the amount of stretching and bending that is needed to deform one surface into another. We show examples of geodesic paths and distances between complex surfaces for both cases. Given the ability to compute geodesics in the shape space, we define the first two moments, the Karcher mean and covariance, and use them for random sampling from Gaussian-type shape models. Finally, we showcase the applicability of these methods in an ADHD medical study.

Acknowledgements This work was partially supported by a grant from the Simons Foundation (#317865 to Eric Klassen).

References

1. Abe K, Erbacher J (1975) Isometric immersions with the same Gauss map. Math Ann 215(3):197–201
2. Almhdie A, Léger C, Deriche M, Lédée R (2007) 3D registration using a new implementation of the ICP algorithm based on a comprehensive lookup matrix: application to medical imaging. Pattern Recogn Lett 28(12):1523–1533
3. Bauer M, Bruveris M (2011) A new Riemannian setting for surface registration. In: International workshop on mathematical foundations of computational anatomy, pp 182–193
4. Beg M, Miller M, Trouvé A, Younes L (2005) Computing large deformation metric mappings via geodesic flows of diffeomorphisms. Int J Comput Vis 61(2):139–157
5. Bouix S, Pruessner JC, Collins DL, Siddiqi K (2001) Hippocampal shape analysis using medial surfaces. NeuroImage 25:1077–1089
6. Brechbühler C, Gerig G, Kübler O (1995) Parameterization of closed surfaces for 3D shape description. Comput Vis Image Underst 61(2):154–170
7. Burden MJ, Jacobson JL, Westerlund AJ, Lundahl LH, Klorman R, Nelson CA, Avison MJ, Jacobson SW (2010) An event-related potential study of response inhibition in ADHD with and without prenatal alcohol exposure. Alcohol Clin Exp Res 34(4):617–627
8. Cates J, Meyer M, Fletcher P, Whitaker R (2006) Entropy-based particle systems for shape correspondence. In: International workshop on mathematical foundations of computational anatomy, pp 90–99
9. Christensen GE, Johnson H (2001) Consistent image registration. IEEE Trans Med Imaging 20(7):568–582
10. Clarke B (2010) The metric geometry of the manifold of Riemannian metrics over a closed manifold. Calc Var Partial Differ Equ 39(3):533–545
11. Csernansky JG, Wang L, Jones DJ, Rastogi-Cru D, Posener G, Heydebrand JA, Miller JP, Grenander U, Miller MI (2002) Hippocampal deformities in schizophrenia characterized by high dimensional brain mapping. Am J Psychiatry 159(12):1–7
12. Davatzikos C, Vaillant M, Resnick S, Prince JL, Letovsky S, Bryan RN (1996) A computerized method for morphological analysis of the corpus callosum. J Comput Assist Tomogr 20:88–97
13. Davies RH, Twining CJ, Cootes TF, Taylor CJ (2010) Building 3-D statistical shape models by direct optimization. IEEE Trans Med Imaging 29(4):961–981
14. DeWitt BS (1967) Quantum theory of gravity. I. The canonical theory. Phys Rev 160(5):1113–1148
15. Dryden IL, Mardia KV (1998) Statistical shape analysis. Wiley, New York

16. Ebin D (1970) The manifold of Riemannian metrics. In: Symposia in pure mathematics (AMS), vol 15, pp 11–40
17. Gerig G, Styner M, Shenton ME, Lieberman JA (2001) Shape versus size: improved understanding of the morphology of brain structures. In: MICCAI, pp 24–32 (2001)
18. Gil-Medrano O, Michor PW (1991) The Riemannian manifold of all Riemannian metrics. Q J Math 42:183–202
19. Gorczowski K, Styner M, Jeong JY, Marron JS, Piven J, Hazlett HC, Pizer SM, Gerig G (2010) Multi-object analysis of volume, pose, and shape using statistical discrimination. IEEE Trans Pattern Anal Mach Intell 32(4):652–666
20. Grenander U, Miller MI (1998) Computational anatomy: an emerging discipline. Q Appl Math LVI(4):617–694
21. Gu X, Vemuri BC (2004) Matching 3D shapes using 2D conformal representations. In: MICCAI, pp 771–780
22. Gu X, Wang S, Kim J, Zeng Y, Wang Y, Qin H, Samaras D (2007) Ricci flow for 3D shape analysis. In: IEEE international conference on computer vision, pp 1–8
23. Jacobson SW, Jacobson JL, Sokol RJ, Chiodo LM, Corobana R (2004) Maternal age, alcohol abuse history, and quality of parenting as moderators of the effects of prenatal alcohol exposure on 7.5-year intellectual function. Alcohol Clin Exp Res 28:1732–1745
24. Jacobson SW, Jacobson JL, Sokol RJ, Martier SS, Chiodo LM (1996) New evidence for neurobehavioral effects of in utero cocaine exposure. J Pediatr 129:581–590
25. Jermyn IH, Kurtek S, Klassen E, Srivastava A (2012) Elastic shape matching of parameterized surfaces using square root normal fields. In: European conference on computer vision, pp 804–817
26. Joshi S, Miller M, Grenander U (1997) On the geometry and shape of brain sub-manifolds. Pattern Recogn Artif Intell 11:1317–1343
27. Joshi SH, Klassen E, Srivastava A, Jermyn IH (2007) A novel representation for Riemannian analysis of elastic curves in $\mathbb{R}^n$. In: IEEE conference on computer vision and pattern recognition, pp 1–7
28. van Kaick O, Zhang H, Hamarneh G, Cohen-Or D (2010) A survey on shape correspondence. In: Eurographics state-of-the-art report, pp 1–24
29. Kelemen A, Szekely G, Gerig G (1999) Elastic model-based segmentation of 3D neurological data sets. IEEE Trans Med Imaging 18(10):828–839
30. Kilian M, Mitra NJ, Pottmann H (2007) Geometric modeling in shape space. ACM Trans Graph 26(3):64
31. Kurtek S, Klassen E, Ding Z, Avison MJ, Srivastava A (2011) Parameterization-invariant shape statistics and probabilistic classification of anatomical surfaces. In: Information processing in medical imaging, pp 147–158
32. Kurtek S, Klassen E, Ding Z, Jacobson SW, Jacobson JL, Avison MJ, Srivastava A (2011) Parameterization-invariant shape comparisons of anatomical surfaces. IEEE Trans Med Imaging 30(3):849–858
33. Kurtek S, Klassen E, Ding Z, Srivastava A (2010) A novel Riemannian framework for shape analysis of 3D objects. In: IEEE conference on computer vision and pattern recognition, pp 1625–1632
34. Kurtek S, Klassen E, Gore J, Ding Z, Srivastava A (2012) Elastic geodesic paths in shape space of parameterized surfaces. IEEE Trans Pattern Anal Mach Intell 34(9):1717–1730
35. Kurtek S, Klassen E, Gore JC, Ding Z, Srivastava A (2011) Classification of mathematics deficiency using shape and scale analysis of 3D brain structures. In: SPIE medical imaging: image processing, vol 7962
36. Kurtek S, Samir C, Ouchchane L (2014) Statistical shape model for simulation of realistic endometrial tissue. In: International conference on pattern recognition applications and methods
37. Kurtek S, Srivastava A, Klassen E, Ding Z (2012) Statistical modeling of curves using shapes and related features. J Am Stat Assoc 107(499):1152–1165

38. Kurtek S, Srivastava A, Klassen E, Laga H (2013) Landmark-guided elastic shape analysis of spherically-parameterized surfaces. Comput Graph Forum 32(2):429–438
39. Malladi R, Sethian JA, Vemuri BC (1996) A fast level set based algorithm for topology-independent shape modeling. J Math Imaging Vis 6:269–290
40. Mio W, Srivastava A, Joshi SH (2007) On shape of plane elastic curves. Int J Comput Vis 73(3):307–324
41. Samir C, Kurtek S, Srivastava A, Canis M (2014) Elastic shape analysis of cylindrical surfaces for 3D/2D registration in endometrial tissue characterization. IEEE Trans Med Imaging 33(5):1035–1043
42. Srivastava A, Klassen E, Joshi SH, Jermyn IH (2011) Shape analysis of elastic curves in Euclidean spaces. IEEE Trans Pattern Anal Mach Intell 33(7):1415–1428
43. Srivastava A, Turaga P, Kurtek S (2012) On advances in differential-geometric approaches for 2D and 3D shape analyses and activity recognition. Image Vis Comput **30**(6–7):398–416
44. Styner M, Oguz I, Xu S, Brechbühler C, Pantazis D, Levitt J, Shenton ME, Gerig G (2006) Framework for the statistical shape analysis of brain structures using SPHARM-PDM. Insight J 1071:242–250
45. Tagare H, Groisser D, Skrinjar O (2009) Symmetric non-rigid registration: a geometric theory and some numerical techniques. J Math Imaging Vis 34(1):61–88
46. Vaillant M, Glaunes J (2005) Surface matching via currents. In: Information processing in medical imaging, pp 381–392
47. Xie Q, Jermyn IH, Kurtek S, Srivastava A (2014) Numerical inversion of SRNFs for efficient elastic shape analysis of star-shaped objects. In: European conference on computer vision, pp 485–499
48. Xie Q, Kurtek S, Christensen GE, Ding Z, Klassen E, Srivastava A (2012) A novel framework for metric-based image registration. In: Workshop on biomedical image registration, pp 276–285
49. Xie Q, Kurtek S, Klassen E, Christensen GE, Srivastava A (2014) Metric-based pairwise and multiple image registration. In: European conference on computer vision, pp 236–250
50. Xie Q, Kurtek S, Le H, Srivastava A (2013) Parallel transport of deformations in shape space of elastic surfaces. In: IEEE international conference on computer vision, pp. 865–872
51. Younes L (1998) Computable elastic distance between shapes. SIAM J Appl Math 58(2):565–586

Part IV
Objects, Humans, and Activity

This part presents applications to several high-level vision problems including object recognition, face recognition, and activity recognition.

Chapter 13
Designing a Boosted Classifier on Riemannian Manifolds

Fatih Porikli, Oncel Tuzel, and Peter Meer

Abstract It is not trivial to build a classifier where the domain is the space of symmetric positive definite matrices such as non-singular region covariance descriptors lying on a Riemannian manifold. This chapter describes a boosted classification approach that incorporates the a priori knowledge of the geometry of the Riemannian space. The presented classifier incorporated into a rejection cascade and applied to single image human detection task. Results on INRIA and DaimlerChrysler pedestrian datasets are reported.

13.1 Introduction

Detecting and locating different types of objects in visual data is one of the fundamental tasks in computer vision. Object detection be considered as a classification problem where each candidate image region is evaluated by a learned classifier for being from the specific object class or not. This can be accomplished by generative and discriminative learning [6, 23]—two of the major paradigms for solving prediction problems in machine learning, each offering distinct advantages.

In generative approaches [7, 28], one models conditional densities of object and non-object classes, and parameters are typically estimated using a likelihood-based criterion. In discriminative approaches, one directly models the mapping from inputs to outputs (often via a prediction function); parameters are estimated by optimizing objectives related to various loss functions. Discriminative approaches have shown better performance given enough data, as they are better tailored to the prediction task and appear more robust to model mismatches. Most of the leading

F. Porikli (✉)
ANU/NICTA, Canberra, ACT, Australia
e-mail: fatih.porikli@anu.edu.au

O. Tuzel
MERL, Cambridge, MA, USA
e-mail: oncel@merl.com

P. Meer
Rutgers University, Newark, NJ, USA
e-mail: meer@cronos.rutgers.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_13

approaches in object detection can be categorized as discriminative models such as neural networks (NNs) [13], support vector machines (SVMs) [3] or boosting [22], and convolutional neural nets (CNNs) [1, 24]. These methods became increasingly popular since they can cope with high-dimensional state spaces and are able to select relevant descriptors among a large set. In [20] NNs and in [16] SVMs were utilized as single strong classifiers for detection of various categories of objects. The NNs and SVMs were also utilized for intermediate representations [5, 15] for final object detectors. In [27], multiple weak classifiers trained using AdaBoost were combined to form a rejection cascade.

In this chapter, we apply local object descriptors, namely region covariance descriptors, to human detection problem. Region covariance features were first introduced in [25] for matching and texture classification problems and later were extended to many applications from tracking [19], event detection [12], and video classification successfully [29]. We represent a human with several covariance descriptors of overlapping image regions where the best descriptors are determined with a greedy feature selection algorithm combined with boosting. A region covariance descriptor is a covariance matrix that measures of how much pixel-wise variables, such as spatial location, intensity, color, derivatives, and pixel-wise filter responses, change together within the given image region. The space of these covariance matrices does not form a vector space. For example, it is not closed under multiplication with negative scalars. Instead, they constitute a connected Riemannian manifold. More specifically, non-singular covariance matrices form a symmetric positive definite manifold that has Riemannian geometry.

It is not possible to use classical machine learning techniques to design the classifiers in this space. Consider a simple linear classifier that makes a classification decision by dividing the Euclidean space based on the value of a linear combination of input coefficients. For example, the simplest form a linear classifier in $\mathbb{R}^2$, which is a point and a direction vector in $\mathbb{R}^2$, defines a line which separates $\mathbb{R}^2$ into two. A function that divides the manifold is rather a complicated notion compared with the Euclidean space. For example, if we consider the image of the lines on the 2-torus, the curves never divide the manifold into two. Typical approaches map such manifolds to higher-dimensional Euclidean spaces, which corresponds to flattening of the manifold. They map the points on the manifold to a tangent space where traditional learning techniques can be used for classification. A tangent space is an Euclidean space relative to a point. Processing a manifold through a single tangent space is restrictive, as only distances to the original point are true geodesic distances. Distances between arbitrary points on the tangent space do not represent true geodesic distances. In general, there is no single tangent space mapping that globally preserves the distances between the points on the manifold. Therefore, a classifier trained on a single tangent space or flattened space does not reflect the global structure of the data points. As a remedy, we take advantage of the boosting framework that consists of iteratively learning weak learners in different tangent spaces to obtain a strong classifier. After a weak learner is added, the training data are reweighted. Misclassified examples are set to gain weight and correctly classified examples to lose weight. Thus, consecutive weak learners focus more on the examples that previous weak learners misclassified. To improve speed, we further structure multiple strong classifiers into a final rejection cascade such that if any previous strong classifier rejects a hypothesis, then it is considered a negative example. This provides an efficient algorithm due to sparse feature selection, besides only a few classifiers are evaluated at most of the regions due to the cascade structure. A previous version of the classification method presented in this book chapter has been published in [26].

For completeness, we present an overview of Riemannian geometry focusing on the space of symmetric positive definite matrices in Sect. 13.2. We explain how to learn a boosted classifier on a Riemannian manifold in Sect. 13.3. Then, we describe the covariance descriptors in Sect. 13.4 and their application to human detection in Sect. 13.5 with experimental evaluations in Sect. 13.6.

13.2 Riemannian Manifolds

We refer to points on a manifold with capital letters $X \in M$, whereas symmetric positive definite matrices with capital bold letters $\mathbf{X} \in \mathrm{Sym}_d^+$ and points on a tangent space with small bold letters $\times \in T_X$. The matrix norms are computed by the Frobenius norm $\|\mathbf{X}\|^2 = \mathrm{trace}(\mathbf{X}\mathbf{X}^T)$, and the vector norms are the ℓ_2 norm.

13.2.1 Manifolds

A manifold M is a topological space which is locally similar to an Euclidean space. Every point on the manifold has a neighborhood for which there exists a homeomorphic mapping the neighborhood to $\mathbb{R}^m$. Technically, a manifold M of dimension d is a connected Hausdorff space for which every point has a neighborhood that is homeomorphic to an open subset of $\mathbb{R}^d$.

A differentiable manifold M^c is a topological manifold equipped with an equivalence class of atlas whose transition maps are c-times continuously differentiable. In case all the transition maps of a differentiable manifold are smooth, i.e., all its partial derivatives exist, then it is a smooth manifold M^∞.

For differentiable manifolds, it is possible to define the derivatives of the curves on the manifold and attach to every point X on the manifold a tangent space T_X, a real vector space that intuitively contains the possible directions in which one can tangentially pass through X. In other words, the derivatives at a point X on the manifold lies in a vector space T_X, which is the tangent space at that point. The tangent space T_X is the set of all tangent vectors at X. The tangent space is a vector space, thereby it is closed under addition and scalar multiplication.

13.2.2 Riemannian Geometry

A Riemannian manifold (M, g) is a differentiable manifold in which each tangent space has an inner product g metric, which varies smoothly from point to point. It is possible to define different metrics on the same manifold to obtain different Riemannian manifolds. In practice, this metric is chosen by requiring it to be invariant to some class of geometric transformations. The inner product g induces a norm for the tangent vectors on the tangent space $\| \times \|_X^2 =< \times, \times >_X = g(\times)$.

The minimum length curve connecting two points on the manifold is called the geodesic, and the distance between the points $d(X, Y)$ is given by the length of this curve. On a Riemannian manifold, a geodesic is a smooth curve that locally joins its points along the shortest path. Suppose $\gamma(r) : [r_0, r_1] \mapsto M$ be a smooth curve on M. The length of the curve $L(\gamma)$ is defined as

$$L(\gamma) = \int_{r_0}^{r_1} \|\gamma'(r)\| dr. \tag{13.1}$$

A smooth curve is called geodesic if and only if its velocity vector is constant along the curve $\|\gamma'(r)\| = \text{const}$. Suppose X and Y be two points on M. The distance between the points $d(X, Y)$ is the infimum of the length of the curves such that $\gamma(r_0) = X$ and $\gamma(r_1) = Y$. For each tangent vector $\times \in T_X$, there exists a unique geodesic γ starting at $\gamma(0) = X$ having initial velocity $\gamma'(0) = \times$. All the shortest length curves between the points are geodesics but not vice versa. However, for nearby points the definition of geodesic and the shortest length curve coincide.

The exponential map, $\exp_X : T_X \mapsto M$, maps the vector $\mathbf{y}$ in the tangent space to the point reached by the geodesic after unit time $\exp_X(y) = 1$. Since the velocity along the geodesic is a constant, the length of the geodesic is given by the norm of the initial velocity $d(X, \exp_X(y)) = \|y\|_X$. An illustration is shown in Fig. 13.1. Under the exponential map, the image of the zero tangent vector is the point itself, $\exp_X(0) = X$. For each point on the manifold, the exponential map is a diffeomorphism (one-to-one, onto, and continuously differentiable mapping in both directions) from a neighborhood of the origin of the tangent space T_X onto a neighborhood of the point X.

In general, the exponential map $\exp_X$ is onto but only one-to-one in a neighborhood of X. Therefore, the inverse mapping $\log_X : X \mapsto T_X$ is uniquely defined only around a small neighborhood of the point X. If for any $Y \in M$, there exists several

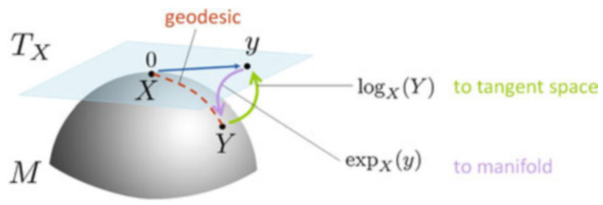


Fig. 13.1 Illustration of a manifold M and the corresponding tangent space T_X at X for point Y

$\mathbf{y} \in T_X$ such that $Y = \exp_X(\mathbf{y})$, then $\log_X(Y)$ is given by the tangent vector with the smallest norm. Notice that both operators are point dependent.

From the definition of geodesic and the exponential map, the distance between the points on manifold can be computed by

$$d(X,Y) = d(X, \exp_X(y)) =< \log_X(Y), \log_X(Y) >_X = \| \log_X(Y)\|_X = \|\mathbf{y}\|_X. \tag{13.2}$$

13.2.3 Space of Symmetric Positive Definite Matrices

The $(d \times d)$-dimensional non-singular covariance matrices, i.e., region covariance descriptors, are symmetric positive definite Sym_d^+ and can be formulated as a connected Riemannian manifold. The set of symmetric positive definite matrices is not a multiplicative group. However, an affine invariant Riemannian metric on the tangent space of Sym_d^+ is given by [17]

$$< \mathbf{y}, \mathbf{z} >_{\mathbf{X}} = \mathrm{trace}\left(\mathbf{X}^{-\frac{1}{2}}\mathbf{y}\mathbf{X}^{-1}\mathbf{z}\mathbf{X}^{-\frac{1}{2}}\right). \tag{13.3}$$

The exponential map associated to the Riemannian metric

$$\exp_{\mathbf{X}}(\mathbf{y}) = \mathbf{X}^{\frac{1}{2}} \exp\left(\mathbf{X}^{-\frac{1}{2}}\mathbf{y}\mathbf{X}^{-\frac{1}{2}}\right)\mathbf{X}^{\frac{1}{2}} \tag{13.4}$$

is a global diffeomorphism. Therefore, the logarithm is uniquely defined at all the points on the manifold

$$\log_{\mathbf{X}}(\mathbf{Y}) = \mathbf{X}^{\frac{1}{2}} \log\left(\mathbf{X}^{-\frac{1}{2}}\mathbf{Y}\mathbf{X}^{-\frac{1}{2}}\right)\mathbf{X}^{\frac{1}{2}}. \tag{13.5}$$

Above, the exp and log are the ordinary matrix exponential and logarithm operators. Not to be confused, $\exp_{\mathbf{X}}$ and $\log_{\mathbf{X}}$ are manifold-specific point-dependent operators, i.e., $\mathbf{X} \in \mathrm{Sym}_d^+$.

For symmetric matrices, these ordinary matrix exponential and logarithm operators can be computed easily. Let $\Sigma = \mathrm{UDU}^T$ be the eigenvalue decomposition of a symmetric matrix. The exponential series is

$$\exp(\Sigma) = \sum_{k=0}^{\infty} \frac{\Sigma^k}{k!} = \mathrm{U}\exp(\mathrm{D})\mathrm{U}^T, \tag{13.6}$$

where exp(D) is the diagonal matrix of the eigenvalue exponentials. Similarly, the logarithm is given by

$$\log(\Sigma) = \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k} (\Sigma - \mathrm{I})^k = \mathrm{U} \log(\mathrm{D}) \mathrm{U}^T. \tag{13.7}$$

The exponential operator is always defined, whereas the logarithms only exist for symmetric matrices with positive eigenvalues, Sym_d^+. From the definition of the geodesic given in the previous section, the distance between two points on Sym_d^+ is measured by substituting (13.5) into (13.3):

$$\begin{aligned} d^2(\mathbf{X}, \mathbf{Y}) &= < \log_{\mathbf{X}}(\mathbf{Y}), \log_{\mathbf{X}}(\mathbf{Y}) >_{\mathbf{X}} \\ &= \operatorname{trace}\left(\log^2(\mathbf{X}^{-\frac{1}{2}}\mathbf{Y}\mathbf{X}^{-\frac{1}{2}})\right). \end{aligned} \tag{13.8}$$

An equivalent form of the affine invariant distance metric was first given in [9], in terms of joint eigenvalues of $\mathbf{X}$ and $\mathbf{Y}$ as

$$d(\mathbf{X}, \mathbf{Y}) = \left(\sum_{k=1}^{d} (\ln \lambda_k(\mathbf{X}, \mathbf{Y}))^2 \right)^{\frac{1}{2}}, \tag{13.9}$$

where $\lambda_k(\mathbf{X}, \mathbf{Y})$ are the generalized eigenvalues of $\mathbf{X}$ and $\mathbf{Y}$, computed from

$$\lambda_k \mathbf{X} \mathbf{v}_k - \mathbf{Y} \mathbf{v}_k = 0 \quad k = 1 \ldots d \tag{13.10}$$

and $\mathbf{v}_k$ are the generalized eigenvectors. This distance measure satisfies the metric axioms, positivity, symmetry, triangle inequality, for positive definite symmetric matrices.

13.2.4 Vectorized Representation for the Tangent Space of Sym_d^+

The tangent space of Sym_d^+ is the space of $d \times d$ symmetric matrices and both the manifold and the tangent spaces are $d(d+1)/2$ dimensional. There are only $d(d+1)/2$ independent coefficients which are the upper triangular or the lower triangular part of the matrix. The off-diagonal entries are counted twice during norm computation.

For classification, we prefer a minimal representation of the points in the tangent space. We define an orthonormal coordinate system for the tangent space with the vector operation. The orthonormal coordinates of a tangent vector $\mathbf{y}$ in the tangent space at point $\mathbf{X}$ is given by the vector operator

$$\mathrm{vec}_{\mathbf{X}}(\mathbf{y}) = \mathrm{vec}_{\mathbf{I}}(\mathbf{X}^{-\frac{1}{2}} \mathbf{y} \mathbf{X}^{-\frac{1}{2}}), \tag{13.11}$$

where $\mathbf{I}$ is the identity matrix and the vector operator at identity is defined as

$$\mathrm{vec}_{\mathbf{I}}(\mathbf{y}) = [y_{1,1}\ \sqrt{2}y_{1,2}\ \sqrt{2}y_{1,3}\ \ldots\ y_{2,2}\ \sqrt{2}y_{2,3}\ \ldots\ y_{d,d}]^T. \tag{13.12}$$

Notice that the tangent vector $\mathbf{y}$ is a symmetric matrix, and with the vector operator $\mathrm{vec}_{\mathbf{X}}(\mathbf{y})$ we get the orthonormal coordinates of $\mathbf{y}$ which is in $\mathbb{R}^d$. The vector operator relates the Riemannian metric (13.3) on the tangent space to the canonical metric defined as

$$< \mathbf{y}, \mathbf{y} >_{\mathbf{X}} = \|\mathrm{vec}_{\mathbf{X}}(\mathbf{y})\|_2^2. \tag{13.13}$$

13.2.5 Mean of the Points on Sym_d^+

Let $\{\mathbf{X}_i\}_{i=1\ldots N}$ be a set of symmetric positive definite matrices on Riemannian manifold M. Similar to Euclidean spaces, the Riemannian center of mass [11] is the point on M which minimizes the sum of squared Riemannian distances

$$\boldsymbol{\mu} = \arg\min_{\mathbf{X}\in M} \sum_{i=1}^{N} d^2(\mathbf{X}_i, \mathbf{X}), \tag{13.14}$$

where in our case d^2 is the distance metric (13.8). In general, the Riemannian mean for a set of points is not necessarily unique. This can be easily verified by considering two points at antipodal positions on a sphere, where the error function is minimal for any point lying on the equator. However, it is shown in several studies that the mean is unique and the gradient descent algorithm is convergent for Sym_d^+ [8, 14, 17].

Differentiating the error function with respect to $\mathbf{X}$, we see that mean is the solution to the nonlinear matrix equation

$$\sum_{i=1}^{N} \log_{\mathbf{X}}(\mathbf{X}_i) = 0, \tag{13.15}$$

which gives the following gradient descent procedure [17]:

$$\boldsymbol{\mu}^{t+1} = \exp_{\boldsymbol{\mu}^t}\left[\frac{1}{N}\sum_{i=1}^{N} \log_{\boldsymbol{\mu}^t}(\mathbf{X}_i)\right]. \tag{13.16}$$

The method iterates by computing first-order approximations to the mean on the tangent space. The weighted mean computation is similar to (13.16). We replace inside of the exponential, the mean of the tangent vectors, with the weighted mean $\frac{1}{\sum_{i=1}^{N} w_i}\sum_{i=1}^{N} w_i \log_{\boldsymbol{\mu}^t}(\mathbf{X}_i)$ as shown in Fig. 13.2.

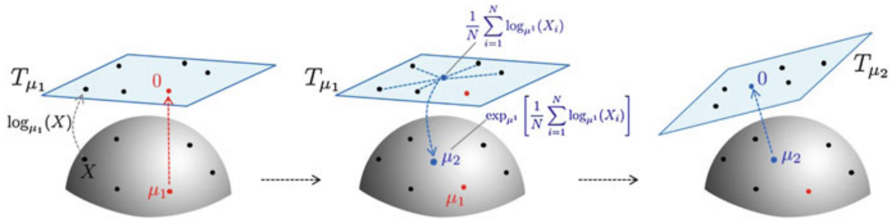

T_{μ_1}
$\log_{\mu_1}(X)$
X
0
μ_1
$\frac{1}{N}\sum_{i=1}^{N}\log_{\mu^1}(X_i)$
$\exp_{\mu^1}\left[\frac{1}{N}\sum_{i=1}^{N}\log_{\mu^1}(X_i)\right]$
μ_2
T_{μ_2}

Fig. 13.2 Illustration of iterative mean computation by mapping back and forth to tangent space

13.3 Classification on Riemannian Manifolds

We use a boosted classification approach that consists of iteratively learning weak learners in different tangent spaces to obtain a strong classifier. After a weak learner is added, the training samples are reweighted such that the weights of the misclassified examples are increased and the weights of the correctly classified examples are increased with respect to a logistic regression rule. Boosting enables future learners focus more on the examples that previous weak learners misclassified.

Furthermore, we adopt a rejection cascade structure such that if any previous strong classifier rejects a hypothesis, then it is considered a negative example. This provides an efficient algorithm as majority of hypotheses in a test image are negatives that are dismissed early in the cascade.

Let $\{(\mathbf{X}_i, l_i)\}_{i=1\ldots N}$ be the training set with class labels, where $l_i \in \{0, 1\}$. We aim to learn a strong classifier $F(\mathbf{X}) : M \mapsto \{0, 1\}$, which divides the manifold into two based on the training set of the labeled items.

13.3.1 Local Maps and Weak Classifiers

We describe an incremental approach by training several weak classifiers on the tangent spaces and combining them through boosting. We start by defining mappings from neighborhoods on the manifold to the Euclidean space, similar to coordinate charts. Our maps are the logarithm maps, $\log_{\mathbf{X}}$, that map the neighborhood of points $\mathbf{X} \in M$ to the tangent spaces $T_{\mathbf{X}}$. Since this mapping is a homeomorphism around the neighborhood of the point, the structure of the manifold is locally preserved. The tangent space is a vector space, and we use standard machine learning techniques to learn the classifiers on this space.

For classification task, the approximations to the Riemannian distances computed on the ambient space should be as close to the true distances as possible. Since we approximate the distances (13.3) on the tangent space $T_{\mathbf{X}}$,

$$d^2(\mathbf{Y}, \mathbf{Z}) \approx \|\mathrm{vec}_{\mathbf{X}}(\log_{\mathbf{X}}(\mathbf{Z})) - \mathrm{vec}_{\mathbf{X}}(\log_{\mathbf{X}}(\mathbf{Y}))\|_2^2 \tag{13.17}$$

is a first-order approximation. The approximation error can be expressed in terms of the pairwise distances computed on the manifold and the tangent space,

$$\epsilon = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(d(\mathbf{X}_i,\mathbf{X}_j) - \left\|\mathrm{vec}_{\mathbf{X}}(\log_{\mathbf{X}}(\mathbf{X}_i)) - \mathrm{vec}_{\mathbf{X}}(\log_{\mathbf{X}}(\mathbf{X}_j))\right\|_2\right)^2, \tag{13.18}$$

which is equal to

$$\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\left\|\log\left(\mathbf{X}_i^{-\frac{1}{2}}\mathbf{X}_j\mathbf{X}_i^{-\frac{1}{2}}\right)\right\|_F - \left\|\log\left(\mathbf{X}^{-\frac{1}{2}}\mathbf{X}_i\mathbf{X}^{-\frac{1}{2}}\right) - \log\left(\mathbf{X}^{-\frac{1}{2}}\mathbf{X}_j\mathbf{X}^{-\frac{1}{2}}\right)\right\|_F\right)^2 \tag{13.19}$$

for the space of symmetric positive definite matrices using (13.5) and (13.13).

The classifiers can be learned on the tangent space at any point $\mathbf{X}$ on the manifold. Best approximation, which preserves the pairwise distances, is achieved at the minimum of ϵ. The error can be minimized with respect to $\mathbf{X}$ which gives the best tangent space to learn the classifier.

Since the mean of the points (13.14) is the minimizer of the sum of squared distances from the points in the set and the mapping preserves the structure of the manifold locally, it is also a good candidate for the minimizer of the error function (13.19). However, for this a theoretical proof does not exist. For some special cases it can be easily verified that the mean is the minimizer. Such a case arises when all the points lie on a geodesic curve, where the approximation error is zero for any point lying on the curve. Since mean also lies on the geodesic curve, the approximation is perfect. Nevertheless, for a general set of points, we only have empirical validation based on simulations. We generated random points on Sym_d^+, many times with varying d. The approximation errors were measured on the tangent spaces at any of the points $T_{\mathbf{X}_{i=1\ldots N}}$ and at the mean $T_{\mathbf{X}\_}$. In our simulations, the errors computed on the tangent spaces at the means were significantly lower than any other choice and counter examples were not observed. The simulations were also repeated for weighted sets of points, where the minimizers of the weighted approximation errors were achieved at the weighted means of the points.

Therefore, the weak learners are learned on the tangent space at the mean of the points. At each iteration, we compute the weighted mean of the points through (13.16), where the weights are adjusted through boosting. Then, we map the points to the tangent space at the weighted mean and learn a weak classifier on this vector space. Since the weights of the samples which are misclassified during the earlier stages of boosting increase, the weighted mean moves towards these points and more accurate classifiers are learned for these points. The process is illustrated in Fig. 13.3. To evaluate a test example, the sample is projected to the tangent spaces at the computed weighted means, and the weak learners are evaluated. The approximation error is minimized by averaging over several weak learners.

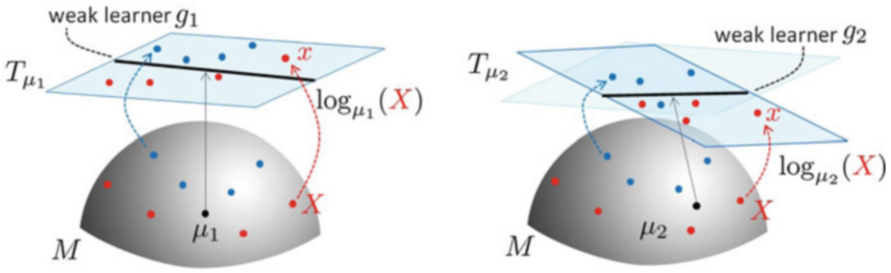


Fig. 13.3 Two iterations of boosting on a Riemannian manifold. The manifold is depicted with the surface of the sphere and the plane is the tangent space at the mean. The samples are projected to tangent spaces at means via $\log_{\mu}$. The weak learners g are learned on the tangent spaces T_{μ}. *Left*: sample $\mathbf{X}$ is misclassified therefore its weight increases. In the second iteration of boosting (*right*), the weighted mean moves towards $\mathbf{X}$

13.3.2 LogitBoost on Riemannian Manifolds

Consider the binary classification problem with labels $l_i \in \{0, 1\}$ on vector spaces. The probability of $\times$ being in class 1 is represented by

$$p(\times) = \frac{e^{F(\times)}}{e^{F(\times)} + e^{-F(\times)}} \qquad F(\times) = \frac{1}{2}\sum_{k=1}^{K} f_k(\times). \tag{13.20}$$

The LogitBoost algorithm learns the set of regression functions $\{f_k(\mathbf{x})\}_{k=1\ldots K}$ (weak learners) by minimizing the negative binomial log-likelihood of the data $(l, p(\times))$

$$-\sum_{i=1}^{N} [l_i \log(p(\times_i)) + (1 - l_i)\log(1 - p(\times_i))] \tag{13.21}$$

through Newton iterations. At the core of the algorithm, LogitBoost fits a weighted least squares regression, $f_k(\times)$ of training points $\times_i \in \mathbb{R}^d$ to response values $z_i \in \mathbb{R}$ with weights w_i where

$$z_i = \frac{l_i - p(\times_i)}{p(\times_i)(1 - p(\times_i))} \qquad w_i = p(\times_i)(1 - p(\times_i)). \tag{13.22}$$

The LogitBoost algorithm [10] on Riemannian manifolds is similar to the original LogitBoost, except a few differences at the level of weak learners. In our case, the domain of the weak learners are in M such that $f_k(\mathbf{X}) : M \mapsto \mathbb{R}$. Following the discussion of the previous section, we learn the regression functions on the tangent space at the weighted mean of the points. We define the weak learners as

$$f_k(\mathbf{X}) = g_k(\mathrm{vec}_{\boldsymbol{\mu}_k}(\log_{\boldsymbol{\mu}_k}(\mathbf{X}))) \tag{13.23}$$

```
Input: Training set $\{(\mathbf{X}_i, l_i)\}_{i=1...N}$, $\mathbf{X}_i \in M$, $l_i \in \{0,1\}$
• Initialize weights $w_i = 1/N$, $i = 1...N$, $F(\mathbf{X}) = 0$ and $p(\mathbf{X}_i) = \frac{1}{2}$
• Repeat for $k = 1...K$
    – Compute the response values and weights
      $z_i = \frac{y_i - p(\mathbf{X}_i)}{p(\mathbf{X}_i)(1-p(\mathbf{X}_i))}$, $w_i = p(\mathbf{X}_i)(1 - p(\mathbf{X}_i))$
    – Compute weighted mean of the points through (13.16)
      $\mu_k = \arg\min_{\mathbf{X} \in M} \sum_{i=1}^{N} w_i d^2(\mathbf{X}_i, \mathbf{X})$ (*)
    – Map the data points to the tangent space at $\mu_k$
      $\times_i = \mathrm{vec}_{\mu_k}(\log_{\mu_k}(\mathbf{X}_i))$ (*)
    – Fit the function $g_k(\times)$ by weighted least-square regression of $z_i$ to $\times_i$ using weights $w_i$
    – Update $F(\mathbf{X}) \leftarrow F(\mathbf{X}) + \frac{1}{2} f_k(\mathbf{X})$ where $f_k$ is defined in (13.23) and $p(\mathbf{X}) \leftarrow \frac{e^{F(\mathbf{X})}}{e^{F(\mathbf{X})} + e^{-F(\mathbf{X})}}$
• Store $F = \{\mu_k, g_k)\}_{k=1...K}$
• Output the classifier
  $\mathrm{sign}[F(\mathbf{X})] = \mathrm{sign}[\sum_{k=1}^{K} f_k(\mathbf{X})]$
```

Fig. 13.4 LogitBoost on Riemannian manifolds

and learn the functions $g_k(\times) : \mathbb{R}^d \mapsto \mathbb{R}$ and the weighted mean of the points $\boldsymbol{\mu}_k \in M$. Notice that the mapping $\mathrm{vec}_{\boldsymbol{\mu}_k}$ (13.11) gives the orthonormal coordinates of the tangent vectors in $T_{\boldsymbol{\mu}_k}$.

The algorithm is presented in Fig. 13.4. The steps marked with (*) are the differences from original LogitBoost algorithm. For functions $\{g_k\}_{k=1...K}$, it is possible to use any form of weighted least squares regression such as linear functions and regression stumps, since the domain of the functions is in $\mathbb{R}^d$.

13.4 Region Covariance Descriptors

Let $\{\mathbf{z}_i\}_{i=1..S}$ be the d-dimensional features (such as intensity, color, gradients, filter responses, etc.) of pixels inside a region R. The corresponding $d \times d$ region covariance descriptor is

$$\mathbf{C}_R = \frac{1}{S-1} \sum_{i=1}^{S} (\mathbf{z}_i - \boldsymbol{\mu})(\mathbf{z}_i - \boldsymbol{\mu})^T, \tag{13.24}$$

where $\boldsymbol{\mu}$ is a vector of the means of the features inside the regions. In Fig. 13.5, we illustrate the construction of region covariance descriptors. The diagonal entries of the covariance matrix represent the variance of each feature and the nondiagonal entries their respective correlations. Region covariance descriptors constitute the

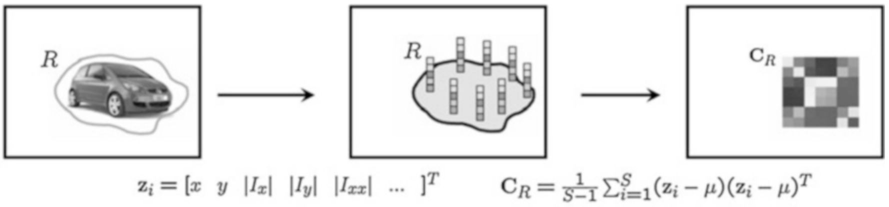

R
R
C_R
z_i = [x y |I_x| |I_y| |I_xx| ...]^T
C_R = 1/(S−1) Σ_{i=1}^S (z_i − μ)(z_i − μ)^T

Fig. 13.5 Region covariance descriptor. The d-dimensional feature image Φ is constructed from input image I. The region R is represented with the covariance matrix, $\mathbf{C}_R$, of the features $\{\mathbf{z}_i\}_{i=1..S}$

space of positive semi-definite matrices $Sym_d^{0,+}$. By adding a small diagonal matrix (or guaranteeing no features in the feature vectors would be exactly identical), they can be transformed into Sym_d^+.

There are several advantages of using covariance matrices as region descriptors. The representation proposes a natural way of fusing multiple features which might be correlated. A single covariance matrix extracted from a region is usually enough to match the region in different views and poses. The noise corrupting individual samples are largely filtered out with the average filter during covariance computation. The descriptors are low dimensional and due to symmetry $\mathbf{C}_R$ has only $d(d+1)/2$ different values (d is often less than 10) as opposed to hundreds of bins or thousands of pixels. Given a region R, its covariance $\mathbf{C}_R$ does not have any information regarding the ordering and the number of points. This implies a certain scale and rotation invariance over the regions in different images. Nevertheless, if information regarding the orientation of the points is represented, such as the gradient with respect to x and y, the covariance descriptor is no longer rotationally invariant. The same argument is also correct for illumination, too.

13.5 Application to Human Detection

For human detection, we define the features as

$$\left[x \;\; y \;\; |I_x| \;\; |I_y| \;\; \sqrt{I_x^2 + I_y^2} \;\; |I_{xx}| \;\; |I_{yy}| \;\; \arctan\frac{|I_x|}{|I_y|}\right]^T, \tag{13.25}$$

where x and y are pixel location, $I_x, I_{xx}, ..$ are intensity derivatives, and the last term is the edge orientation. With the defined mapping, the input image is mapped to a $d = 8$-dimensional feature image. The covariance descriptor of a region is an 8×8 matrix, and due to symmetry only upper triangular part is stored, which has only 36 different values. The descriptor encodes information of the variances of the defined features inside the region, their correlations with each other and spatial layout.

Given an arbitrary-sized detection window R, there are a very large number of covariance descriptors that can be computed from subwindows $r_{1,2,...}$. We perform sampling and consider all the subwindows r starting with minimum size of $1/10$ of the width and height of the detection window R, at all possible locations. The size of r is incremented in steps of $1/10$ along the horizontal or vertical, or both, until $r = R$. Although the approach might be considered redundant due to overlaps, there is significant evidence that the overlapping regions are an important factor in detection performances [4, 30]. The greedy feature selection mechanism, which will be described later, allows us to search for the best regions during learning classifiers.

Although it has been mentioned that the region covariance descriptors are robust towards illumination changes, we would like to enhance the robustness to also include local illumination variations in an image. Let r be a possible feature subwindow inside the detection window R. We compute the covariance of the detection window $\mathbf{C}_R$ and subwindow $\mathbf{C}_r$ using integral representation [18]. The normalized covariance descriptor of region r, $\hat{\mathbf{C}}_r$, is computed by dividing the columns and rows of $\mathbf{C}_r$ with the square root of the respective diagonal entries of $\mathbf{C}_R$:

$$\hat{\mathbf{C}}_r = \text{diag}(\mathbf{C}_R)^{-\frac{1}{2}}\mathbf{C}_r\text{diag}(\mathbf{C}_R)^{-\frac{1}{2}}, \tag{13.26}$$

where $\text{diag}(\mathbf{C}_R)$ is equal to $\mathbf{C}_R$ at the diagonal entries and the rest is truncated to zero. The method described is equivalent to first normalizing the feature vectors inside the region R to have zero mean and unit standard deviation, and after that computing the covariance descriptor of subwindow r. Notice that under the transformation, $\hat{\mathbf{C}}_R$ is equal to the correlation matrix of the features inside the region R. The process only requires d^2 extra division operations.

13.5.1 Training the Cascade for Human Detection

Due to the significantly large number of possible candidate detection windows R in a single image as a result of search in multiple scales and locations, and due to the considerable cost of the distance computation for each weak classifier, we adopt a rejection cascade structure to accelerate the detection process.

The domain of the classifier is the space of eight-dimensional symmetric positive definite matrices, Sym_8^+. We combine $K = 30$ strong LogitBoost classifiers on Sym_8^+ with rejection cascade, as shown in Fig. 13.6. Weak learners $g_{m,k}$ are linear regression functions learned on the tangent space of Sym_8^+. A very large number of covariance descriptors can be computed from a single detection window R. Therefore, we do not have a single set of positive and negative features, but several sets corresponding to each of the possible subwindows. Each weak learner is associated with a single subwindow of the detection window. Let $r_{m,k}$ be the subwindow associated with k-th weak learner of cascade level m.

Let R_i^+ and R_i^- refer to the N_p positive and N_n negative samples in the training set, where $N = N_p + N_n$. While training the mth cascade level, we classify all the negative examples $\{R_i^-\}_{i=1\ldots N_n}$ with the cascade of the previous

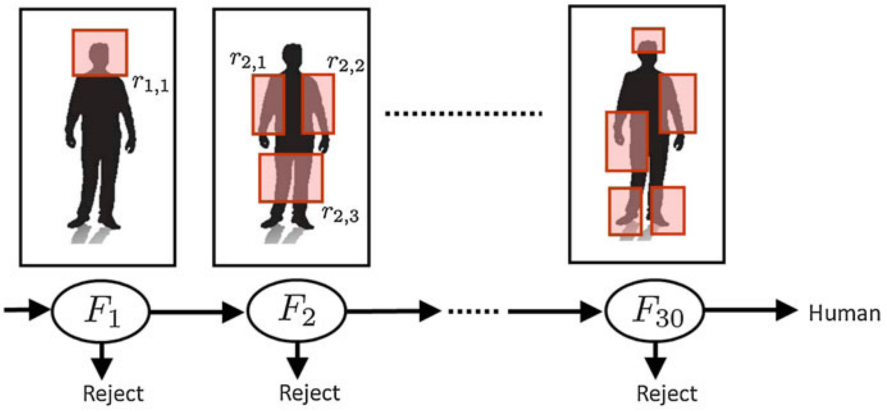


Fig. 13.6 Cascade of LogitBoost classifiers. The *m*th LogitBoost classifier selects normalized covariance descriptors of subwindows $r_{m,k}$

$(m-1)$ LogitBoost classifiers. The samples which are correctly classified (samples classified as negative) are removed from the training set. Any window sampled from a negative image is a negative example; therefore, the cardinality of the negative set, N_n, is very large. During training of each cascade level, we sample $10,000$ negative examples.

We have varying number of weak learners K_m for each LogitBoost classifier m. Each cascade level is optimized to correctly detect at least 99.8 % of the positive examples, while rejecting at least 35 % of the negative examples. In addition, we enforce a margin constraint between the positive samples and the decision boundary. Let $p_m(R)$ be the learned probability function of a sample being positive at cascade level m, evaluated through (13.20). Let R_p be the positive example that has the $(0.998N_p)$th largest probability among all the positive examples. Let R_n be the negative example that has the $(0.35N_n)$th smallest probability among all the negative examples. We continue to add weak classifiers to cascade level m until $p_m(R_p) - p_m(R_n) > 0.2$. When the constraint is satisfied, the threshold (decision boundary) for cascade level m is stored as $\tau_m = F_m(R_n)$.

A test sample is classified as positive by cascade level m if $F_m(R) > \tau_m$ or equivalently $p_m(R) > p_m(R_n)$. With the proposed method, any of the positive training samples in the top 99.8 percentile have at least 0.2 margin more probability than the points on the decision boundary. The process continues with the training of $(m+1)$th cascade level, until $m = 30$.

We incorporate a greedy feature selection method to produce a sparse set of classifiers focusing on important subwindows. At each boosting iteration k of the mth LogitBoost level, we sample 200 subwindows among all the possible subwindows and construct normalized covariance descriptors. We learn the weak classifiers representing each subwindow and add the best classifier that minimizes the negative binomial log-likelihood (13.21) to the cascade level m. The procedure iterates with training the $(k+1)$th weak learner until the specified detection rates are satisfied.

The negative sample set is not well characterized for detection tasks. Therefore, while projecting the points to the tangent space, we compute the weighted mean of only the positive samples. Although rarely happens, if some of the features are fully correlated, there will be singularities in the covariance descriptor. We ignore those cases by adding very small identity matrix to the covariance.

The learning algorithm produces a set of 30 LogitBoost classifiers which are composed of K_m triplets $F_m = \left\{(r_{m,k}, \boldsymbol{\mu}_{m,k}, g_{m,k})\right\}_{k=1\ldots K_m}$ and τ_m, where $r_{m,k}$ is the selected subwindow, $\boldsymbol{\mu}_{m,k}$ is the mean, and $g_{m,k}$ is the learned regression function of the k-th weak learner of the mth cascade. To evaluate a *test region R* with mth classifier, the normalized covariance descriptors constructed from regions $r_{m,k}$ are projected to tangent spaces $T_{\boldsymbol{\mu}_{m,k}}$ and the features are evaluated with $g_{m,k}$

$$\operatorname{sign}\left[F_m(R) - \tau_m\right] = \operatorname{sign}\left[\sum_{k=1}^{K_m} g_{m,k}\left(\operatorname{vec}_{\boldsymbol{\mu}_{m,k}}\left(\log_{\boldsymbol{\mu}_{m,k}}\left(\hat{\mathbf{C}}_{r_{m,k}}\right)\right)\right) - \tau_m\right]. \tag{13.27}$$

The initial levels of the cascade are learned on relatively easy examples, thus there are very few weak classifiers in these levels. Due to the cascade structure, only a few are evaluated for most of the test samples, which produce a very efficient solution.

13.6 Experiments and Discussion

We conduct experiments on INRIA and DaimlerChrysler datasets. Since the sizes of the pedestrians in a scene are not known a priori, the images are searched at multiple scales. There are two searching strategies. The first strategy is to scale the detection window and apply the classifier at multiple scales. The second strategy is to scale the image and apply the classifier at the original scale. In covariance representation we utilized gradient-based features which are scale dependent. Therefore, evaluating classifier at the original scale (second strategy) produces the optimal result. However, in practice up to scales of $2\times$ we observed that the detection rates were almost the same, whereas in more extreme scale changes the performance of the first strategy degraded. The drawback of the second strategy is slightly increased search time, since the method requires computation of the filters and the integral representation at multiple scales.

INRIA pedestrian dataset [4] contains 1774 pedestrian annotations (3548 with reflections) and 1671 person free images. The pedestrian annotations were scaled into a fixed size of 64×128 windows which include a margin of 16 pixels around the pedestrians. The dataset was divided into two, where 2416 pedestrian annotations and 1218 person free images were selected as the training set, and 1132 pedestrian annotations and 453 person free images were selected as the test set. Detection on INRIA pedestrian dataset is challenging since it includes subjects with a wide range of variations in pose, clothing, illumination, background, and partial occlusions.

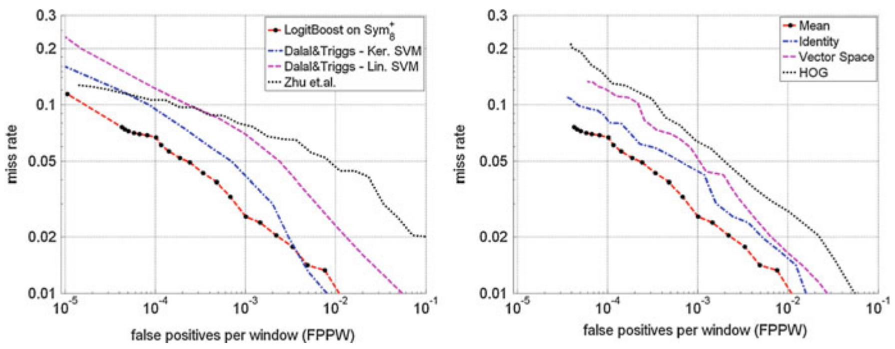


Fig. 13.7 *Left*: comparison with methods of Dalal and Triggs [4] and Zhu et al. [30] on INRIA dataset. The curves for other approaches are generated from the respective papers. *Right*: detection rates of different approaches for our method on INRIA dataset

First, we compare our results with [4] and [30]. Although it has been noted that kernel SVM is computationally expensive, we consider both the linear and kernel SVM method of [4]. In [30], a cascade of AdaBoost classifiers was trained using HOG features, and two different results were reported based on the normalization of the descriptors. Here, we consider only the best performing result, the ℓ_2-norm. In Fig. 13.7-left, we plot the detection error trade-off curves on a log–log scale. The vertical axis corresponds to the miss rate $\frac{\text{FalseNeg}}{\text{FalseNeg}+\text{TruePos}}$, and the horizontal axis corresponds to false positives per window (FPPW) $\frac{\text{FalsePos}}{\text{TrueNeg}+\text{FalsePos}}$. The curve for our method is generated by adding one cascade level at a time. For example, in our case the rightmost marker at 7.5×10^{-3} FPPW corresponds to detection using only the first 11 levels of cascade, whereas the marker positioned at 4×10^{-5} FPPW corresponds to cascade of all 30 levels. The markers between the two extremes correspond to a cascade of between 11 and 30 levels.

To generate the result at 10^{-5} FPPW (leftmost marker), we shifted the decision boundaries of all the cascade levels, τ_m, to produce less false positives at the cost of higher miss rates. We see that at almost all the false positive rates, our miss rates are significantly lower than the other approaches. The closest result to our method is the kernel SVM classifier of [4], which requires kernel evaluation at 1024 dimensional space to classify a single detection window. If we consider 10^{-4} as an acceptable FPPW, our miss rate is 6.8 %, where the second best result is 9.3 %.

Since the method removes samples which were rejected by the previous levels of cascade, during the training of last levels, only very small amount of negative samples, of order 10^2, remained. At these levels, the training error did not generalize well, such that the same detection rates are not achieved on the test set. This can be seen by the dense markers around FPPW $< 7\times10^{-5}$. We believe that better detection rates can be achieved at low false positive rates with introduction of more negative images. In our method, 25 % of false positives are originated from a single image which contains a flower texture, where the training set does not include a similar example. We note that, in [21], a pedestrian detection system utilizing shapelet features is described which has 20–40 % lower miss rates at equal FPPWs on INRIA dataset, compared to our approach. The drawback of the method is the significantly higher computational requirement.

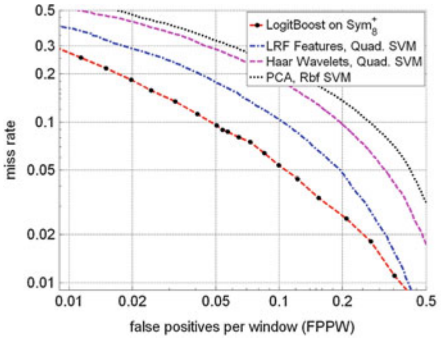
We also consider an empirical validation of the presented classification algorithm on Riemannian manifolds. In Fig. 13.7-right, we present the detection error trade-off curves for four different approaches: (1) The original method, which maps the points to the tangent spaces at the weighted means. (2) The mean computation step is removed from the original algorithm and points are always mapped to the tangent space at the identity. (3) We ignore the geometry of Sym_8^+, and stack the upper triangular part of the covariance matrix into a vector, such that learning is performed on the vector space. (4) We replace the covariance descriptors with HOG descriptors, and perform original (vector space) LogitBoost classification.

The original method outperforms all the other approaches significantly. The second best result is achieved by mapping points to the tangent space at the identity matrix followed by the vector space approaches. Notice that our LogitBoost implementation utilizing HOG descriptors has 3 % more miss rate at 10^{-4} FPPW than [30] which trains an AdaBoost classifier. The performance is significantly degraded beyond this point.

DaimlerChrysler dataset [15] contains 4000 pedestrian (24,000 with reflections and small shifts) and 25,000 non-pedestrian annotations. As opposed to INRIA dataset, non-pedestrian annotations were selected by a preprocessing step from the negative samples, which match a pedestrian shape template based on average Chamfer distance score. Both annotations were scaled into a fixed size of 18×36 windows, and pedestrian annotations include a margin of 2 pixels around. The dataset was organized into three training and two test sets, each of them having 4800 positive and 5000 negative examples. The small size of the windows combined with a carefully arranged negative set makes detection on DaimlerChrysler dataset extremely challenging. In addition, 3600 person free images with varying sizes between 360×288 and 640×480 were also supplied.

In [15], an experimental study was described comparing three different feature descriptors and various classification techniques. The compared feature descriptors were the PCA coefficients, Haar wavelets, and local receptive fields (LRFs) which are the output of the hidden layer of a specially designed feed forward NN. We compare our method with the best results for each descriptor in [15]. The same training configuration is prepared by selecting two out of three training sets. Since the number of non-pedestrian annotations was very limited for training of our method, we adapted the training parameters. A cascade of $K = 15$ LogitBoost classifiers on Sym_8^+ is learned, where each level is optimized to detect at least 99.75 % of the positive examples, while rejecting at least 25 % negative samples.

In Fig. 13.8-left, we plot the detection error trade-off curves. The cascade of 15 LogitBoost classifiers produced a FPPW rate of 0.05. The detection rates with lower FPPW are generated by shifting the decision boundaries of all the cascade levels gradually, until FPPW $=$ 0.01. We see that our approach has significantly lower miss rates at all the false positive rates. This experiment should not be confused with the experiments on INRIA dataset, where much lower FPPW rates were observed. Here, the negative set consists of hard examples selected by a preprocessing step.



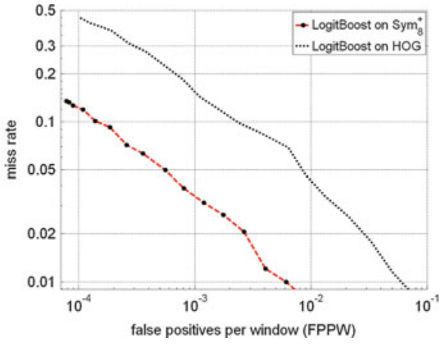


Fig. 13.8 *Left*: comparison with [15] on DaimlerChrysler dataset. The curves for other approaches are generated from the original paper. Comparison of covariance and HOG descriptors on DaimlerChrysler dataset

We also set up a different test configuration on DaimlerChrysler dataset. The 3600 person free images are divided into two: 2400 images are selected as the negative training set and 1200 images are selected for the negative test set. For both the covariance descriptors and the HOG descriptors, we trained cascade of 25 classifiers. We observed that the object sizes were too small for HOG descriptors to separate among positive and negative examples at the later levels of cascade. The classifiers trained utilizing HOG descriptors failed to achieve the specified detection (99.8 %) and the rejection rates (35.0 %). We stopped adding weak learners to a cascade level after reaching $K_m = 100$. The detection error trade-off curves are given in Fig. 13.8-right where we see that the covariance descriptors significantly outperform HOG descriptors.

Utilizing the classifier trained on the INRIA dataset, we generated several detection examples for crowded scenes with pedestrians having variable illumination, appearance, pose, and partial occlusion. The results are shown in Fig. 13.9. The images are searched at five scales using the first strategy, starting with the original window size 64×128 and two smaller and two larger scales of ratio 1.2. The white dots are all the detection results and we filtered them with adaptive bandwidth mean shift filtering [2] with bandwidth 0.1 of the window width and height. Black dots show the modes and ellipses are generated by averaging the detection window sizes converging to the modes.

13.7 Remarks

The presented LogitBoost learning algorithm is not specific to Sym_d^+ and can be used to train classifiers for points lying on any connected Riemannian manifold. In addition, the approach can be combined with any boosting method including

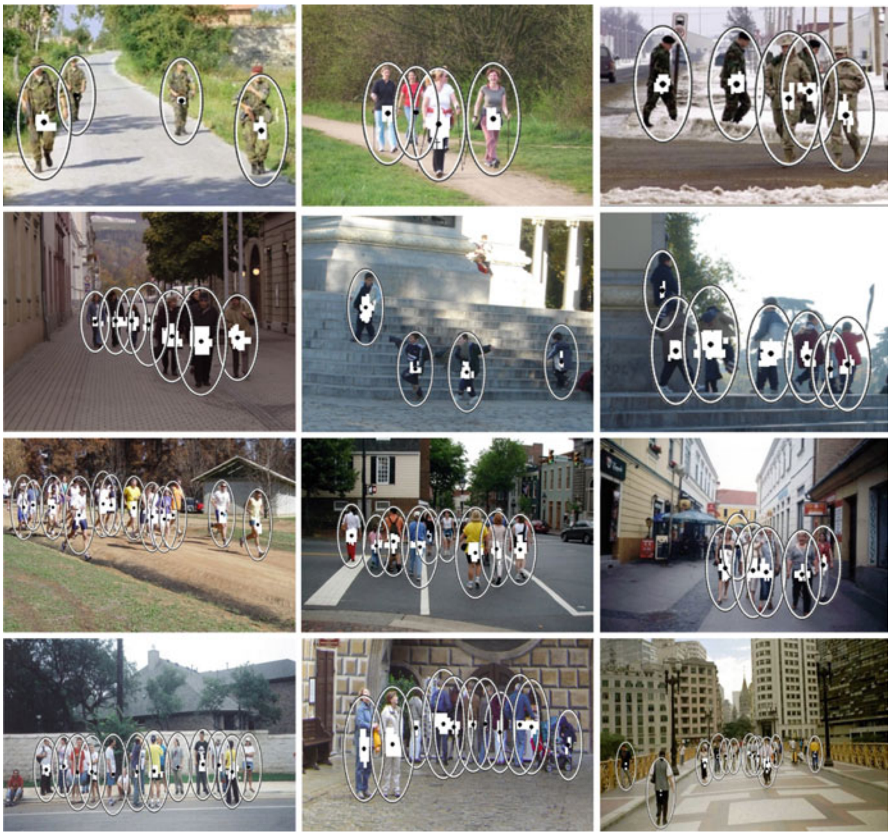


Fig. 13.9 Detection examples. *White dots* show all the detection results. *Black dots* are the modes generated by mean shift smoothing and the ellipses are average detection window sizes. There are extremely few false positives and negatives

GentleBoost and AdaBoost classifiers on Riemannian manifolds using LDA, decision stumps, and linear SVMs as weak learners. In our experiments, the results of the methods were comparable.

References

1. Bengio Y, Goodfellow I, Courville A (2014) Deep learning. MIT Press, Cambridge
2. Comaniciu D, Meer P (2002) Mean shift: a robust approach toward feature space analysis. IEEE Trans Pattern Anal Mach Intell 24:603–619
3. Cortes C, Vapnik V (1995) Support vector networks. Mach Learn 20:273–297
4. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE conference on computer vision and pattern recognition, pp 886–893

5. Dorkó G, Schmid C (2003) Selection of scale-invariant parts for object class recognition. In: International conference on computer vision, pp 634–640
6. Efron B (1975) The efficiency of logistic regression compared to normal discriminant analysis. J Am Stat Assoc 70(352):892–898
7. Fergus R, Perona P, Zisserman A (2003) Object class recognition by unsupervised scale-invariant learning. In: IEEE conference on computer vision and pattern recognition, pp 264–271
8. Fletcher PT, Joshi S (2007) Riemannian geometry for the statistical analysis of diffusion tensor data. Signal Process 87(2):250–262
9. Förstner W, Moonen B (1999) A metric for covariance matrices. Technical report, Dept. of Geodesy and Geoinformatics, Stuttgart University
10. Friedman J, Hastie T, Tibshirani R (2000) Additive logistic regression: a statistical view of boosting. Ann Stat 28(2):337–407
11. Grove K, Karcher H (1973) How to conjugate C^1-close group actions. Math Z 132:11–20
12. Guo K, Ishwar P, Konrad J (2010) Action recognition using sparse representation on covariance manifolds of optical flow. In: IEEE international conference advanced video and signal based surveillance (AVSS), pp 188–195
13. Haykin S (1998) Neural networks: a comprehensive foundation, 2nd edn. Prentice Hall, Upper Saddle River
14. Moakher M (2005) A differential geometric approach to the geometric mean of symmetric positive-definite matrices. SIAM J Matrix Anal Appl 26:735–747
15. Munder S, Gavrila DM (2006) An experimental study on pedestrian classification. IEEE Trans Pattern Anal Mach Intell 28:1863–1868
16. Papageorgiou P, Poggio T (2000) A trainable system for object detection. Int J Comput Vis 38(1):15–33
17. Pennec X, Fillard P, Ayache N (2006) A Riemannian framework for tensor computing. Int J Comput Vis 66(1):41–66
18. Porikli F (2005) Integral histogram: a fast way to extract histograms in Cartesian spaces. In: IEEE conference on computer vision and pattern recognition, pp 829–836
19. Porikli F, Tuzel O, Meer P (2006) Covariance tracking using model update based on Lie algebra. In: IEEE conference on computer vision and pattern recognition, pp 728–735
20. Rowley H, Baluja S, Kanade T (1998) Neural network-based face detection. IEEE Trans Pattern Anal Mach Intell 20:22–38
21. Sabzmeydani P, Mori G (2007) Detecting pedestrians by learning shapelet features. In: IEEE conference on computer vision and pattern recognition
22. Schapire RE (2002) The boosting approach to machine learning, an overview. In: MSRI workshop on nonlinear estimation and classification
23. Schmah T, Hinton GE, Zemel R, Small SL, Strother S (2009) Generative versus discriminative training of RBMs for classification of fMRI images. In: Advances in neural information processing systems, 21
24. Sermanet P, Kavukcuoglu K, Chintala S, LeCu Y (2013) Pedestrian detection with unsupervised multi-stage feature learning. In: IEEE conference on computer vision and pattern recognition
25. Tuzel O, Porikli F, Meer P (2006) Region covariance: a fast descriptor for detection and classification. In: European conference on computer vision, pp 589–600
26. Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on Riemannian manifolds. IEEE Trans Pattern Anal Mach Intell 30(10):1713–1727
27. Viola P, Jones M (2001) Rapid object detection using a boosted cascade of simple features. In: IEEE conference on computer vision and pattern recognition, pp 511–518
28. Weber M, Welling M, Perona P (2000) Unsupervised learning of models for recognition. In: European conference on computer vision, pp 18–32

29. Zhang S, Kasiviswanathan S, Yuen PC, Harandi M (2015) Online dictionary learning on symmetric positive definite manifolds with vision applications. In: AAAI conference on artificial intelligence
30. Zhu Q, Avidan S, Yeh MC, Cheng KT (2006) Fast human detection using a cascade of histograms of oriented gradients. In: IEEE conference on computer vision and pattern recognition, pp 1491–1498

Chapter 14
A General Least Squares Regression Framework on Matrix Manifolds for Computer Vision

Yui Man Lui

Abstract Least squares regression is one of the fundamental approaches for statistical analysis. And yet, its simplicity has often led to researchers overlooking it for complex recognition problems. In this chapter, we present a nonlinear regression framework on matrix manifolds. The proposed method is developed based upon two key attributes: underlying geometry and least squares fitting. The former attribute is vital since geometry characterizes the space for classification while the latter exhibits a simple estimation model. Considering geometric properties, we formulate the least squares regression as a composite function. The proposed framework can be naturally generalized to many matrix manifolds. We show that this novel formulation is applicable to matrix Lie groups, SPD(n), Grassmann manifolds, and the product of Grassmann manifolds for a number of computer vision applications ranging from visual tracking, object categorization, and activity recognition to human interaction recognition. Our experiments reveal that the proposed method yields competitive performance, including state-of-the-art results on challenging activity recognition benchmarks.

14.1 Introduction

Multiple regression is one of the fundamental tools in statistical analysis and machine learning. It manifests the relationship between observation and training data. Among many regression methods, the most straightforward model may be least squares regression. It is simple and often outperforms complicated models when the number of training samples is small [12]. However, researchers often prefer complicated models when the recognition tasks are intricate, resulting in a larger set of parameters.

Alternatively, we can focus on a simple statistical model with emphasis on the geometrical aspect of data. In particular, high-dimensional data often arise in various computer vision applications and may be modeled on matrix manifolds.

Y.M. Lui (✉)
Department of Mathematics and Computer Science, University of Central Missouri, Warrensburg, MO 64093, USA
e-mail: lui@ucmo.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_14

For instance, Lin et al. [21] have modeled visual flows through geometric transformations which are the elements of matrix Lie group. Furthermore, the separation between foreground and background of a video stream has been realized on the Grassmann manifold [13]. Lighting and alignment variations have also been characterized using domain adaptation on a product manifold for face recognition [14]. Su et al. [37] have analyzed temporal evolution patterns on the Riemannian manifold for visual speech recognition.

Among many statistical tools, regression analysis is a popular technique for estimating relationships among data and has been widely used in pattern recognition. While traditional regression methods perform regression analysis in Euclidean space, these techniques may not be applicable to matrix manifolds where data are represented as matrices; in other words, such data are sampled from a manifold. Recently, many regression paradigms have been introduced on matrix manifolds [4, 8, 27, 28, 30, 31, 38]. Nevertheless, a unified scheme for matrix manifolds is still lacking. Here, we propose a unified framework for least squares regression analysis. The key element of our regression framework is accounting for the underlying geometry on matrix manifolds. In doing so, we formulate least squares regression as a composite function; our regression framework naturally generalizes to Euclidean and non-Euclidean spaces as long as the mappings between manifolds and tangent spaces are well defined. This makes our method suitable to many matrix manifolds and applications.

We evaluate the proposed regression framework on four matrix manifolds including matrix Lie groups, SPD(n), Grassmann manifolds, and the product of Grassmann manifolds for visual tracking, object categorization, activity recognition, and human interaction recognition. We illustrate the merits of our regression framework by underpinning the importance of latent geometry using simple feature representations. For visual tracking, we model the underlying motion as a matrix Lie group and utilize a particle filter for state transition and PCA for appearance characterization. For object categorization, we employ position, color, and gradient vectors to form a symmetric positive definite matrix on a Riemannian cone; in addition, the proficiency of using Grassmann manifolds is studied. For human activity and interaction recognition, video sequences are expressed on a product manifold where we exploit grayscale pixels as the feature representation. As such, we do not extract silhouette images or apply dynamic time warping, which simplifies feature selection. All of these applications are modeled using the proposed least squares regression framework.

The primary contribution of this work is revealing the generalizability of least squares regression on a number of matrix manifolds and its applicability to a variety of computer vision applications. We demonstrate that competitive results can be achieved using the proposed regression framework when the intrinsic geometry is appropriately addressed.

The remainder of this chapter is organized as follows: Related work and a mathematical preliminary are given in Sects. 14.2 and 14.3, respectively. The regression framework in non-Euclidean spaces is introduced in Sect. 14.4. Application descriptions are provided in Sect. 14.5. Experimental results are reported in Sect. 14.6. Finally, discussion and conclusions are given in Sect. 14.7.

14.2 Related Work

Regression techniques have been studied and applied to a variety of machine learning and computer vision applications. Here, we review some related work.

Human detection is an integral part of human action recognition. Schwartz et al. [35] combined HOG, color frequency, and co-occurrence matrices and employed partial least squares (PLS) fitting for dimension reduction. QDA was then employed for classification. Unlike traditional methods like PCA, PLS takes class labels into account while performing dimension reduction.

Liu et al. [22] proposed ordinal regression via manifold learning. The order information from a neighborhood graph was optimized by maximizing the margin between two consecutive ranks while preserving the underlying geometry. This framework was extended to a multilinear representation and applied to face and digit classification using a *k*-NN classifier.

The Lie group theory has been widely used in motion estimation. Davis et al. [4] characterized the shape changes in anatomy using kernel regression in which the intrinsic shapes were modeled by SO(3). Tuzel et al. [38] proposed a regression forest model while Porikli and Pan [31] employed importance sampling for object tracking. Both of these methods updated the motion parameters in the Lie algebra. Recently, Fletcher [8] has introduced a geodesic regression method in the Kendall's shape space where curves are fitted on a manifold. This geodesic regression was applied to fit the shape of the corpus callosum as it changes at different ages.

Pham and Venkatesh [30] studied the applicability of multivariate lasso regression on Stiefel manifolds for face recognition. This method employed dual projections for dimension reduction and data fitting. Because of the orthogonality constraint of the projection matrix, the steepest descent method was applied to find the optimal projection on the Stiefel manifold.

Meyer et al. [27] addressed a regression model under fixed-rank constraints on a Riemannian manifold. Since fixed-rank matrices do not reside in Euclidean space, quotient geometry was considered. Matrices were first factorized using balanced and polar factorizations. Line-search algorithms were then employed to seek the optimal projection. This regression model was applied to the Movie-Lens collaborative filtering data.

While previous methods have demonstrated the effectiveness of regression analysis on some specific problems, we introduce a unified view of least squares regression on a variety of matrix manifolds and show its applicability to visual tracking, object categorization, activity recognition, and human interaction recognition. Before discussing the proposed regression framework, we briefly review the preliminary mathematics on matrix manifolds used in this chapter.

14.3 Mathematical Preliminary

In this section, we briefly illustrate the definitions of some matrix manifolds and their operators. These manifolds include matrix Lie groups, SPD(n), Grassmann manifolds, and the product of Grassmann manifolds. For details of matrix manifolds see [1, 2, 6, 19, 29].

14.3.1 Matrix Lie Groups

A matrix Lie group, G_M, is both a group and a smooth manifold and is a subgroup of general linear group GL(n) [2]. A group G is defined as a nonempty set together with group operations

$$\texttt{Multiplication} : G \times G \longrightarrow G$$
$$\texttt{Inversion} : G \longrightarrow G$$

where the multiplication is associative and the inversion is bijective. The GL(n) is closed under group operations.

While smooth manifolds are globally curved, they are locally like $\mathbb{R}^n$. As such, one may map an element from a matrix Lie group to the Lie algebra and from the Lie algebra back to the matrix Lie group. These maps are called logarithmic and exponential maps defined as

$$\log_X(Y) = \log(X^{-1}Y) \tag{14.1}$$
$$\exp_X(\Delta) = X \exp(\Delta) \tag{14.2}$$

where X and Y are elements in a matrix Lie group, Δ is the tangent vector in the Lie algebra which is locally Euclidean, and log and exp are the matrix logarithmic and exponential operators.

The geodesic distance on a smooth manifold is measured by the length of a curve. Using the Baker–Campbell–Hausdorff (BCH) formula and taking the first-order term, the intrinsic distance in the matrix Lie group can be approximated by

$$d(X, Y) = \| \log(X^{-1}Y) \|_F \tag{14.3}$$

14.3.2 SPD(n)

The set of $n \times n$ symmetric positive definite (*SPD*) matrices can be endowed with a geometric structure on a Riemannian manifold [29]. Formally,

$$\mathrm{Sym}_n^+ = \{Y \in \mathbb{R}^{n\times n} : Y^T = Y, Y > 0\} \tag{14.4}$$

where $Y > 0$ denotes $x^T Yx > 0$ for any $x \in \mathbb{R}^n \setminus \{0\}$. As such, the set of all SPD matrices forms a space of an open convex cone. Sometimes, this space is referred to a Riemannian cone or manifold of *SPD* matrices.

Similar to matrix Lie groups previously discussed, we can map an element from Sym_n^+ to its tangent space and from the tangent space back to Sym_n^+ using logarithmic and exponential maps, respectively. The logarithmic and exponential maps for Sym_n^+ may be defined as

$$\log_X(Y) = X^{\frac{1}{2}} \log(X^{\frac{-1}{2}} Y X^{\frac{-1}{2}}) X^{\frac{1}{2}} \tag{14.5}$$

$$\exp_X(\varDelta) = X^{\frac{1}{2}} \exp(X^{\frac{-1}{2}} \varDelta X^{\frac{-1}{2}}) X^{\frac{1}{2}} \tag{14.6}$$

where X and Y are elements of Sym_n^+ and $\varDelta$ is the tangent vector at X. Because elements of Sym_n^+ are symmetric matrices, the log and exp operators can be efficiently computed using spectral decomposition. Furthermore, the geodesic distance on the space of Sym_n^+ may be computed as

$$d(X, Y) = \mathrm{trace}(\log^2(X^{\frac{-1}{2}} Y X^{\frac{-1}{2}})) \tag{14.7}$$

14.3.3 Grassmann Manifolds

The set of p-dimensional linear subspaces of $\mathbb{R}^n$ may be viewed as points on a Grassmann manifold $\mathcal{G}_{n,p}$ where elements are considered as being equivalent if there exists a $p \times p$ orthogonal matrix mapping one point into the other. This map may be identified by

$$(Y, Q_p) \longmapsto YQ_p \in \mathrm{span}(Y) \tag{14.8}$$

where Y is an $n \times p$ orthogonal matrix, Q_p is a $p \times p$ orthogonal matrix, and span(Y) is an element on a Grassmann manifold [6].

Similar to matrix Lie groups and SPD(n), logarithmic and exponential maps are used to chart elements between the Grassmann manifold and its tangent space. These maps can be defined as

$$\log_X(Y) = U_1 \varTheta_1 V_1^T \tag{14.9}$$

$$\exp_X(\varDelta) = XV_2 \cos(\varSigma_2) + U_2 \sin(\varSigma_2) \tag{14.10}$$

where X and Y are elements on a Grassmann manifold, $X_\perp X_\perp^T Y(X^T Y)^{-1} = U_1 \varSigma_1 V_1^T$, $\varTheta_1 = \arctan(\varSigma_1)$, $X_\perp$ is the orthogonal complement to X, and $\varDelta$ is the tangent vectors at X and $\varDelta = U_2 \varSigma_2 V_2^T$.

While many metrics can be placed on the Grassmannian topology, we chose the chordal distance as the measure of geodesic distance since it is differentiable everywhere [3]. The chordal distance is defined as

$$d(X, Y) = \| \sin\theta \|_2 \tag{14.11}$$

where θ are canonical angles (also known as principal angles) and each angle is recursively computed as

$$\theta_k = \min_{\substack{x \in \mathrm{span}(X) \\ y \in \mathrm{span}(Y)}} \cos^{-1}(x^T y) = \cos^{-1}(x_k^T y_k) \tag{14.12}$$

subject to

$$\begin{aligned} \| x \| = \| y \| &= 1 \\ x^T x_i = 0, \quad y^T y_i = 0, &\quad i = 1, \ldots, k-1. \end{aligned}$$

14.3.4 Product of Grassmann Manifolds

The product of Grassmann manifolds has recently been introduced in [24, 25] for the characterization of high-order data tensors, particularly in video data. Considering a video represented as a third-order tensor $\mathcal{T}$, we can apply a high order singular value decomposition [18] (HOSVD) to factorize the data tensor as

$$\mathcal{T} = \mathcal{S} \times_1 V^{(1)}_{\text{v-motion}} \times_2 V^{(2)}_{\text{h-motion}} \times_3 V^{(3)}_{\text{appearance}} \tag{14.13}$$

where $\times_k$ denotes the mode-k multiplication, $\mathcal{S}$ is the core tensor, and $V^{(k)}$ is the orthogonal matrix spanning the row space of the unfolded matrix $T_{(k)}$ associated with nonzero singular values. In the context of video data, $V^{(1)}$, $V^{(2)}$, and $V^{(3)}$ express the horizontal motion, vertical motion, and appearance, respectively. By imposing a quotient geometry of an orthogonal group to the orthogonal matrices $V^{(k)}$, the spanning sets of these matrices are elements on three Grassmann manifolds. The product of these factor manifolds forms a product manifold formulated as

$$\begin{aligned} M_\Pi &= M_1 \times M_2 \times M_3 \\ &= \left\{ (x_1, x_2, x_3) : \begin{matrix} x_1 \in M_1, \\ x_2 \in M_2, \\ x_3 \in M_3 \end{matrix} \right\} \end{aligned} \tag{14.14}$$

where $\times$ denotes the Cartesian product, M_k represents a factor manifold, and x_k is an element in M_k. Here, the Cartesian product establishes a smooth manifold whose topology is equivalent to the product topology [19]. Thus, the geodesic distance may be characterized using the chordal distance given in Eq. (14.11) where the canonical angles are computed from each factor manifold. Consequently, higher data tensors are abstracted to points and their intrinsic distances may be computed accordingly.

14.4 Regression in Non-Euclidean Spaces

Linear regression is one of the fundamental techniques in data analysis and machine learning. It often outperforms complicated models when the number of training data is insufficient [12]. The aim of this work is to formulate a least squares regression model on matrix manifolds.

14.4.1 Linear and Nonlinear Least Squares Regressions

Perhaps the most straightforward data prediction technique is least squares regression. Here, we demonstrate the effectiveness of regression by considering the underlying geometry. Our regression framework can achieve competitive performance on a variety of computer vision applications. Before we discuss the geometric extension, we will first review the standard form of least squares fitting.

Consider a regression problem $y = A\beta$ where $y \in \mathbb{R}^n$ is the regression value, $A([a_1|a_2|\cdots|a_k]) \in \mathbb{R}^{n\times k}$ is the training set, and $\beta \in \mathbb{R}^k$ is the fitting parameter. The residual sum-of-squares can be written as

$$R(\beta) = \| y - A\beta \|^2 \tag{14.15}$$

and the fitting parameter β can be obtained by minimizing the residual sum-of-squares error from Eq. (14.15). Then, we have

$$\hat{\beta} = (A^T A)^{-1} A^T y \tag{14.16}$$

The fitted pattern from the training set has the following form:

$$\hat{y} = A\hat{\beta} = A(A^T A)^{-1} A^T y \tag{14.17}$$

We can further extend the linear least squares regression from Eq. (14.17) to a nonlinear form by incorporating a kernel function. Here, the multiplications in Eq. (14.16) are replaced by a nonlinear operator shown in the following

$$A(A \star A)^{-1}(A \star y) \tag{14.18}$$

where $\star$ is a nonlinear similarity operator. Obviously, $\star$ is equal to $x^T y$ in the linear space. Here, the elements of $A \star A$ and $A \star y$ are computed using the RBF kernel given as

$$x \star y = \exp\left(-\frac{\|\cdot\|}{\sigma}\right) \tag{14.19}$$

where σ is set to 2 in all our experiments, and $\|\cdot\|$ is a distance measure associated with a particular matrix manifold discussed in Sect. 14.5.

14.4.2 Nonlinear Least Squares Regression in Non-Euclidean Spaces

Non-Euclidean geometry often arises in many real-world applications. The key contribution of this work is the characterization of least squares regression in non-Euclidean spaces modeled by matrix manifolds. In traditional applications, the patterns A in Eq. (14.18) generally represent a data matrix from a training set, i.e., a set of training instances. To incorporate high-dimensional data on matrix manifolds, the patterns $\mathcal{A}$ will represent the training set composed by a set of matrices, i.e., a high-order tensor. Similarly, the regression value Y is also a matrix.

We now take a closer look to what the computation of a similarity operator gives us. Given p training instances, $(\mathcal{A}\star\mathcal{A})^{-1}$ produces a $p\times p$ matrix from the training set and $(\mathcal{A}\star Y)$ would create a $p\times 1$ vector. Thus, the similarity map yields a weighting vector that characterizes the training samples on a manifold as

$$\mathbf{w} = (\mathcal{A}\star\mathcal{A})^{-1}(\mathcal{A}\star Y) \tag{14.20}$$

where the weighting vector, $\mathbf{w}$, is in a vector space $\mathcal{V}$.

Recall that $\mathcal{A}$ is a set of training instances residing on a matrix manifold $\mathcal{M}$. To join the manifold data with the weighting vector in a vector space, we propose a composite function $\mathcal{F}\circ\mathcal{H}$ where $\mathcal{H}:\mathcal{M}\longrightarrow\mathcal{V}$ and $\mathcal{F}:\mathcal{V}\longrightarrow\mathcal{M}$. As a result, the least squares regression on matrix manifolds is formulated as

$$\Psi(Y) = \mathcal{A}\bullet\{(\mathcal{A}\star\mathcal{A})^{-1}(\mathcal{A}\star Y)\} \tag{14.21}$$

where $\bullet$ is an operator realizing the composite function that maps data between a matrix manifold and a vector space. By introducing an additional operator, we ensure that both the domain data Y and the range data $\Psi(Y)$ reside on a matrix manifold.

One possible way to accomplish such mapping $\mathcal{F}\circ\mathcal{H}$ is to modify the Riemannian center of mass (a.k.a. Grove–Karcher mean) computation [16]. The calculation of Grove–Karcher mean takes the geometry aspect into account and iteratively minimizes the distance between the updated mean and all data samples. Since w is the weighting vector, it naturally produces the weight between training data. All we

Algorithm 1 Least Squares Regression on Matrix Manifolds

```
1: Compute the weighting vector
2: Initialize a base point on a matrix manifold
3: while not converge do
4:     Apply the logarithmic map to the training samples to the base point
5:     Compute the weighted average on the tangent space at the base point
6:     Update the base point by applying the exponential map on the weighted average
7: end while
8: Output the base point
```

need is to use the weighting vector to weight the training data on a matrix manifold. This is equivalent to computing the weighted Grove–Karcher mean which is indeed an element on the manifold; therefore, the composite function is realized.

To illustrate the computation of least squares regression on matrix manifolds, we sketch the pseudo-code in Algorithm 1. As Algorithm 1 reveals, the first step is to compute the weighting vector. Second, we need to initialize a base point on a manifold. In our experiments, we set it to the identity matrix. Then, we iteratively update the base point on the manifold. The updated procedure involves the standard logarithmic and exponential maps defined in the previous section for a specific matrix manifold. Finally, the converged base point is the regressed output.

For classification, the class label is determined from the regression models as

$$l^* = \underset{l}{\operatorname{argmin}} \; d(Y, \Psi_l(Y)) \tag{14.22}$$

where l is the class regressed model, Y is the test instance, and d is the geodesic distance associated with a particular matrix manifold which we will discuss in the next section.

Finally, we note that when the kernel operator is $\star = x^T y$, $\log_x(y) = y$, and $\exp_x(\Delta) = x + \Delta$, the regression model in Eq. (14.21) becomes the canonical least squares regression in Euclidean space. This example reinforces the generalizability of the proposed regression framework.

14.5 Computer Vision Applications

We demonstrate the effectiveness and applicability of the proposed regression framework on four matrix manifolds, the matrix Lie group G_M, the Riemannian manifold Sym_n^+, the Grassmann manifold $\mathcal{G}_{n,p}$, and the product of Grassmann manifold M_Π, with four computer vision applications. The applications span the areas of visual tracking, object categorization, activity recognition, and human interaction recognition. Recent advances in matrix manifolds for computer vision can be found in [23].

Our regression framework can be generalized to various matrix manifolds as long as the operator that realizes the composite function given in Eq. (14.21) is well defined. To specify the proposed regression framework for a particular matrix manifold or application, we need to determine two elements. They are the measure of intrinsic distance on a matrix manifold $\| \cdot \|$ given in Eq. (14.19) and the feature representation for the application. The following subsections discuss how these elements may be constructed.

14.5.1 Visual Tracking

For visual tracking, we model the underlying motion as affine transformations. The group of affine transformations can be written as $\begin{bmatrix} A & T \\ 0 & 1 \end{bmatrix}$ where A is a 2×2 non-singular matrix and $T \in \mathbb{R}^2$. The set of all affine transformations forms the matrix Lie group representation of SE(2) characterizing scale, rotation, skew, and translation. As far as visual tracking is concerned, such space is our state space which is obviously nonlinear.

Next, we need to employ the distance measure $\| \cdot \|$ associated with the intrinsic distance. That is, $\sqrt{\| \log(X^{-1}Y) \|_F}$ where X and Y are elements of affine transformations and log is the matrix logarithmic operator. The visual tracking is realized via a sequential Monte Carlo technique [9] also known as a particle filter. This tracking algorithm is particularly effective for non-Gaussian data. The aim here is to track the affine transformation represented by a particle (state) that characterizes the underlying motion. The tracked particle is obtained by applying our least squares regression in the matrix Lie group. In addition, we employ the standard PCA for appearance characterization. Consequently, both motion and appearance are considered. We note that such formulation can also be casted to a Bayesian subspace tracking problem [36].

14.5.2 Object Categorization

For object categorization, we consider two sorts of matrix manifolds. First, we take a set of simple features $[x\ y\ R\ G\ B\ |I_x|\ |I_y|]$ where x and y are the pixel location, R, G, and B are the color channels, and I_x and I_y are the intensity derivatives. These features can be used to form a covariance matrix which is a 7×7 symmetric positive definite (SPD) matrix. Thus, these features are characterized on the space of Sym_n^+. In addition, we choose the distance measure $\| \cdot \|$ as $\| \log_I(x) - \log_I(y) \|_F$ where $\log_I$ stands for a logarithmic map at the identity matrix and $\log_I(x)$ is computed using Eq. (14.5).

Second, we consider grayscale pixels as the feature pattern represented by an element of the Grassmann manifold $\mathcal{G}_{n,p}$. By imposing a rotation invariant to the patterns, elements on the Grassmann manifold are subspaces spanned by the columns of orthogonalized data. We choose the distance measure $\| \cdot \|$ as $\| \sin\theta \|_2$ shown in Eq. (14.11) where θ are the canonical angles between subspaces. Thus, this distance measure characterizes the intrinsic geometry of quotient space.

Once the feature patterns and the intrinsic distances are provided, we can plug these elements into our regression framework and perform object categorization using Eq. (14.22).

14.5.3 Human Activity and Interaction Recognition

Human activity and interaction are often subjects of interest due to the large number of potential applications. Such data are typically collected from video. This gives rise to spatiotemporal characterizations that may be modeled using the product of Grassmann manifolds M_{Π}. The key idea here is to use three Grassmann manifolds to depict horizontal motion, vertical motion, and appearance [24], and consider the product topology. Thus, the setup for human activity and interaction recognition is similar to the Grassmann manifold where the distance measure $\| \cdot \|$ is chosen as $\| \sin\theta \|_2$. On the other hand, the feature patterns are obtained from the unfolded matrices of a video tensor. This operation can be achieved via a higher-order singular value decomposition [18] yielding three orthogonal matrices and may be imposed on three Grassmann manifolds. For more details of this realization refer to [24].

After extracting the feature patterns from video, regression is performed using three sub-least squares regression models $\Psi^{(j)}$, where j is the sub-regression model index [see Eq. (14.22)], on three Grassmann manifolds. The final geodesic distance is computed from the product manifold.

We have discussed the primary aspects of our regression framework on matrix manifolds for visual tracking, object categorization, and human activity and interaction recognition. Next, we will examine the effectiveness of the proposed framework.

14.6 Experiments

This section summarizes our empirical results and demonstrates the proficiency of the proposed regression framework on a number of computer vision applications including visual tracking, object categorization, activity recognition, and human interaction recognition. We evaluate our performance using four public data sets including the Dudek sequence [32], ETH-80 [20], UT-tower activity [33], and UT-interaction [33] data sets.

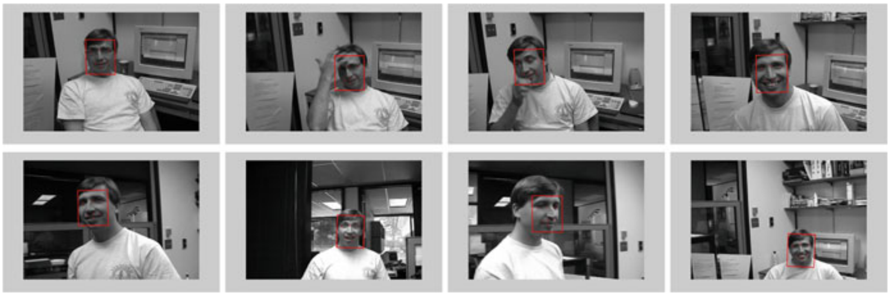


Fig. 14.1 Samples of tracking results using the proposed least squares regression framework on the matrix Lie group

Before discussing our experimental findings, we remind the readers that the notations of the matrix Lie group, SPD(n), Grassmann manifold, and product of Grassmann manifold are denoted as G_M, Sym_n^+, $\mathcal{G}_{n,p}$, and M_Π, respectively.

14.6.1 Visual Tracking

Visual tracking is one of the key preprocessing steps in video analysis. We demonstrate our regression framework on the Dudek video [32]. The video is acquired using a handheld camera exhibiting both camera motion and human motion concurrently. The Dudek video is annotated with seven hand-labeled ground-truth positions for 573 frames which facilitates quantitative comparison.

We model the underlying motion as affine transformations which are elements of G_M. Since our tracker is particle based, we benchmark our regression framework against IVT [32] using 600 particles on both methods. We select the top 50 particles for least squares fit due to speed considerations. The samples of tracking results are shown in Fig. 14.1 and the quantitative results are given in Fig. 14.2.[1] The average RMSE for IVT is 18.2 while our regression framework achieves 8.6. This result is encouraging because IVT exploits incremental learning with PCA whereas our tracker employs a standard PCA.

To further illustrate the effectiveness of our method, we test two variants in the matrix Lie group: Greedy and Lie group average. The Greedy method selects the particle with the highest probability resulting in 20.9 average RMSE. On the other hand, the Lie group average computes the average in the Lie algebra from the resampled population; so it is a weighted average resulting in 13.6 average RMSE. Thus, performing regression on the matrix Lie group yields a better result than the conventional scheme for visual tracking.

[1]Figure 14.2 is best viewed in color.

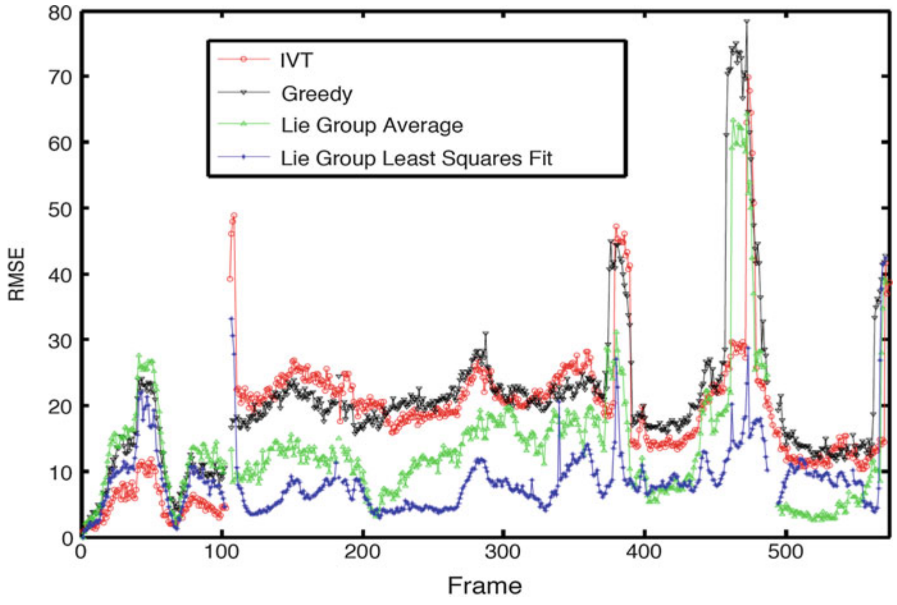


Fig. 14.2 Root mean square error (RMSE) on the Dudek video sequence

14.6.2 Object Categorization

Object categorization is an important research area in computer vision. We demonstrate the effectiveness of our regression framework using the ETH-80 data set. The ETH-80 data set contains eight categories of natural and artificial objects including apple, car, cow, cup, dog, horse, pear, and tomato. Each category consists of ten sets and each set has 41 orientations. Sample images are shown in Fig. 14.3. We follow the experimental protocol from [11] which is tenfold cross validation for image-set matching.

We assess the proposed regression framework on two matrix manifolds, Sym_n^+ and $\mathcal{G}_{n,p}$. First, all images are resized to 32×32. For Sym_n^+, we extract seven standard features $[x\ y\ R\ G\ B\ |I_x|\ |I_y|]$ to compute the covariance matrix. For $\mathcal{G}_{n,p}$, we use the grayscale pixels. The experimental results are given in Table 14.1. As Table 14.1 indicates, the least squares regression on $\mathcal{G}_{n,p}$ outperforms the one on Sym_n^+ and is comparable to state-of-the-art methods. We note that the performance of GDA (projective kernel) is superior on the ETH-80 data set, but the recognition result is selected from the best number of eigenvectors empirically.

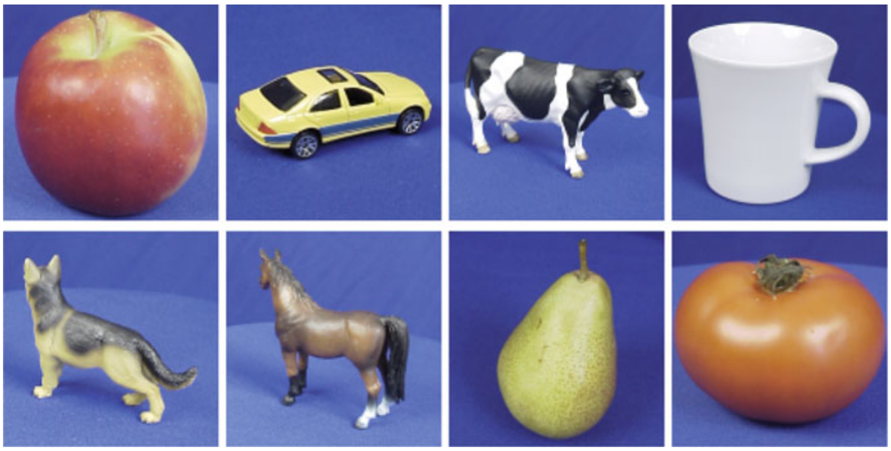


Fig. 14.3 Sample images (apple, car, cow, cup, dog, horse, pear, and tomato) on the ETH-80 data set

Table 14.1 Categorization results on the ETH-80 data set where the best result is highlighted in bold

Method	Accuracy (%)
GDA (projective kernel) [11]	**97.5**
GDA (Binet–Cauchy kernel) [11]	93.8
Discriminant CCA [17]	88.8
Our least squares regression on Sym_n^+	87.5
Our least squares regression on $\mathcal{G}_{n,p}$	95.0

14.6.3 Activity Recognition

Activity recognition has received attention in recent years due in part to its potential applications. We report the proficiency of our regression model on the product manifold M_Π using the UT-Tower activity data set. Unlike most activity data sets, the UT-tower activity data set is collected from a low-resolution aerial view camera presenting another avenue of challenge.

The UT-tower activity data set was used in the aerial view activity classification challenge which is a part of the semantic description of human activities (SDHA) challenge [33] in 2010. The data set comprises nine different activities performed 12 times by six different people. These video sequences exhibit extremely low resolution in which the average height of human figures is around 20 pixels. The data set comes with bounding boxes for localization as well as foreground masks for activities. All localized videos are resized to $32 \times 32 \times 24$ where the middle 24 frames from a video sequence are extracted due to the lower frame rate (10 fps). In our experiments, we do not exploit the foreground masks. Sample frames on the UT-tower activity data set are shown in Fig. 14.4.

The experimental protocol [33] for the UT-tower activity data set is the leave-one-out (LOO) cross validation. Table 14.2 reports our experimental results with

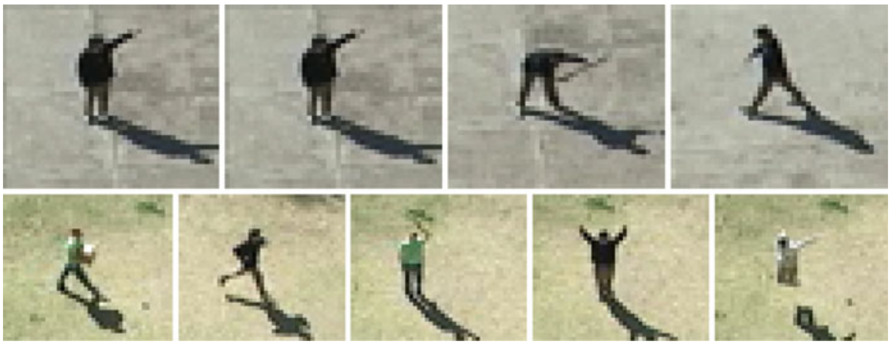


Fig. 14.4 Sample frames (stand, point, dig, walk, carry, run, wave1, wave2, and jump) on the UT-tower activity data set

Table 14.2 Recognition results on the UT-tower activity data set where the best result is highlighted in bold

Method	Accuracy (%)
Covariance manifold+sparsity (Guo et al. [10])	**97.2**
HMM+histogram (Vezzani et al. [39])	96.3
Hough-Voting (Waltisberg et al. [40])	95.4
Our least squares regression on M_{Π}	**97.2**

The benchmark results are obtained from the SDHA report [33]

three final contestants. Empirical results show that our method obtains 97.2 % recognition rate which is very competitive with the best result in this contest. While Guo's method [10] employs silhouette tunnels as feature representation, our feature patterns are extracted from a video tensor making it easier to apply. The confusion matrix given in Fig. 14.5 reveals that our method performs well across all activities.

14.6.4 Human Interaction Recognition

Another challenge of the SDHA [33] was the high-level human interaction recognition. While the aim of activity recognition is for single person recreation, human interaction focuses on recognizing activity that involves more than one person. The UT-interaction data set consists of six human interactions ranging from handshaking, hugging, kicking, and pointing to punching and pushing. The data set was further divided into two subsets: set 1 and set 2. For each subset, there are six interactions per sequence and a total of ten sequences. The interactions of set 1 are collected from mostly static backgrounds whereas the interactions of set 2 have more camera jittering. In addition, there is pedestrian interference in the scene

	Stand	Point	Dig	Walk	Carry	Run	Wave1	Wave2	Jump
Stand	12								
Point		12							
Dig			11	1					
Walk				12					
Carry					12				
Run						12			
Wave1							11	1	
Wave2							1	11	
Jump									12

Fig. 14.5 Confusion matrix on the UT-tower activities

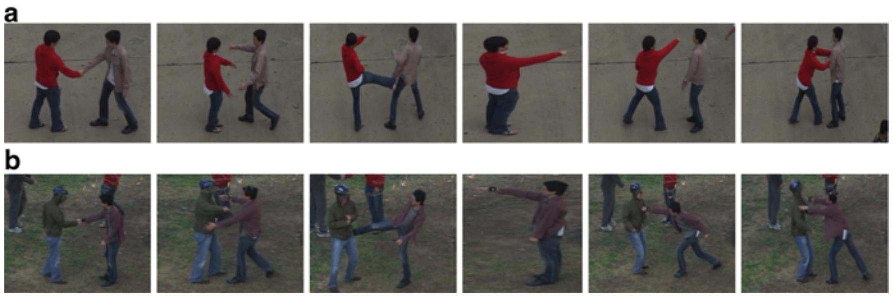


Fig. 14.6 UT-interactions where (**a**) and (**b**) show the sample frames from set 1 and set 2, respectively

during some interactions in the set 2 data set making it a more challenge data set (Fig. 14.6).

The UT-interaction data set provides segmented video clips containing one interaction per sequence. In both subsets, the experimental protocol is the LOO cross validation (tenfold cross validation). We resize all video sequences to $32 \times 32 \times 150$ because of the higher frame rate (30 fps). We extract the middle 150 frames for the longer sequences; otherwise, linear interpolation is applied to standardize the frame

Table 14.3 Recognition results on the UT-interaction data sets (set 1 and set 2)

Method	Accuracy (set 1) (%)	Accuracy (set 2) (%)
Hough-Voting (Waltisberg et al. [40])	88.3	76.7
Cuboid [5] + SVM (best)	85.0	70.0
Laptev [34] + SVM (best)	68.3	65.0
Our least squares regression on M_{Π}	91.7	75.0

The benchmark results are obtained from the SDHA report [33] where the (best) indicates the use of best codebook generated from ten trial runs

size. Since the interaction could occur either from left to right or from right to left, we also flip the video horizontally and find the nearest match.

The results of UT-interaction set 1 and set 2 are summarized in Table 14.3. Two baseline algorithms, Cuboid [5] + SVM and Laptev [34] + SVM, are the classic bag-of-features methods which employ cuboids and spatiotemporal features. The reported results were obtained from the best codebook generated from 10 trial runs. The best reported result for the UT-interaction challenge is the Hough-Voting method [40]. Table 14.3 reveals that our regression framework on M_{Π} also performs well on these data sets achieving 91.7 % recognition rate on set 1 and 75 % recognition rate on set 2. The proposed regression framework significantly outperforms the traditional bag-of-features models and is competitive with the Hough-Voting method. Our confusion matrices for set 1 and set 2 are given in Figs. 14.7 and 14.8, respectively. While our regression model performs perfectly for shaking, hugging, kicking, and pointing in set 1, there is more confusion in set 2 except for hugging and pointing. Similar findings have been reported from other methods. Further research is needed in such interaction recognition.

14.7 Discussion and Conclusions

The principle of our regression framework is based upon latent geometry in the data. Taking geometric aspects to the least squares regression framework, we formulate it as a composite function. Here, we require the computation of logarithmic and exponential maps associated with a given matrix manifold. We note that these differential operators for some spaces are easier to compute than for others. The proposed regression framework is applicable to many matrix manifolds as long as the logarithmic and exponential maps are well defined.

As far as statistical fitting is concerned, a regularization parameter may be augmented to the proposed regression framework. Such regularization provides shrinkage to reduce the effect of high variance of a statistical model easing the effect of overfitting. Furthermore, one of the key operators in our regression framework is the kernel operator which maps a distance to a similarity measure. While we employ the standard RBF kernel in our experiments, other kernels over matrix manifolds

	Shake	Hug	Kick	Point	Punch	Push
Shake	10					
Hug		10				
Kick			10			
Point				10		
Punch		1			7	1
Push		1			1	8

Fig. 14.7 Confusion matrix on the UT-interaction set 1

	Shake	Hug	Kick	Point	Punch	Push
Shake	6	2			1	1
Hug	1	9				
Kick			7		3	
Point				10		
Punch			2	1	7	
Push					4	6

Fig. 14.8 Confusion matrix on the UT-interaction set 2

can also be applied. In fact, inducing a valid Mercer kernel on a matrix manifold is an intriguing and important research topic. The generalization between Euclidean space and Riemannian manifolds via RBF kernels can be found in [15].

There are still many interesting matrix manifolds and applications that we have not discussed. We briefly summarize some recent developments here. High-dimensional data visualization is a crucial tool for data analysis. Fiori [7] employed the multidimensional scaling (MDS) technique to visualize data on Riemannian manifolds; in particular, SO(n), S^{n-1}, and Sym_n^+. The key notion of MDS is to reduce the dimensionality of patterns while keeping the proximity structures. The special orthogonal group SO(3) may also be utilized to characterize spherical displacement. Ma et al. [26] employed both SO(3) and so(3) to model the coplanar constraints which lead to a product of a Stiefel manifold. Newton's method is then performed to seek the optimal matrix for rigid motion recovery. Furthermore, visual dynamics has been depicted using geometric flow analysis. Lin et al. [21] consider motion of points over a spatial region and a temporal interval on a Lie algebra. By examining the underlying geometry, the flow of motion estimation is more consistent than optical flows. Recently, the characterization of deformations has been modeled using parallel transport. Wei et al. [41] approximated the deformations of an object on a matrix Lie group where the deformation is adapted via the parallel transport to the associated Lie algebra. Given so much active research focusing on matrix manifolds, we shall expect more fruitful results when the latent geometry is emphasized.

In conclusion, we have shown that our regression framework is generalizable to many matrix manifolds including matrix Lie groups, SPD(n), Grassmann, and product manifolds for visual tracking, object categorization, activity recognition, and human interaction recognition. While we employ fairly standard feature representations, the proposed least squares regression gives competitive performance on four public data sets. This realization is achieved by considering the underlying geometry of matrix manifolds. We conclude that the role of manifold geometry is vital for computer vision.

References

1. Absil PA, Mahony R, Sepulchre R (2008) Optimization algorithms on matrix manifolds. Princeton University Press, Princeton
2. Belinfante J, Kolman B (1972) A survey of lie groups and lie algebras with applications and computational methods. SIAM, Philadelphia
3. Conway J, Hardin R, Sloane N (1996) Packing lines, planes, etc.: packings in grassmannian spaces. Exp Math 5(2):139–159
4. Davis BC, Fletcher PT, Bullitt E, Joshi S (2007) Population shape regression from random design data. In: International conference on computer vision
5. Dollar P, Rabaud V, Cottrell G, Belongie S (2005) Behavior recognition via sparse spatio-temporal features. In: IEEE VS-PETS workshop, pp 65–72
6. Edelman A, Arias R, Smith S (1999) The geometry of algorithms with orthogonal constraints. SIAM J Matrix Anal Appl 2:303–353
7. Fiori S (2011) Visualization of riemannian-manifold-valued elements by multidimensional scaling. Neurocomputing 174:983–992
8. Fletcher PT (2011) Geodesic regression on riemannian manifolds. In: Third international workshop on mathematical foundations of computational anatomy
9. Gordon N, Salmond D, Smith A (1993) A novel approach to non-linear and non-gaussian Bayesian state estimation. IEE Proc F 140:107–113

10. Guo K, Ishwar P, Konrad J (2010) Action recognition using sparse representation on covariance manifold of optical flow. In: IEEE international conference on advanced video and signal-based surveillance, Boston
11. Hamm J, Lee D (2008) Grassmann discriminant analysis: a unifying view on subspace-based learning. In: International conference on machine learning, Helsinki, pp 376–383
12. Hastie T, Tibshirani R, Friedman J (2001) The elements of statistical learning: data mining, inference, and prediction. Springer, New York
13. He J, Balzano L, Szlam A (2012) Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In: IEEE conference on computer vision and pattern recognition
14. Ho H, Gopalan R (2014) Model-driven domain adaptation on product manifolds for unconstrained face recognition. Int J Comput Vis 109:110–125
15. Jayasumana S, Hartley R, Salzmann M, Li H, Harandi M (2015) Kernels on riemannian manifolds. In: Turaga P, Srivastava A (eds.) Riemannian computing in computer vision. Springer, New York (2015)
16. Karsten G, Hermann K (1973) How to conjugate c1-close group actions. Math Z 132:11–20
17. Kim TK, Kittler J, Cipolla R (2007) Discriminative learning and recognition of image set classes using canonical correlations. IEEE Trans Pattern Anal Mach Intell 29(6):1–14
18. Lathauwer LD, Moor BD, Vandewalle J (2000) A multilinear singular value decomposition. SIAM J Matrix Anal Appl 21:1253–1278
19. Lee J (2003) Introduction to smooth manifolds. Springer, New York
20. Leibe B, Schiele B (2003) Analyzing appearance and contour based methods for object categorization. In: IEEE conference on computer vision and pattern recognition
21. Lin D, Grimson E, Fisher J (2009) Learning visual flows: a lie algebraic approach. In: IEEE conference on computer vision and pattern recognition
22. Liu Y, Liu Y, Chan KCC (2011) Ordinal regression via manifold learning. In: Twenty-fifth AAAI conference on artificial intelligence, pp 398–403
23. Lui YM (2012) Advances in matrix manifolds for computer vision. Image Vis Comput 30:380–388
24. Lui YM (2012) Human gesture recognition on product manifolds. J Mach Learn Res 13:3297–3321
25. Lui YM, Beveridge JR, Kirby M (2010) Action classification on product manifolds. In: IEEE conference on computer vision and pattern recognition, San Francisco, pp 833–839
26. Ma Y, Kosecka J, Sastry S (1998) Motion estimation in computer vision: Optimization on stiefel manifolds. In: IEEE conference on decision and control, Tempa, pp 3751–3756
27. Meyer G, Bonnabel S, Sepulchre R (2011) Linear regression under fixed-rank constraints: a riemannian approach. In: International conference on machine learning
28. Pelletier B, Bataillon PE (2006) Non-parametric regression estimation on a closed riemannian manifold. J Nonparametric Stat **18**:57–67
29. Pennec X, Fillard P, Ayache N (2006) A riemannian framework for tensor computing. Int J Comput Vis 66(1):728–735
30. Pham DS, Venkatesh S (2008) Robust learning of discriminative projection for multicategory classification on the stiefel manifold. In: IEEE conference on computer vision and pattern recognition
31. Porikli F, Pan P (2009) Regressed importance sampling on manifolds for efficient object tracking. In: IEEE international conference on advanced video and signal-based surveillance
32. Ross D, Lim J, Lin RS, Yang MH (2008) Incremental learning for robust visual tracking. Int J Comput Vis 77:125–141
33. Ryoo MS, Chen CC, Aggarwal JK, Roy-Chowdhu A (2010) An overview of contest on semantic description of human activities (sdha). Http://cvrc.ece.utexas.edu/SDHA2010/
34. Schüldt C, Laptev I, Caputo B (2004) Recognizing human actions: a local svm approach. In: International conference on pattern recognition, Cambridge
35. Schwartz WR, Kembhavi A, Harwood D, Davis LS (2009) Human detection using partial least squares analysis. In: International conference on computer vision

36. Srivastava A, Klassen E (2004) Bayesian, geometric subspace tracking. Adv Appl Probab 36(1):43–56
37. Su J, Srivastava A, de Souza F, Sarkar S (2014) Rate-invariant analysis of trajectories on riemannian manifolds with application in visual speech recognition. In: IEEE conference on computer vision and pattern recognition
38. Tuzel O, Porikli F, Meer P (2008) Learning on lie groups for invariant detection and tracking. In: IEEE conference on computer vision and pattern recognition
39. Vezzani R, Baltieri D, Cucchiara R (2010) Hmm based action recognition with projection histogram features. In: Recognizing patterns in signals, speech, images and videos, pp 286–293
40. Waltisberg D, Yao A, Gall J, Gool LV (2010) Variations of a hough-voting action recognition system. In: Recognizing patterns in signals, speech, images and videos, pp 306–312
41. Wei D, Lin D, Fisher J (2012) Learning deformations with parallel transport. In: European conference on computer vision (2012)

Chapter 15
Domain Adaptation Using the Grassmann Manifold

David A. Shaw and Rama Chellappa

Abstract Modern data analysis is flush with large databases and high-dimensional inference problems. Often the size of these problems directly relates to the fact that given data may have variations that can be difficult to incorporate into well-known, classical methods. One of these sources of variation is that of differing data sources, often called domain adaptation. Many domain adaptation techniques use the notion of a shared representation to attempt to remove domain-specific variation in given observations. One way to obtain these representations is through dimension reduction using a linear projection onto a lower-dimensional subspace of the predictors. Estimating linear projections is intrinsically linked with parameters lying on the Grassmannian. We present some historical approaches and their relationship to recent advances in domain adaptation that exploit the Grassmannian through subspace estimation.

15.1 Introduction

Modern data sets are, almost by definition, big. Continuous streams of data result in a large number of data points being observed, while the increasing computing power and storage capacity allow us to take an increasing number of measurements of each data point. "Data deluge," "big data," and the "curse of dimensionality" characterize various ways in which analysis can be hindered by large databases, though these difficulties can lead to methods that handle many different types of inference problems. The large size of these modern data sets often indicates that data may come from multiple sources with differing distributions, and data may even be highly multimodal within a single source. Domain adaptation has come

D.A. Shaw (✉)
Department of Electrical and Computer Engineering,
Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: dashaw@andrew.cmu.edu

R. Chellappa
Department of Electrical and Computer Engineering, University of Maryland,
College Park, MD, USA
e-mail: rama@umiacs.umd.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_15

into view in recent years as an attempt to overcome the problems inherent in these differing variations in data sources and distributions.

Many classical problems relate to domain adaptation and big data. In the statistics literature, survey methodology [24] seeks to extrapolate inferences from small samples to the population as a whole. Sample selection bias [15] focuses on econometric problems, though tackles a general problem in assuming sampled observations are nonrandomly selected. Length-biased sampling [42] poses the problem of multiple data sources with a known, parametric difference between the distributions. Covariate shift [35] proposes importance weighting methods to improve predictive tasks when data may not arise from a single source. Meta-analysis [7], popular in many clinical fields, seeks methods to combine multiple inferences from potentially differing sources.

Unfortunately, many of these classical techniques were designed for univariate problems, and most big data problems are not only multivariate but also high dimensional. Additionally, we are not necessarily interested in estimating population totals from a subsample, but perhaps transporting model parameters to an entirely separate domain [29], and we do not wish to assume a large amount of data from any source. We explore methods of overcoming these problems through dimension reduction in the following sections.

15.2 High-Dimensional Inference

We assume access to features $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^p$, each independent and distributed as some random variable $\mathbf{X}$. Corresponding to each feature we have response values $y_1, \ldots, y_n \in \mathcal{Y}$ distributed as some random variable Y. We assume further that $\mathcal{Y}$ is the discrete set $\{1, \ldots, C\}$ for C a fixed, finite constant, meaning all inference is classification based.

We assume the features are high dimensional, that is, the dimension p is large, though in all cases we assume the features have a structured dependency of a much lower dimension than p. It is not often standard what a "large" p means specifically, and we only assume p is large in a qualitative sense, possibly with $p \gg n$ though not necessarily. In many cases, we say $\mathbf{X}$ comes from a *manifold* of dimension $d \ll p$ [39], which in much of the following we assume to be a lower-dimensional space in $\mathbb{R}^p$ that has the property of local linearity—that is, for any sufficiently small neighborhood around a point $\mathbf{x}$, all distances can be approximated using Euclidean distance.

We assume features arise from a globally linear manifold throughout, a strict assumption that aids in analysis and exposition. In other words, we seek a linear projection of points from the p-dimensional ambient space onto a d-dimensional subspace of $\mathbb{R}^p$. Often this assumption is too strict, and methods can be extended to overcome nonlinearity, though for purposes of illustration we operate under the linear assumption.

15.2.1 The Grassmannian

A slightly different notion of a manifold is that of the Grassmannian. As opposed to globally linear manifolds, the *Grassmannian* $\mathcal{G}(r, s)$ is a special manifold: the space of all s-dimensional subspaces of $\mathbb{R}^r$ for $r \geq s$. This intuitive formulation shows that the Grassmannian is useful to use as a parameter space when seeking lower-dimensional representations of high-dimensional data, as we see in the following sections.

Formally, $\mathcal{G}(r, s)$ is the quotient space

$$\mathcal{G}(r, s) = \mathcal{R}(r, s)/ \sim,$$

where $\mathcal{R}(r, s)$ is the space of all $r \times s$ matrices of rank s, and, for $\mathbf{U}, \mathbf{V} \in \mathbb{R}^{r \times s}$, $\mathbf{U} \sim \mathbf{V}$ if there exists a nonsingular $\mathbf{L} \in \mathbb{R}^{s \times s}$ such that $\mathbf{V} = \mathbf{U}\mathbf{L}$ [3]. As an example, note $\text{span}(\mathbf{A}) \in \mathcal{G}(r, s)$ if $\mathbf{A}$ is an $r \times s$ full rank matrix.

We define the underlying structure of $\mathcal{G}(r, s)$ as many methods seek to exploit it. We fix representations $\mathbf{Q}, \mathbf{R} \in \mathbb{R}^{r \times s}$, which can be thought of bases for s-dimensional subspaces of $\mathbb{R}^r$, with $\mathbf{Q}^\perp, \mathbf{R}^\perp \in \mathbb{R}^{r \times (r-s)}$ the orthogonal complements, and assume

$$\mathbf{Q}^\top \mathbf{Q} = \mathbf{I}_s,\ \mathbf{R}^\top \mathbf{R} = \mathbf{I}_s,\ \mathbf{Q}^\top \mathbf{Q}^\perp = \mathbf{0},\ \mathbf{R}^\top \mathbf{R}^\perp = \mathbf{0}.$$

We write the *geodesic flow* $\Phi : [0, 1] :\rightarrow \mathcal{G}(r, s)$ as [12]

$$\Phi(t; \mathbf{Q}, \mathbf{R}) = \mathbf{Q}\,\mathbf{U}_1\,\Gamma(t) - \mathbf{Q}^\perp\,\mathbf{U}_2\,\Sigma(t), \tag{15.1}$$

where $\mathbf{U}_1, \mathbf{U}_2, \Gamma$, and Σ are given by the generalized singular value decomposition

$$\mathbf{Q}^\top \mathbf{R} = \mathbf{U}_1\,\Gamma\,\mathbf{V}^\top,\ (\mathbf{Q}^\perp)^\top \mathbf{R} = -\,\mathbf{U}_2\,\Sigma\,\mathbf{V}^\top\,. \tag{15.2}$$

We define $\Gamma(t)$ and $\Sigma(t)$ as diagonal matrices with $\cos(t \cdot \theta_i)$ and $\sin(t \cdot \theta_i)$ on the diagonal for $i = 1, \ldots, s$ and Γ and Σ are $\Gamma(1)$ and $\Sigma(1)$, respectively.

The geodesic flow defines a path between two points on the Grassmannian that is related to the “shortest path” between these points. In general we may not always find a unique geodesic flow between two points, but in practical situations the above formulation yields a method for obtaining Φ.

Along with the geodesic flow, the Grassmannian structure admits another useful construction: the exponential and inverse exponential maps. The exponential map is a mapping from the tangent space of the Grassmannian about a point to a point on the manifold itself. To motivate the idea of the tangent space of $\mathcal{G}(r, s)$, we first look into the tangent space of a simpler ambient space: $\mathbb{R}^n$. We can think of the tangent space to $\mathbb{R}^n$ as the set of all pairs in $\mathbb{R}^n \times \mathbb{R}^n$ describing the locations and orientations of vectors in $\mathbb{R}^n$ [36]. For more exotic submanifolds $\mathcal{M}$ of $\mathbb{R}^n$, the tangent space is defined instead by collections of points on $\mathcal{M}$ with the associated tangent vectors in

$\mathbb{R}^d$ where d is the intrinsic dimension of $\mathcal{M}$. We can specifically think of the tangent space about a point $\mathbf{p} \in \mathcal{M}$ as the collection of all vectors tangent to $\mathcal{M}$ anchored at the point $\mathbf{p}$. The benefit of considering tangent spaces of manifolds is that they are Euclidean spaces, and thus standard operations such as addition can be defined and executed on them, a property most manifolds do not afford us.

Since $\mathcal{G}(r,s)$ is a manifold of dimension $s(r-s)$, we can think of the tangent space about a point on $\mathcal{G}(r,s)$ as a collection of vectors in $\mathbb{R}^{s(r-s)}$ centered at a point on $\mathcal{G}(r,s)$. Then we have

$$\exp(\cdot;\mathbf{p}) : T_{\mathbf{p}}\mathcal{G}(r,s) \to \mathcal{G}(r,s)$$

and

$$\exp^{-1}(\cdot;\mathbf{p}) : \mathcal{G}(r,s) \to T_{\mathbf{p}}\mathcal{G}(r,s),$$

where $T_{\mathbf{p}}\mathcal{G}(r,s) \subset \mathbb{R}^{s(r-s)}$ is the tangent space of our Grassmannian about a point $\mathbf{p}$. We can efficiently compute both exp and $\exp^{-1}$ using values obtained from (15.2) (see [14]). The exponential and inverse exponential maps are useful in that they operate on our assumption of local linearity of manifolds. We may now map points on a non-Euclidean object into standard Euclidean space, and this can simplify certain methods that require a Euclidean structure, e.g., assuming observations are Gaussian.

15.2.2 Inference in Computer Vision

The nature of images is inherently high dimensional with a lower-dimensional intrinsic structure. For example, naïve estimation using images as features could treat the grayscale level of each pixel as a separate feature, meaning for each observation we have a measurement from $[0,1]^p$. As an example, a relatively small 100×100 grayscale image indicates an ambient predictor dimension of 10,000, which can possibly be unwieldy for many classical approaches. See the left column of Fig. 15.2 in Sect. 15.4 for examples of 100×100 grayscale images. We could gain more information using color images, though this would increase the ambient dimension further, and video sequences become even more of a problem. Fortunately visual data typically have a much lower intrinsic dimension than ambient dimension. It is often problem specific how low this intrinsic dimension is [17, 28], but even the most complicated images of interest have a strong dependence between each pixel.

Structured Data

We consider a special case of visual data in which the structure of the data is known beforehand: features that lie on the previously defined Grassmannian. As an example, a set of r instances of two-dimensional landmark points contained in a matrix $\mathbf{A} \in \mathbb{R}^{r\times 2}$ can be used in various problems, such as age estimation [11, 40] and Procrustes analysis [8]. Unfortunately, a database of landmark points will typically have large amounts of unwanted variation, like rotations and shears. We can remove affine transformations of shape by considering instead an orthogonal basis for span($\mathbf{A}$) [13, 40]. In other words, for these points contained in $\mathbf{A}$, all affine transformations of shape can be obtained by right multiplication of $\mathbf{A}$ by a 2×2 full rank matrix $\mathbf{B}$. After normalization through a singular value decomposition so that $\mathbf{A}^\top \mathbf{A} = \mathbf{I}_2$, each set of landmark points will lie on $\mathcal{G}(r, 2)$—that is, the space of all two-dimensional subspaces of $\mathbb{R}^r$.

The FG-NET database is a typical source for benchmarking tasks like age estimation. The database consists of 1002 images of individuals' faces, as well as landmark points for each individual. Additionally, attributes such as an individual's age or gender are given for each observation. See Fig. 15.1 for examples. The landmark points in the bottom half of Fig. 15.1 represent a point on $\mathcal{G}(r, 2)$ and will be equivalent to the upright, rotated versions corresponding to the images in the top half.

For databases with observations lying on the Grassmannian, we can use the methods defined in the previous section to aid analysis. We solve the problem of prior structure by applying the inverse exponential map as described previously, that is,

$$\exp^{-1}(\cdot; \mu) : \mathcal{G}(r, 2) \to \mathbb{R}^{2(r-2)}$$

defined on the Grassmannian that allows mapping between $\mathcal{G}(r, 2)$ at a specified point $\mu \in \mathcal{G}(r, 2)$ and the corresponding tangent space.

One problem with this method is we must choose μ, often thought of as the "mean" of the given points. However, since $\mathbf{x}_1, \ldots, \mathbf{x}_n$ do not lie in Euclidean space, straightforward addition does not apply and we turn to the *Fréchet mean*. This "mean" is defined as the point $\hat{\mu}$ that satisfies (provided it exists)

$$\hat{\mu} = \operatorname*{argmin}_{\mu \in \mathcal{G}(r,2)} \frac{1}{n} \sum_{i=1}^{n} d^2(\mathbf{x}_i, \mu),$$

where $d^2(\cdot, \cdot) : \mathcal{G}(r, 2) \times \mathcal{G}(r, 2) \to \mathbb{R}^+$ is a distance function. Note however that d^2 requires representative projection matrices as input, thus implying an extrinsic approach to estimating $\hat{\mu}$. One extrinsic method involves fixing the distance function d^2 as

$$d^2(\mathbf{x}_i, \mu) = \operatorname{tr}\{\mathbf{I}_d - \mathbf{x}_i^\top \mu \, \mu^\top \mathbf{x}_i\},$$

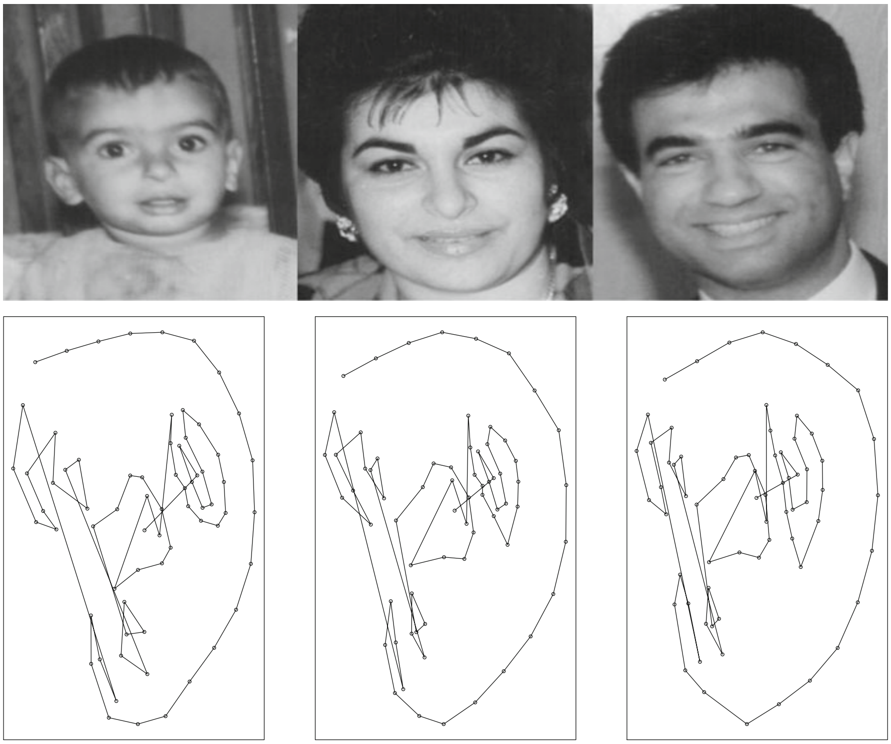


Fig. 15.1 Sample images from FG-NET originals (*top*) and landmark points (*bottom*). Images taken from [10]. The landmark points have had affine transformations factored out, and so each configuration corresponds to the same upright configuration

and in most cases we will use this to define a distance between two points on $\mathcal{G}(r, 2)$. For technical details, see [1, 3]. Similarly, one can obtain rough estimates of $\hat{\mu}$ by computing the sample mean directly and projecting to $\mathcal{G}(r, 2)$ through singular value decomposition or Gram–Schmidt procedures [30]. For data in a concentrated subset of $\mathcal{G}(r, 2)$, this can reduce computation while still obtaining a useful mean estimate. While all of these approaches detail extrinsic estimates of a mean value, one can calculate $\hat{\mu}$ intrinsically, e.g., through an iterative procedure [41].

15.2.3 Dimension Reduction

One of the main goals in assuming a manifold structure in the predictors is to find a lower-dimensional embedding or representation that can be used in analysis. Typically, we use linear and nonlinear dimension reduction methods to estimate these representations.

We define the data matrix of the features as

$$\mathbb{X} = \begin{bmatrix} \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_n^\top \end{bmatrix}.$$

Often features are mean centered to simplify certain analyses, though we shall not do so in order to outline specific properties of some techniques.

Principal component analysis (PCA, [19]) is a classical dimension reduction tool in which we seek a linear dimension reduction parameter $\eta \in \mathbb{R}^{p\times d}$, such that η projects the predictors into directions of maximum variation, i.e.,

$$\hat{\eta} = \arg\max_{\eta}\; \mathrm{tr}\{\eta^\top \Sigma^x \eta\}, \text{ such that } \eta^\top\eta = \mathbf{I}_d,$$

where $\Sigma^x = \mathbb{X}^\top \mathbb{C}\mathbb{X}$ is proportional to the covariance matrix of the data $\mathbb{X}$, $\mathbf{I}_d$ is the d-dimensional identity matrix, $\mathbb{C} = \mathbf{I}_n - \mathbf{1}\mathbf{1}^\top/n$ is a centering operator, and $\mathrm{tr}\{\cdot\}$ denotes matrix trace. Note that the constraint $\eta^\top\eta = \mathbf{I}_d$ assumes that η is a basis for a d-dimensional subspace of $\mathbb{R}^p$, effectively seeking coefficients η that lie on the Grassmannian $\mathcal{G}(p,d)$.

We can write the PCA estimate as a maximum likelihood estimator, which will enable us to generalize to other, potentially more useful methods. We pose the error model [4]

$$\mathbf{X} = \mu^x + \eta\,\nu + \epsilon,\ \epsilon \sim N(\mathbf{0}, \Delta), \tag{15.3}$$

where ϵ is a vector of random errors, ν are unknown coefficients, here equal to $\eta^\top(\mathbf{X} - \mu^x)$, and $\Delta > 0$ is a covariance matrix. Setting $\Delta = \sigma^2\,\mathbf{I}_p$, we see that the log-likelihood function for η is

$$L(\eta;\mathbb{X}) = -\frac{np}{2}\log(2\pi) - \frac{np}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}[\tilde{\mathbb{X}}^\top(\mathbf{I}_p - \eta\,\eta^\top)\tilde{\mathbb{X}}],$$

where $\tilde{\mathbb{X}}$ is the mean-centered version of $\mathbb{X}$. Suppressing constant terms we see that

$$\hat{\eta} = \arg\max_{\eta}\; L(\eta), \text{ such that } \eta^\top\eta = \mathbf{I}_d,$$

is equivalent to

$$\hat{\eta} = \arg\max_{\eta}\; \mathrm{tr}\{\eta^\top \Sigma^x \eta\}, \text{ such that } \eta^\top\eta = \mathbf{I}_d.$$

The estimate $\hat{\eta}$ here coincides with the PCA estimate above.

We are typically interested in the subspace $\mathrm{span}(\eta)$, with η simply a specific basis for this subspace. In this case, η can be thought of as an element of the Grassmannian

$\mathcal{G}(p,d)$. This fact is often used to construct various optimization problems for dimension reduction [4], and we write the above optimization compactly as

$$\hat{\eta} = \underset{\eta \in \mathcal{G}(p,d)}{\arg\max}\ \operatorname{tr}\{\eta^\top \Sigma^x \eta\}.$$

We note that, while PCA gives a useful initial estimate of a lower-dimensional subspace for our features, it makes a number of strict assumptions. For example, we assume no relation of the dimension reduction parameter η on the label variable Y, as well as the assumption that the dependence of $\mathbf{X}$ on η is linear. For this second assumption, we have many generalizations in the machine learning literature, LLE [27] and ISOMAP [38] being early instances. A number of these techniques are outlined in [21]. We analyze the first assumption in the following section.

Label-Dependent Dimension Reduction

Many techniques desire transformations that exploit label information, seeking lower-dimensional representations that are useful for classification problems. One such example of this is *linear discriminant analysis* (LDA, [26]), in which we seek

$$\hat{\eta} = \underset{\eta}{\arg\max}\ \operatorname{tr}\{\eta^\top \Sigma^{xb} \eta\},\ \text{such that } \eta^\top \Sigma^x \eta = \mathbf{I}_d,$$

as the solution to a generalized eigenvalue problem where Σ^{xb} is the "between-class" covariance matrix, that is,

$$\Sigma^{xb} = \sum_y (\mu_y^x - \mu^x)(\mu_y^x - \mu^x)^\top,$$

where μ_y^x is the within-class mean of $\mathbf{X}$ for class label y. This method is somewhat different from the previously outlined dimension reduction methods in that d can be no larger than $C-1$, meaning we are not necessarily finding a subspace that exploits the structure of $\mathbf{X}$ but rather estimating a predictive model based on $\mathbf{X}|Y$. Another key difference is that the constraint depends on the random variable $\mathbf{X}$, indicating our parameter does not directly lie on the Grassmannian as before. This can be seen in the modified case in which predictors are premultiplied by $\Sigma^{-1/2}$ and we seek

$$\hat{\eta} = \underset{\eta \in \mathcal{G}(p,d)}{\arg\max}\ \operatorname{tr}\{\eta^\top \tilde{\Sigma}^{xb} \eta\},$$

where

$$\tilde{\Sigma}^{xb} = \Sigma^{-1/2} \Sigma^{xb} \Sigma^{-1/2}.$$

However, note that in many high-dimensional cases Σ^x will not be full rank and so Σ^{-1} will be ill defined, requiring methods such as regularization or using a pseudoinverse.

Sliced inverse regression (SIR, [22]) attempts to incorporate response information through the within-class first moments. Estimates for η are obtained as the top d eigenvectors of $(\Sigma^x)^{-1}\mathbf{M}\mathbf{M}^\top$ where $\mathbf{M}$ is the $\mathbb{R}^{p\times C}$ matrix of within-class means. The dependence of the parameter estimate on the inverse of Σ^x is a problem in many high-dimensional problems and indicates SIR similarly has its constraint set depending on $\mathbf{X}$.

Sufficient dimension reduction [4] estimates η so that

$$\{Y|\eta^\top \mathbf{X}\} \sim \{Y|\mathbf{X}\}.$$

Likelihood-acquired directions (LAD, [5]), a type of sufficient dimension reduction method, are estimated by maximizing

$$L(\eta; \mathbb{X}, \mathbf{y}) = \frac{1}{2}\log|\eta^\top \Sigma^x \eta| - \frac{1}{2}\sum_{y=1}^{C}\frac{n_y}{n}\log|\eta^\top \Sigma^x_y \eta| \tag{15.4}$$

for $\eta \in \mathcal{G}(p,d)$, where $L(\eta)$ is proportional to a likelihood function, n_y is the number of observations in $\mathbb{X}$ with label y, Σ^x_y is the within-class covariance corresponding to label y, and $|\cdot|$ denotes the absolute value of the determinant. This optimization is done through conjugate gradient ascent on $\mathcal{G}(p,d)$, with details given in [9]. A benefit of this likelihood-based approach is that we can consider estimating parameters directly on the Grassmannian and do not have any constraints on the intrinsic dimension d, while estimating a lower-dimensional subspace that exploits the structure of $\mathbf{X}$ and incorporates information about the response.

15.3 Domain Adaptation Through Dimension Reduction

In many practical problems, we assume homogeneity of distributions between training and testing—that is, we assume both training and testing data come from the same distribution, or that our observations are identically distributed, perhaps conditional on some other variable. For example, parameter selection and error reporting are typically done by assuming we have training data $\mathbf{x}_1,\ldots,\mathbf{x}_n$ to estimate model parameters with additional data $\mathbf{x}^*_1,\ldots,\mathbf{x}^*_m$ to use for testing. We could use $\mathbf{x}^*_i$ for selecting optimal parameters by finding those parameters that minimize some cross-validation criterion; we could similarly use these test data to report how well a method can predict response values for unseen data. In most classical settings, we assume that both data $\mathbf{x}_i$ and $\mathbf{x}^*_j$ are distributed similarly, namely as the random variable $\mathbf{X}$. Often in practical situations the distributions between the training and testing phases will not be homogeneous, which can result in poor predictive performance and present difficulties in determining optimal tuning parameters through cross-validation.

Problems in which there exist discrepancies between training and testing data are common and have seen a lot of recent attention. Most domain adaptation problems in computer vision either use the covariate shift assumption [6, 35] or the transfer learning assumption [25]. The methods presented here attempt to solve the domain adaptation problem operating under the covariate shift assumption where we assume the joint distribution between the labels Y and features $\mathbf{X}$ changes from training to testing by way of the marginal distribution of the features. In other words, we assume that while the feature distribution might change between two domains, the distribution $Y|\mathbf{X}$ does not. Several approaches have been devised to tackle this problem, such as methods seeking features invariant under certain domain shifts [2, 23, 34], metric learning methods [20, 29], and dictionary methods [33]. Many methods in domain adaptation are closely linked to the dimension reduction approaches that have been outlined previously in that they attempt to estimate a "shared" or "common" representation in which the distribution of the lower-dimensional features is similarly distributed.

In domain adaptation literature, data from $\mathbf{X}$ is known as "source" or "training" data, while data from $\mathbf{X}^*$ is known as "target" or "testing" data. As we assume knowledge of some data from $\mathbf{X}^*$ for training, we call the distribution $\mathbf{X}^*$ the "target" distribution. Often interest lies in adapting to multiple different domains, a problem which we do not consider presently, but one that is a relatively straightforward extension.

Many domain adaptation methods that do not use dimension reduction techniques attempt to bring the source and target distributions together through instance weighting [18, 35, 37]. These importance weighting methods have issues in handling source and target data that arise from potentially different underlying structures. For example, the "covariate shift" approach assumes that $\mathrm{supp}(\mathcal{X}^*) \subset \mathrm{supp}(\mathcal{X})$, which is often not the case in computer vision problems. Presently we only consider dimension reduction techniques for domain adaptation.

15.3.1 Intermediate Subspaces

The *intermediate subspace approach* (IS, [14]) seeks a latent feature representation by obtaining intermediate feature spaces that help to quantify the shift from the source to the target space. In IS, the latent variables are obtained by sampling points along a geodesic on the Grassmannian $\mathcal{G}$ between the d-dimensional subspace spanned by the source data set and the d-dimensional subspace spanned by the target data set. First, we estimate descriptive subspaces for both the source and the target data sets; this is done through PCA. Using this method, we obtain an element of the Grassmannian $\mathcal{G}(p, d)$ for each data source, called η_{x} and η_{x^*}.

Given these two elements of $\mathcal{G}(p, d)$, we recall the geodesic flow

$$\Phi(t; \eta_{\mathrm{x}}, \eta_{\mathrm{x}^*}) = \eta_{\mathrm{x}} \mathbf{U}_1 \Gamma(t) - \eta_{\mathrm{x}}^{\perp} \mathbf{U}_2 \Sigma(t). \tag{15.5}$$

We then use Φ to sample points along the "path" between the source and target subspaces. Though different sampling strategies can be used, we take a uniformly spaced sampling along Φ.

After we sample a fixed number of intermediate subspaces, we concatenate all representations into one overall representation. For example, if we sample $\eta_1, \dots, \eta_k$ from Φ, we create the overall representation as

$$\tilde{\eta} = [\eta_{\mathrm{x}} \; \eta_1 \; \dots \; \eta_k \; \eta_{\mathrm{x}^*}] \in \mathbb{R}^{p \times d(k+2)}.$$

We perform *partial least squares* (PLS, [43]) on the extrapolated data using $\tilde{\eta}$ to obtain a low-dimensional model operating on these expanded data sets.

The IS method suffers from a few drawbacks, the main one being that the method of sampling intermediate subspaces is not necessarily optimal. Moreover, one must choose a large number of tuning parameters, such as the number of intermediate subspaces, the intrinsic dimension, and the initial representations η_{x} and η_{x^*}. Additionally, extrapolating the source and target data results in a high dimensionality that must be overcome through PLS.

15.3.2 Geodesic Flow Kernel

Geodesic flow kernel (GFK, [12]) is an extension of the IS method. It attempts to remove the need for the uniform sampling along the geodesic between the source and target subspaces. In this case, we again take the geodesic flow between η_{x} and η_{x^*} as in (15.5). However, instead of sampling directly from Φ, we wish to use all $t \in (0, 1)$. Unfortunately, this is computationally infeasible, so we instead proceed through a kernel by writing

$$< \mathbf{u}, \mathbf{v} >_{\Phi} = \int_0^1 (\Phi(t)^{\top} \mathbf{u})^{\top} (\Phi(t)^{\top} \mathbf{v}) \, dt = \mathbf{u}^{\top} \mathbf{G} \, \mathbf{v}.$$

Here, $\mathbf{G} \in \mathbb{R}^{p \times p}$ is positive semi-definite and by definition taken as

$$\mathbf{G} = \int_0^1 \Phi(t) \Phi(t)^{\top} \, dt.$$

We can define $\mathbf{G}$ with the elements obtained from the generalized singular value decomposition in (15.2).

We perform prediction using $\mathbf{G}$ in a kernel nearest-neighbor classifier, i.e., we use $< \cdot, \cdot >_{\Phi}$ as a distance metric in a standard nearest-neighbor classifier. We take as the label for a single test point $\mathbf{x}^*$ as the label corresponding to $\hat{\mathbf{x}}$ taken as

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \; < \mathbf{x}^*, \mathbf{x} >_\Phi$$

using our data-dependent kernel.

GFK, similar to IS, is highly dependent on an initial choice of η_{x} and η_{x^*}, which in practice is typically estimated through PCA or PLS, depending on whether or not label information is available. Additionally, it is possible that the geodesic between these two subspaces may not be the optimal choice for inference tasks.

15.3.3 Combined Direction Estimation

Combined direction estimation (CDE, [31]), another extension of the IS method, seeks a model-based approach that can incorporate a number of useful constructions into the method, such as directly including the conditional distribution, assuming more general distributions on the subspaces (as opposed to the Gaussian distribution assumed by the initial PCA), and including additional penalties on the lower-dimensional transformation (e.g., those that encourage sparsity).

We seek the transformation η as the solution to the regularized optimization problem

$$\underset{(\beta,\eta)}{\operatorname{argmin}} \; E_{Y,\mathbf{X}} \mathcal{L}(Y, \eta^\top \mathbf{X}; \beta) + \mu_0 \cdot J(\eta),$$

where $\mathcal{L}$ is a loss function depending on η and a model parameter $\beta \in \mathbb{R}^d$, μ_0 is a positive regularization parameter, and

$$J(\eta) = -\frac{1}{2}\mathrm{tr}\left\{ \frac{1}{\sigma_{x*}^2} \Sigma^x \, \eta \, \eta^\top + \frac{1}{\sigma_x^2} \Sigma^{x*} \, \eta \, \eta^\top \right\}. \tag{15.6}$$

Here, J is obtained through placing a normal model on both random variables $\mathbf{X}$ and $\mathbf{X}^*$ and taking the Kullback–Leibler divergence between the two distributions.

We can extend this method to incorporate local structure through assuming a nonparametric density for the random errors. For example, in (15.6) we make the assumption that the data come from a normal distribution and take the KL-divergence. A nonparametric approach assumes instead that the error model in (15.3) can be replaced with a Nadaraya–Watson-style kernel estimate as opposed to a normal distribution. We achieve computational speedups by assuming a mixed Gaussian framework for a fixed, small number of mixtures instead of an entirely nonparametric one.

We can use this method in cases where data have a prior structure. Here, a normal distribution is not conformable, and we consider a matrix Langevin error model

[3] instead of a Gaussian, as it operates directly on Grassmannian data points. We assume the data are generated from

$$f(\mathbf{X};\mu,k_x)=a(k_x)\cdot \mathrm{etr}\{k_x\cdot \mu^{x\top}\mathbf{X}\},$$

where etr$\{\cdot\}$ denotes exp[tr$\{\cdot\}$], $k_x\in\mathbb{R}$ is a scale parameter, and μ^x is a mean point, estimated through methods outlined in section "Structured Data". Most alternative methods cannot directly handle data on the Grassmannian, and first map all points to the tangent space about a fixed point in order to use standard Euclidean methods.

The CDE method overcomes some of the drawbacks from IS and GFK but still has a number of its own issues. As there is no closed-form expression to estimate η, we must perform gradient ascent on $\mathcal{G}(p,d)$ [9]. Unfortunately, this is not always guaranteed to converge to a global maximum and can potentially be affected by poor initial estimates. CDE also requires a large number of tuning parameters, including the additional regularization parameter μ_0.

15.3.4 Regularized Likelihood Directions

Regularized likelihood directions (RLD, [32]) is a modified approach to LAD (see section "Label-Dependent Dimension Reduction") that is useful in cases where the dimension between the source and target distribution changes [20]. It seeks η and γ so that

$$(Y,\eta^\top\mathbf{X})\sim(Y,\gamma^\top\mathbf{X}^*).$$

We attempt to estimate subspaces in which the joint distributions of the labels and features are similarly distributed between source and target in the reduced space, similar to most other dimension reduction techniques for domain adaptation.

We make a distributional assumption on the within-class observations, namely that for each label y the features $\mathbf{X}$ are Gaussian with differing means and covariances per class. We then recall the likelihood $L(\eta)$ in Eq. (15.4) and proceed by estimating

$$\underset{(\eta,\gamma)}{\arg\max}\ L(\eta)+\Gamma_\lambda(\eta,\gamma),$$

where

$$\Gamma_\lambda(\eta,\gamma)=\frac{\lambda}{2}\sum_{y=1}^{C}||\eta^\top\mu_y^x-\gamma^\top\mu_y^{x*}||^2$$

for a fixed $\lambda > 0$ and μ_y^x and μ_y^{x*} are the means for $\mathbf{X}$ and $\mathbf{X}^*$ given class y. As we do not assume our within-class covariances are equal, ideally we would want to constrain the second moments as well; however, in practice these constraints often do not yield any improvements in estimation. Additionally, though we define Γ as requiring within-class target means, we can simply replace μ_y^{x*} with μ^{x*} for unsupervised estimation.

RLD is similar to CDE—both are penalized maximum likelihood approaches—as well as KMM [18] as it attempts to constrain within-class means to be similar to one another. RLD can handle cases with nonlinearity by localizing about a given observation and constructing local versions of the means and covariances, though this approach is computationally intensive.

Drawbacks to RLD are similar to those from CDE. As we depend on optimization along the Grassmannian, we must rely on an iterative method for estimating optimal parameters, and the number of tuning parameters is still large. Additionally, now we must estimate two parameters of interest, η and γ, as opposed to the single η. While this allows the method more flexibility in handling features of differing dimension, it also requires more data as a greater number of parameters must be estimated.

15.4 Illustration

We illustrate the outlined domain adaptation techniques on two data sets.

15.4.1 Object Recognition

The object recognition data set from [29] comprises 4110 images of 31 object classes, where each image comes from one of three data sources: `amazon.com`, DSLR captures, and webcam captures. We use standard feature extraction methods to obtain histograms of oriented gradients (HOG) with eight bins on 8×8 patches, then standardize features to have zero mean and standard deviation one. While we can obtain better performance with more bins [12] or with domain adaptation-specific features from the data [16], all of our techniques assume a similar difference in distribution between the source and target observations; we use these features as a demonstration of the relative value of each method. We also consider the above task where we rescale each observation to a 10×10 grayscale image for the target data set. We give example realizations in Fig. 15.2.

We randomly sample 20 observations per class from the source data and three observations per class from the target, both with replacement, and replicate the study 10 times, reporting the mean improvement of the classification rate over PCA. All methods except GFK use a least-squares classifier, as it is computationally efficient and yields a general model [31]. For GFK, we use one nearest neighbor as suggested by the authors and outlined in Sect. 15.3.2.

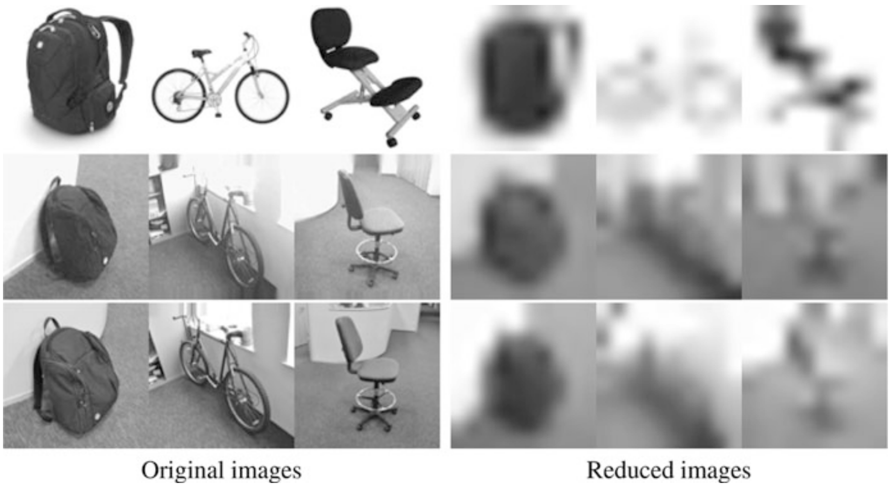


Fig. 15.2 Sample images from `amazon.com` (*top*), webcam (*middle*), and a DSLR camera (*bottom*). Reduced images are scaled to 100 × 100 from 10 × 10 for visualization. Images taken from [29]

Table 15.1 Object recognition results, source: HOG features, target: HOG features

	A:W	A:D	W:A	W:D	D:A	D:W
IS	2.7821	1.1272	1.9546	2.7127	1.4456	4.4222
GFK	1.6028	0.5742	1.9092	3.7403	1.4630	8.0992
CDE	2.9466	1.1705	2.0048	2.4862	1.4116	3.7660

Numbers reported are the relative improvement over PCA. All results are on unseen target data. Here, A:W denotes `amazon.com` as source and webcam as target, A:D denotes `amazon.com` as source and DSLR as target, etc.

For both source and target as HOG data, we provide results in Table 15.1. We see that all methods are able to gain improvements over PCA, with some methods performing better than others in different cases. As RLD is designed to handle cases in which the source and target dimensions differ, we do not estimate it in this example. We see the largest improvements in the DSLR-to-webcam study, which is reasonable as the data have similar backgrounds and lighting, with DSLR data having a higher resolution.

When the dimension between source and target domains differs, in all methods except RLD, we use PCA to obtain features that have the same dimension, and then use the domain adaptation methods as described. Table 15.2 shows that this preprocessing can hinder many approaches, yielding results that are in fact worse than simply using PCA on the source and target. RLD, however, shows consistent improvement over PCA in all cases, most likely because of its construction of two separate transformations and attempt to exploit label information throughout estimation.

Table 15.2 Object recognition results, source: HOG features, target: raw image data

	A:W	A:D	W:A	W:D	D:A	D:W
IS	0.2479	−0.2287	−0.0714	−0.1572	−0.0275	−0.1026
GFK	0.1538	−0.1741	−0.0159	−0.2174	−0.0296	−0.0064
CDE	0.1880	−0.0887	0.0760	−0.3445	−0.0137	−0.4359
RLD	1.3846	1.3823	1.7120	1.2475	1.6015	0.9872

Numbers reported are the relative improvement over PCA. All results are on unseen target data. Here, A:W denotes `amazon.com` as source and webcam as target, A:D denotes `amazon.com` as source and DSLR as target, etc.

Table 15.3 Age classification results

	Same dimension		Different dimension	
	Source: Centered	Source: Rotated	Source: Centered	Source: Rotated
IS	0.5933	0.5371	−0.0713	0.1112
GFK	0.3022	0.2023	−0.0491	0.0567
CDE	0.5968	0.5447	0.2514	0.2458
RLD	—	—	0.1575	0.2821

Numbers reported are the relative improvement over PCA. All results are on unseen target data

15.4.2 Age Classification

We perform a similar study to the one above on the previously defined landmark point data set, examples given in Fig. 15.1. This data set is interesting because of its prior Grassmannian structure, and in order to apply many standard techniques we must first map points to a particular tangent space. We again randomly sample with replacement 20 observations per class from the source data and three observations per class from the target, where classes here have been defined by thresholding an individual's age at the one-third and two-thirds quantiles. We replicate the study 10 times and report the relative performance in classification rate over PCA in Table 15.3.

The first column of Table 15.3 corresponds to source data as landmark points and target data as the landmark points artificially rotated. These artificial rotations can be removed using methods described in section "Structured Data", but for illustration we assume they have not yet been factored out. The second column is the reverse—that is, artificially rotated points are used as source data. The third and fourth columns correspond to the same study, though with the artificially rotated data having one-fourth of landmark points removed, yielding a case of differing dimension. We see all methods perform well when compared with PCA, with the exception of IS and GFK in the first case where the dimension between source and target differs, and no method consistently outperforms all others.

15.5 Discussion

We have outlined a variety of techniques for reducing the dimension of high-dimensional data and extensions of those techniques to the problem of domain adaptation. All techniques are inherently related to the Grassmannian in that they attempt to estimate low-dimensional subspaces of high-dimensional data. Moreover, some data in computer vision problems themselves lie on the Grassmannian, requiring similar techniques. We use the Grassmannian to exploit the structure of parameter spaces for domain adaptation problems and can see improvements over classical dimension reduction techniques. The Grassmannian itself is useful mainly as a parameterization of subspaces—that is, the structure of the Grassmannian is typically seen as a technical hurdle to overcome for specific optimization tasks. We see this in both the classical label-based dimension reduction techniques and the more recent domain adaptation methods, and considering this underlying structure yields better results on some visual classification tasks.

References

1. Bhattacharya R, Patragenaru V (2003) Large sample theory of intrinsic and extrinsic sample means on manifolds I. Ann Stat. 31(1):1–29
2. Blitzer J, McDonald R, Pereira F (2006) Domain adaptation with structural correspondence learning. In: Proceedings of the 2006 conference on empirical methods in natural language processing - EMNLP ‘06. Association for Computational Linguistics, July 2006
3. Chikuse Y (2003) Statistics on special manifolds. Lecture notes in statistics. Springer, Berlin
4. Cook RD, Adragni KP (2009) Sufficient dimension reduction and prediction in regression. Philos Trans R Soc A Math Phys Eng Sci 367(1906):4385–4405
5. Cook RD, Forzani L (2009) Likelihood-based sufficient dimension reduction. J Am Stat Assoc 104(485):197–208
6. Daumé III H (2007) Frustratingly easy domain adaptation. In: Annual meeting—association for computational linguistics, vol 45, pp 256–263
7. DerSimonian R, Laird N (1986) Meta-analysis in clinical trials. Control Clin Trials 7(3): 177–188
8. Dryden IL, Mardia KV (1998) Statistical shape analysis. Wiley, New York
9. Edelman A, Arias TA, Smith ST (1998) The geometry of algorithms with orthogonality constraints. SIAM J Matrix Anal Appl 20:303–353
10. FG-NET Aging Database (2011) Face and gesture recognition research network. http://www.fgnet.rsunit.com/. Accessed April 2011
11. Fu Y, Huang TS (2008) Human age estimation with regression on discriminative aging manifold. IEEE Trans Multimedia 10(4):578–584
12. Gong B, Shi Y, Sha F, Grauman K (2012) Geodesic flow kernel for unsupervised domain adaptation. In: IEEE conference on computer vision and pattern recognition
13. Goodall CR, Mardia KV (1999) Projective shape analysis. J Comput Graph Stat 8(2):143–168
14. Gopalan R, Li R, Chellappa R (2011) Domain adaptation for object recognition : an unsupervised approach. In: International conference on computer vision
15. Heckman JJ (1979) Sample selection bias as a specification error. Econometrica J Econ Soc 47(1):153–161

16. Ho HT, Gopalan R (2014) Model-driven domain adaptation on product manifolds for unconstrained face recognition. Int J Comput Vis 109(1–2):110–125
17. Huang J, Mumford DB (1999) Statistics of natural images and models. In: IEEE Computer Society (ed) Proceedings of the IEEE computer society conference on computer vision and pattern recognition, vol 1. IEEE Computer Society Press, Los Alamitos, CA, pp 541–547
18. Huang J, Smola AJ, Gretton A, Borgwardt KM, Schölkopf B (2007) Correcting sample selection bias by unlabeled data. Adv Neural Inf Process Syst 19:601
19. Jolliffe IT (2005) Principal component analysis. Wiley Online Library, New York
20. Kulis B, Saenko K, Darrell T (2011) What you saw is not what you get: domain adaptation using asymmetric kernel transforms. In: IEEE conference on computer vision and pattern recognition
21. Lee JA, Verleysen M (2007) Nonlinear dimensionality reduction. Springer, Berlin
22. Li K-C (1991) Sliced inverse regression for dimension reduction. J Am Stat Assoc 86(141): 316–327
23. Liu J, Shah M, Kuipers B, Savarese S (2011) Cross-view action recognition via view knowledge transfer. In: IEEE conference on computer vision and pattern recognition
24. Mahalanobis PC (1944) On large-scale sample surveys. Philos Trans R Soc Lond B 231:329–451
25. Pan SJ, Yang Q (2010) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10): 1345–1359
26. Radhakrishna Rao C (1948) The utilization of multiple measurements in problems of biological classification. J R Stat Soc Ser B 10(2):159–203
27. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290:2323–2326
28. Ruderman DL, Bialek W (1994) Statistics of natural images: scaling in the woods. Phys Rev Lett 73(6):814–817
29. Saenko K, Kulis B, Fritz M, Darrell T (2010) Adapting visual category models to new domains. In: Daniilidis K, Maragos P, Paragios N (eds) European conference on computer vision 2010. Lecture notes in computer science. Springer, Berlin/Heidelberg, pp 213–226
30. Shaw DA, Chellappa R (2013) Regression on manifolds using data-dependent regularization with applications in computer vision. Stat Anal Data Min. (Special issue: joint statistical meetings 2012) 6(6):519–528
31. Shaw DA, Chellappa R (2015) Combined direction estimation for dimension reduction in the presence of inhomogeneous data. J Am Stat Assoc. Under review
32. Shaw DA, Chellappa R (2014) Domain adaptation for robust pattern recognition. In: Information theory and applications workshop (ITA), 2014, pp 1–7, Feb 2014
33. Shekhar S, Patel V, Nguyen H, Chellappa R (2013) Generalized domain-adaptive dictionaries. In: IEEE conference on computer vision and pattern recognition
34. Shi Y, Sha F (2012) Information-theoretical learning of discriminative clusters for unsupervised domain adaptation. In: International conference on machine learning
35. Shimodaira H (2000) Improving predictive inference under covariate shift by weighting the log-likelihood function. J Stat Plann Inference 90(2):227–244
36. Spivak M (1999) A comprehensive introduction to differential geometry vol 1, 3rd edn. Publish or Perish Inc., Houston, TX
37. Sugiyama M, Suzuki T, Nakajima S, Kashima H, von Bünau P, Kawanabe M (2008) Direct importance estimation for covariate shift adaptation. Ann Inst Stat Math 60(4):699–746
38. Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science (New York, NY) 290(5500):2319–23
39. Thorpe JA (1979) Elementary topics in differential geometry. Springer, New York
40. Turaga P, Biswas S, Chellappa R (2010) The role of geometry for age estimation. In: IEEE international conference on acoustics, speech and signal processing. University of Michigan Library, Ann Arbor, pp 946–949

41. Turaga PK, Veeraraghavan A, Srivastava A, Chellappa R (2011) Statistical computations on grassmann and stiefel manifolds for image and video-based recognition. IEEE Trans Pattern Anal Mach Intell 33(11):2273–2286
42. Vardi Y (1982) Nonparametric estimation in the presence of length bias. Ann Stat 10(2): 616–620
43. Wold H (1985) Partial least squares. In: Kotz S, Johnson N (eds) Encyclopedia of statistical sciences. Wiley, New York, pp 581–591

Chapter 16
Coordinate Coding on the Riemannian Manifold of Symmetric Positive-Definite Matrices for Image Classification

Mehrtash Harandi, Mina Basirat, and Brian C. Lovell

16.1 Introduction

Over the years, **coding**—in its broadest definition—has proven a crucial step in visual recognition systems [4, 7]. Many techniques have been investigated, such as bag of words [1, 9, 16, 18, 19, 31], sparse coding [21, 34], and locality-based coding [33, 35]. All these techniques follow a similar flow: Given a dictionary of code words, a query is associated to one or multiple dictionary elements with different weights (i.e. let@tokeneonedot, binary or real). These weights, or *codes*, act as the new representation for the query and serve as input to a classifier (i.e., support vector machine (SVM)) after an optional pooling step.

This work introduces techniques to perform coding on symmetric positive-definite (SPD) matrices. More specifically, unlike traditional sparse coding schemes that work on vectors, in this study we discuss how SPD matrices can be described by combination of dictionary atoms, where the atoms are also SPD matrices. Our motivation stems from pervasive role of SPD matrices in machine learning, computer vision, and related areas. For example, SPD matrices have been used in

M. Harandi (✉)
NICTA, Canberra and College of Engineering and Computer Science,
Australian National University, ACT 2601, Australia
e-mail: mehrtash.harandi@nicta.com.au

M. Basirat
Faculty of Electrical, Computer and IT Engineering,
Qazvin Islamic Azad University, Qazvin, Iran

B.C. Lovell
School of Information Technology and Electrical Engineering,
University of Queensland, Brisbane, QLD 4072, Australia
e-mail: lovell@itee.uq.edu.au

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_16

medical imaging, texture classification [10–12, 29], action recognition, and gesture categorization [27], as well as face recognition [10, 22].

Extending coding methods to SPD matrices is not trivial, as such matrices form the interior of the positive-semidefinite cone. In other words, simply vectorizing SPD matrices and employing Euclidean geometry (e.g., Euclidean norms) do not lead to accurate representations [15, 23, 30]. To overcome the drawbacks of Euclidean structure, Pennec et al. [23] introduced a Riemannian structure, referred to as SPD or tensor manifold, to analyze SPD matrices. Explicitly taking into account the geometry of SPD manifolds can be highly beneficial for discrimination ability [10, 12, 15, 23, 30].

In this work, we extend the notion of coordinate coding [35] to the SPD manifolds. In coordinate coding, nearby atoms to a query determine the coding weights. This as discussed in [33, 35] can result in sparsity (which is widely used for coding images and videos). To this end, we propose an intrinsic solution to perform coordinate coding on the SPD manifolds. Interestingly, the proposed coding scheme, unlike sparse coding, has a closed-form solution. In an attempt to reduce the computation load of the intrinsic method, we propose to flatten the SPD manifold prior to coding. We consider two types of flattening here. First, we use the tangent space of the manifold to flatten the manifold. Second, we propose to embed the SPD manifold in an infinite-dimensional reproducing kernel Hilbert space (RKHS) by exploiting two types of the Bregman divergences.

We continue this chapter as follows. Section 16.2 briefly reviews the geometry of SPD manifolds, the Bregman divergences and their properties. Section 16.3 which elucidates various coordinate coding schemes can be performed on the SPD manifolds. This includes the intrinsic, log-Euclidean, and kernel coordinate coding (kCC). In Sect. 16.4 the performance of the proposed methods is assessed on the task of classifying face images. The main findings and possible future directions are presented in Sect. 16.5.

16.2 Riemannian Geometry of SPD Manifolds

In this section, we discuss some notions of geometry of SPD manifolds. Throughout this chapter we will use the following notation: $\mathcal{S}_{++}^n$ is the space of real $n \times n$ SPD matrices; $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ is the identity matrix; $\mathrm{GL}(n)$ is the general linear group, i.e., the group of real invertible $n \times n$ matrices. The principal matrix logarithm $\log(\cdot) : \mathcal{S}_{++}^n \to \mathrm{Sym}(n)$ is defined as

$$\log(\boldsymbol{X}) = \sum_{r=1}^{\infty} \frac{(-1)^{r-1}}{r} (\boldsymbol{X} - \mathbf{I}_n)^r = \boldsymbol{U}\mathrm{Diage}\left(\ln(\lambda_i)\right)\boldsymbol{U}^T, \tag{16.1}$$

with $\boldsymbol{X} = \boldsymbol{U}\mathrm{Diage}\left(\lambda_i\right)\boldsymbol{U}^T$. The principal matrix exponential $\exp(\cdot) : \mathrm{Sym}(n) \to \mathcal{S}_{++}^n$ is defined as

$$\exp(\boldsymbol{X}) = \sum_{r=0}^{\infty} \frac{1}{r!}\boldsymbol{X}^r = \boldsymbol{U}\text{Diage}\left(\exp(\lambda_i)\right)\boldsymbol{U}^T, \tag{16.2}$$

with $\boldsymbol{X}$=$\boldsymbol{U}\text{Diage}\left(\lambda_i\right)\boldsymbol{U}^T$. The vector representation of a symmetric matrix $\boldsymbol{X} \in \text{Sym}(n)$ is

$$\text{vec}_{\mathbf{I}}(\boldsymbol{X}) = \Big([\boldsymbol{X}]_{1,1}, \sqrt{2}[\boldsymbol{X}]_{1,2}, \ldots, \sqrt{2}[\boldsymbol{X}]_{1,n}, [\boldsymbol{X}]_{2,2}, \sqrt{2}[\boldsymbol{X}]_{2,3}, \ldots, [\boldsymbol{X}]_{n,n}\Big)^T . \tag{16.3}$$

Definition 16.1. A real and symmetric matrix $\boldsymbol{X} \in \mathbb{R}^{n\times n}$ is said to be SPD if $\boldsymbol{v}^T\boldsymbol{X}\boldsymbol{v}$ is positive for any nonzero $\mathbf{0} \neq \boldsymbol{v} \in \mathbb{R}^n$.

The space of $n\times n$ SPD matrices is obviously not a vector space since multiplying an SPD matrix by a negative scalar results in a matrix which does not belong to S^n_{++}. Instead, S^n_{++} forms the interior of a convex cone in the $n(n+1)/2$-dimensional Euclidean space. The S^n_{++} space is mostly studied when endowed with a Riemannian metric and thus forms a Riemannian manifold [23]. A natural way to measure closeness on a manifold is by considering the geodesic distance between two points on the manifold. Such a distance is defined as the length of the shortest curve connecting the two points. The geodesic curves are analogous to straight lines in $\mathbb{R}^n$. The affine invariant Riemannian metric (AIRM) is probably the most popular Riemannian structure for analyzing SPD matrices [23]. Let $\boldsymbol{P}$ be a point on S^n_{++}. The tangent space of S^n_{++} is $\text{Sym}(n)$ and the AIRM for two tangent vectors $\boldsymbol{V}, \boldsymbol{W} \in T_{\boldsymbol{P}}S^n_{++}$ is defined as

$$\langle \boldsymbol{V}, \boldsymbol{W}\rangle_{\boldsymbol{P}} := \langle \boldsymbol{P}^{-1/2}\boldsymbol{V}\boldsymbol{P}^{-1/2}, \boldsymbol{P}^{-1/2}\boldsymbol{W}\boldsymbol{P}^{-1/2}\rangle = \text{Tr}\left(\boldsymbol{P}^{-1}\boldsymbol{V}\boldsymbol{P}^{-1}\boldsymbol{W}\right) . \tag{16.4}$$

Definition 16.2. The geodesic distance $\delta_g : \mathcal{S}^n_{++} \times \mathcal{S}^n_{++} \to [0, \infty)$ induced by the AIRM is defined as

$$\delta_g(\boldsymbol{X}, \boldsymbol{Y}) = \|\log(\boldsymbol{X}^{-1/2}\boldsymbol{Y}\boldsymbol{X}^{-1/2})\|_F , \tag{16.5}$$

where $\log(\cdot)$ is the matrix principal logarithm.

The AIRM has several useful properties which are discussed in Sect. 16.2.1. For the AIRM, the exponential and logarithms maps are given by

$$\exp_{\boldsymbol{P}}(\boldsymbol{X}) = \boldsymbol{P}^{1/2}\exp(\boldsymbol{P}^{-1/2}\boldsymbol{X}\boldsymbol{P}^{-1/2})\boldsymbol{P}^{1/2} , \tag{16.6}$$

$$\log_{\boldsymbol{P}}(\boldsymbol{X}) = \boldsymbol{P}^{1/2}\log(\boldsymbol{P}^{-1/2}\boldsymbol{X}\boldsymbol{P}^{-1/2})\boldsymbol{P}^{1/2} . \tag{16.7}$$

Beside the AIRM, the Bregman divergences have been successfully employed for analyzing SPD matrices. Here, we are interested in two divergences derived from the Bregman matrix divergence.

Definition 16.3. Let $\zeta : \mathcal{S}_{++}^n \to \mathbb{R}$ be a strictly convex and differentiable function defined on the symmetric positive cone $\mathcal{S}_{++}^n$. The Bregman matrix divergence $d_\zeta : \mathcal{S}_{++}^n \times \mathcal{S}_{++}^n \to [0, \infty)$ is defined as

$$d_\zeta(\boldsymbol{X}, \boldsymbol{Y}) = \zeta(\boldsymbol{X}) - \zeta(\boldsymbol{Y}) - \langle \nabla_\zeta(\boldsymbol{Y}), \boldsymbol{X} - \boldsymbol{Y} \rangle \ , \tag{16.8}$$

where $\langle \boldsymbol{X}, \boldsymbol{Y} \rangle = \operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{Y}\right)$ is the Frobenius inner product and $\nabla_\zeta(\boldsymbol{Y})$ represents the gradient of ζ evaluated at $\boldsymbol{Y}$.

The Bregman divergence is asymmetric, nonnegative, and definite (i.e., $d_\zeta(\boldsymbol{X}, \boldsymbol{Y}) = 0$, iff $\boldsymbol{X} = \boldsymbol{Y}$). While the Bregman divergence enjoys a variety of useful properties [17], its asymmetric behavior is often a hindrance. In this chapter we are interested in two types of symmetrized Bregman divergences, namely the *Jeffrey* and the *Stein* divergences.

Definition 16.4 (Symmetric Positive-Definite Matrices: Jeffrey Divergence). The J divergence (also known as Jeffrey or symmetric KL divergence) is obtained from the Bregman divergence of Eq. (16.8) by using $\zeta(\boldsymbol{X}) = -\ln\det(\boldsymbol{X})$ as the seed function where $|\cdot|$ denotes determinant:

$$\begin{aligned} J(\boldsymbol{X}, \boldsymbol{Y}) &\triangleq \frac{1}{2} d_\zeta(\boldsymbol{X}, \boldsymbol{Y}) + \frac{1}{2} d_\zeta(\boldsymbol{Y}, \boldsymbol{X}) \\ &= \frac{1}{2}\operatorname{Tr}(\boldsymbol{X}^{-1}\boldsymbol{Y}) - \frac{1}{2}\ln\det(\boldsymbol{X}^{-1}\boldsymbol{Y}) + \frac{1}{2}\operatorname{Tr}(\boldsymbol{Y}^{-1}\boldsymbol{X}) - \frac{1}{2}\ln\det(\boldsymbol{Y}^{-1}\boldsymbol{X}) - n \\ &= \frac{1}{2}\operatorname{Tr}(\boldsymbol{X}^{-1}\boldsymbol{Y}) + \frac{1}{2}\operatorname{Tr}(\boldsymbol{Y}^{-1}\boldsymbol{X}) - n \ . \end{aligned} \tag{16.9}$$

Definition 16.5 (Symmetric Positive-Definite Matrices: Stein Divergence). The Stein or S divergence (also known as Jensen–Bregman LogDet divergence [6]) is obtained from the Bregman divergence of Eq. (16.8) by again using $\zeta(\boldsymbol{X}) = -\ln\det(\boldsymbol{X})$ as the seed function but through *Jensen–Shannon* symmetrization:

$$\begin{aligned} S(\boldsymbol{X}, \boldsymbol{Y}) &\triangleq \frac{1}{2} d_\zeta\left(\boldsymbol{X}, \frac{\boldsymbol{X}+\boldsymbol{Y}}{2}\right) + \frac{1}{2} d_\zeta\left(\boldsymbol{Y}, \frac{\boldsymbol{X}+\boldsymbol{Y}}{2}\right) \\ &= \ln\det\left(\frac{\boldsymbol{X}+\boldsymbol{Y}}{2}\right) - \frac{1}{2}\ln\det(\boldsymbol{X}\boldsymbol{Y}) \ . \end{aligned} \tag{16.10}$$

16.2.1 Properties of J and S Divergences

The J and S divergences have a variety of properties which are akin to those of AIRM. We present the pertinent properties which inspired us to seek coding on $\mathcal{S}_{++}^n$ using such divergences. The key message worth noting is the Hilbert space embedding property of the J and S divergences, which does not hold for AIRM [10, 15].

Invariance Property

An especially attractive property for the computer vision community is the invariance of J and S divergences to affine transforms. More specifically (and similar to AIRM), for $\boldsymbol{A} \in \mathrm{GL}(n)$, we have

$$J(\boldsymbol{X}, \boldsymbol{Y}) = J(\boldsymbol{A}\boldsymbol{X}\boldsymbol{A}^T, \boldsymbol{A}\boldsymbol{Y}\boldsymbol{A}^T),$$
$$S(\boldsymbol{X}, \boldsymbol{Y}) = S(\boldsymbol{A}\boldsymbol{X}\boldsymbol{A}^T, \boldsymbol{A}\boldsymbol{Y}\boldsymbol{A}^T).$$

This property postulates that the metric between two SPD matrices is unaffected by the action of the affine group. In the specific case where the SPD matrices are region covariance descriptors [29], this implies that the distance between two descriptors will remain unchanged after an affine transformation of the image features, such as a change of illumination when using RGB values. Furthermore, similar to AIRM, both divergences are invariant to inversion, i.e.,

$$J(\boldsymbol{X}, \boldsymbol{Y}) = J(\boldsymbol{X}^{-1}, \boldsymbol{Y}^{-1}),$$
$$S(\boldsymbol{X}, \boldsymbol{Y}) = S(\boldsymbol{X}^{-1}, \boldsymbol{Y}^{-1}).$$

Proofs for the above statements can be readily obtained by plugging the affine representations (e.g., $\boldsymbol{A}\boldsymbol{X}\boldsymbol{A}^T$) or inverses into the definition of J and S divergences.

Averaging Property

Definition 16.6. Given a metric δ and two tensors $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{S}_{++}^n$, the geometric mean is defined as

$$\boldsymbol{A} \sharp_\delta \boldsymbol{B} = \operatorname*{arg\,min}_{\boldsymbol{X} \in \mathcal{S}_{++}^n} \delta(\boldsymbol{X}, \boldsymbol{A}) + \delta(\boldsymbol{X}, \boldsymbol{B}). \tag{16.11}$$

Theorem 16.1. *For two matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{S}_{++}^n$, the geometric mean of J divergence $\boldsymbol{A} \sharp_J \boldsymbol{B}$ and AIRM $\boldsymbol{A} \sharp_g \boldsymbol{B}$ are the same.*

Proof. For the J divergence, we note that

$$\frac{\partial \{J(\boldsymbol{X}, \boldsymbol{A}) + J(\boldsymbol{X}, \boldsymbol{B})\}}{\partial \boldsymbol{X}} = \frac{1}{2}\Big(\boldsymbol{A}^{-1} + \boldsymbol{B}^{-1} - \boldsymbol{X}^{-1}(\boldsymbol{A} + \boldsymbol{B})\boldsymbol{X}^{-1}\Big).$$

Therefore, $\boldsymbol{A} \sharp_J \boldsymbol{B}$ is the solution of

$$\boldsymbol{X}(\boldsymbol{A}^{-1} + \boldsymbol{B}^{-1})\boldsymbol{X} = \boldsymbol{A} + \boldsymbol{B}, \tag{16.12}$$

which is a *Riccati* equation with only one positive-definite solution [3]. We note that

$$\begin{aligned} A\sharp_g B &= \exp_B\left(\frac{1}{2}\log_B(A)\right) = \exp_A\left(\frac{1}{2}\log_A(B)\right) \\ &= A^{\frac{1}{2}}\left(A^{-\frac{1}{2}}BA^{-\frac{1}{2}}\right)^{\frac{1}{2}}A^{\frac{1}{2}}. \end{aligned} \tag{16.13}$$

It can be readily shown that $A\sharp_g B$ satisfies Eq. (16.12) which concludes the proof.

Theorem 16.2. *The average of a set of points* $\{X_i\}_{i=1}^N$, $X_i \in S_{++}^n$ *based on J divergence is defined as*

$$M \triangleq \underset{X \in S_{++}^n}{\arg\min} \sum_{i=1}^{N} J(X_i, X) \tag{16.14}$$

and admits a closed-form solution in the form of

$$M = \mathbf{L}^{-1/2}\left(\mathbf{L}^{1/2}\boldsymbol{\Gamma}\mathbf{L}^{1/2}\right)^{1/2}\mathbf{L}^{-1/2} \tag{16.15}$$

with $\mathbf{L} = \sum_{i=1}^N X_i^{-1}$ *and* $\boldsymbol{\Gamma} = \sum_{i=1}^N X_i$.

Proof. The theorem can be readily proved. To this end, the solution of

$$\frac{\partial \sum_{i=1}^N J(X_i, X)}{\partial X} = 0 \tag{16.16}$$

must be obtained. Similar to the proof of previous theorem, Eq. (16.16) has the form of *Riccati* equation with a unique and closed-form solution. A slightly different proof is also provided in [32].

For the S divergence and similar to AIRM, the average of a set of points has no closed-form solution and can be obtained by an iterative scheme using the convex–concave procedure [36] as explained in [6]. Nevertheless, the average of two points based on S-divergence coincides with the AIRM mean as the following theorem states.

Theorem 16.3. *For two matrices* $A, B \in S_{++}^n$, *the geometric mean of S divergence* $A\sharp_S B$ *and AIRM* $A\sharp_g B$ *are the same.*

Proof.

$$\begin{aligned} &\frac{\partial\{S(X,A) + S(X,B)\}}{\partial X} = 0 \\ &\Rightarrow \left(X + A\right)^{-1} - \frac{1}{2}X^{-1} + \left(X + B\right)^{-1} - \frac{1}{2}X^{-1} = 0 \\ &\Rightarrow X^{-1}\left(X + A\right) = \mathbf{I}_n + \left(X + B\right)^{-1}\left(X + A\right) \end{aligned}$$

$$\Rightarrow \big(\boldsymbol{X}+\boldsymbol{B}\big)\boldsymbol{X}^{-1}\big(\boldsymbol{X}+\boldsymbol{A}\big) = \big(\boldsymbol{X}+\boldsymbol{B}\big) + \big(\boldsymbol{X}+\boldsymbol{A}\big)$$

$$\Rightarrow \boldsymbol{X}\boldsymbol{A}^{-1}\boldsymbol{X} = \boldsymbol{B} \tag{16.17}$$

$$\Rightarrow \boldsymbol{X} = \boldsymbol{A}^{\frac{1}{2}}\left(\boldsymbol{A}^{-\frac{1}{2}}\boldsymbol{B}\boldsymbol{A}^{-\frac{1}{2}}\right)^{\frac{1}{2}}\boldsymbol{A}^{\frac{1}{2}}. \tag{16.18}$$

Again, we note that Eq. (16.17) *is a Riccati equation with the unique solution as depicted in Eq.* (16.18) *which is exactly the AIRM mean in Eq.* (16.13).

Hilbert Space Embedding Property (SPD Kernels)

Both J and S divergences admit a Hilbert space embedding in the form of RBF kernel. More specifically, for the J divergence it has been shown that the kernel

$$k_J(\boldsymbol{X},\boldsymbol{Y}) = \exp\{-\beta J(\boldsymbol{X},\boldsymbol{Y})\} \tag{16.19}$$

is conditionally positive definite (CPD) [13]. The class of CPD kernels corresponds to Hilbertian metrics and has wide applications in machine learning. For example, it has been shown that the solution and optimization problem of the SVMs only depend on Hilbertian metrics [13]. We note that in [20] the kernel $k_J(\cdot,\cdot)$ was claimed to be positive definite. However, a formal proof is not available according to our best knowledge. For the Stein divergence, the kernel

$$k_S(\boldsymbol{X},\boldsymbol{Y}) = \exp\{-\beta S(\boldsymbol{X},\boldsymbol{Y})\} \tag{16.20}$$

is not positive definite for $\beta > 0$ [28]. However, bounds on β do exist which guarantee the positive definiteness of the kernel. The following theorem states the condition under which Stein kernel is positive definite.

Theorem 16.4. *Let $\Omega = \{\boldsymbol{X}_1,\boldsymbol{X}_2,\ldots,\boldsymbol{X}_N\}; \boldsymbol{X}_i \in S^n_{++}$ be a set of Riemannian points. The $N\times N$ matrix $\boldsymbol{K}_\beta = [k_\beta(i,j)]; 1 \le i,j \le N$, with $k_\beta(i,j) = k_S(\boldsymbol{X}_i,\boldsymbol{X}_j)$, defined in Eq.* (16.20)*, is positive definite iff*

$$\beta \in \left\{\frac{1}{2},\frac{2}{2},\ldots,\frac{n-1}{2}\right\} \cup \left\{\tau \in \mathbb{R} : \tau > \frac{1}{2}(n-1)\right\}. \tag{16.21}$$

Interested readers can follow the proof in [28]. For values of β outside of the above set, it is possible to convert a pseudo kernel into a true kernel, as discussed for example in [5].

16.3 Riemannian Coordinate Coding

Given a query $\boldsymbol{X} \in \mathcal{S}_{++}^n$, such as an image descriptor, we are interested in transforming $\boldsymbol{X}$ into a more compact, and hopefully discriminative, representation, hereafter referred to as *code*. A general formulation that englobes many different coding techniques can be expressed as

$$\min_{\boldsymbol{y}} \left\| \boldsymbol{X} \ominus \biguplus_{j=1}^{N} [\boldsymbol{y}]_j \odot \boldsymbol{D}_j \right\|_{\delta}^{2} + \lambda \gamma(\boldsymbol{y}; \boldsymbol{X}, \mathbb{D}) \text{ s.t. } \ \boldsymbol{y} \in \mathcal{C}. \tag{16.22}$$

Here, $\mathbb{D} = \{\boldsymbol{D}_i\}_{i=1}^N$, $\boldsymbol{D}_i \in \mathcal{S}_{++}^n$ is the N dictionary elements, $\gamma(\cdot)$ is a prior on the codes $\boldsymbol{y}$, and $\mathcal{C}$ is a set of constraints on $\boldsymbol{y}$. Note that this formulation allows the prior to be dependent on both the query $\boldsymbol{X}$ and the dictionary $\mathbb{D}$. Before proceeding on possible solutions of Eq. (16.22), we take a detour and explain how the dictionary $\mathbb{D}$ can be obtained through k-Means algorithm.

16.3.1 Creating Visual Dictionary

In computer vision, k-Means is a prevalent technique in generating visual dictionaries. Let $(\mathcal{M}, \delta)$ be a metric space. Given a set of samples $\{\boldsymbol{X}_i\}_{i=1}^m$, $\boldsymbol{X}_i \in \mathcal{M}$, k-Means minimizes the following cost function to determine N cluster centers $\{\boldsymbol{D}_j\}_{j=1}^N$, $\boldsymbol{D}_j \in \mathcal{M}$:

$$e(\mathbb{X}, \mathbb{D}) = \sum_{i=1}^{m} \min_{j=1}^{N} \delta^2(\boldsymbol{X}_i, \boldsymbol{D}_j). \tag{16.23}$$

This is achieved by first selecting the initial centers randomly from $\{\boldsymbol{X}_i\}_{i=1}^m$. The k-Means algorithm then iterates through two main steps. In the first step, given the current estimation of the cluster centers, the closest samples to each center are determined. Then the cluster centers are updated by computing the mean of the closest samples. In our case, $\mathcal{M}$ is $\mathcal{S}_{++}^n$ and the metric could be the AIRM, S or J divergence. As for AIRM, the mean of a set of points can be obtained through computing the Karcher mean [23] which uses the exponential and logarithm maps. The mean of a set of points on $\mathcal{S}_{++}^n$ is obtained using methods described in Sect. 16.2.1. More specifically, for the J divergence, a closed-form solution for computing the mean of a set of points is available as described in Sect. 16.2.1. As for the Stein divergence, an efficient procedure is proposed in [26] which can be employed to obtain the mean. Algorithm 1 illustrates k-Means++ clustering on $\mathcal{S}_{++}^n$ which benefits from a more involved procedure in initializing the cluster centers.

Algorithm 1 k-Means++ algorithm for dictionary learning on $\mathcal{S}^n_{++}$

```
Input: Training data: ${X_i}_{i=1}^m$, $X_i \in \mathcal{S}^n_{++}$; number of clusters: $k$;
Output: The cluster centers: ${D_i}_{i=1}^k$, $D_i \in \mathcal{S}^n_{++}$;
Initialization.
Take first center $D_1$, chosen uniformly at random from ${X_i}_{i=1}^m$;
$\mathbb{D} \leftarrow D_1$
for $j \leftarrow 2$ to $k$ do
    for $i \leftarrow 1$ to $m$ do
        $d(X_i) = \min_{\mathbb{D}} \delta^2(X_i, \mathbb{D})$
    end for
    for $i \leftarrow 1$ to $m$ do
        $p(X_i) \leftarrow \frac{d(X_i)}{\sum_l d(X_l)}$
    end for
    Take center $D_j$ randomly from the distribution $p(X_i)$;
    $\mathbb{D} \leftarrow \mathbb{D} \cup D_j$
end for
Processing.
repeat
for $i = 1 \leftarrow 1$ to $m$ do
    $a[i] \leftarrow \arg\min_j \delta^2(X_i, D_j)$ {//assignment; $\delta^2$ is either J or S-divergence (see Eq. (16.9) and Eq. (16.10))//}
end for
for $j = 1 \leftarrow 1$ to $N$ do
    $D_j \leftarrow \arg\min_D \sum_{a[i]==j} \delta^2(X_i, D)$ {//updating centers//}
end for
until Convergence;
```

16.3.2 *Intrinsic Coordinate Coding*

Given a dictionary $\mathbb{D} = \{D_i\}_{i=1}^N$, $D_i \in \mathcal{S}^n_{++}$, and a query $X \in \mathcal{S}^n_{++}$, we propose to write Eq. (16.22) as

$$\min_{y} \Big\| \sum_i y_i \log_X(D_i) \Big\|_X^2 + \lambda \gamma(y; X, \mathbb{D}) \text{ s.t. } \ y \in \mathcal{C}. \tag{16.24}$$

To see the logic behind Eq. (16.24), consider the logarithm map on $\mathcal{S}^n_{++}$. We note that $\log_X(X) = \mathbf{0}$ and $\delta_g^2(X, D_i) = \|\log_X(D_i)\|_X^2$. Moreover, the tangent space at X is a vector space. Therefore, one can seamlessly choose vector space operators (addition, subtraction, and scalar product) to perform $\uplus$, $\ominus$, and $\odot$. As such, Eq. (16.24) is effectively the same as Eq. (16.22). However, Eq. (16.24) suffers from a trivial solution if no constraint is envisioned for coding. That is, without a proper $\mathcal{C}$, $y = \mathbf{0}$ will be the solution of Eq. (16.24).

To this end, we propose to use the constraint $y^T\mathbf{1} = 1$ to solve Eq. (16.24). This affine constraint results in independency to the origin of the coordinate system defined on T_X and has been used for example in [14] for sparse coding on Riemannian manifolds and in [8, 25] for the purpose of dimensionality reduction and manifold learning.

To see the independency to coordinate system, assume $\boldsymbol{x} \in \mathbb{R}^d$ is encoded by three atoms $\boldsymbol{d}_1$, $\boldsymbol{d}_2$, and $\boldsymbol{d}_3$ with weight vector $\boldsymbol{y} = ([y]_1, [y]_2, [y]_3)^T$. That is, $\boldsymbol{x} = [y]_1\boldsymbol{d}_1 + [y]_2\boldsymbol{d}_2 + [y]_3\boldsymbol{d}_3$. Changing the coordinates by translating the origin to $\boldsymbol{t}$ results in having $\boldsymbol{x} - \boldsymbol{t}$, $\boldsymbol{d}_1 - \boldsymbol{t}$, $\boldsymbol{d}_2 - \boldsymbol{t}$ and $\boldsymbol{d}_3 - \boldsymbol{t}$ as query and dictionary atoms, respectively. Under this change, $\boldsymbol{x} - \boldsymbol{t}$ cannot be encoded by the weight vector $\boldsymbol{y}$. However, if we enforce the weight vector to be affine invariant, i.e., $\boldsymbol{y}^T\mathbf{1} = 1$, then it is easy to verify that $\boldsymbol{x} - \boldsymbol{t} = [y]_1(\boldsymbol{d}_1 - \boldsymbol{t}) + [y]_2(\boldsymbol{d}_2 - \boldsymbol{t}) + [y]_3(\boldsymbol{d}_3 - \boldsymbol{t})$.

Turning our attention to the function $\gamma(\boldsymbol{y}; \boldsymbol{X}, \mathbb{D})$, we note that sparse coding can be achieved by choosing $\gamma(\boldsymbol{y}; \boldsymbol{X}, \mathbb{D}) = \|\mathbf{y}\|_1$.

This is of course what Ho et al. proposed in their work [14]. However, some studies favor to perform coding based on the distances between dictionary atoms and the query [33, 35]. That is, it is preferable to have coding schemes where only nearby atoms to a query contribute to the coding weights. This, which we call coordinate coding, can be achieved with the following penalty function:

$$\gamma(\boldsymbol{y}; \boldsymbol{X}, \mathbb{D}) = \|\boldsymbol{E}\boldsymbol{y}\|_2^2\,, \text{ with } \boldsymbol{E}_{ii} = \exp\left(\sigma\delta_g^2(\boldsymbol{X}, \boldsymbol{D}_i)\right)\,, \text{ and } \boldsymbol{E}_{ij} = 0,\ i \neq j\,, \tag{16.25}$$

Obviously, Eq. (16.25) will incur a heavy penalty if $\delta_g^2(\boldsymbol{X}, \boldsymbol{D}_i)$ is large and hence force the coding weights to be dependent more on nearby atoms. This ultimately can result in sparsity (as a parameter of σ) while the other way around is not necessarily true, i.e., sparsity cannot guarantee locality.

A special case, which we will use later in our experiments, is the case where a hard thresholding is performed on the prior $\|\boldsymbol{E}\boldsymbol{y}\|_2^2$. More specifically, to enforce locality, we could construct a local dictionary $\mathbb{B}_{\boldsymbol{X}}$ by considering the N_B nearest atoms from $\mathbb{D}$, i.e., $\mathbb{B}_{\boldsymbol{X}} \ni \boldsymbol{D}_i$, iff $\delta^2(\boldsymbol{X}, \boldsymbol{D}_i) \leq \delta^2(\boldsymbol{X}, \boldsymbol{D}_j)$, $\boldsymbol{D}_j \notin \mathbb{B}_{\boldsymbol{X}}$. Having $\mathbb{B}_{\boldsymbol{X}}$, we define the following scheme as intrinsic coordinate coding (iCC) on $\mathcal{S}_{++}^n$

$$\min_{\boldsymbol{y}} \Big\| \sum_{\boldsymbol{D}_i \in \mathbb{B}_{\boldsymbol{X}}} \boldsymbol{y}_i \log_{\boldsymbol{X}}(\boldsymbol{D}_i) \Big\|_{\boldsymbol{X}}^2 \text{ s.t. } \boldsymbol{y}^T\mathbf{1} = 1. \tag{16.26}$$

Interestingly, the solution of Eq. (16.26) can be obtained in closed form and as a least-squares problem. Hence, unlike sparse coding where codes are obtained through an iterative algorithm, the iCC method achieves its solution in a single step. Algorithm 2 provides all the details for performing iCC on $\mathcal{S}_{++}^n$.

16.3.3 Log-Euclidean Coordinate Coding

The aforementioned intrinsic method requires one to project all atoms in the local dictionary onto the tangent space of a query point through the logarithm map. This is computationally expensive for high-dimensional manifolds as well as large local dictionaries. To reduce the computational load, it is possible to flatten the manifold through its identity tangent space which identifies the Lie algebra of $\mathcal{S}_{++}^n$. More

Algorithm 2 Intrinsic coordinate coding (iCC)

```
Input: Dictionary $\mathbb{D} = \{\boldsymbol{D}_i\}_{i=1}^N$, $\boldsymbol{D}_i \in \mathcal{S}_{++}^n$; the query $\boldsymbol{X} \in \mathcal{S}_{++}^n$; size of local dictionary $N_B$.
Output: The iCC codes $\boldsymbol{y}^*$
Processing.
$\boldsymbol{y}^* \leftarrow \mathbf{0}_{N\times 1}$
for $i \leftarrow 1$ to $N$ do
    $\delta(i) \leftarrow \delta_g^2(\boldsymbol{X}, \boldsymbol{D}_i)$
end for
active_set $\leftarrow$ indexes of the $N_B$ smallest $\delta(i), 1 \leq i \leq N$
for $i \leftarrow 1$ to $N_B$ do
    $\boldsymbol{b}_i \leftarrow \mathrm{vec}_{\mathbf{I}}\Big(\log\big(\boldsymbol{X}^{-1/2}\boldsymbol{D}_{\text{active\_set}(i)}\boldsymbol{X}^{-1/2}\big)\Big)$
end for
$\boldsymbol{B} \leftarrow [\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_{N_B}]$ {// create the local dictionary $\boldsymbol{B}_{n(n+1)/2\times N_B}$//}
Solve the linear equation system $\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{y} = \mathbf{1}$
$\hat{\boldsymbol{y}} \leftarrow \boldsymbol{y}/\mathbf{1}^T\boldsymbol{y}$
$\boldsymbol{y}^*(\text{active\_set}) \leftarrow \hat{\boldsymbol{y}}$
```

specifically, we can rewrite Eq. (16.22) with the affine constraint as

$$\min_{\boldsymbol{y}} \left\| \nu(\boldsymbol{X}) - \sum_{j=1}^{N} [\boldsymbol{y}]_j \nu(\boldsymbol{D}_j) \right\|^2 + \lambda\gamma\big(\boldsymbol{y}; \nu(\boldsymbol{X}), \nu(\mathbb{D})\big) \text{s.t. } \boldsymbol{y}^T\mathbf{1} = 1. \tag{16.27}$$

Here, $\nu(\cdot) : \mathcal{S}_{++}^n \to \mathbb{R}^{n(n+1)/2}$ is defined as $\nu(\boldsymbol{X}) = \mathrm{vec}_{\mathbf{I}}\big(\log(\boldsymbol{X})\big)$. Similar to Eq. (16.25), we utilize the following prior for coding

$$\gamma\big(\boldsymbol{y}; \nu(\boldsymbol{X}), \nu(\mathbb{D})\big) = \|\boldsymbol{E}\boldsymbol{y}\|_2^2, \text{ with } \boldsymbol{E}_{ii} = \exp\Big(\sigma\|\nu(\boldsymbol{X}) - \nu(\boldsymbol{D}_i)\|^2\Big), \text{ and } \boldsymbol{E}_{ij} = 0,\ i \neq j\,. \tag{16.28}$$

The closed-form solution of Eq. (16.27) with the prior described in Eq. (16.28) can be obtained in two steps: First, one needs to solve the system of equations

$$\big(\nu(\mathbb{D})^T\nu(\mathbb{D}) - \mathbf{1}^T \otimes \nu(\boldsymbol{X}) + \gamma\boldsymbol{E}^2\big)\boldsymbol{y} = \mathbf{1}, \tag{16.29}$$

where $\otimes$ denotes the Kronecker product. The solution obtained from Eq. (16.29) is then normalized to have unit ℓ_1 norm.

This, as compared to the solution described in Sect. 16.3.2, is computationally less demanding (only one logarithm is computed per query) but appreciates the true geometry of $\mathcal{S}_{++}^n$ less. With the hard thresholding on prior (which will be used in the experiments), one first needs to determine a local dictionary $\mathbb{B}_{\boldsymbol{X}}$ by stacking the N_B closest atoms from $\nu(\mathbb{D})$ to $\nu(\boldsymbol{X})$. Then the solution can be obtained using $\mathbb{B}_{\boldsymbol{X}}$ as illustrated in Algorithm 3. We shall refer to this straightforward approach as Log-Euclidean coordinate coding (leCC), following the terminology used in [2].

Algorithm 3 Log-Euclidean coordinate coding (leCC)

```
**Input:** Dictionary $\mathbb{D} = \{\boldsymbol{D}_i\}_{i=1}^N$, $\boldsymbol{D}_i \in \mathcal{S}_{++}^n$; the query $\boldsymbol{X} \in \mathcal{S}_{++}^n$; size of local dictionary $N_B$.
**Output:** The leCC codes $\boldsymbol{y}^*$
**Initialization.**
**for** $i \leftarrow 1$ **to** $N$ **do**
    $\boldsymbol{d}_i \leftarrow \mathrm{vec}_{\mathbf{I}}\big(\log(\boldsymbol{D}_i)\big)$
**end for**
**Processing.**
$\boldsymbol{x} \leftarrow \mathrm{vec}_{\mathbf{I}}\big(\log(\boldsymbol{X})\big)$
$\boldsymbol{y}^* \leftarrow \mathbf{0}_{N\times 1}$
**for** $i \leftarrow 1$ **to** $N$ **do**
    $\delta(i) \leftarrow \|\boldsymbol{x} - \boldsymbol{d}_i\|^2$
**end for**
active_set $\leftarrow$ indexes of the $N_B$ smallest $\delta(i), 1 \le i \le N$
$\boldsymbol{B} \leftarrow \bigcup_{i \in \text{active\_set}} \boldsymbol{d}_i$ {//create the local dictionary $\boldsymbol{B}_{n(n+1)/2\times N_B}$//}
Solve the linear equation system $\big(\boldsymbol{B}^T\boldsymbol{B} - \mathbf{1}^T \otimes \boldsymbol{x}\big)\boldsymbol{y} = \mathbf{1}$
$\hat{\boldsymbol{y}} \leftarrow \boldsymbol{y}/\mathbf{1}^T\boldsymbol{y}$
$\boldsymbol{y}^*(\text{active\_set}) \leftarrow \hat{\boldsymbol{y}}$
```

16.3.4 Kernel Coordinate Coding

The idea of coordinate coding can be kernelized as follows. Given an embedding ϕ from $\mathcal{S}_{++}^n$ to an RKHS $\mathcal{H}$, i.e., $\phi : \mathcal{S}_{++}^n \to \mathcal{H}$ a Riemannian dictionary $\mathbb{D} = \{\boldsymbol{D}_i\}_{i=1}^N$, $\boldsymbol{D}_i \in \mathcal{S}_{++}^n$, we first kernelize the prior γ as

$$\gamma(\boldsymbol{y}; \boldsymbol{X}, \mathbb{D}) = \|\boldsymbol{E}\boldsymbol{y}\|_2^2\,, \text{ with } \boldsymbol{E}_{ii} = \exp\left(\sigma\|\phi(\boldsymbol{X}) - \phi(\boldsymbol{D}_i)\|_2\right), \text{ and } \boldsymbol{E}_{ij} = 0,\ i \neq j. \tag{16.30}$$

We note that

$$\boldsymbol{E}_{ii} = \exp\left(\sigma\sqrt{k(\boldsymbol{X},\boldsymbol{X}) - 2k(\boldsymbol{X},\boldsymbol{D}_i) + k(\boldsymbol{D}_i,\boldsymbol{D}_i)}\right). \tag{16.31}$$

Now, the problem of coordinate coding on $\mathcal{S}_{++}^n$ can be cast as

$$\min_{\boldsymbol{y}} \left\|\phi(\boldsymbol{X}) - \sum_{j=1}^N [\boldsymbol{y}]_j \phi(\boldsymbol{D}_j)\right\|_2^2 + \lambda\gamma(\boldsymbol{y}; \boldsymbol{X}, \mathbb{D}) \text{ s.t. } \boldsymbol{y}^T\mathbf{1} = 1. \tag{16.32}$$

Following similar steps to Sect. 16.3.3, we can find a closed-form solution for Eq. (16.32). More specifically, by expanding Eq. (16.32)

$$\begin{aligned}
&\left\|\phi(\boldsymbol{X}) - \sum_{j=1}^N [\boldsymbol{y}]_j \phi(\boldsymbol{D}_j)\right\|_2^2 + \lambda\left\|\boldsymbol{E}\boldsymbol{y}\right\|^2 \\
&= k(\boldsymbol{x},\boldsymbol{x}) - 2\boldsymbol{y}^T\boldsymbol{k}(\boldsymbol{x},\mathbb{D}) + \boldsymbol{y}^T\big(\boldsymbol{K}(\mathbb{D},\mathbb{D}) + \lambda\boldsymbol{E}^2\big)\boldsymbol{y},
\end{aligned}$$

Algorithm 4 Kernel coordinate coding (kCC)

```
Input: Dictionary $\boldsymbol{D} = \{\boldsymbol{d}_i\}_{i=1}^N$, $\boldsymbol{d}_i \in \mathbb{R}^d$; the query $\boldsymbol{x} \in \mathbb{R}^d$, a positive-definite kernel
$k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$.
Output: The kCC codes $\boldsymbol{y}^*$
Processing.
$\boldsymbol{y}^* \leftarrow \boldsymbol{0}_{N\times 1}$
for $i \leftarrow 1$ to $N$ do
    $\delta_i \leftarrow k(\boldsymbol{d}_i, \boldsymbol{d}_i) - 2k(\boldsymbol{x}, \boldsymbol{d}_i)$
end for
active_set $\leftarrow$ indexes of the $N_{LLC}$ smallest $\delta_i$, $1 \leq i \leq N$
for $i \leftarrow 1$ to $N_B$ do
    $\boldsymbol{b}_i \leftarrow \boldsymbol{d}_{\text{active\_set}(i)}$
end for
for $i \leftarrow 1$ to $N_B$ do
    $[\boldsymbol{k}(\boldsymbol{x}, \boldsymbol{B})]_i \leftarrow k(\boldsymbol{x}, \boldsymbol{b}_i)$
end for
for $i, j \leftarrow 1$ to $N_B$ do
    $[\boldsymbol{K}(\boldsymbol{B}, \boldsymbol{B})]_{i,j} \leftarrow k(\boldsymbol{b}_i, \boldsymbol{b}_j)$
end for
Solve the linear equation system $\Big(\boldsymbol{K}(\boldsymbol{B}, \boldsymbol{B}) - \big(\boldsymbol{1}^T \otimes \boldsymbol{k}(\boldsymbol{x}, \boldsymbol{B})\big)\Big)\boldsymbol{y} = \boldsymbol{1}$
$\hat{\boldsymbol{y}} \leftarrow \boldsymbol{y}/\boldsymbol{1}^T\boldsymbol{y}$
$\boldsymbol{y}^*(\text{active\_set}) \leftarrow \hat{\boldsymbol{y}}$
```

we see that again the solution is obtained by first solving the equation system

$$\big(K(\mathbb{D}, \mathbb{D}) - \boldsymbol{1}^T \otimes \boldsymbol{k}(\boldsymbol{x}, \mathbb{D}) + \gamma \boldsymbol{E}^2\big)\boldsymbol{y} = \boldsymbol{1}, \tag{16.33}$$

followed by normalizing the solution to have unit ℓ_1 norm.

As for hard thresholding the prior, the local dictionary $\boldsymbol{B}$ by performing kernel nearest neighbor between the dictionary elements and the query. This lets us write the kCC in $\mathcal{H}$ as

$$\min_{\boldsymbol{y}} \|\phi(\boldsymbol{x}) - \phi(\boldsymbol{B})\boldsymbol{y}\|_2^2 \quad \text{s.t.} \quad \boldsymbol{1}^T\boldsymbol{y} = 1, \tag{16.34}$$

which has a form similar to Eq. (16.32) with no prior. Algorithm 4 provides the pseudo-code for performing kCC.

16.4 Empirical Evaluation

We used the "b" subset of the FERET data set [24], which includes 1800 images from 200 subjects for the face recognition task. The images were closely cropped around the face and downsampled to 64×64. Examples are shown in Fig. 16.1.

Fig. 16.1 Examples from the FERET face data set [24]. (**a**) ba, (**b**) bj, (**c**) bk, (**d**) bd, (**e**) be, (**f**) bf, (**g**) bg

We performed six tests with various pose angles. Training data were composed of images marked "ba," "bj," and "bk" (i.e., frontal faces with expression and illumination variations). Images with "bc," "bd," "be," "bf," "bg," and "bh" labels (i.e., non-frontal faces) were used as test data.

Each face image is described by a set of 11 × 11 RCMs using the following features:

$$\boldsymbol{f}_{x,y} = \left(I(x,y),\ \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial x}\right)^2}, \right.$$
$$\left. \arctan\left(\frac{\partial I(x,y)}{\partial y} \Big/ \frac{\partial I(x,y)}{\partial x}\right),\ G_{0,0}(x,y),\ \ldots,\ G_{1,3}(x,y) \right)^T ,$$

where $I(x,y)$ is the intensity value at position (x,y), $\frac{\partial I(x,y)}{\partial \cdot}$ is gradient along x or y, and $G_{u,v}(x,y)$ is the response of a difference of Gaussian (DOG) filter centered at (x,y) with orientation v and scale u. In particular, we extracted 49 RCMs from an image where each RCM corresponded to a region of size 16 × 16. The descriptor for an image is obtained by averaging the coordinate codes of its 49 RCMs, i.e., a simple averaging was used for pooling.

Table 16.1 shows the performance of all the studied methods for the task of face recognition. This includes the iCC, **leCC**, as described in Sect. 16.3.3, the intrinsic coordinate coding, **iCC**, as described in Sect. 16.3.2, and the iCC, **kCC**, as described in Sect. 16.3.4. We show the kCC algorithm with the Jeffrey and the Stein divergences by **kCC-J** and **kCC-S**, respectively. For each method, three dictionaries of size 128, 256, and 512 were learned using the k-Means++ algorithm as described in Sect. 16.3.1.

First we note that exploiting the Riemannian structure of SPD manifolds boosts the performance significantly as evidenced when *iCC* algorithm is compared against *leCC*. For example, the difference between *iCC* and *leCC* surpasses 12 % points when a dictionary of size 512 was utilized. By increasing the size of the dictionary, the performance of both *leCC* and *iCC* increases at the price of heavier computation load.

Comparing *kCC* algorithm against *iCC* shows that the kernel method is preferable. We conjecture that this is a by-product of embedding SPD manifolds into

Table 16.1 Recognition accuracy (in %) for the FERET face data set [24]

Method	bc (%)	bd (%)	be (%)	bf (%)	bg (%)	bh (%)
leCC (128 atoms)	18.5	44.0	78.5	83.0	52.0	19.0
leCC (256 atoms)	21.0	55.0	90.0	91.5	65.5	30.5
leCC (512 atoms)	21.0	55.0	90.0	94.0	66.0	26.0
iCC (128 atoms)	21.0	42.0	74.0	85.0	49.5	17.5
iCC (256 atoms)	26.5	59.0	88.0	92.0	67.0	29.0
iCC (512 atoms)	32.0	69.0	94.0	97.5	78.5	35.0
kCC-J (128 atoms)	23.0	58.0	77.0	88.0	73.0	41.5
kCC-J (256 atoms)	34.5	74.0	94.0	96.5	86.5	46.5
kCC-J (512 atoms)	40.0	**85.0**	96.5	**98.0**	**91.0**	**62.0**
kCC-S (128 atoms)	34.5	72.5	94.0	92.0	76.0	37.5
kCC-S (256 atoms)	38.5	80.0	97.0	97.5	82.5	46.5
kCC-S (512 atoms)	**42.0**	82.0	**98.5**	**98.0**	90.0	53.5

infinite-dimensional spaces through kernels which might have increased the discriminatory power among classes. Among the Jeffrey and the Stein divergences, there is no clear winner as *kCC-S* achieves the highest recognition accuracy in three experiments. However, in general *kCC-J* seems to be slightly better.

16.5 Main Findings and Future Work

With the aim of addressing coding on SPD manifolds, we proposed iCC which exploits the tangent bundle of the SPD manifolds to perform coding. We also proposed to seek the solution through embedding the manifolds into RKHS with the aid of two Bregman divergences, namely Stein and Jeffrey divergences. This was motivated by the success of many learning algorithms arises from their use of kernel methods. Therefore, one could expect embedding a Riemannian manifold into higher dimensional RKHS, where linear geometry applies, facilitates inference.

Experiments on the task of face recognition show that the proposed approaches achieve notable improvements in discrimination accuracy. Future venues of exploration include devising structured coding and learning.

Acknowledgements NICTA is funded by the Australian Government as represented by the *Department of Broadband, Communications and the Digital Economy*, as well as the Australian Research Council through the *ICT Centre of Excellence* program.

References

1. Agarwal A, Triggs B (2006) Hyperfeatures-multilevel local coding for visual recognition. In: Proceedings of European conference on computer vision (ECCV). Springer, Berlin, pp 30–43
2. Arsigny V, Fillard P, Pennec X, Ayache N (2006) Log-Euclidean metrics for fast and simple calculus on diffusion tensors. Magn Reson Med 56(2):411–421
3. Bhatia R (2007) Positive definite matrices. Princeton University Press, Princeton
4. Boureau YL, Bach F, LeCun Y, Ponce J (2010) Learning mid-level features for recognition. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), pp 2559–2566
5. Chen Y, Garcia EK, Gupta MR, Rahimi A, Cazzanti L (2009) Similarity-based classification: concepts and algorithms. J Mach Learn Res 10:747–776
6. Cherian A, Sra S, Banerjee A, Papanikolopoulos N (2013) Jensen-Bregman logdet divergence with application to efficient similarity search for covariance matrices. IEEE Trans Pattern Anal Mach Intell 35(9):2161–2174
7. Coates A, Ng AY (2011) The importance of encoding versus training with sparse coding and vector quantization. In: Proceedings of international conference on machine learning (ICML), pp 921–928
8. Goh A, Vidal R (2008) Clustering and dimensionality reduction on Riemannian manifolds. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), pp 1–7
9. Grauman K, Darrell T (2005) The pyramid match kernel: discriminative classification with sets of image features. In: Proceedings of international conference on computer vision (ICCV), vol 2, pp 1458–1465
10. Harandi MT, Sanderson C, Hartley R, Lovell BC (2012) Sparse coding and dictionary learning for symmetric positive definite matrices: a kernel approach. In: Proceedings of European conference on computer vision (ECCV), pp 216–229. Springer, Berlin
11. Harandi M, Salzmann M, Porikli F (2014) Bregman divergences for infinite dimensional covariance matrices. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR)
12. Harandi MT, Salzmann M, Hartley R (2014) From manifold to manifold: geometry-aware dimensionality reduction for spd matrices. In: Proceedings of European conference on computer vision (ECCV), p 1. Springer, Berlin
13. Hein M, Bousquet O (2005) Hilbertian metrics and positive definite kernels on probability measures. In: Proceedings of international conference on artificial intelligence & statistics, pp 136–143
14. Ho J, Xie Y, Vemuri B (2013) On a nonlinear generalization of sparse coding and dictionary learning. In: Proceedings of international conference on machine learning (ICML), pp 1480–1488
15. Jayasumana S, Hartley R, Salzmann M, Li H, Harandi M (2013) Kernel methods on the Riemannian manifold of symmetric positive definite matrices. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR)
16. Koenderink J, Van Doorn A (1999) The structure of locally orderless images. Int J Comput Vis 31(2–3):159–168
17. Kulis B, Sustik MA, Dhillon IS (2009) Low-rank kernel learning with bregman matrix divergences. J Mach Learn Res 10:341–376
18. Lazebnik S, Schmid C, Ponce J (2006) Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), vol 2, pp 2169–2178
19. Liu L, Wang L, Liu X (2011) In defense of soft-assignment coding. In: Proceedings of international conference on computer vision (ICCV), pp 2486–2493
20. Moreno PJ, Ho PP, Vasconcelos N (2003) A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In: Proceedings of advances in neural information processing systems (NIPS)

21. Olshausen BA et al (1996) Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature 381(6583):607–609
22. Pang Y, Yuan Y, Li X (2008) Gabor-based region covariance matrices for face recognition. IEEE Trans Circuits Syst Video Technol 18(7):989–993
23. Pennec X, Fillard P, Ayache N (2006) A Riemannian framework for tensor computing. Int J Comput Vis 66(1):41–66
24. Phillips PJ, Moon H, Rizvi SA, Rauss PJ (2000) The FERET evaluation methodology for face-recognition algorithms. IEEE Trans Pattern Anal Mach Intell 22(10):1090–1104
25. Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2326
26. Salehian H, Cheng G, Vemuri B, Ho J (2013) Recursive estimation of the stein center of spd matrices and its applications. In: Proceedings of international conference on computer vision (ICCV), pp 1793–1800
27. Sanin A, Sanderson C, Harandi M, Lovell B (2013) Spatio-temporal covariance descriptors for action and gesture recognition. In: IEEE workshop on applications of computer vision (WACV), pp 103–110
28. Sra S (2012) A new metric on the manifold of kernel matrices with application to matrix geometric means. In: Proceedings of advances in neural information processing systems (NIPS), pp 144–152
29. Tuzel O, Porikli F, Meer P (2006) Region covariance: a fast descriptor for detection and classification. In: Proceedings of European conference on computer vision (ECCV). Springer, Berlin, pp 589–600
30. Tuzel O, Porikli F, Meer P (2008) Pedestrian detection via classification on Riemannian manifolds. IEEE Trans Pattern Anal Mach Intell 30(10):1713–1727
31. van Gemert JC, Veenman CJ, Smeulders AW, Geusebroek JM (2010) Visual word ambiguity. IEEE Trans Pattern Anal Mach Intell 32(7):1271–1283
32. Wang Z, Vemuri BC (2004) An affine invariant tensor dissimilarity measure and its applications to tensor-valued image segmentation. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), vol 1, pp I–228
33. Wang J, Yang J, Yu K, Lv F, Huang T, Gong Y (2010) Locality-constrained linear coding for image classification. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), pp 3360–3367
34. Yang J, Yu K, Gong Y, Huang T (2009) Linear spatial pyramid matching using sparse coding for image classification. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), pp 1794–1801
35. Yu K, Zhang T, Gong Y (2009) Nonlinear learning using local coordinate coding. In: Proceedings of advances in neural information processing systems (NIPS), vol 9, p 1
36. Yuille AL, Rangarajan A (2003) The concave-convex procedure. Neural Comput 15(4): 915–936

Chapter 17
Summarization and Search Over Geometric Spaces

Nitesh Shroff, Rushil Anirudh, and Rama Chellappa

Abstract The last decade has seen an explosion in the amount of data being generated, in part due to the prevalence of image and video sensors. As a result, searching through these data for relevant information or even getting a gist of the data is increasingly becoming difficult. The task is further complicated when the data have a non-Euclidean geometric interpretation. In this chapter, we address these limitations by discussing techniques to (a) summarize the data and (b) search through the data to find the nearest neighbor, in the general case of data lying on non-Euclidean manifolds. First, we consider the "précis" problem of sampling K representative yet diverse data points from a large data set. We formulate a general theory which encompasses not only traditional techniques devised for vector spaces but also non-Euclidean manifolds, thereby enabling these techniques for shapes, human activities, textures, and many other image and video-based data sets. We discuss the intrinsic manifold measures for measuring the quality of a selection of points with respect to their representative power, and their diversity. We also extend our formulation to the infinite-dimensional manifolds. We then address the problem of nearest-neighbor search in curved spaces. Towards this end, we discuss geodesic hashing which employs intrinsic geodesic-based functions to hash the data for realizing approximate but fast nearest-neighbor retrieval. The proposed family of hashing functions, although intrinsic, is optimally selected to empirically satisfy the locality sensitive hashing property.

N. Shroff (✉)
Light, Co, Palo Alto, CA
e-mail: nitesh@light.co

R. Anirudh
Department of Electrical, Computer, and Energy Engineering,
Arizona State University, Tempe, AZ, USA
e-mail: ranirudh@asu.edu

R. Chellappa
Center for Automation Research, UMIACS, University of Maryland,
College Park, MD, USA
e-mail: rama@umiacs.umd.edu

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7_17

17.1 Introduction

Large databases of images and videos are increasingly becoming commonplace due to the growth of personal collections and Internet archives. Getting a gist of these data or searching through the enormous amounts of data to find the relevant information is increasingly becoming difficult as the size of the data sets is constantly increasing. Several applications require us to overcome this limitation and make these data sets more easily accessible to users. This has necessitated the development of methods that allow fast retrieval and access to information.

For instance, in applications such as content-based image retrieval, texture classification, biometrics, and video mining, the problem is frequently reduced to searching for exemplars similar to a query in a large database. This is more formally known as similarity search or the nearest-neighbor (NN) problem. The problem of NN search has been studied for many years in the database and algorithms communities involving searching in an n-dimensional Euclidean space. Several advances have been made here in the last couple of decades. However, the research has mainly focused on utilizing the geometry of the Euclidean space for a faster NN search. Hence, several of these techniques are not immediately applicable to data which has an underlying geometry that is non-Euclidean.

Another set of applications frequently requires us to sample K representative data points from a large data set. For instance, consider analyzing large data sets of shapes, objects, documents or large video sequences, etc. Analysts spend a large amount of time sifting through the acquired data to familiarize themselves with the content, before using them for their application-specific tasks. This has necessitated the problem of optimal selection of a few representative exemplars from the data set as an important step in exploratory data analysis. Other applications include Internet-based video summarization, where providing a quick overview of a video is important to improve the browsing experience. Similarly, in medical image analysis, picking a subset of K anatomical shapes from a large population helps in identifying the variations within and across shape classes, providing an invaluable tool for analysts.

In this chapter, we address these two problems and discuss techniques that are tailored to utilizing the geometry of the data. We first discuss a diverse sampling technique on the geometric spaces called the manifold précis. Subsequently, we generalize this to infinite-dimensional manifolds. Later in this chapter, we discuss an efficient algorithm for the nearest-neighbor search on non-Euclidean manifolds. Before getting into this, let us first look into various scenarios where such geometric representations are used and the various statistical learning techniques that have been developed.

Computations on Nonlinear Manifolds In the computer vision literature, over the past several years there has been significant research about what are good features and models for representing images and videos. Images and videos are usually represented by a concise set of features or models—such as shape, color/intensity histograms, SIFT, histogram of gradients, linear dynamical systems, and covariance matrices. Many of these features and models do not lie in the Euclidean space. What this means is that the underlying distance function on the space is not the usual ℓ_2/ℓ_p norm but a highly nonlinear function which is also in most cases computationally hard to compute.

Over the years, many advances have been made to understand the geometric properties of these varied spaces, and they have been utilized to devise more accurate inference and classification algorithms [18, 24, 39, 42, 45]. These works have addressed inferences, clustering, dimensionality reduction, etc. in non-Euclidean spaces. Understanding the geometry allows one to define distances, leading to geodesics, etc. on these manifolds. While one could try to solve the problem by obtaining an embedding of a given manifold into a larger ambient Euclidean space, it is desirable to have a solution that is more intrinsic in nature. This is because the chosen embedding is often arbitrary and introduces peculiarities that result from such extrinsic approaches. Further manifolds such as the Grassmannian or the manifold of infinite-dimensional diffeomorphisms do not admit a natural embedding into a vector space.

In this chapter, we develop techniques for summarizing and searching over the data sets. Summarization of a collection of data points is formulated as a subset selection problem. An efficient technique that optimizes for two conflicting criteria—representation and diversity—is developed. We then extend this formulation to infinite-dimensional manifolds. We then turn our attention towards the problem of searching over the non-Euclidean manifolds. Specifically, we focus on the problem of approximate nearest-neighbor search on the manifold. We discuss a technique called the geodesic hashing which extends the idea of locality sensitive hashing on the manifolds.

17.2 Subset Selection on Analytic Manifolds

In this section we discuss the problem of selecting a subset of K exemplars from a data set of N points when the data have an underlying manifold structure. We formulate the notion of representational error and diversity measure of exemplars while utilizing the non-Euclidean structure of the data points, followed by an efficient annealing-based optimization algorithm to solve for the exemplars. Towards this end, we develop a mathematical framework to concisely represent videos with an objective to gain a quick overview of the video while minimizing the loss of details. The problem is formulated as a subset selection problem with the objective to select representative yet diverse exemplars.

17.2.1 Related Work

Depending upon the application, several subset selection criteria have been proposed in the literature. However, there seems to be a consensus in selecting exemplars that are representative of the data set while minimizing the redundancy between the exemplars. Liu et al. [33] proposed that the summary of a document should satisfy the "coverage" and "orthogonality" criteria. Shroff et al. [36] extended this idea to selecting exemplars from videos that maximize "coverage" and "diversity." Simon et al. [37] formulated scene summarization as one of picking interesting and important scenes with minimal redundancy. Similarly, in statistics, stratified sampling techniques sample the population by dividing the data set into mutually exclusive and exhaustive "strata" (subgroups) followed by a random selection of representatives from each stratum [11]. The splitting of population into strata ensures that a diverse selection is obtained. The need to select diverse subsets has also been emphasized in information retrieval applications [6, 49].

Column Subset Selection (CSS) [5, 17, 26] has been one of the popular techniques to address this problem. The goal of CSS is to select the K most "well-conditioned" columns from the matrix of data points. One of the key assumptions behind this and other techniques is that objects or their representations lie in the Euclidean space. Unfortunately, this assumption is not valid in several cases when the data lie in non-Euclidean manifolds.

The problem of subset selection has also been studied by the communities of numerical linear algebra and theoretical computer science. Most work in the former community is related to the *Rank-Revealing* QR factorization (RRQR) [7, 25, 26]. Given a data matrix Y, the goal of RRQR factorization is to find a permutation matrix Π such that the QR factorization of $Y\Pi$ reveals the numerical rank of the matrix. The resultant matrix $Y\Pi$ has as its first K columns the most "well-conditioned" columns of the matrix Y. On the other hand, the latter community has focused on Column Subset Selection (CSS). The goal of CSS is to pick K columns forming a matrix $C \in \mathbb{R}^{m\times K}$ such that the residual $|| \ Y - \ P_C Y \ ||_\zeta$ is minimized over all possible choices for the matrix C. Here $P_C = CC^\dagger$ denotes the projection onto the K-dimensional space spanned by the columns of C, and ζ can represent the spectral or Frobenius norm. $C^\dagger$ indicates the pseudo-inverse of matrix C. Along these lines, different randomized algorithms have been proposed [5, 15, 17, 20]. Various approaches include a two-stage approach [5] and subspace sampling methods [17].

Clustering techniques [22] have also been applied for subset selection [16, 19]. In order to select K exemplars, data points are clustered into ℓ clusters with ($\ell \leq K$) followed by the selection of one or multiple exemplars from each cluster to obtain the best representation or low-rank approximation of each cluster. Affinity Propagation [19] is a clustering algorithm that takes similarity measures as input and recursively passes message between nodes until a set of exemplars emerges. As we discuss in this chapter, the problems with these approaches are that (a) the objective functions optimized by the clustering functions do not incorporate the diversity of the exemplars, hence can be biased towards denser clusters, and also by outliers, and (b) seeking low-rank approximation of the data matrix or clusters individually is not always an appropriate subset selection criterion. Furthermore, these techniques are largely tuned towards addressing the problem in a Euclidean setting and cannot be applied for data sets in non-Euclidean spaces.

We overcome these limitations by formulating the subset selection as the optimization of two criteria—representation and diversity. Both the criteria are individually formulated in a manner that utilizes the geometry. Let us first look into the formulation of the representational error.

17.2.2 Representational Error on Manifolds

Let us assume that we are given a set of points $X = \{x_1, x_2, \ldots x_n\}$ which belong to a manifold $\mathcal{M}$. The goal is to select a few exemplars $E = \{e_1, \ldots e_K\}$ from the set X, such that the exemplars provide a good representation of the given data points and are minimally redundant. For the special case of vector spaces, two common approaches for measuring representational error are in terms of linear spans and nearest-exemplar error. The linear span error is given by

$$\min_z \|\mathbf{X} - \mathbf{E}z\|_F^2 \tag{17.1}$$

where $\mathbf{X}$ is the matrix form of the data and $\mathbf{E}$ is a matrix of chosen exemplars. The nearest-exemplar error is given by

$$\sum_i \sum_{x_k \in \Phi_i} \|x_k - e_i\|^2 \tag{17.2}$$

where e_i is the ith exemplar and Φ_i corresponds to its Voronoi region.

Of these two measures, the notion of linear span, while appropriate for matrix approximation, is not particularly meaningful for general data set approximation problems since the "span" of a data set item does not carry perceptually meaningful information. For example, the linear span of a vector $x \in \mathbb{R}^n$ is the set of points $\alpha x, \alpha \in \mathbb{R}$. However, if x were an image, the linear span of x would be the set of images obtained by varying the global contrast level. All elements of this set however are perceptually equivalent, and one does not obtain any representational advantage from considering the span of x. Further, points sampled from the linear span of few images would not be meaningful images. This situation is further complicated for manifold-valued data such as shapes, where the notion of linear span does not exist. One could attempt to define the notion of linear spans on the manifold as the set of points lying on the geodesic shot from some fixed pole towards the given data set item. But, points sampled from this linear span might not be very meaningful, e.g., samples from the linear span of a few shapes would give physically meaningless shapes.

Hence, it is but natural to consider the representational error of a set X with respect to a set of exemplars E as follows:

$$J_{\text{rep}}(E) = \sum_i \sum_{x_j \in \Phi_i} d_g^2(x_j, e_i) \tag{17.3}$$

Here, d_g is the geodesic distance on the manifold and Φ_i is the Voronoi region of the ith exemplar. This boils down to the familiar K-means or K-medoids cost function for Euclidean spaces. In order to avoid combinatorial optimization involved in solving this problem, we use efficient approximations, i.e., we first find the mean followed by the selection of e_i as data point that is closest to the mean. The algorithm for optimizing J_{rep} is given in Algorithm 1. Similar to K-means clustering, a cluster label is assigned to each x_j followed by the computation of the mean μ_i for each cluster. This is further followed by selecting representative exemplar e_i as the data point closest to μ_i.

17.2.3 Diversity Measures on Manifolds

The next question we consider is to define the notion of diversity of a selection of points on a manifold. We first begin by examining equivalent constructions for $\mathbb{R}^n$. One of the ways to measure diversity is simply to use the sample variance of the points. This is similar to the construction used recently in [36]. For the case of manifolds, the sample variance can be replaced by the sample Karcher variance, given by the function:

Algorithm 1 Algorithm to minimize J_{rep}

```
Input :$X \in \mathcal{M}$, $k$, index vector $\omega$, $\Gamma$
Output: Permutation Matrix $\Pi$
Initialize $\Pi \leftarrow \mathcal{I}_{n\times n}$
for $\gamma \leftarrow [1 \dots \Gamma]$ do
    Initialize $\Pi^{(\gamma)} \leftarrow \mathcal{I}_{n\times n}$
    $e_i \leftarrow x_{\omega_i}$ for $i = \{1,2,\dots,k\}$
    for $i \leftarrow [1 \dots k]$ do
        $\Phi_i \leftarrow \{x_p : \text{argmin}_j\, d_g(x_p, e_j) = i\}$
        $\mu_i \leftarrow$ mean of $\Phi_i$
        $\hat{j} \leftarrow \text{argmin}_j\, d_g(x_j, \mu_i)$
        Update: $\Pi^{(\gamma)} \leftarrow \Pi^{(\gamma)}\, \Pi_{i\leftrightarrow\hat{j}}$
    end for
    Update: $\Pi \leftarrow \Pi\, \Pi^{(\gamma)}$, $\omega \leftarrow \omega\Pi^{(\gamma)}$
    if $\Pi^{(\gamma)} = \mathcal{I}_{n\times n}$ then
        break
    end if
end for
```

$$\rho(E) = \frac{1}{K}\sum_{i=1}^{K} d_g^2(\mu, e_i) \tag{17.4}$$

where μ is the Karcher mean [29] and the function value is the Karcher variance. However, this construction leads to highly inefficient optimization routines, essentially boiling down to a combinatorial search over all possible K-sized subsets of X.

An alternate formulation for vector spaces that results in highly efficient optimization routines is via Rank-Revealing QR (RRQR) factorizations. For vector spaces, given a set of vectors $\mathbf{X} = \{x_i\}$, written in matrix form $\mathbf{X}$, RRQR [26] aims to find Q, R, and a permutation matrix $\Pi \in \mathbb{R}^{n\times n}$ such that $\mathbf{X}\Pi = QR$ reveals the numerical rank of the matrix $\mathbf{X}$. This permutation

$$\mathbf{X}\Pi = (\mathbf{X}_K \;\; \mathbf{X}_{n-K}) \tag{17.5}$$

gives $\mathbf{X}_K$, the K most linearly independent columns of $\mathbf{X}$. This factorization is achieved by seeking Π which maximizes

$$\Lambda(\mathbf{X}_K) = \prod_i \sigma_i(\mathbf{X}_K) \tag{17.6}$$

the product of the singular values of the matrix $\mathbf{X}_K$.

For the case of manifolds, we adopt an approximate approach in order to measure diversity in terms of the "well-conditioned" nature of the set of exemplars projected on the tangent space at the mean. In particular, for the data set $\{x_i\} \subseteq \mathcal{M}$, with intrinsic mean μ, and a given selection of exemplars $\{e_j\}$, we measure the diversity of exemplars as follows: matrix $\mathbf{T}_E = [\log_\mu(e_j)]$ is obtained by projecting the exemplars $\{e_j\}$ on the tangent space at mean μ. Here, log() is the inverse exponential map on the manifold and gives tangent vectors at μ that point towards e_j. Diversity can then be quantified as

$$J_{\text{div}}(E) = \Lambda(\mathbf{T}_E) \tag{17.7}$$

where $\Lambda(\mathbf{T}_E)$ represents the product of the singular values of the matrix $\mathbf{T}_E$. For vector spaces, this measure is related to the sample variance of the chosen exemplars. For manifolds, this measure is related to the sample Karcher *variance*. If we denote $\mathbf{T}_X = [\log_\mu(x_i)]$, the matrix of tangent vectors corresponding to all data points, and if Π is the permutation matrix that orders the columns such that the first K columns of $\mathbf{T}_X$ correspond to the most diverse selection, then

$$J_{\text{div}}(E) = \Lambda(\mathbf{T}_E) = \det(R_{11}), \text{where, } \mathbf{T}_X\Pi = QR = Q\begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} \tag{17.8}$$

Algorithm 2 Algorithm for diversity maximization

```
**Input:** Matrix $V \in \mathbb{R}^{d\times n}$, $k$, Tolerance $tol$
**Output:** Permutation Matrix $\Pi$
Initialize $\Pi \leftarrow \mathcal{I}_{n\times n}$
**repeat**
    Compute $QR$ decomposition of $V$ to obtain $R_{11}, R_{12}$ and $R_{22}$ s.t., $V = Q\begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22}\end{pmatrix}$
    $\beta_{ij} \leftarrow \sqrt{(R_{11}^{-1}R_{12})_{ij}^2 + ||R_{22}\alpha_j||_2^2||\alpha_i^T R_{11}^{-1}||_2^2}$
    $\beta_m \leftarrow \max_{ij} \beta_{ij}$
    $(\hat{i}, \hat{j}) \leftarrow \arg\max_{ij} \beta_{ij}$
    Update: $\Pi \leftarrow \Pi \; \Pi_{i\leftrightarrow(j+k)}$
    $V \leftarrow V\Pi_{i\leftrightarrow(j+k)}$
**until** $\beta_m < tol$
```

Algorithm 3 Annealing-based alternation algorithm for subset selection on manifolds

```
**Input:** Data points $X = \{x_1, x_2, \dots, x_n\} \in \mathcal{M}$, Number of exemplars $k$, Tolerance step $\delta$
**Output:** $E = \{e_1, \dots e_k\} \subseteq X$
**Initial setup:**
Compute intrinsic mean $\mu$ of $X$
Compute tangent vectors $v_i \leftarrow \log_\mu(x_i)$
$V \leftarrow [v_1, v_2, \dots, v_n]$
$\omega \leftarrow [1, 2, \dots, n]$ be the $1 \times n$ index vector of $X$
$tol \leftarrow 1$
Initialize: $\Pi \leftarrow$ Randomly permute columns of $\mathcal{I}_{n\times n}$
Update: $V \leftarrow V\Pi, \omega \leftarrow \omega\Pi.$
**while** $\Pi \neq \mathcal{I}_{n\times n}$ **do**
    **Diversity:**
    $\Pi \leftarrow$ Div($V, k, tol$) as in Algorithm 2.
    Update: $V \leftarrow V\Pi$, $\omega \leftarrow \omega\Pi.$
    **Representative Error:**
    $\Pi \leftarrow$ Rep($X, k, \omega$,1) as in Algorithm 1
    Update: $V \leftarrow V\Pi$, $\omega \leftarrow \omega\Pi.$
    $tol \leftarrow tol + \delta$
**end while**
$e_i \leftarrow x_{\omega_i}$ for $i = \{1,2,\dots,k\}$
```

Here, $R_{11} \in \mathbb{R}^{K\times K}$ is the upper triangular matrix of $R \in \mathbb{R}^{n\times n}$, $R_{12} \in \mathbb{R}^{K\times(n-K)}$, and $R_{22} \in \mathbb{R}^{(n-K)\times(n-K)}$. The advantage of viewing the required quantity as the determinant of a sub-matrix on the right-hand side of the above equation is that one can obtain efficient techniques for optimizing this cost function. The algorithm for optimizing J_{div} is adopted from [26] and described in Algorithm 2. Input to the algorithm is a matrix V created by the tangent-space projection of X and output is the K most "well-conditioned" columns of V. This is achieved by first decomposing V into QR and computing β_{ij}, which indicates the benefit of swapping ith and jth columns [26]. Algorithm 3 then selects pair $(\hat{i}, \hat{j})$ corresponding to the maximum benefit swap β_m and if $\beta_m > tol$, this swap is accepted. This is repeated until either $\beta_m < tol$ or maximum number of iterations are completed.

17.2.4 Representation and Diversity Trade-Offs for Subset Selection

From (17.3) and (17.8), it can be seen that we seek a solution that represents a trade-off between two conflicting criteria. As an example, in Fig. 17.1a we show two cases, where J_{rep} and J_{div} are individually optimized. We can see that the solutions look quite different in each case. One way to write the global cost function is as a weighted combination of the two. However, such a formulation does not lend itself to efficient optimization routines (c.f. [36]). Further, the choice of weights is often left unjustified. Instead, we propose an annealing-based alternating technique of optimizing the conflicting criteria J_{rep} and J_{div}. Optimization algorithms for J_{rep} and J_{div} individually are given in Algorithms 1 and 2, respectively. We first optimize J_{div} to obtain an initial set of exemplars and use this set as an initialization for optimizing J_{rep}. The output of this stage is used as the current solution to further optimize J_{div}. However, with each iteration, we increase the tolerance parameter *tol* in Algorithm 2. This has the effect of accepting only those permutations that increase the diversity by a higher factor as iterations progress. This is done to ensure that the algorithm is guided towards convergence. If the *tol* value is not increased at each iteration, then optimizing J_{div} will continue to provide a new solution at each iteration that modifies the cost function only marginally. This is illustrated in Fig. 17.1c, where we show how the cost functions J_{rep} and J_{div} exhibit an oscillatory behavior if annealing is not used.

As shown in Fig. 17.1b, the convergence of J_{div} and J_{rep} is obtained very quickly on using the proposed annealing alternation technique. The complete annealing-based alternation algorithm is described in Algorithm 3. A technical detail to be noted here is that for Algorithm 2, input matrix $V \in \mathbb{R}^{d \times n}$ should have $d \geq k$. For cases where $d < k$, Algorithm 2 can be replaced by its extension proposed in [5]. Table 17.1(a) shows the notations introduced in Algorithms 1–3. $\Pi_{i \leftrightarrow j}$ is obtained by permuting i and j columns of the identity matrix.

17.2.5 Complexity, Special Cases, and Limitations

In this section, we discuss how the proposed method relates to the special case of $\mathcal{M} = \mathbb{R}^n$ and to sub-manifolds of $\mathbb{R}^n$ specified by a large number of samples. For the case of $\mathbb{R}^n$, the cost functions J_{rep} and J_{div} boil down to the familiar notions of clustering and low-rank matrix approximation, respectively. In this case, Algorithm 3 reduces to alternation between clustering and matrix approximation with the annealing ensuring that the algorithm converges. This results in a new algorithm for subset selection in vector spaces.

For the case of manifolds implicitly specified using samples, one can approach the problem in one of two ways. The first would be to obtain an embedding of the space into a Euclidean space and apply the special case of the algorithm for

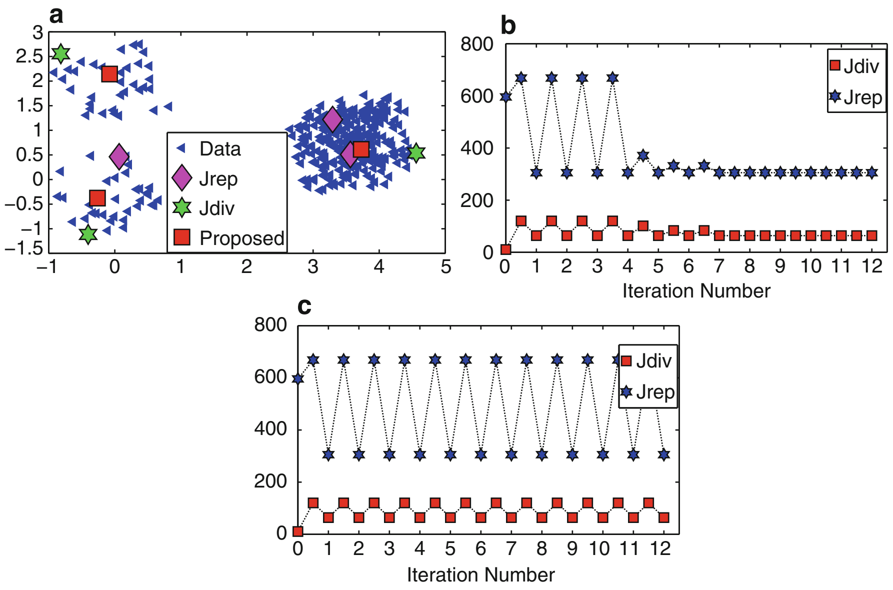


Fig. 17.1 Subset selection for a simple data set consisting of unbalanced classes in $\mathbb{R}^4$. (**a**) Data projected on $\mathbb{R}^2$ for visualization using PCA. While minimizing the representational error, J_{rep} picks two exemplars from the dominant class. J_{div} picks diverse exemplars but from the boundaries. The proposed approach strikes a balance between the two and picks one "representative" exemplar from each class. Convergence analysis of algorithm 3: (**b**) with annealing and (**c**) without annealing

$\mathcal{M} = \mathbb{R}^n$. The embedding here needs to preserve the geodesic distances between all pairs of points. Multi dimensional scaling can be used for this purpose. However, recent methods have also focused on estimating logarithmic maps numerically from sampled data points [32]. This would make the algorithm directly applicable for such cases, without the need for a separate embedding. Thus the proposed formalism can accommodate manifolds with known and unknown geometries.

Computational Complexity The computational complexity of computing exponential map and its inverse is specific to each manifold. Let n be the number of data points and K be the number of exemplars to be selected. Table 17.1(b) enumerates the complexity of different computational step of the algorithm. The last two rows show the complexity of an efficient algorithm proposed by [21] to compute the exponential map and its inverse for the case of Grassmann manifold $\mathcal{G}_{m,p}$.

17.2.6 Extension to Infinite-Dimensional Manifolds

The formalism described previously works well in cases where data lie on finite-dimensional manifolds, but extending it to infinite-dimensional manifolds poses

Table 17.1 (a) Notations used in Algorithms 1–3 (b) Complexity of Computational Steps

(a)

Symbol	Represents
Γ	Maximum number of iterations
$\mathcal{I}_{n\times n}$	Identity matrix
Φ_i	Voronoi region of ith exemplar
$\Pi_{i\leftrightarrow j}$	Permutation matrix that swaps columns i and j
$\Pi^{(\gamma)}$	Π in the γth iteration
V	Matrix obtained by tangent-space projection of X
H_{ij}	(i,j) element of matrix H
α_j	jth column of the identity matrix
$H\alpha_j, \alpha_j^T H$	jth column and row of matrix H, respectively

(b)

Computational step	Complexity
$\mathcal{M}$ exp. map (assume)	$O(\nu)$
$\mathcal{M}$ inverse exp. map (assume)	$O(\chi)$
Intrinsic mean of X	$O((n\chi + \nu)\Gamma)$
Projection of X to tangent space	$O(n\chi)$
Geodesic distances in Algorithm 1	$O(nK\chi)$
K intrinsic means	$O((n\chi + K\nu)\Gamma)$
Algorithm 2	$O(mnK \log n)$
$\mathcal{G}_{m,p}$ exponential map	$O(p^3)$
$\mathcal{G}_{m,p}$ inverse exponential map	$O(p^3)$

challenges due to the difficulty in formulating the diversity cost function. Examples of such manifolds are diffeomorphisms [43], space of closed curves [34], and trajectories on manifolds [41], which find uses in several applications. For instance, most smoothly varying features extracted from videos can be interpreted as trajectories on the corresponding feature manifolds.

The diversity measure J_{div} can be formulated purely in terms of pairwise geodesics, making it extensible to infinite-dimensional manifolds but it would have made the optimization a significant bottleneck, as already discussed earlier. Instead, we employ the Transport Square Root Velocity Function (TSRVF) [41]—a recent development in statistics, which models trajectories on manifolds and continuous curves in the Euclidean space [40] as functions. In both the cases, the resulting metric between two sequences on the functional space remains unchanged to identical time warpings. We use the TSRVF representation to model all trajectories as deviations from an "average trajectory" using tangent vectors.

Algorithm 4 describes the procedure to obtain exemplar sequences using précis. Here, the term $V(i)$ can be interpreted as a "sequence tangent," in that it takes us from the average activity, $\mu(t)$, to the ith activity in unit time. $V(i)$ contains the shooting vectors, which are the tangent vectors one would travel along, starting from the average sequence $\mu(t)$ at $\tau = 0$ to reach the ith action $\tilde{\alpha}_i(t)$ at time $\tau = 1$. Note here that τ is the time in the TSRVF space which is different from t, which is time in the original shape space. In comparison to traditional manifold précis, the matrix V is equivalent to $\mathbf{T}_X$ in Eq. (17.8). Since V can be high dimensional, depending on the choice of the feature and length of each sequence, solving for exemplars from a large set of sequences can still be computationally intractable.

An alternative solution could be to project V onto a lower-dimensional space, which can be achieved with any of the existing algorithms due to the Euclidean nature of V. Finally, one can solve for exemplars in the projected space efficiently.

17.2.7 Experiments

Baselines We compare the proposed algorithm with two baselines. The first baseline is a clustering-based solution to subset selection, where we cluster the dataset into K clusters and pick as exemplar the data point that is closest to the cluster centroid. Since clustering optimizes only the representation cost function, we do not expect it to have the diversity of the proposed algorithm. This corresponds to the special case of optimizing only J_{rep}. The second baseline is to apply a tangent-space approximation to the entire data set at the mean of the data set and then apply a subset-selection algorithm such as RRQR. This corresponds to optimizing only J_{div} where the input matrix is the matrix of tangent vectors. Since minimization of J_{rep} is not explicitly enforced, we do not expect the exemplars to be the best representatives, even though the set is diverse.

Synthetic Illustration To gain some intuition, we first perform experiments on a simple synthetic data set. For easy visualization and understanding, we generated a data set with three unbalanced classes in Euclidean space $\mathbb{R}^4$. Individual cost functions J_{rep} and J_{div} were first optimized to pick three exemplars using Algorithms 1 and 2, respectively. Selected exemplars are shown in Fig. 17.1a, where the four-dimensional data set has been projected into two dimensions for visualization using Principal Component Analysis (PCA). Despite unbalanced class sizes, when optimized individually, J_{div} seeks to select exemplars from diverse classes but tends to pick them from the class boundaries, while unbalanced class sizes cause J_{rep} to pick two exemplars from the dominant cluster. Algorithm 3 iteratively optimizes for both these cost functions and picks an exemplar from every class. These exemplars are closer to the centroid of the individual classes.

Figure 17.1b shows the convergence of the algorithm for this simple data set and compares it with the case when no annealing is applied (Fig. 17.1c). J_{rep} and J_{div} plots are shown as the iterations of Algorithm 3 progresses. When annealing is applied, the tolerance value (tol) is increased by 0.05 in each iteration. It can be noted that in this case the algorithm converges to a steady state in 7 iterations ($tol = 1.35$). If no annealing is applied, the algorithm does not converge.

Shape Sampling/Summarization We conducted a real shape summarization experiment on the MPEG data set [31]. This dataset has 70 shape classes with 20 shapes per class. For our experiments, we created a smaller data set of ten shape classes with ten shapes per class. Figure 17.2a shows the shapes used in our experiments. We use an affine-invariant representation of shapes based on landmarks. Shape boundaries are uniformly sampled to obtain m landmark points. These points are concatenated in a matrix to obtain the landmark matrix $\mathcal{L} \in \mathbb{R}^{m\times 2}$. Left singular vectors ($U_{m\times 2}$), obtained by the singular value decomposition of

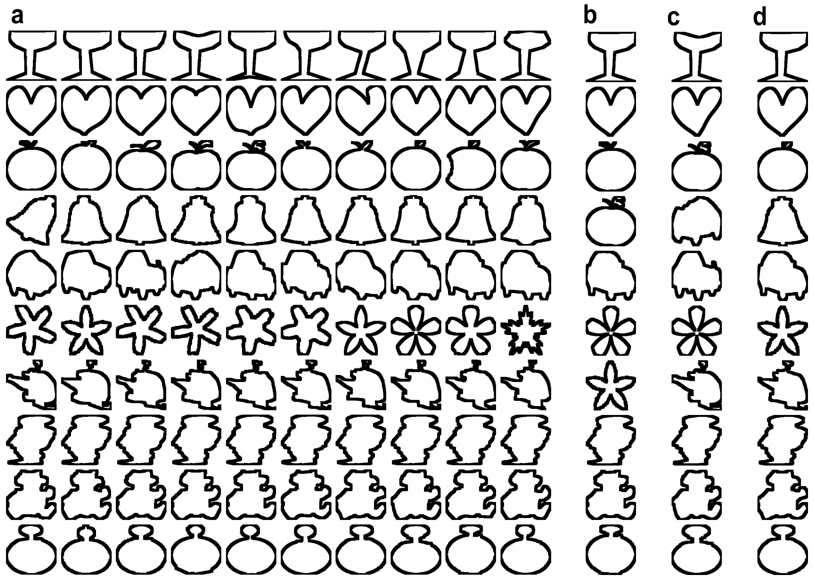


Fig. 17.2 (**a**) 10 classes from MPEG data set with ten shapes per class. Comparison of ten exemplars selected by (**b**) J_{rep}, (**c**) J_{div}, and (**d**) proposed approach. J_{rep} picks two exemplars each from two classes ("apple" and "flower") and misses "bell" and "chopper" classes. J_{div} picks one from eight different classes, two exemplars from class "car," and none from class "bell." It can be observed that exemplars chosen by J_{div} for classes "glass," "heart," "flower" and "apple" tend to be unusual members of the class. It also picks up the flipped car, while the proposed approach picks one representative exemplars from each class as desired

matrix $\mathcal{L} = U\Sigma V^T$, give the affine-invariant representation of shapes [3]. This affine shape space U of m landmark points is a two-dimensional subspace of $\mathbb{R}^m$. These p-dimensional subspaces in $\mathbb{R}^m$ constitute the Grassmann manifold $\mathcal{G}_{m,p}$. Details of the algorithms for the computation of exponential and inverse exponential maps on $\mathcal{G}_{m,p}$ can be found in [21].

Cardinality of the subset in the experiment was set to ten. As the number of shape classes is also ten, one would ideally seek one exemplar from each class. Algorithms 1 and 2 were first individually optimized to select the optimal subset. Algorithm 1 was applied intrinsically on the manifold with multiple initializations. Figure 17.2b shows the output with the least cost among these initializations. For Algorithm 2, data points were projected on the tangent space at the mean using the inverse exponential map and the selected subset is shown in Fig. 17.2c. Individual optimization of J_{rep} results in one exemplar each from six classes, two each from two classes ("apple" and "flower") and misses two classes ("bell" and "chopper"), while, individual optimization of J_{div} alone picks one each from eight classes, two from the class "car," and none from the class "bell." It can be observed that exemplars chosen by J_{div} for classes "glass," "heart," "flower," and "apple" tend to be unusual members of the class. It also picks up the flipped car. Optimizing for both J_{div} and J_{rep} using Algorithm 3 picks one "representative" exemplar from each class as shown in Fig. 17.2d.

Diversely Sampling Human Actions To further demonstrate the generalization of précis to infinite-dimensional manifolds, we look at diversely sampling human actions from the UMD actions data set [46] which contains ten unique actions such as *bend, jog, squat, throw, etc.* repeated ten times by the same actor. Due to a relatively static background, we are able to extract the shape silhouette of the human, following which we sample the silhouettes uniformly to represent each silhouette as a point on the Grassmann manifold as described earlier. The smooth nature of human actions allows us to model them as trajectories on the Grassmann manifold. Finally, we choose exemplars, actions using Algorithm 4.

To show the effectiveness of précis, we constructed a collection of actions that were chosen such that different classes had significantly different populations, a distribution of the action classes is shown in Fig. 17.3a. As our baseline, we first sampled this data set using K-medoids as shown in Fig. 17.3c. As expected, classes

Algorithm 4 Functional manifold précis

```
Input: $\alpha_1(t), \alpha_2(t) \dots \alpha_N(t)$
Output: $K$ Exemplar Sequences
Compute Karcher mean [41] $\mu(t)$, which also aligns $\tilde{\alpha}_1(t), \tilde{\alpha}_2(t) \dots \tilde{\alpha}_N(t)$.
for $i \leftarrow [1 \dots N]$ do
    for $t \leftarrow [1 \dots T]$ do
        Compute shooting vectors $v(i,t) \in T_{\mu(t)}(M)$ as $v(i,t) = exp^{-1}_{\mu(t)}(\tilde{\alpha}_i(t))$
    end for
    Define $V(i) = [v(i,1)^T \; v(i,2)^T \; \dots \; v(i,T)^T]^T$
end for
Pass $V$ to algorithm 2 to maximize diversity and to algorithm 1 to minimize representational error.
```

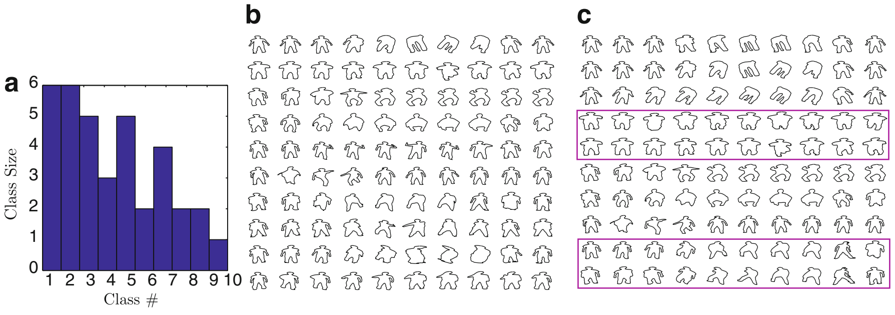


Fig. 17.3 *Sampling effectively from a skewed collection of sequences*, particularly when the proportion of classes is skewed. Our proposed Précis algorithm is able to sample uniformly from all classes as compared to K-medoids which picks more samples (marked) from classes that have a higher representation which are actions #1,#2, and #7 here. (**a**) A dataset with skewed class proportions. (**b**) Row-wise: actions sampled by functional-Précis. (**c**) Row-wise: centers obtained using functional K-medoids

which have a higher population are overrepresented in the chosen samples. On the other hand, précis samples these collection of actions without getting biased by the distribution of the classes as shown in Fig. 17.3b.

Effect of Parameters and Initialization In our experiments, the effect of tolerance steps (δ) for smaller values (< 0.1) has very minimal effect. After a few attempts, we fixed this value to 0.05 for all our experiments. In the first iteration, we start with $tol = 1$. With this value, Algorithm 2 accepts any swap that increases J_{div}. This makes output of Algorithm 2 after first iteration almost insensitive to initialization. While, in the later iterations, swaps are accepted only if they increase the value of J_{div} significantly and hence input to Algorithm 2 becomes more important with the increase in *tol*.

17.3 Fast Nearest Neighbor on Analytic Manifolds

Let us now switch gears and focus on the problem of searching for the nearest neighbor over these Geometric spaces. Nearest-Neighbor search is a widely used technique in pattern recognition and computer vision. It is a lazy learning technique where all the training points are stored, and during testing, distance of the query point from each training point is evaluated to identify the nearest training point. This causes exhaustive NN search to grow linearly with the number of training data points. This becomes significantly more computationally intense when distance computation between every pair of points is intensive. For instance, consider the Grassmann manifold ($\mathcal{G}_{m,p}$) which is the space of p-dimensional subspaces embedded in $\mathbb{R}^m$. Geodesic computation between a pair of points on $\mathcal{G}_{m,p}$ requires $O(p^3)$ time. This intensive nature of NN has led to interest in developing fast NN search techniques both for Euclidean and non-Euclidean spaces.

In the relatively better understood domain of searching in Euclidean spaces, numerous techniques that drastically reduce the search time (both algorithms and structures) have been developed over the last couple of decades. These techniques have been studied in many different fields and can be broadly categorized as (a) exact and (b) approximate NN search techniques. Some recent surveys of these techniques can be found in [9, 27, 35]. Another notable survey, especially from computer vision perspective, is by Kumar et al. [30] which provides a comprehensive comparison of exact search methods applied to patch-based image searching.

Exact methods usually rely on space partitioning. Examples of this approach include quad-trees, $k - d$ trees, and ball trees. The resulting data structure is represented as a tree with the root node being the entire data set, child nodes representing partitions and leaf nodes representing individual data points. PCA trees and $k - d$ trees are "projective trees" as they split points based on their projection into some lower-dimensional space. These techniques require the coordinates of the points to split data hierarchically. This restricts the application of these techniques in metric spaces where only a metric is defined on the space. Another disadvantage with $k-d$ trees is that for dimensions larger than ten, the algorithm becomes very inefficient and tends towards a linear scan algorithm. PCA trees overcome the restriction of $k-d$ trees by splitting data points such that the boundary is aligned to one of the axes. So, now it can find the variance of the data and split according to the hyperplane that splits the maximum variance.

Approximate methods in Euclidean spaces became popular when locality sensitive hashing (LSH) algorithms [23, 28] were introduced. The original LSH algorithm was proposed by Gionis et al. [23] which was tuned for “Hamming” spaces, i.e., spaces where the underlying distance function is the Hamming distance. This was extended to Euclidean spaces in [14]. A good introduction and survey of these methods can be found in [12]. The core idea of LSH techniques is to efficiently construct binary codes for high-dimensional points while ensuring that points that are nearby in high-dimensional space are close even in the binary code space. Brief overview of LSH technique is discussed later in section “Hashing”.

All these methods rely heavily on the assumption that points are in $\mathbb{R}^n$ with the underlying metric being the ℓ_2 norm. These methods can also be applied directly to non-Euclidean data points if the manifold of interest can somehow be embedded into the Euclidean space. In this part of the chapter, we discuss that it is important to consider the Riemannian geometry of manifolds to more systematically address this problem. This is because there exist several analytic manifolds which cannot in any easy way be embedded into an ambient Euclidean space. For instance, the space of linear subspaces or the Grassmann manifold is best treated as a quotient space of the orthogonal group and there is no easy natural embedding into an ambient vector space [38].

Nearest-neighbor search in metric spaces when only an underlying metric is known has also been studied in literature. Several NN algorithms have been devised for indexing data with arbitrary distance functions such as vantage point trees, metric trees, and multi-vantage point trees. An excellent survey of these methods can be found in [9]. These are exact methods of searching in metric spaces. Unfortunately, tree-based indexing methods such as these often suffer from the curse of dimensionality. For large dimensions the performance is not significantly different from simple brute-force search. Approximate search methods in arbitrary metric spaces have also been proposed [1]. Indyk and Motwani [28] provided algorithms for approximate searches when the underlying distance function is the ℓ_∞ norm. In some cases the underlying non-Euclidean distance measure can be replaced with another which can be efficiently computed. Several methods have been proposed for embedding arbitrary spaces into Euclidean or pseudo-Euclidean space [4, 47]. Embedding methods substitute a fast approximate distance for the original distance and hence are approximate search methods.

We note that fast NN retrieval for manifold data points is still an emerging area and hence the literature on it is scarce. The work of [13] shows how to adapt the standard k-d tree algorithm to low-dimensional manifolds whose structure is unknown. The work on approximate nearest subspace search [2] is an example of indexing on non-Euclidean manifolds. However, this work is limited in applicability to subspaces; moreover, it does not exploit the Riemannian geometric interpretation of subspaces. Recently, there has been two works addressing this problem of adapting NN techniques from vector space to non-Euclidean manifolds. Turaga and Chellappa [44] proposed an approximate NN technique via embedding of data into the tangent space at a pole using the log-Euclidean mapping. Similarly, Chaudhry and Ivanov [8] used log-Euclidean embedding to develop a spectral hashing method. However, both these techniques are based on Euclidean embedding of the Riemannian manifold through tangent spaces.

We overcome this limitation by proposing an approximate NN technique that does not require the Euclidean embedding of the data. Before diving into the approximate NN search, it is worth noting that the generalized clustering technique discussed in the first half of the chapter can be easily extended to a hierarchical clustering technique. This immediately provides us with an exact NN search technique. However, the complexity of this technique would be still high. But, if one would be willing to accept a small reduction in the accuracy for a large gain in speed, efficient approximate NN techniques can be developed on non-Euclidean manifolds. Towards this, we discuss a Geodesic Hashing technique which employs intrinsic geodesic-based functions to hash the training data set.

17.3.1 Intrinsic Approximate Nearest-Neighbor Search

In this section, we introduce Geodesic Hashing, an intrinsic approach for approximate NN search on manifolds. But before that, we review the hashing technique and the property of Locality Sensitive Hashing (LSH) method.

Hashing

Hashing was originally studied in the field of cryptography and text databases which involved entities such as passwords, names, addresses, etc. where finding exact matches was the key requirement. Hashing image and video data brought additional challenges since no two semantically related images or videos are exactly the same. This brought about a new class of techniques—LSH [23]—that could answer approximate NN queries in a fast manner.

A good introduction and survey of LSH can be found in [12]. Here, we briefly review the basic concepts of Euclidean LSH. LSH attempts to solve a problem called the (r, ϵ)-neighbor problem. The problem is described as follows: Given a database of points $X = \{x_i\}$ in $\mathbb{R}^n$ and a query q, if there exists a point $x \in X$ such that $d(x, q) \leq r$, then with high probability, a point $x' \in X$ is retrieved such that $d(x', q) \leq (1 + \epsilon)r$. LSH solves this problem by constructing a family of hash functions $\mathcal{H}$ over X called (r_1, r_2, p_1, p_2) locality sensitive, if for any $u, v \in X$

$$d(u, v) \leq r_1 \quad \Rightarrow \quad Pr_{h\in\mathcal{H}}(h(u) = h(v)) \geq p_1 \tag{17.9}$$

$$d(u, v) \geq r_2 \quad \Rightarrow \quad Pr_{h \in \mathcal{H}}(h(u) = h(v)) \leq p_2 \tag{17.10}$$

A (r_1, r_2, p_1, p_2) locality-sensitive family of hashing function solves the (r, ϵ)-neighbor problem by choosing $r_1 = r$ and $r_2 = (1 + \epsilon)r$. Popular choices of h include random projections, i.e., $h(v) = \text{sgn}(v^T r)$ where r is a randomly chosen unit vector and sgn is the signum function. In this case, h is binary valued taking values in $\{+1, -1\}$. A generalization of this is termed random projections using "p-stable" distributions [14], where

$$h(v) = \left\lfloor \frac{v^T r + b}{w} \right\rfloor$$

where r is a randomly chosen direction whose entries are chosen independently from a stable distribution and b is a random number chosen between $[0, w]$. In this case, the hash function takes on integer values. A k-bit hash is constructed by appending k randomly chosen hash functions $H(x) = [h_1(x), h_2(x), \ldots, h_k(x)]$. Thus, $H \in \mathcal{H}^k$. Then, L hash tables are constructed by randomly choosing $H_1, H_2 \ldots H_L \in \mathcal{H}^k$. All the training examples are hashed into the L hash tables. For a query point q, an exhaustive search is carried out among the examples in the union of the L hash buckets indexed by q. Appropriate choices of k and L ensure that the algorithm succeeds in finding a (r, ϵ)-NN of the query q with a high probability.

Geodesic Hashing

We now discuss Geodesic Hashing (GH) which utilizes geodesics on the manifold to hash data lying on a non-Euclidean manifold. Let $\mathcal{M}$ represent the manifold, and $\mathcal{X} = \{x_1, x_2, \ldots x_n\} \in \mathcal{M}$ be the database of n points. The first step in our formulation is to define a family $\mathcal{H}$ of hash functions over $\mathcal{M}$ which is (r_1, r_2, p_1, p_2)- sensitive. We obtain this by seeking the generalization of the binary-valued random projection $(h(v) = \text{sgn}(v^T r))$ hashing function to the Riemannian manifold.

In Euclidean space, this family of hash function is created by randomly picking a projection direction r. Binary hash value for a data point v is then obtained by looking at the orientation of v w.r.t. r. In Euclidean geometry, the notion of "origin" is very well defined. Hence, hashing family is defined well by selecting a set of directions. On the other hand, non-Euclidean manifolds do not have a point that naturally serves as the "origin" for the manifold. One could attempt to define an analog of the origin for manifolds by computing the intrinsic mean of the data and use it as a "pole" to project data points on the tangent space of the pole. This approach was recently taken by [44]. As the tangent space is Euclidean, LSH techniques can directly be applied on the projected data. This approximation works reasonably well if the data spread is small. But, as the data-spread increases, the approximation increasingly becomes crude. In order to overcome this, Chaudhry and Ivanov [8] proposed computing poles for individual clusters. Individual data points are then hashed in the tangent space of the closest pole. This reduces the error in data approximation. However, this method relies heavily on finding the nearest pole and incurs large errors for points near the boundary of two clusters. Moreover, the choice of number of clusters could be critical to the performance of the hashing technique.

Attempting to define one or few poles which serve as the analog of “origin” for non-Euclidean manifolds followed by Euclidean embedding of data leads to crude approximation of data. We can better generalize hashing for manifolds by looking at the configuration of data points w.r.t. a randomly chosen geodesic on the manifold. Geodesics on manifolds are analogs of “lines” in Euclidean spaces, and computing a query point’s orientation w.r.t. the geodesic does not require us to make any approximation. Consequently, this helps us to get rid of the reliance of LSH techniques on the artificially created notion of origin. This, in turn, leads to a better way of hashing antipodal points.

In order to formulate this, let $g_{a,b}$ represent a geodesic specified by the pair of points (x_a, x_b) s.t. both $x_a, x_b \in \mathcal{M}$. Now, given a query point $q \in \mathcal{M}$, one can look at the projection of q on $g_{a,b}$. In Euclidean space, projection of a point on a vector is a relatively cheaper operation. But, in Riemannian manifolds, projecting a query point on a geodesic is a costly optimization problem [18]. Hence, choosing any projection-based functions would significantly increase the query time. So, instead we can look at the orientation of the point q w.r.t. the geodesic $g_{a,b}$. Specifically, this is obtained by defining a binary function $h_{(x_a,x_b)}(q) : \mathcal{M} \to \{-1, +1\}$ such that

$$h_{(x_a,x_b)}(q) = \mathrm{sgn}(\mathrm{Log}_{x_a}(x_b)^T \mathrm{Log}_{x_a}(q)) \tag{17.11}$$

So, given a set of geodesics $\mathbb{G}$, this provides us a family of hashing functions

$$\mathcal{H}_{\mathrm{GH}}(\mathbb{G}) = \{h_{(x_a,x_b)} \mid g_{a,b} \in \mathbb{G}\} \tag{17.12}$$

It can be noted here that the family of functions defined using Eq. (17.11) is a very rich family and is specified completely by $\mathbb{G}$. Having defined the family of hash functions, we can append k-random binary hash functions h sampled from $\mathcal{H}_{\mathrm{GH}}(\mathbb{G})$ to define a k-bit hash function. Then, L hash tables are constructed by randomly choosing $H_1, H_2, \ldots, H_L \in \mathcal{H}^k$. This way, we obtain an intrinsic indexing and retrieval method on the manifold in the LSH framework.

Now, in order to verify the above-mentioned intuition, we synthetically generated a dataset on $\mathcal{G}_{9,3}$. We adopted the procedure described in [10] to generate this dataset which has 4 randomly generated clusters with 200 points each. We empirically computed the probability of collision between two points (u, v)

$$Pr_{h\in\mathcal{H}}\big(h(u) = h(v)\big)$$

Fig. 17.4a shows the probability of collision with increase in the geodesic distance between the points averaged over 5 such randomly generated data sets. We computed this for both tangent space LSH [44] (blue curve) and Geodesic Hashing

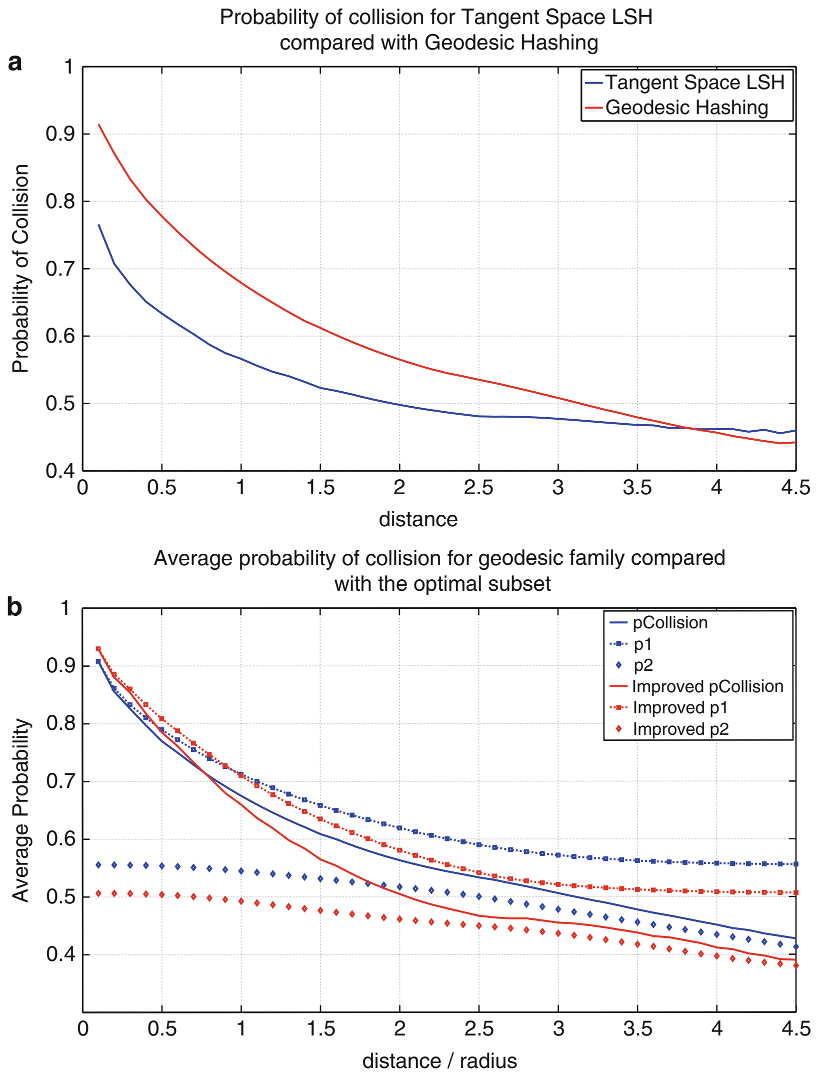


Fig. 17.4 (**a**) Comparison of probability of collision for tangent space LSH [44] with that for geodesic hashing. It can be noted that Geodesic hashing has the desirable properties of (i) higher probability of collision when the distance is smaller and (ii) smaller probability of collision when the distance is large. (**b**) LSH property evaluation. *Curves in blue* show the empirical evaluation of LSH properties for geodesic hashing. The *blue curve* marked as "pCollision" shows the monotonic decrease in probability of collision with increase in distance. Similarly, p_1 and p_2 values, as defined in Eqs. (17.9)–(17.10), are shown with radius values (r_1, r_2), respectively on the x-axis. The *red curves* show the improvement in collision probabilities along with improvements in p_1 and p_2 by optimally selecting the set of geodesics

(red curve). It can be noted that tangent space LSH due to approximate embedding of data sees a deterioration in the collision probability even for points that are nearby.

Satisfying LSH Property In the previous section, given a set of geodesics $\mathbb{G}$ from the manifold $\mathcal{M}$, we created a family $\mathcal{H}_{\text{GH}}(\mathbb{G})$ of hashing functions. Now, we verify if the proposed family satisfies the LSH properties which requires that $\mathcal{H}_{\text{GH}}(\mathbb{G})$ should satisfy Eqs. (17.9)–(17.10) over $\mathcal{M}$. For this, similar to previous section, we generated a synthetic data set on $\mathcal{G}_{9,3}$ and empirically evaluated the probability of collision for a randomly chosen set of geodesics $\mathbb{G}$. The collision probability, thus computed, has been averaged over 20 random selection of $\mathbb{G}$ and is shown in Fig. 17.4b. This figure shows that the probability of collision monotonically decreases with increase in pairwise distance thus satisfying an important LSH property. Moreover, as defined in Eq. (17.9), we computed the probability of collision p_1 for points within a ball of radius r_1. Figure 17.4b shows the variation of p_1 with the increasing radius r_1. Similarly, this figure also shows the variation of p_2 with r_2 defined in Eq. (17.10). It can be seen here that for any choice of (r_1, r_2) such that $r_2 > r_1$, corresponding collision probabilities satisfy $p_1 > p_2$ and $p_1 > 0.5$, hence guaranteeing the usefulness of this LSH family [23].

17.3.2 Reducing the Hashing Cost

When compared to the exact NN techniques, LSH techniques require us to compute distance of a query point with a very small number of data points. Hence, in Euclidean space, where the pairwise distance computation is relatively cheaper, it suffices to have defined a family of hashing functions. However, as discussed earlier in the case of non-Euclidean manifolds, geodesic distance computations are costly operations. This implies that a further reduction in the average number of points that collide with a query point for a hashing family $\mathcal{H}_{\text{GH}}(\mathbb{G})$ is required. This can be achieved by optimally selecting the set of geodesics $\mathbb{G}$, with an eventual goal to significantly reduce the query time. Although optimal selection of $\mathbb{G}$ requires more processing during training, it can significantly reduce the query time.

This requires us to address various criteria that the set of geodesics should meet w.r.t. the statistics of the database $\mathfrak{X}$, in order to allow fewer collisions while retaining high accuracy. On an individual level, a good geodesic should satisfy the following criteria:

1. The binary sequence generated by a geodesic for $\mathfrak{X}$ should have equal probability of ones and zeros.
2. Geodesics should have large value of p_1 and smaller value of p_2.

Moreover, the group of geodesics $\mathbb{G}$ should satisfy an additional criterion that the binary sequence generated by a geodesic should be independent of that generated by another geodesic. Weiss et al. [48] presented a similar discussion from the perspective of a good code. Formally, these criteria can be written as the following:

Let us denote the probability of collision of nearby points by

$$C_r(q,p) = Pr_{h \in \mathcal{H}_{\mathrm{GH}}(\mathbb{G})}\,[h(q) = h(p)] \qquad \text{when} \qquad d_g(q,p) \le r$$

Then, the optimization problem can be written as

$$\max_{\mathbb{G}} \quad C_r(q,p)$$

s.t. if $g_{a,b} \in \mathbb{G}$, then

$$Pr_{q \in \mathfrak{X}}[h_{(x_a,x_b)}(q) = 1] = 0.5$$

and for $g_{a',b'} \in \mathbb{G}$

$$Pr_{q \in \mathfrak{X}}[h_{(x_a,x_b)}(q) = h_{(x_{a'},x_{b'})}(q)] = 0.5$$

Optimization An optimal selection of $\mathbb{G}$, with $|\mathbb{G}| = \tau \ge kL$, that satisfies the above criteria can be achieved by the following procedure: The set of geodesics on the manifold $\mathcal{M}$ is very large. So, in order to reduce the search space, we restrict ourselves to the geodesics formed by randomly selecting a pair of points from $\mathfrak{X}$. Let $\mathbb{G}_{\mathfrak{X}}$ represent this set of geodesics. It should be noted here that even after this reduction in search space, the number of possible geodesics $\binom{|\mathbb{G}_{\mathfrak{X}}|=n}{2}$ is still very large. So, we randomly select a set of geodesics $\mathbb{G}_R \subseteq \mathbb{G}_{\mathfrak{X}}$ with $|\mathbb{G}_R| = m >> \tau$. This is followed by evaluating criterion 1 for individual geodesics over a smaller uniformly sampled subset $\mathcal{X} \subseteq \mathfrak{X}$. This provides us $\mathbb{G}_E \subseteq \mathbb{G}_R$, the set of geodesics with equal probabilities of ones and zeros. For the set $\mathbb{G}_E$, p_1 on $\mathcal{X}$ is computed. We then greedily populate the set $\mathbb{G}_O$ with geodesics of high p_1 values on the condition that they are not correlated to any geodesic $g_{a,b}$ already in $\mathbb{G}_O$.

Figure 17.4b shows the improvement in the probability of collision w.r.t. distance, and p_1 w.r.t. r_1, and p_2 w.r.t. r_2. As expected, the optimized family of geodesics increases the collision probability for nearby points and reduces it for far away points. In other words, points close to a query point will have better probability of colliding and points far away will have lesser probability of colliding. This signifies that for the same accuracy, the number of colliding points is reduced.

17.4 Conclusion

In this chapter, we addressed the problems of searching and summarizing large data sets which has an underlying non-Euclidean geometry to it. We utilized the geometric structure of the manifold to formulate the notion of picking exemplars which minimize the representational error while maximizing the diversity of exemplars. An iterative alternation optimization technique based on annealing has been discussed. We demonstrated the application of the subset selection in the sampling of shapes and human actions. We then turned our attention towards the nearest-neighbor search on the non-Euclidean manifolds. We discussed an intrinsic geodesic hashing technique for the problem of approximate NN search on non-Euclidean manifolds. This technique utilizes the orientation of a data point w.r.t. a set of geodesics to hash the data. A technique is devised to optimally select the set of geodesics, which in turn defines the hashing family.

Acknowledgements Authors would like to acknowledge Prof. Pavan Turaga for their collaboration in the work.

References

1. Athitsos V, Potamias M, Papapetrou P, Kollios G (2008) Nearest neighbor retrieval using distance-based hashing. In: IEEE international conference on data engineering
2. Basri R, Hassner T, Zelnik Manor L (2007) Approximate nearest subspace search with applications to pattern recognition. In: IEEE conference on computer vision and pattern recognition, pp 1–8
3. Begelfor E, Werman M (2006) Affine invariance revisited. In: IEEE Conference on Computer Vision and Pattern Recognition
4. Bourgain J (1985) On Lipschitz embedding of finite metric spaces in hilbert space. Israel Journal of Mathematics 52(1):46–52
5. Boutsidis C, Mahoney M, Drineas P (2009) An improved approximation algorithm for the column subset selection problem. In: ACM-SIAM Symposium on Discrete Algorithms.
6. Carbonell J, Goldstein J (1998) The use of MMR, diversity-based reranking for reordering documents and reproducing summaries. In: Special Interest Group on Information Retrieval
7. Chan T (1987) Rank revealing QR factorizations. Linear Algebra Appl 88:67–82
8. Chaudhry R, Ivanov Y (2010) Fast approximate nearest neighbor methods for non-euclidean manifolds with applications to human activity analysis in videos. In: European Conference on Computer Vision
9. Chávez E, Navarro G, Baeza-Yates R, Marroquín J (2001) Searching in metric spaces. ACM Computing Surveys 33(3):273–321
10. Chikuse Y (2003) Statistics on special manifolds. Lecture notes in statistics. Springer, New York
11. Cochran W (1977) Sampling techniques. Wiley, New York
12. Darrell T, Indyk P, Shakhnarovich GE (2006) Nearest neighbor methods in learning and vision: theory and practice. MIT Press, Cambridge
13. Dasgupta S, Freund Y (2008) Random projection trees and low dimensional manifolds. In: ACM symposium on theory of computing, pp 537–546
14. Datar M, Immorlica N, Indyk P, Mirrokni VS (2004) Locality-sensitive hashing scheme based on p-stable distributions. In: Proceedings of the symposium on computational geometry, pp 253–262
15. Deshpande A, Rademacher L (2010) Efficient volume sampling for row/column subset selection. In: Foundations of computer science (FOCS)
16. Dhillon I, Modha D (2001) Concept decompositions for large sparse text data using clustering. Mach Learn 42(1):143–175
17. Drineas P, Mahoney M, Muthukrishnan S (2008) Relative-error CUR matrix decompositions. SIAM J Matrix Anal Appl 30:844–881

18. Fletcher PT, Lu C, Pizer SM, Joshi SC (2004) Principal geodesic analysis for the study of nonlinear statistics of shape. IEEE Transactions on Medical Imaging 23(8):995–1005
19. Frey BJ, Dueck D (2007) Clustering by passing messages between data points. Science 315:972–976
20. Frieze A, Kannan R, Vempala S (2004) Fast Monte-Carlo algorithms for finding low-rank approximations. Journal of the ACM 51(6):1025–1041
21. Gallivan K, Srivastava A, Liu X, Van Dooren P (2003) Efficient algorithms for inferences on grassmann manifolds. In: IEEE workshop on statistical signal processing
22. Gan G, Ma C, Wu J (2007) Data clustering: theory, algorithms, and applications. Society for Industrial and Applied Mathematics, Philadelphia
23. Gionis A, Indyk P, Motwani R (1999) Similarity search in high dimensions via hashing. In: Proceedings of the international conference on very large data bases, pp 518–529
24. Goh A, Vidal R (2008) Clustering and dimensionality reduction on riemannian manifolds. In: International Conference on Computer Vision and Pattern Recognition
25. Golub G (1965) Numerical methods for solving linear least squares problems. Numerische Mathematik 7(3):206–216
26. Gu M, Eisenstat S (1996) Efficient algorithms for computing a strong rank-revealing QR factorization. SIAM Journal on Scientific Computing 17(4):848–869
27. Hjaltason G, Samet H (2003) Index-driven similarity search in metric spaces (survey article). ACM Transactions on Database Systems 28(4):517–580
28. Indyk P, Motwani R (1998) Approximate nearest neighbors: towards removing the curse of dimensionality. In: ACM Symposium on Theory of Computing, pp 604–613
29. Karcher H (1977) Riemannian center of mass and mollifier smoothing. Communications on Pure and Applied Mathematics 30(5):509–541
30. Kumar N, Zhang L, Nayar S (2008) What is a good nearest neighbors algorithm for finding similar patches in images? In: European conference on computer vision, pp 364–378
31. Latecki L, Lakamper R, Eckhardt T (2000) Shape descriptors for non-rigid shapes with a single closed contour. In: IEEE Conference on Computer Vision and Pattern Recognition
32. Lin T, Zha H (2008) Riemannian manifold learning. IEEE Transactions on Pattern Analysis and Machine Intelligence 30:796–809
33. Liu K, Terzi E, Grandison T (2008) ManyAspects: a system for highlighting diverse concepts in documents. Proc Very Large Data Bases Endowment 1(2):1444–1447
34. Mio W, Srivastava A, Joshi S (2007) On shape of plane elastic curves. International Journal of Computer Vision 73(3):307–324
35. Shakhnarovich G, Darrell T, Indyk P (2006) Nearest-neighbor methods in learning and vision. MIT Press, Cambridge
36. Shroff N, Turaga P, Chellappa R (2010) Video précis: highlighting diverse aspects of videos. IEEE Transactions on Multimedia 12(8):853 –868
37. Simon I, Snavely N, Seitz S (2007) Scene summarization for online image collections. In: International Conference on Computer Vision
38. Spivak M (1999) A comprehensive introduction to differential geometry, vol 1, 3rd edn. Publish or Perish, Houston
39. Srivastava A, Joshi SH, Mio W, Liu X (2005) Statistical shape analysis: clustering, learning, and testing. Transactions on Pattern Analysis and Machine Intelligence 27(4):590–602
40. Srivastava A, Klassen E, Joshi SH, Jermyn IH (2011) Shape analysis of elastic curves in euclidean spaces. IEEE Transactions on Pattern Analysis and Machine Intelligence 33:1415–1428
41. Su J, Kurtek S, Klassen E, Srivastava A (2014) Statistical analysis of trajectories on Riemannian manifolds: bird migration, hurricane tracking, and video surveillance. Ann Appl Stat 8(1):530–552
42. Subbarao R, Meer P (2006) Nonlinear mean shift for clustering over analytic manifolds. In: International Conference on Computer Vision and Pattern Recognition
43. Trouvé A (1998) Diffeomorphisms groups and pattern matching in image analysis. Int J Comput Vis 28:213–221

44. Turaga P, Chellappa R (2010) Nearest-neighbor search algorithms on non-euclidean manifolds for computer vision applications. In: Indian Conference on Computer Vision, Graphics and Image Processing
45. Turaga PK, Veeraraghavan A, Chellappa R (2008) Statistical analysis on stiefel and Grassmann manifolds with applications in computer vision. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 1–8
46. Veeraraghavan A, Chellappa R, Roy-Chowdhury AK (2006) The function space of an activity. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 959–968
47. Wang X, Wang JT, Lin KI, Shasha D, Shapiro BA, Zhang K (2000) An index structure for data mining and clustering. Knowl Inf Syst 2(2):161–184
48. Weiss Y, Torralba A, Fergus R (2008) Spectral hashing. In: Neural Information on Processing Systems
49. Yue Y, Joachims T (2008) Predicting diverse subsets using structural svms. In: International Conference on Machine Learning

Index

P.K. Turaga, A. Srivastava (eds.), *Riemannian Computing in Computer Vision*,
DOI 10.1007/978-3-319-22957-7