Online Learning on Spheres

Krzysztof A. Krakowski[1], Robert E. Mahony[1],
Robert C. Williamson[2], and Manfred K. Warmuth[3]

[1] Department of Engineering, Australian National University,
Canberra, ACT 0200, Australia
[2] National ICT Australia Ltd (NICTA), and
Research School of Information Sciences and Engineering,
Australian National University, Canberra, ACT 2601, Australia
[3] Department of Computer Science, University of California,
Santa Cruz, CA 95064, USA (work performed whilst at NICTA)

Abstract. The classical on-line learning algorithms such as the LMS (Least Mean Square) algorithm have attracted renewed interest recently because of many variants based on non-standard parametrisations that bias the algorithms in favour of certain prior knowledge of the problem. Tools such as link functions and Bregman divergences have been used in the design and analysis of such algorithms. In this paper we reconsider the development of such variants of classical stochastic gradient descent (SGD) in a purely geometric setting. The Bregman divergence is replaced by the (squared) Riemannian distance. The property of convexity of a loss function is replaced by a more general notion of compatibility with a metric. The ideas are explicated in the development and analysis of an algorithm for online learning on spheres.

1 Introduction

Stochastic Gradient Descent (SGD) algorithms for online learning have been very widely studied. Recently much work has focussed on different parameterisations which build a bias into the algorithm. Examples such as exponentiated gradient descent [1] perform much better when their bias (sparsity of the target weight vector) is satisfied compared to the classical LMS algorithm. A systematic theory has been developed to understand these algorithms using the tools of link functions and Bregman divergences [2].

These variants of stochastic gradient descent seem to take account of some geometry, but do not fully respect it. For example, the normalised EG algorithm (in which the parameter vector is restricted to a simplex) [3,1] does not (even approximately) follow a geodesic in its updates (A geodesic is the shortest path between two points on the parameter manifold). Furthermore, they do not allow a clear connection with a Bayesian interpretation and motivation of the algorithms.

Mahony & Williamson [4] have shown how one can derive all of the algorithms available in the link-function framework directly from a Bayesian prior. This works by noting that a distribution (the prior) and a metric can both be used to assign a value to the volume of a set. They posited that the distribution and the metric both give the same value to the volume of an arbitrary set thus allowing a choice of metric which matches the distribution. The algorithms utilise the natural gradient and (approximately) follow geodesics. Whilst a nice connection, the development of performance bounds eluded the authors.

In this paper we start from scratch and develop stochastic gradient descent with respect to a general metric. Such an idea has been considered before, notably by Amari [5] who was motivated to match an online learning algorithm to the probabilistic generative model which generates the data. We do not assume the data is generated probabilistically, but do assume there are some constraints on the parameter vector that can be expressed in terms of restrictions to a manifold. We develop algorithms fully respecting the geometry imposed and do all of the analysis in terms of Riemannian distances rather than Bregman divergences.

Given a class of functions (linear) and a parameterisation (restriction to a manifold) it turns out that one is not completely free to choose a loss function. Perhaps more surprisingly, the classical assumption of convexity of the loss function is not sufficient to ensure convergence of the algorithm. Convexity *is* sufficient for algorithms in "flat" metrics where the geodesics are all straight lines. In Section 4 we present a general notion of compatibility of a loss function that reduces to convexity when the metric is flat.

We illustrate the new framework by developing and analysing an algorithm on the simplest curved manifold — the sphere. Whilst there is a classical literature on statistics on spheres (e.g., [6,7]), to the best of our knowledge this is the first SGD algorithm specifically developed on the sphere. Our hope is that the new framework will lead to improved algorithms and analyses for existing constraints (such as on a simplex or even the space of semidefinite matrices [8]) as well as new algorithms for new problems.

2 Preliminaries and Notation

Consider the standard on-line learning setting where the learning algorithm maintains a parameter p_t. In each trial the algorithm receives an instance $x_t \in \mathbb{R}^n$, forms a prediction $\hat{y}_t$, and receives a label y_t. Finally the algorithm incurs a loss $\mathcal{L}(\hat{y}_t, y_t)$ and updates its parameter vector based on this feedback.

For the sake of concreteness we restrict ourselves to linear regression, i.e.,

$$\hat{y}_t = f_p(x) := \langle p, x \rangle \,. \tag{1}$$

Now assume you know that the target parameter lies in a parameter set $\mathbf{M} \subseteq \mathbb{R}^n$. Two main cases have been studied: $\mathbf{M} = \mathbb{R}^n$ (i.e., no restriction) and $\mathbf{M} = \mathcal{P}^{n-1}$ is the $n-1$ dimensional probability simplex in $\mathbb{R}^n$ (or in general any subset of $\mathbb{R}^n$ defined by linear constraints). The work of Kivinen and Warmuth [1] and others has shown that algorithms that exploit the knowledge that the target lies on the simplex can have radically better performance.

So what about other subsets of $\mathbb{R}^n$? Since we want to use variational techniques it is reasonable to assume that $\mathbf{M}$ is a Riemannian manifold. Arguably the simplest natural case to study (besides the simplex $\mathcal{P}^{n-1}$) is restricting the parameter space $\mathbf{M}$ to the n dimensional sphere $\mathbf{S}^{n-1}$. We use $\mathbf{M} = \mathbf{S}^{n-1}$ as the main example of this paper even though our methodology is developed for arbitrary manifolds.

So how do we best take advantage for the fact that the target lies on the sphere? First we can do a standard gradient descent update (such as the Widrow Hoff update). Now the updated parameter p'_t will be off the sphere. However the following heuristic (or kludge) will again produce a parameter on the sphere: Replace the intermediate parameter p_t' by the point p_{t+1} on the sphere that is closest to p_t'. This method only weakly respects the geometry of the parameter space.

A second method involves Bregman divergences between pairs of parameters and link functions (which are closely related to the derivatives of Bregman divergences) [9]. A suitable link function has $\mathbf{M}$ as its domain and $\mathbb{R}^n$ as its range and the update has the form:

$$p_{t+1} = h^{-1}(h(p_t) - \eta \boldsymbol{\partial}_p L(\underbrace{\langle p_t, x_t \rangle}_{\hat{y}_t}, y_t).$$

Since h^{-1} is applied at the end, the updated parameter stays in $\mathbf{M}$. For the case of the simplex, the method has been quite successful: One can use as the Bregman divergence the relative entropy. In this case the link h is essentially a componentwise logarithm and the inverse h^{-1} the softmax function. However, the choice of the link function is still rather at hoc and the resulting update (the EG update) does not follow a geodesic path. Moreover, we don't know have any natural link functions for the sphere.

In this paper we replace the Bregman divergence by the squared Riemannian distance between parameters on the manifold $\mathbf{M}$. We plug this squared distance into the usual cost function [1] for deriving on-line updates. Before we do this we need to discuss loss functions that are amenable to our differential geometry viewpoint.

In this section we propose a general structure for parametric on-line learning problems that incorporates the possibility of nonlinear geometry.

An on-line learning problem consists of several components as follows:

Input Space Ω typically a subset of $\mathbb{R}^n$.
Sampling process Σ**:** At each time-step of the algorithm the sampling process $(x_t, y_t) = \Sigma(t)$ provides a data point $(x_t, y_t) \in \Omega \times \mathbb{R}$.
Parametrised model class (PMC) $(f_p, \mathbf{M})$**:** The parametrised model class is the parametric class of functions $\{x \mapsto f_p(x) \colon p \in \mathbf{M}\}$. The function $f_p \colon \Omega \to \mathbb{R}$, is a continuously differentiable real valued function for each p. The parameter set $\mathbf{M}$ is a Riemannian manifold. The predictions of the algorithm are $\hat{y} = f_p(x)$.
Loss function $\mathcal{L}$**:** A continuously differentiable function $\mathcal{L}(p, (x, y))$, $\mathcal{L} \colon \mathbf{M} \times \Omega \times \mathbb{R} \to \mathbb{R}$. Given Σ, we sometimes will use the *instantaneous loss* $\mathcal{L}_t(p) := \mathcal{L}(p, (x_t, y_t))$.

Typically, the sampling process $\Sigma(t)$ is derived from a generative noise model

$$y_t = f(x_t) + \mu_t, \tag{2}$$

where μ_t is a given noise process and $f \colon \Omega \to \mathbb{R}$ is an unknown function. The input x_t may itself be a realisation of a stochastic process on $\Omega \subset \mathbb{R}^n$ or a deterministic process provided by the user. In this paper we will deal only with the noise free case to simplify the treatment of the underlying geometry. The most common on-line learning problem involves the class of linear predictors as a parametrised model class (1). The parameter $p \in \mathbb{R}^n$ lies in Euclidean space and is called the *weight vector* or just *weight*. A wide range of general learning problems can be reformulated into the structure of a linear predictor model.

It is important to separate the choice of loss function from the specification of the PMC. The most common on-line learning problem involves a Gaussian generative noise model. The associated loss function is the squared error loss

$$\mathcal{L}(p_t, (x_t, y_t)) = (y_t - \langle p_t, x_t \rangle)^2 \,. \tag{3}$$

This is associated with the log likelihood of the probability distribution associated with the parameter p for the given measured data point. Observe that in this case $\mathcal{L}(p_t, (x_t, y_t)) = \widetilde{\mathcal{L}}(y_t, \hat{y}_t)$, which is the more traditional way of defining the loss function.

The information theoretic structure of the Gaussian generative noise model is geometrically flat [10]. Assuming for the moment that a true parameter $p_\star$ exists, the squared loss function can be interpreted as a scaled degenerate squared distance measure associated with the flat structure of the generative noise model

$$\mathcal{L}(p_t, (x_t, y_t)) = \|x_t\|^2 \left\langle p_\star - p_t, \frac{x_t}{\|x_t\|} \right\rangle^2 ,$$

where $\|x_t\|$ is the standard 2-norm in $\mathbb{R}^n$. Here, the projected component of the distance $p_\star - p_t$ in the direction x_t is measured by the loss function. Once the flat geometric structure associated with the generative noise model is no longer valid it is clear that the squared error loss function loses much of its motivation.

The goal of a (parametric) on-line learning algorithm is to progressively refine an estimate $f_{p_t}(x)$, $t = 0, 1, \dots, k-1$ to minimize the expected or cumulative value of the loss $\mathcal{L}(p_t, (x_t, y_t))$. A key concept in the design and analysis of on-line learning algorithms is that there is only a small change in parameter estimate made at any one time step. Typically, there is a small positive constant η at each step of the algorithm, called the *learning rate* or *step-size* that limits the change in p_t. In noisy environments the step-size is chosen small to limit the effect of noise disturbing the averaged convergence properties of the algorithm. In less noisy environments the step-size can be chosen larger.

Consider a learning problem specified by $\Omega, \Sigma, f_p, \mathbf{M}$ and $\mathcal{L}$. In the online learning algorithm data points (x_t, y_t) are given one at the time. The algorithm derives a new weight p_{t+1} based on the latest data (x_t, y_t) and the current weight p_t via the *update rule* $\mathfrak{A}\colon \mathbf{M} \times \Omega \times \mathbb{R} \to \mathbf{M}$. Hence the rule of the on-line learning algorithm may be written as

$$p_{t+1} = \mathfrak{A}(p_t, (x_t, y_t)).$$

The simplest and most widely known algorithm is the *Widrow-Hoff* algorithm given by

$$p_{t+1} = p_t - \eta\, \boldsymbol{\partial}_p \mathcal{L}(p_t, (x_t, y_t)), \tag{4}$$

where $\boldsymbol{\partial}_p$ is the vector of partial differentials $\dfrac{\partial}{\partial p^i}$. A number of generalisations of the Widrow-Hoff algorithm were proposed in the literature. Amari [5] investigates the *natural gradient* algorithm

$$p_{t+1} = p_t - \eta\, G^{-1}\, \boldsymbol{\partial}_p \mathcal{L}(p_t, (x_t, y_t)), \tag{5}$$

where $G^{-1} = \left(g^{ij}\right)$ is the inverse of the metric g in the parameter space. The *natural gradient* $G^{-1}\, \boldsymbol{\partial}_p$ is the "right" intrinsic notion of a gradient in a non-Euclidean space. It is the steepest direction of a function on a manifold. Another approach is taken by Warmuth *et al.* [2,9] who introduced a concept of *link functions*. A link function h maps parameter space $\mathbf{M}$ to $\mathbb{R}^n$ and hence the update rule (4) is modified to

$$p_{t+1} = h^{-1}(h(p_t) - \eta\, \boldsymbol{\partial}_p \mathcal{L}(p_t, (x_t, y_t))), \tag{6}$$

where h^{-1} is the inverse of the link function h. Whilst allowing the use of different $\mathbf{M}$, the update (6) can be shown to not follow a geodesic in general.

3 Two Simple Examples

In this section we present two simple examples of on-line learning algorithms. First, we review the well known stochastic gradient descent algorithm posed on Euclidean space. Second, we propose a learning problem posed on the sphere. The non-Euclidean problem considered has a simple geometric structure that clearly demonstrates the key ideas in the paper. All derivations are made formally in the noise free case to simplify the development and focus on the geometric structure.

3.1 Linear Predictor Model in $\mathbb{R}^n$

We assume there is a true $p_\star$ and the sample process $\Sigma(t)$ is given by $(\langle p_\star, x_t\rangle, x_t) = \Sigma(t) \in \mathbb{R}\times\mathbb{R}^n$ where $x_t \in \mathbb{R}^n$ is sampled as desired. The PMC is $\hat{y}_t = \langle p_t, x_t\rangle, \quad p_t \in \mathbf{M} = \mathbb{R}^n$. The loss function is the squared error loss

$$\mathcal{L}_t(p) = \mathcal{L}(p, (x_t, y_t)) = (\langle p, x_t\rangle - y_t)^2 \tag{7}$$

The Widrow-Hoff learning algorithm 4 takes the familiar form

$$p_{t+1} = p_t - 2\eta\, (\langle p, x_t\rangle - y_t)\, x_t.$$

3.2 Scale Invariant Linear Predictor

We first motivate the choice of $\mathbf{M} = \mathbf{S}^{n-1}$. Suppose the sample process $\Sigma(t)$ generates $x_t \in \mathbf{S}^{n-1}$ so $\|x_t\| = 1$ One can think of the data as scale invariant; that is, the generative model gives that same output for any scaling of the data point $y_t = f(x_t) = f(\alpha x_t), \quad \forall \alpha \in \mathbb{R}, x_t \in \mathbb{R}^n$.

The PMC for the scale invariant linear predictor is $f_p(x) = \langle p_t, x_t\rangle, \quad p_t \in \mathbf{M} = \mathbf{S}^{n-1}$. Again we suppose a true parameter $p_\star$ exists: i.e., $y_t = \langle x_t, p_\star\rangle$, for all (x_y, y_t) generated by $\Sigma(t)$. Due to the scale invariance one has $\langle p_\star, \alpha x_t\rangle = \langle p_\star, x_t\rangle$ and consequently $\langle p_\star, x_t\rangle = 0$. This does not imply that $p_\star = 0$, simply that the data x_t is always orthogonal to $p_\star$. Clearly, any $\alpha p_\star$ for $\alpha \in \mathbb{R}$ is also a valid estimate of the parametric model. It is thus natural to constrain the parameter estimates to a constant norm $\|p\| = 1$. Furthermore, $p = 0$ will generate a perfect estimate of the true output and provide no information about the true parameter value $p_\star$. Thus, it is necessary to apply a norm constraint to the parameter p to avoid the on-line learning algorithm learning the null hypothesis.

Problems of the above nature are common in identification of data dependencies. For example in identification of an ARMA model the data received can be entered into a Hankel matrix. An estimate of the underlying model, expressed as a weight vector, is characterised as the zero eigenvector of the Hankel matrix. In terms of an on-line learning problem, the columns of the Hankel matrix $h_t \in \mathbb{R}^n$ may be taken as data samples. The goal of the learning algorithm is to refine the parameter estimate p_t that makes $\langle h_t, p\rangle$ small.

The instantaneous loss function that we propose here is

$$\mathcal{L}_t(p) = \mathcal{L}(p, (x_t)) = \langle p, x_t \rangle^2 . \tag{8}$$

Note that $\mathcal{L}_t \colon \mathbf{S}^{n-1} \to \mathbb{R}$. Thus, although it appears to be a Euclidean squared error loss, the underlying geometry of the problem will result in interesting structure. The choice of loss function is discussed in detail in Section 4.

The algorithm update on the unit sphere $\mathbf{S}^{n-1}$ can be written as

$$p_{t+1} = p_t \, \cos\left(\eta \, \|V_t\|\right) - \frac{V_t}{\|V_t\|} \sin\left(\eta \, \|V_t\|\right),$$

where $V_t = 2 \langle p_t, x_t \rangle \left(x_t - \langle p_t, x_t \rangle \, p_t\right)$ and the norm $\|V_t\|$ is the standard norm of the vector V_t in $\mathbb{R}^n$. A detailed derivation for this algorithm is given in Section 5.

4 Parametrised Model Classes and Loss Functions

In this section we propose a definition for compatibility of a loss function and a parameterised model class (PMC). It is interesting to note that the concept of convexity that is often taken for granted in the Euclidean case does not generalise to the case of non-flat PMCs. The weaker concept of compatibility is justified by reference to the examples introduced in Section 3.

Definition 1. *Consider a parameterised model class* $(f_p, \mathbf{M})$. *The* solution set *is defined to be*

$$S_t(0) := \{ p \in \mathbf{M} \; : \; y_t = f_p(x_t) \} .$$

The solution set of a PMC depends explicitly on the data sample at each time instant. The solution set characterises the set of parameters for which a given data sample provides no further information on the parameter performance. An important (although trivial) observation is that for a single output learning algorithm $y_t \in \mathbb{R}$ solution set is generically an $(n-1)$-dimensional hyper-surface in $\mathbf{M}$. For $\lambda \in \mathbb{R}$ we define a *generalised solution set*

$$S_t(\lambda) := \{ p \in \mathbf{M} \; : \; \lambda = y_t - f_p(x_t) \} .$$

Let $p \in \mathbf{M}$ be a regular point of the function $g_t(p) = y_t - f_p(x_t)$. The parameter $\lambda \in \mathbb{R}$ is a *regular value* if $dg_t \colon \mathcal{T}_p\mathbf{M} \to \mathbb{R}$ is surjective at any $p \in g_t^{-1}(\lambda)$. If λ is a regular value of $g_t(p)$ then the preimage $S_t(\lambda)$ is a submanifold of $\mathbf{M}$ of codimension 1, a hyper-surface of dimension $n-1$. If λ_0 is a regular value of $g_t(p)$ then locally the sets $S(\lambda)$, $|\lambda - \lambda_0|$ small, provide a foliation of $\mathbf{M}$ (*cf.* Figure 1). That is, there exist local coordinates $\xi = (\xi_1, \ldots, \xi_{n-1}, \lambda)$ for $\mathbf{M}$ such that the nth coordinate characterises the change in parameter λ and the first $(n-1)$ coordinates parameterise the surface S_λ.

The following definition is important in understanding and choosing good loss functions for non-linear PMCs.

Definition 2 (Compatibility).

Weak Compatibility *A parametrised model class and loss function $\mathcal{L}$ are called* weakly compatible *if the minimal level set of $\mathcal{L}_t(p)$ is the solution set $S_t(0)$.*

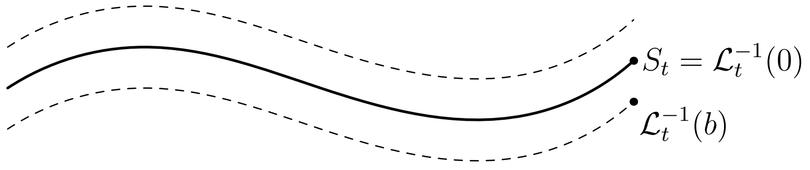

Fig. 1. For a fixed data sample (x_t, y_t) the solution set $S_t(0)$ is a hyper-surface of $\mathbf{M}$ if 0 is a regular point of the function $g_t(p) = y_t - f_p(x_t)$. In this case parameter space has a local foliation into the sets $S_t(\lambda)$. A loss function $\mathcal{L}$ is weakly compatible if its level sets generate the same foliation of $\mathbf{M}$.

Output Compatibility *A parametrised model class and loss function $\mathcal{L}$ are called* output compatible *if they are weakly compatible and the level sets of $\mathcal{L}_t(p)$ are the generalised solution sets $S_t(\lambda)$.*

Compatibility *A parametrised model class and loss function $\mathcal{L}$ are called* compatible *if they are weakly compatible and if for any given data sample and any geodesic $\gamma\colon [0,1] \to \mathbf{M}$ such that $\dot{\gamma}(0)$ is orthogonal to the level set of $\mathcal{L}_t$ passing through $\gamma(0)$, then $\dot{\gamma}(t)$ is orthogonal to the level set of $\mathcal{L}_t$ passing through $\gamma(t)$ for all t.*

Strong Compatibility *A parametrised model class and loss function $\mathcal{L}$ are called* strongly compatible *if they are compatible and if the level set $\mathcal{L}_t(p) = 0$ is a totally geodesic submanifold in $\mathbf{M}$.*

Convex Compatibility *A parametrised model class and loss function $\mathcal{L}$ are called* convexly compatible *if they are weakly compatible and if the loss function $\mathcal{L}_t(p) = 0$ is convex with respect to the geometry of $\mathbf{M}$.*

The goal of a loss function is to measure an error associated with a given parameter for the present measured data. The minimum that should be required of a loss function is that it measures its minimum cost when the parameter estimate produces the best output possible over parameter space. This is the concept of weak compatibility in Definition 2. Thus, an algorithm that acts to reduce the loss at each sample time will drive the parameter estimate closer to the best parameter estimate for the model at the intersection of all solution sets for the observed data. In both the examples posed, Section 3.1 and 3.2, it is clear that the PMC and loss function are weakly compatible.

Output compatibility is a natural consequence of the squared error structure of classical loss functions. For a generative noise model $y_t = f_p(x_t) + \mu_t$ consider a loss of the form $\mathcal{L}_t(p) = F(|y_t - f_p(x_t)|)$, where $F : \mathbb{R} \to \mathbb{R}$ is a monotonic increasing function with $F(0) = 0$. For the classical squared error loss functions $F(x) = x^2$. Any loss function of this form will be output compatible with the PMC. In both the examples posed, Section 3.1 and 3.2, the structure of the loss function is based on a least squares error criterion and the PMC and loss function are output compatible.

The definition of compatibility is important because it links the underlying geometry of the PMC to the structure of the loss function. The concept of compatibility is stronger than weak and output compatibility since the earlier definitions do not refer to the underlying Riemannian geometry of the PMC in any way. Compatibility plays an important role in the derivation of the on-line learning algorithms presented in Section 5. Anticipating the formal details given in the following section, the update at a point p_t is made along the geodesic passing through p_t with velocity $-\mathrm{grad}\mathcal{L}$. If the PMC and loss function are compatible then the gradient vector field of the loss is parallel to the geodesic along the update step. Thus, the instantaneous decrease in loss along the update step is maximised and the update step generates the maximal total decrease in loss for a step of that length. In both the examples posed, Section 3.1 and 3.2, the PMC and the loss function are compatible. This is clear in the Euclidean case as both the geodesics and the level sets of the loss function are straight lines. In the case of the sphere, the geodesics are great circles while the level sets of the loss function are small circles. Only the solution set, at the equator for a given data sample x_t, is both a geodesic and a level set of the cost function (*cf.* Figure 2).

It is possible to hypothesise situations where the PMC and loss are 'output compatible' but not 'compatible'. In such a case, either the PMC itself is not well posed or the underlying concept of a loss measuring output error is irrelevant for the particular learning problem considered. If the geometric structure is inherited from the generative noise model [10] then we would expect that a problem would be both compatible and output compatible.

One of the consequences of compatibility is that it leads naturally to a 'best' choice of loss function. Consider the instantaneous loss function

$$\mathcal{L}_t(p_t) = \min_{p \in S_t(0)} \mathrm{dist}(p, p_t)^2.$$

This loss function is a least squares distance loss based on geodesic distance to the solution set. It is clear that a loss function defined in this way will be compatible with the underlying PMC. Other loss functions that scale the squared distance using any monotonic scaling function are also compatible. The authors conjecture that this characterises the possible compatible loss functions for a given PMC.

Strong compatibility is a small increment on compatibility. In the Euclidean example, Section 3.1, the solution set is a hyper-plane and is clearly a totally geodesic submanifold. In the case of the sphere, Section 3.2, the solution set is the equator of the sphere with respect to the data point x_t (Figure 2). Thus, the PMC and loss function are strongly compatible.

Convexity has been at the core of many of the earlier developments of on-line learning algorithms. The concept of convexity is certainly well defined on a Riemannian manifold. It imposes some strong conditions on the structure of the underlying cost function. For example, the solution set is a subset of a totally geodesic submanifold of the parameter space. To see this, choose any two points in solution set and generate the geodesic between the two points. The cost function is minimal at either end point of the geodesic and convexity implies it is minimal at all points along the geodesic, which must be contained in solution set. The Euclidean problem considered Section 3.1 is convexly compatible as a direct consequence of the convexity of the least squares error loss.

Proposition 3. *The PMC on the sphere of Section 3.2 is not convexly compatible.*

Proof omitted. For more on convex functions on the sphere see [11].

5 Deriving an On-Line Learning Algorithm for non-Euclidean PMC

Let $\mathcal{L}_t : \mathbf{M} \to \mathbb{R}$ be a non-constant smooth function. If $\mathbf{M}$ is compact then $\mathcal{L}_t$ has at least one point of minimum at a critical point $p \in \mathbf{M}$. Following Kivinen and Warmuth [1], define a cost function $U_t : \mathbf{M} \to \mathbb{R}$ by

$$U_t(p) := \mathit{\Delta}(p, p_t) + \eta\, \mathcal{L}_t(p),$$

where $\mathit{\Delta}(p, p_t)$ is a measure of how far the two parameters are apart. U_t consists of two terms: the regularisation and the optimisation. The former term introduces a penalty if p is far from p_t. The latter one is proportional to the instantaneous loss $\mathcal{L}_t$. The learning rate $\eta > 0$ determines the relative importance of the two terms. In the framework of Kivinen and Warmuth [9] $\mathit{\Delta}$ is the Bregman divergence. Here we take $\mathit{\Delta}(p, p_t) = \frac{1}{2}\operatorname{dist}(p, p_t)^2$, where $\operatorname{dist}(p, p_t)$ is the Riemannian distance between p and q. The advantage of making $\mathit{\Delta}$ depend solely on the distance is that such a measure is independent of a parametrisation of $\mathbf{M}$. Let $p \in \mathbf{M}$ be a point of minimum of U_t, which does not have to be unique. Because the function U_t is smooth, p is the critical point, where the differential dU_t is zero; more precisely,

$$\begin{aligned} dU_t(p)(V) &= d\mathit{\Delta}(p, p_t)(V) + \eta\, d\mathcal{L}_t(p)(V) \\ &= \left\langle -\operatorname{Exp}_p^{-1} p_t, V \right\rangle_g + \eta \left\langle \operatorname{grad}\mathcal{L}_t(p), V \right\rangle_g = 0, \end{aligned}$$

for any tangent vector $V \in \mathcal{T}_p\mathbf{M}$. Here $\langle\ ,\ \rangle_g$ denotes the inner product with respect to the Riemannian metric g. The map Exp is called the *exponential map* and can be described as follows. Suppose that there exists a geodesic $\gamma : [0,1] \to \mathbf{M}$ satisfying the initial conditions $\gamma(0) = p$ and $\dot\gamma(0) = V$. Then $\operatorname{Exp}_p(V) := \gamma(1)$ is the end point of the geodesic γ. Consequently the inverse $\operatorname{Exp}_p^{-1} p_t$ denotes the velocity vector of the geodesic from p to p_t calculated at p, *cf.* [12]. Here, and throughout this paper, grad is the *covariant gradient* operator extended to Riemannian manifolds by

$$dh(w)(V) = \langle \operatorname{grad} h(w), V \rangle_g\,,$$

for any $V \in \mathcal{T}_w\mathbf{M}$ and function $h : \mathbf{M} \to \mathbb{R}$. Such a definition of the gradient of a function on $\mathbf{M}$ makes it a vector field on $\mathbf{M}$, *cf.* [13]. Let p_{t+1} be a point of minimum of U_t. Then

$$\operatorname{Exp}_{p_{t+1}}^{-1} p_t = \eta\, \operatorname{grad}\mathcal{L}_t(p_{t+1}), \tag{9}$$

but for our algorithm we take the approximation

$$-\operatorname{Exp}_{p_t}^{-1} p_{t+1} = \eta\, \operatorname{grad}\mathcal{L}_t(p_t). \tag{10}$$

The two above equations define two different update rules called the *implicit update* (since to determine p_{t+1} one needs to solve the equation (9)) and the *explicit update* (10). The explicit update (10) says that point p_{t+1} is the end point of the geodesic starting from p_t with initial velocity vector equal to $-\eta \operatorname{grad}\mathcal{L}_t(p_t)$. Note that (9) does not necessarily define p_{t+1} uniquely. On the other hand the explicit update has the property of decreasing the value of $\mathcal{L}_t$ as the following Proposition 4 shows.

Recall that the manifold $\mathbf{M}$ is *complete* if the exponential map $\mathrm{Exp}_p(V)$ is defined for all $p \in \mathbf{M}$ and all vectors $V \in \mathcal{T}_p\mathbf{M}$. By the Hopf-Rinow theorem [13] a connected manifold is complete if and only if it is complete as a metric space. For example a sphere $\mathbf{S}^{n-1}$ is a complete manifold but any proper open subset of $\mathbb{R}^n$ with Euclidean metric is not.

Proposition 4. *Let $\mathbf{M}$ be a complete manifold and $\mathcal{L}_t\colon \mathbf{M} \to \mathbb{R}$ be a continuously differentiable function with a finite number of isolated critical points. Then there exists $\xi > 0$ such that for any learning rate $0 < \eta < \xi$*

$$\mathcal{L}_t(p_{t+1}) \leq \mathcal{L}_t(p_t),$$

where p_{t+1} is given by explicit update (10), and equality holds if and only if $p_{t+1} = p_t$.

Proof. Let $\gamma_V\colon [0,c] \to \mathbf{M}$ be a geodesic with initial point $p_t = \gamma_V(0)$ and initial velocity vector $V = \dot{\gamma}_V(0) = -\mathrm{grad}\mathcal{L}_t(p_t)$, and let $f = \mathcal{L}_t \circ \gamma_V$. Then by the chain rule

$$\begin{aligned} f'(0) &= \langle \mathrm{grad}\mathcal{L}_t(\gamma_V(0)), \dot{\gamma}_V(0) \rangle \\ &= \langle \mathrm{grad}\mathcal{L}_t(p_t), -\mathrm{grad}\mathcal{L}_t(p_t) \rangle = -\left\| \mathrm{grad}\mathcal{L}_t(p_t) \right\|^2 \leq 0, \end{aligned}$$

where equality holds if and only if p_t is a critical point of $\mathcal{L}_t$, in which case (10) gives $p_{t+1} = p_t$. If p_t is not a critical point of $\mathcal{L}_t$ then, since f' is continuous, f is monotonically decreasing in some interval $[0, \xi)$. Since the number of isolated critical points of f is finite then clearly $\xi > 0$. Henceforth, for any $0 < \eta < \xi$ there is $f(\eta) < f(0)$. By the rescaling lemma (*cf.* [13]) and by (10) it follows that $\gamma_V(\eta) = \gamma_{\eta V}(1) = p_{t+1}$. Hence $\mathcal{L}_t(p_{t+1}) < \mathcal{L}_t(p_t)$, for any $0 < \eta < \xi$, what was to show. $\square$

Proposition 4 shows that under weak assumption for the loss function, the explicit update always leads towards decreasing the loss, for a sufficiently small learning rate. Although it is impossible to tell, in general, how efficient this algorithm is, that is how fast the sequence of derived parameters approaches the true parameter and how large the cumulative loss is, we may say that for any finite sequence of data points such learning rate always exists.

We conclude this section with showing that both update rules: the Widrow-Hoff algorithm (4) of the linear predictor model in $\mathbb{R}^n$, and the geometrical update of the scale invariant linear predictor, described in Sections 3.1 and 3.2, are examples of the explicit update (10).

5.1 Examples of the Explicit Update

Example 5. Let $\mathbf{M} = \mathbb{R}^n$. The covariant gradient grad is equal to the standard gradient $\boldsymbol{\partial}$. Considering the standard Euclidean connection, we find that geodesics in $\mathbb{R}^n$ are line segments and $\mathrm{Exp}_{p_t}^{-1} p_{t+1} = p_{t+1} - p_t$. The explicit update (10) then becomes

$$-\left(p_{t+1} - p_t\right) = \eta\, \boldsymbol{\partial}_p \mathcal{L}_t(p_t),$$

which is precisely the Widrow-Hoff learning algorithm.

Example 6. Let us now assume that $\mathbf{M} = \mathbf{S}^{n-1}$ is the $n-1$-unit sphere embedded in $\mathbb{R}^n$ with its standard metric induced from the Euclidean metric on $\mathbb{R}^n$. The covariant gradient $\mathrm{grad}\mathcal{L}_t(p)$ on $\mathbf{S}^{n-1}$ is equal to the projection of the standard gradient onto the tangent space at p. Thus, the gradient of the scale invariant linear predictor (8) is given by

$$\mathrm{grad}\mathcal{L}_t(p) = 2\left\langle p, x_t\right\rangle\left(x_t - \left\langle p, x_t\right\rangle p\right). \tag{11}$$

The exponential mapping on the unit sphere is given by

$$\mathrm{Exp}_p(V) = p\,\cos\|V\| + \frac{V}{\|V\|}\sin\|V\|, \tag{12}$$

for any $p \in \mathbf{S}^{n-1}$ and $V \in \mathcal{T}_p\mathbf{S}^{n-1}$, where the norm is taken with respect to the Euclidean metric on $\mathbb{R}^n$. Given the exponential mapping (12) the the explicit update (10) takes the form

$$\begin{aligned} p_{t+1} &= \mathrm{Exp}_{p_t}\left(-\eta\,\mathrm{grad}\mathcal{L}_t(p_t)\right) \\ &= p_t\,\cos\left(\eta\,\|V_t\|\right) - \frac{V_t}{\|V_t\|}\sin\left(\eta\,\|V_t\|\right), \end{aligned}$$

where $V_t = \mathrm{grad}\mathcal{L}_t(p_t)$ is given by (11). To complete the calculation we give the formula for $\|V_t\|$

$$\|V_t\| = \|\mathrm{grad}\mathcal{L}_t(p_t)\| = 2\left|\left\langle p_t, x_t\right\rangle\right|\|x_t - \left\langle p_t, x_t\right\rangle p_t\| = 2\left|\left\langle p_t, x_t\right\rangle\right|\sqrt{1 - \left\langle p_t, x_t\right\rangle^2},$$

or in short $\|V_t\| = |\sin 2d|$, where $d = \arccos\left\langle p_t, x_t\right\rangle$ is the spherical distance between p_t and x_t.

6 Analysis

This section investigates performance of the learning algorithm on the unit sphere. To begin with a geometrical relationship between input and output points of the algorithm is established. Afterwards it is shown that for any learning rate $0 < \eta < 1$ the algorithm always progresses, in terms of spherical distances, towards a (closer) true parameter $p_\star$, except for the pathological case when $\mathrm{dist}(p_t, p\star) = \pi/2$. Finally cumulative mistake bound is investigated.

6.1 Description of the Algorithm on the Sphere

In order to analyse the learning algorithm (10) we first describe its single step in terms of spherical geometry. Observe that the three points x_t, p_t and p_{t+1} aways lie along a geodesic, a great arc on $\mathbf{S}^{n-1}$ and the order of the points depends on the sign of $\left\langle p_t, x_t\right\rangle$. When the inner product $\left\langle p_t, x_t\right\rangle$ is positive then $d < \pi/2$ and the order of x_t, p_t, p_{t+1} is as illustrated on Figure 2, while for $\left\langle p_t, x_t\right\rangle$ negative $d > \pi/2$ and the order of the points changes to x_t, p_{t+1}, p_t. Consider the two spherical triangles $\triangle x_t, p_{t+1}, p_\star$ and $\triangle x_t, p_t, p_\star$, as in Figure 2. By the law of cosines on the unit sphere, the following identities hold

$$\begin{cases} \cos a = \cos\left(b + d\right)\,\cos e + \sin\left(b + d\right)\,\sin e\,\cos\beta \quad \text{and} \\ \cos c = \cos d\,\cos e + \sin d\,\sin e\,\cos\beta, \end{cases} \tag{13}$$

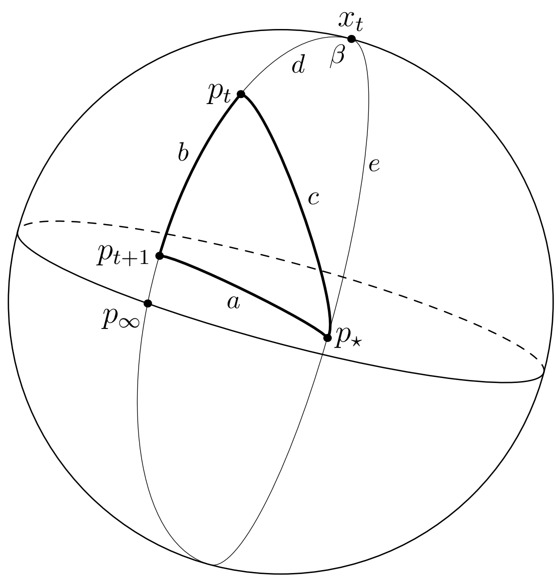

Fig. 2. Update step on the unit sphere, where $n = 3$.

where

$$c = \mathrm{dist}(p_t, p_\star) \quad \text{and} \quad a = \mathrm{dist}(p_{t+1}, p_\star).$$

It is easy to check that the system of equations (13) still holds for $d > \pi/2$ if b is replaced by $-b$. That is the situation with the explicit update since now the gradient $\mathrm{grad}\mathcal{L}_t$ points in the opposite direction. Thus we have shown that the geometrical relationship between the points x_t, p_t, p_{t+1} and $p_\star$ of the learning algorithm (10) on the sphere is captured by (13), where $b = \eta \sin 2d$.

6.2 Measure of Progress

In Section 6.3 we analyse the upper bound of the cumulative loss of the explicit algorithm on $\mathbf{S}^{n-1}$, which gives a measure of how the algorithm performs. For the present we consider step-wise progress towards the true parameter.

It is useful to analyse each step of a learning algorithm in terms of progress towards a *comparator parameter* p, *cf.* [8]. Using the notation of the previous section we denote $a = \mathrm{dist}(p_{t+1}, p)$ and $c = \mathrm{dist}(p_t, p)$, for any parameter $p \in \mathbf{S}^{n-1}$. It is convenient here to measure the progress in terms of the difference $\sin^2 c - \sin^2 a$. With a little algebra it can be found from (13) that

$$\begin{aligned} \sin^2 c - \sin^2 a = {} & (\mathcal{L}_t(p_t) - \mathcal{L}_t(p_{t+1}))\, \cos^2\beta && (14)\\ & - \mathcal{L}_t(p)\, (\mathcal{L}_t(p_t) - \mathcal{L}_t(p_{t+1}))\, \big(1 + \cos^2\beta\big) && (15)\\ & + \cos(b + 2d)\, \cos\beta\, \sin b\, \sin 2e. && (16) \end{aligned}$$

Proposition 4 asserts that the difference $\mathcal{L}_t(p_t) - \mathcal{L}_t(p_{t+1})$ remains positive for sufficiently small η. Therefore term (14) is positive and depends on the loss at p_t and β. The remaining terms (15) and (16) depend on both $\mathcal{L}_t(p)$ and $\mathcal{L}_t(p_t)$, where (15) is negative and (16) is usually relatively small. Further analysis is restricted to the case when $\mathcal{L}_t(p) = 0$. We have the following.

Lemma 7. *Let $p_\star$ be a parameter such that $f_{p_\star}(x_t) = 0$. Then the algorithm (10) makes progress towards $p_\star$, for any $\beta \neq \pi/2$ and any $0 < \eta < 1$. If $\beta = \pi/2$ then the algorithm makes no progress, that is* $\mathrm{dist}(p_{t+1}, p_\star) = \mathrm{dist}(p_t, p_\star)$.

Proof. By the hypothesis $e = \pi/2$ and hence the measure of progress becomes

$$\sin^2 c - \sin^2 a = \cos^2 a - \cos^2 c = \cos^2 \beta \, \sin b \, \sin(b + 2d).$$

Clearly if $\beta = \pi/2$ then $\sin^2 c = \sin^2 a$ and by (13) it follows that $\cos c = \cos a$ hence $c = a$, what was to show. Assume now that $\beta \neq \pi/2$. We will show that $\sin b \, \sin(b + 2d) \geq 0$, where the equality holds if and only if $d = \pi/2$. We assume the opposite that $\sin b \, \sin(b + 2d) < 0$ and derive a contradiction. Consider the three cases separately.

If $d = \pi/2$, then $b = \eta \sin 2d = \eta \sin \pi = 0$ and the case is proved.

If $d < \pi/2$, then since $b = \eta \sin 2d$ it follows from the hypothesis that $0 < b < 1$ and therefore $\sin b > 0$. Hence it must be that $\sin(b + 2d) \leq 0$ then $b + 2d \geq \pi$ and $b = \eta \sin 2d \geq \pi - 2d$, hence $\sin 2d \geq \pi - 2d$, but $\sin 2d = \sin(\pi - 2d) < \pi - 2d$. A contradiction.

The case $d > \pi/2$ is proven in a similar way. $\square$

One consequence of Lemma 7 is that if the learning rate satisfies $0 < \eta < 1$ and the spherical distance $\mathrm{dist}(p_t, p_\star) < \pi/2$ then also $\mathrm{dist}(p_{t+1}, p_\star) < \pi/2$, for otherwise, since p_{t+1} continuously depends on η, there would be $0 < \eta_0 < 1$ such that $\mathrm{dist}(p_{t+1}, p_\star) = \pi/2$ which is impossible in view of Lemma 7. Thus $\mathrm{dist}(p_{t+1}, p_\star) \leq \mathrm{dist}(p_t, p_\star)$.

6.3 Upper Bound for the Cumulative Loss

We conclude the analysis of the explicit algorithm on the sphere with derivation of cumulative mistake upper bounds. Such bounds provide a measure of performance of an on-line algorithm. For a given sequence of k instances $\{x_t\}_{t=0}^{k-1}$ one compares the loss accumulated by the algorithm with the accumulated loss for a fixed comparator parameter.

Here we choose comparator to be the true parameter $p_\star$ so that the sequence of k instances satisfy $\langle p_\star, x_t \rangle = 0$, for all t. In other words, all x_t lie on a great circle orthogonal to $p_\star$ and the loss is always $\mathcal{L}_t(p_\star) = 0$. The following fact is needed.

Proposition 8. *If* $0 \leq a \leq c \leq \pi/3$ *then*

$$\frac{\cos a - \cos c}{\cos c} \leq c^2 - a^2.$$

Let $\mathbf{loss}_k(\mathfrak{A})$ denote the cumulative loss of the algorithm after k iterations, defined by

$$\mathbf{loss}_k(\mathfrak{A}) := \sum_{t=0}^{k-1} \mathcal{L}_t(p_t).$$

Lemma 9. *Let* $\operatorname{dist}(p_0, p_\star) \leq \pi/3$ *then for any* $0 < \eta < 1$ *the following upper bound for the cumulative loss holds*

$$\mathbf{loss}_k(\mathfrak{A}) \leq \frac{\operatorname{dist}(p_0,p_\star)^2 - \operatorname{dist}(p_k,p_\star)^2}{\eta\left(1-\eta+\sqrt{(1-\eta)^2+\frac{2}{3k}\left(3-(2-\eta)\,\eta\right)\left(\operatorname{dist}(p_0,p_\star)^2-\operatorname{dist}(p_k,p_\star)^2\right)}\right)}$$
$$< \frac{\operatorname{dist}(p_0,p_\star)^2 - \operatorname{dist}(p_k,p_\star)^2}{2\eta\,(1-\eta)}.$$

Proof. We shall use the notation of the previous section, where $a = \operatorname{dist}(p_{t+1}, p_\star)$, $c = \operatorname{dist}(p_t, p_\star)$ and $e = \operatorname{dist}(x_t, p_\star)$. By the hypothesis $e = \pi/2$ and hence the system of equations (13) yields

$$\frac{\cos a - \cos c}{\cos c} = \frac{\sin(b+d) - \sin d}{\sin d}.$$

One checks that with the hypothesis of the lemma the following estimation holds

$$\frac{\sin(b+d)-\sin d}{\sin d} > 2\eta\,(1-\eta)\ \mathcal{L}_t(p_t) + \frac{2}{3}\,\eta^2\ \left(3-(2-\eta)\,\eta\right)\ \frac{1}{k}\left(\mathcal{L}_t(p_t)\right)^2.$$

Thus by Proposition 8 and the remark after Lemma 7 the following inequality holds

$$\operatorname{dist}(p_t,p_\star)^2 - \operatorname{dist}(p_{t+1},p_\star)^2 > 2\eta\,(1-\eta)\ \mathcal{L}_t(p_t) + \frac{2}{3}\,\eta^2\left(3-(2-\eta)\,\eta\right)\mathcal{L}_t(p_t)^2.$$

Summing up for $t = 0, \ldots, k-1$ yields

$$\operatorname{dist}(p_0,p_\star)^2 - \operatorname{dist}(p_k,p_\star)^2 > 2\eta\,(1-\eta)\ \sum_{t=0}^{k-1}\mathcal{L}_t(p_t) + \frac{2}{3}\,\eta^2\left(3-(2-\eta)\,\eta\right)\ \sum_{t=0}^{k-1}\mathcal{L}_t(p_t)^2.$$

By Hölder's inequality

$$\sum_{t=0}^{k-1}\mathcal{L}_t(p_t)^2 \geq \frac{1}{k}\left(\sum_{t=0}^{k-1}\mathcal{L}_t(p_t)\right)^2,$$

and hence

$$\operatorname{dist}(p_0,p_\star)^2 - \operatorname{dist}(p_k,p_\star)^2 > 2\eta\,(1-\eta)\ \mathbf{loss}_k(\mathfrak{A}) + \frac{2}{3}\,\eta^2\ \left(3-(2-\eta)\,\eta\right)\ \frac{1}{k}\left(\mathbf{loss}_k(\mathfrak{A})\right)^2.$$

Solving the above quadratic inequality completes the proof. □

One of the goals in machine learning is to design an on-line algorithm with the best possible cumulative loss. Lemma 9 assures that given a sequence of examples such that $\langle p_\star, x_t\rangle = 0$, for all t, we may choose an optimal η so that cumulative loss $\mathbf{loss}_k(\mathfrak{A}) < 2\left(\operatorname{dist}(p_0,p_\star)^2 - \operatorname{dist}(p_k,p_\star)^2\right) < 2\operatorname{dist}(p_0,p_\star)^2 \leq 2\pi^2/9$.

7 Conclusion

The paper reconsidered the basic ideas of stochastic gradient descent in a setting more general than usual. We showed how to respect the underlying geometry induced by restrictions to the parameter space for linear predictors. We also showed that the classical and seemingly obvious notion of convexity for loss functions does not map across to the more general situation so simply. We developed notions of compatibility between a loss function and a metric. We illustrated our ideas on the problem of learning linear functions with the parameters restricted to a sphere.

We also analysed the algorithm in terms of a cumulative mistake bound. Unsurprisingly (given the non-flat geometry) we did not obtain results of the same structural form as those obtained for flat metrics. Specifically it matters significantly where the algorithm starts (how far it is from $p_\star$).

It is reasonable to expect that the analysis of the algorithm presented can be considerably refined. More interesting is the prospect of applying the general framework here to a wide range of problems.

Acknowledgments: Krzysztof Krakowski and Robert Mahony were supported by the Australian Research Council. National ICT Australia is funded by the Australian Government's Backing Australia's Ability initiative, in part through the Australian Research Council.

References

1. Kivinen, J., Warmuth, M.K.: Exponentiated gradient versus gradient descent for linear predictors. Information and Computation **132** (1997) 1–63
2. Jagota, A., Warmuth, M.K.: Continuous and discrete time nonlinear gradient descent: relative loss bounds and convergence. In: International Symposium on Artificial Intelligence and Mathematics. (1998)
3. Cesa-Bianchi, N.: Analysis of two gradient-based algorithms for on-line regression. Journal of Computer and System Sciences **59** (1999) 392–411
4. Mahony, R.E., Williamson, R.C.: Prior knowledge and preferential structures in gradient descent learning algorithms. Journal of Machine Learning Research **1** (2001) 311–355
5. Amari, S.: Natural gradient works efficiently in learning. Neural Computation **10** (1998) 251–276
6. Watson, G.S.: Statistics on Spheres. Volume 6 of The University of Arkansas Lecture Notes in the Mathematical Sciences. John Wiley and Sons, New York (1983)
7. Mardia, K.V.: Statistics of Directional Data. Academic Press, New York (1972)
8. Tsuda, K., Rätsch, G., Warmuth, M.K.: Matrix exponentiated gradient updates for on-line learning and Bregman projection. Journal of Machine Learning Research **5** (2004) 1–48
9. Kivinen, J., Warmuth, M.K.: Relative loss bounds for multidimensional regression problems. Machine Learning **45** (2001) 301–329
10. Amari, S.: Differential-geometrical methods in statistics. Number 28 in Lecture notes in statistics. Springer-Verlag, New York (1985)
11. Kendall, W.S.: Convexity and the hemisphere. Journal of the London Mathematical Society **43** (1991) 567–576
12. Karcher, H.: Riemannian center of mass and mollifier smoothing. Communications on Pure and Applied Mathematics **30** (1977) 509–541
13. Lee, J.M.: Riemannian Manifolds: An Introduction to Curvature. Number 176 in Graduate Texts in Mathematics. Springer-Verlag, New York (1997)