Applications of Information Geometry to Machine Learning

Jason d'Eon

August 2019

A research project submitted to the
University of Waterloo in partial fulfillment
of the requirements for the degree of
Master of Mathematics in Pure Mathematics

Abstract

Information geometry is the application of differential geometry to the study of statistical objects. In particular, it gives ways of predicting the information change in a parametric probability distribution when the parameters have been modified by a small amount. This is relevant to neural networks, where these techniques can be used to predict how the output of the model will change as the parameters are updated. The predictions can be used to analyze and improve the optimization process in supervised learning. In this paper, we present an overview of the theory of information geometry and highlight some key applications to machine learning.

Contents

1 Introduction

Information geometry sits at the intersection of differential geometry, statistics, and probability theory. Exponential families of probability distributions, such as normal distributions, come with standard parameterizations. Therefore, it is natural to interpret a space of probability distributions as a Riemannian manifold, where the coordinates are given by the parameters of the distribution family and the metric is induced by a measurement of information discrepancy. This gives a way of imposing a distance between probability distributions. The first use of differential geometry in statistics is generally attributed to Harold Hotelling, in the late twenties [43]. C. R. Rao is well-known for his research of Riemannian metrics on spaces of probability distributions. In particular, he studied the Fisher information matrix and interpreted it as a metric [34]. Claude Shannon introduced what is now known as Shannon Entropy, which is a way of describing the amount of information stored in a distribution [39]. Solomon Kullback and Richard Leibler independently introduced a different way of measuring the difference between distributions, which they called the KL-divergence [24]. In this paper, we describe these concepts at an introductory level and show how they relate to each other.

We now summarize the contents of this paper. Chapter 2 is an overview of Riemannian geometry. Basic knowledge of manifolds and tangent spaces is assumed. The results from this chapter are taken from Lee's *Riemannian Manifolds: an Introduction to Curvature* [27] and Tu's *Differential Geometry: Connections, Curvature, and Characteristic Classes* [45]. The exception is the section on dual affine connections, since this topic is not covered in these references. For this chapter we referred to [4] and [29].

Chapter 3 discusses how parameter spaces of probability distributions can be interpreted as Riemannian manifolds. In this context, they are called statistical manifolds. This theory was the primary topic of Shun-ichi Amari's research [1, 4]. He also demonstrated applications of information geometry to machine learning [5, 4] and blind signal separation [5, 6]. He described two important classes of divergences on parameter spaces, which are the f-divergence and Bregman divergence classes. The f-divergences satisfy information monotonicity and they induce a Riemannian divergence given by the Fisher information matrix. It follows that this is the only Riemannian metric on these parameter spaces which is invariant under transformations of the random variable. Bregman divergences are in a one-to-one correspondence with dually flat structures on the parameter space. That is, they induce a pair of flat, dual affine connections on the parameter space. Amari also showed that a canonical divergence can be derived from a dually flat structure, which gives a Bregman divergence.

Chapter 4 shows examples of how the previous topics can be used to improve the optimization of neural networks. The optimization algorithm typically used in machine learning is stochastic gradient descent. Amari introduced the so-called natural gradient, mentioned in [5], which is meant to better capture the direction of steepest descent in a parameter space of probability distributions, compared to the standard gradient. In [7], it was shown that even when gradient descent is used, one can compute the expected change in output with respect to a change in the parameters by leveraging the Fisher information matrix. We also show that momentum-based gradient descent algorithms can be extended to a Riemannian setting. Finally, we mention the drawbacks of these techniques, and why they are not widespread.

2 Riemannian Geometry

2.1 Riemannian Manifolds

Manifolds are a way of generalizing Euclidean space. The most basic definition of a manifold is not very restrictive, so it does not necessarily inherit many properties of Euclidean spaces. For example, in $\mathbb{R}^n$, we have a notion of length of a vector, as well as angles between two vectors, both of which can be encapsulated by an inner product. Riemannian manifolds are generalizations of $\mathbb{R}^n$ that inherit this concept by equipping each tangent space with an inner product. This feature is captured by the Riemannian metric tensor.

Definition 2.1. A **Riemannian metric tensor** on a smooth manifold M is a symmetric 2-tensor field on M that is positive-definite for all $p \in M$.

In its coordinate-free form, we typically denote a Riemannian metric by g, and the tensor at a given point by g_p. Since $g_p(\cdot, \cdot)$ satisfies the properties of an inner product on T_pM, we also denote it by $\langle \cdot, \cdot \rangle_g$. If M is an n-dimensional smooth manifold and $x = (x^1, ..., x^n)$ are local coordinates on M then g can be written in terms of the basis $\{dx^1|_p, ..., dx^n|_p\}$ of T_p^*M:

$$g_p = \sum_i \sum_j g_{ij} dx^i|_p \otimes dx^j|_p$$

where $G = (g_{ij})$ is a symmetric, positive definite matrix.

Definition 2.2. A **Riemannian manifold**, denoted by (M, g), is a smooth manifold M paired with a Riemannian metric g.

The inner product on T_pM induced by g can be written in terms of the matrix G. If

$\{\frac{\partial}{\partial x^1}|_p, ..., \frac{\partial}{\partial x^n}|_p\}$ is a local basis of T_pM, then tangent vectors $v, w \in T_pM$ can be written as linear combinations of the $\frac{\partial}{\partial x^i}$, and the inner product on T_pM is:

$$\langle v, w \rangle_g = v^T G w,$$

the length of v is given by:

$$\|v\|_g = \sqrt{\langle v, v \rangle_g},$$

and if v and w are non-zero, the angle between v and w can be computed by the formula:

$$\cos\theta = \frac{\langle v, w \rangle_g}{\|v\|_g \cdot \|w\|_g}.$$

Example 2.3. $\mathbb{R}^n$ can be equipped with the Euclidean metric. In terms of the standard coordinate functions on $\mathbb{R}^n$, $x^i : \mathbb{R}^n \to \mathbb{R}$, where $x^i(\xi_1, ..., \xi_n) = \xi_i$, this is written as:

$$g_{ij} = \delta_{ij}.$$

In other words, the matrix $G = (g_{ij})$ is the identity matrix, for every point $p \in \mathbb{R}^n$.

Example 2.4. Let f_{ij} be any smooth bounded functions on $\mathbb{R}^n$, for $i, j \in \{1, ..., n\}$, such that $f_{ij} = f_{ji}$. Then if $C > 0$ is sufficiently large, the metric on $\mathbb{R}^n$ given by:

$$g_{ij} = C\delta_{ij} + f_{ij}.$$

is a Riemannian metric. By construction, $g_{ij} = g_{ji}$. If we let $G = (g_{ij})$, $p \in \mathbb{R}^n$, and $(\xi_1, ..., \xi_n)^T \in T_p\mathbb{R}^n$ then by expanding:

$$(\xi_1, ..., \xi_n) G (\xi_1, ..., \xi_n)^T,$$

we get:

$$C\left(\sum_i \xi_i^2\right) + \sum_{i,j} f_{ij}(p)\xi_i\xi_j.$$

By boundedness, there exists $C > 0$ such that if $(\xi_1, ..., \xi_n)^T$ is non-zero:

$$C > \frac{-\sum_{i,j} f_{ij}(p)\xi_i\xi_j}{\sum_i \xi_i^2},$$

for all $p \in \mathbb{R}^n$.

2.2 Affine Connections

The directional derivative in $\mathbb{R}^n$ allows us to define how components of a vector field change in a particular direction. We denote this operator by D, which takes in two vector fields $X, Y \in \mathfrak{X}(\mathbb{R}^n)$ and returns a single vector field $D_X Y = D(X, Y)$, which captures the rate of change of each component of Y in the direction $X(p)$ for all $p \in \mathbb{R}^n$. This is defined by the limit:

$$(D_X Y)(p) = \lim_{h \to 0} \frac{Y(p + hX(p)) - Y(p)}{h}.$$

There are two notable properties of the operator D:

1. $C^\infty(\mathbb{R}^n)$-linearity in the first component: $D_{fX} Y = f D_X Y$;
2. Leibniz Rule: $D_X(fY) = (Xf)Y + f D_X Y$.

There is no obvious way to directly extend the definition of the directional derivative of vector fields to manifolds. The addition of a vector to a point, $p + hX(p)$, has no defined meaning, and the subtraction of two tangent vectors in different tangent spaces is also not defined. To solve this issue, we instead define the operator abstractly, so that it already comes with the properties we want. It is not obvious that every Riemannian manifold even admits an affine connection, but this turns out to be the case, as we will see later in this section. On the other hand, a Riemannian manifold generally admits many affine connections.

Definition 2.5. An **affine connection** on a manifold M is an $\mathbb{R}$-bilinear operator:

$$\nabla : \mathfrak{X}(M) \times \mathfrak{X}(M) \to \mathfrak{X}(M)$$

that satisfies the following two properties. Here $\nabla_X Y := \nabla(X, Y)$:

1. $\nabla_{fX} Y = f \nabla_X Y$;
2. $\nabla_X(fY) = (Xf)Y + f \nabla_X Y$.

It is worth pointing out that the choice of Riemannian metric on a manifold has no impact on the existence of affine connections, however, we will see later (Section 2.6) that certain connections can exist that interact with the metric in a desirable way.

If we fix the first component of ∇ to be $X \in \mathfrak{X}(M)$, and think of ∇ as having a single input, the affine connection is referred to as the **covariant derivative**:

$$\nabla_X : \mathfrak{X}(M) \to \mathfrak{X}(M).$$

Example 2.6. The directional derivative on $\mathbb{R}^n$ is an example of an affine connection.

Let $M \subset \mathbb{R}^n$ be a embedded submanifold, and let $Y \in \mathfrak{X}(M)$. Because of the embedding, for each $p \in M$, we can think of Y_p as lying in T_pM or in $T_p\mathbb{R}^n$. Therefore, we can write Y in terms of the standard coordinates (x^i) in $\mathbb{R}^n$:

$$Y = \sum_i Y^j(p)\partial_i|_p.$$

Where ∂_i denotes $\frac{\partial}{\partial x^i}$. Then if $X_p \in T_pM$, we can take the directional derivative along direction X_p by computing:

$$D_{X_p}Y = \sum_i X_p(Y^i(p))\partial_i|_p.$$

The resulting vector may not be tangent to M. So we must project the tangent vector $D_{X_p}Y$ from $T_p\mathbb{R}^n$ to T_pM. In this sense, M inherits an affine connection ∇^T from the directional derivative D on $\mathbb{R}^n$ by setting:

$$(\nabla^T_X Y)_p := \pi((D_X Y)_p),$$

where π is the orthogonal projection from $T_p\mathbb{R}^n \to T_pM$. Since the Nash embedding theorem guarantees that every manifold can be isometrically embedded in $\mathbb{R}^n$ for some n, this means that every Riemannian manifold admits at least one connection.

Theorem 2.7. *∇^T defines an affine connection.*

Proof. Since π is linear, then ∇^T will be $\mathbb{R}$-linear in each component, and $C^\infty(M)$-linear in the first component. Now let $f \in C^\infty(M)$, and $X, Y \in \mathfrak{X}(M)$. Note that:

$$\begin{aligned}
\nabla^T_X(fY) &= \pi(D_X(fY)) \\
&= \pi((Xf)Y + fD_XY) \\
&= (Xf)\pi(Y) + f\pi(D_XY) \\
&= (Xf)Y + f\nabla^T_X Y.
\end{aligned}$$

□

Example 2.8. Consider $S^1 \subset \mathbb{R}^2$. Define the following vector field on $\mathbb{R}^2$:

$$X = y\partial_x - x\partial_y$$

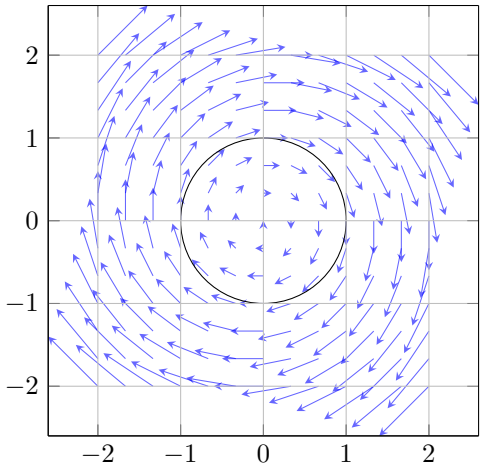

Figure 1: A visualization of the flow of the vector field $X = y\partial_x - x\partial_y$ and S^1.

We can also think of this as a vector field on S^1, since for any point p on the unit circle, the vector $X(p)$ is tangent to the unit circle, and therefore can be thought of as a tangent vector of S^1. Let ∇^T be the affine connection on S^1, inherited from the directional derivative on $\mathbb{R}^2$, given by the orthogonal projection. We will compute $\nabla^T_X X$. The integral curve of X that coincides with the unit circle is:

$$\gamma(t) = (\sin t, \cos t).$$

In other words:

$$X(\gamma(t)) = \gamma'(t) = (\cos t, -\sin t).$$

Then the directional derivative at $p = (p_1, p_2)^T$ gives:

$$\begin{aligned}
(D_X X)(p) &= \lim_{h\to 0} \frac{1}{h}\left[X\left(\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} + hX\begin{pmatrix} p_1 \\ p_2 \end{pmatrix}\right) - X\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \right] \\
&= \lim_{h\to 0} \frac{1}{h}\left[X\begin{pmatrix} p_1 + hp_2 \\ p_2 - hp_1 \end{pmatrix} - \begin{pmatrix} p_2 \\ -p_1 \end{pmatrix} \right] \\
&= \lim_{h\to 0} \frac{1}{h}\left[\begin{pmatrix} p_2 - hp_1 \\ -p_1 - hp_2 \end{pmatrix} - \begin{pmatrix} p_2 \\ -p_1 \end{pmatrix} \right] \\
&= \lim_{h\to 0} \frac{1}{h}\left[\begin{pmatrix} -hp_1 \\ -hp_2 \end{pmatrix} \right] \\
&= \begin{pmatrix} -p_1 \\ -p_2 \end{pmatrix},
\end{aligned}$$

meaning that for $p = \gamma(t)$:

$$(D_X X)(\gamma(t)) = \gamma''(t) = (-\sin t, -\cos t).$$

However, note that $\gamma'(t) \cdot \gamma''(t) = 0$. So $\nabla^T_X X = 0$.

2.3 Christoffel Symbols

Let $(U, \varphi = (x^1, ..., x^n))$ be a local chart on M, and for $p \in M$, let $\{\partial_1 = \frac{\partial}{\partial x^1}, ..., \partial_n = \frac{\partial}{\partial x^n}\}$ be the basis for T_pM. To define an affine connection ∇ on M, at least locally on U, it is sufficient to define each of the n components of $(\nabla_{\partial_i}\partial_j)$ as smooth functions on U. That is, for each i, j, k, we have a smooth function $\Gamma^k_{ij} := (\nabla_{\partial_i}\partial_j)^k$. The properties of the connection will force how $(\nabla_X Y)_p$ will be evaluated for any arbitrary $X, Y \in \mathfrak{X}(M)$ and $p \in U$. Consider Y in terms of the given basis of T_pM:

$$Y(p) = \sum_j Y^j(p)\partial_j|_p.$$

Then we can compute $\nabla_{\partial_i} Y$ in terms of $\nabla_{\partial_i}\partial_j$ by the following:

$$\begin{aligned}
\nabla_{\partial_i} Y &= \nabla_{\partial_i}\left(\sum_j Y^j \partial_j\right) \\
&= \sum_j \nabla_{\partial_i}(Y^j\partial_j) \\
&= \sum_j \left[\partial_i(Y^j) \cdot \partial_j + Y^j \cdot \nabla_{\partial_i}\partial_j\right].
\end{aligned}$$

And we can also compute $\nabla_X Y$ in terms of $\nabla_{\partial_i} Y$ by:

$$\nabla_X Y = \sum_j X^j \nabla_{\partial_j} Y.$$

Definition 2.9. The smooth functions $\Gamma^k_{ij} := (\nabla_{\partial_i}\partial_j)^k$ are referred to as the **Christoffel symbols** of the affine connection ∇.

Example 2.10. On $\mathbb{R}^n$, the Christoffel symbols of the directional derivative are $\Gamma^k_{ij}(p) = 0$ for all $p \in \mathbb{R}^n$, since $D_{\partial_i}\partial_j = 0$ for all $i, j \in \{1, ..., n\}$.

Example 2.11. Let $M = \mathbb{R}^2$ and $a, b \in \mathbb{R}$. Define ∇ by setting for all p:

$$(\nabla_{\partial_x}\partial_x)_p = \begin{pmatrix} a \\ b \end{pmatrix},$$

and letting $\Gamma_{ij}^k(p) = 0$ otherwise. Suppose $\gamma : \mathbb{R} \to \mathbb{R}^2$ is defined by:

$$\gamma(t) = \begin{pmatrix} t \\ 0 \end{pmatrix},$$

which is simply the constant-speed curve along the x-axis in the positive direction. If we were working with the directional derivative, then clearly:

$$D_{\gamma'(t)}\gamma'(t) = D_{\partial_x|_{\gamma(t)}}\partial_x|_{\gamma(t)} = 0.$$

However, with our affine connection ∇:

$$\nabla_{\gamma'(t)}\gamma'(t) = \nabla_{\partial_x|_{\gamma(t)}}\partial_x|_{\gamma(t)} = \begin{pmatrix} a \\ b \end{pmatrix}.$$

So the acceleration of the curve can be made to change with respect to t, and these values can be modified with a lot of freedom. This also means the acceleration of a curve depends on an affine connection, whereas velocity of a curve does not.

2.4 Geodesics

Affine connections allow us to generalize the notion of a straight line. In $\mathbb{R}^n$, this can be represented linearly in terms of a single parameter t:

$$c(t) = vt + p,$$

for $p, v \in \mathbb{R}^n$.

Now consider a curve $\gamma : [a, b] \to M$. On a general manifold, the velocity of a curve $\gamma'(t)$ is a vector that lies in the tangent space $T_{\gamma(t)}M$. Similarly to how affine connections generalize the directional derivative, we can generalize differentiating a vector field along a curve.

Theorem 2.12. *Let $\gamma : [a, b] \to M$ be a curve on a manifold M with affine connection ∇. We will use $\mathfrak{X}(\gamma)$ to denote vector fields in $\mathfrak{X}(M)$ that are restricted to points on $\gamma(t)$. There exists a unique map:*

$$\frac{D}{dt} : \mathfrak{X}(\gamma) \to \mathfrak{X}(\gamma),$$

such that for all $V \in \mathfrak{X}(\gamma)$ and all C^∞ functions f on $[a, b]$:

1. $\frac{D(\alpha V + \beta W)}{dt} = \alpha\frac{DV}{dt} + \beta\frac{DW}{dt}$.

2. $\frac{D(fV)}{dt} = \frac{df}{dt}V + f\frac{DV}{dt}$.

3. *If* $\widetilde{V} \in \mathfrak{X}(M)$ *is such that, for all* t, $V(t) = \widetilde{V}(\gamma(t))$, *then* $\frac{DV}{dt}(t) = \nabla_{\gamma'(t)}\widetilde{V}$ *for all* t.

Definition 2.13. The map $\frac{D}{dt}$ given from the above theorem is called the **covariant derivative along a curve**.

We will postulate that γ is "straight" if the velocity of $\gamma(t)$ does not change as we move along the curve. This brings us to the definition of a geodesic:

Definition 2.14. Let $\gamma : [a, b] \to M$ be a curve on a manifold M with affine connection ∇. Then γ is a **geodesic** if $\frac{D\gamma'(t)}{dt} = 0$ for all t.

Example 2.15. When $M = \mathbb{R}^n$, g is the Euclidean metric, and ∇ is the directional derivative, then $\nabla_{\gamma'(t)}\gamma'(t) = \gamma''(t)$. Therefore, $\gamma(t)$ being a geodesic is equivalent to saying the acceleration of $\gamma(t)$ is 0. So $\gamma(t)$ is a geodesic if and only if $\gamma(t) = vt + p$ for $v, p \in \mathbb{R}^n$.

Example 2.16. Let $M = S^1$, embedded as the unit circle in $\mathbb{R}^2$, and ∇^T be the affine connection on S^1, inherited from the directional derivative on $\mathbb{R}^2$. As an embedded submanifold, S^1 can also inherit the Euclidean metric g from $\mathbb{R}^n$, which is simply the restriction to vectors that are tangent to S^1. As we will see in a later section (Section 2.6) affine connections can satisfy a certain kind of compatibility with the metric. In $\mathbb{R}^n$ we have that:

$$\frac{d}{dt}\langle\gamma'(t), \gamma'(t)\rangle_g = 2\langle\gamma''(t), \gamma'(t)\rangle_g.$$

(See also Theorem 2.26). Since ∇^T is inherited from the directional derivative, $\frac{D\gamma'(t)}{dt}$ is the tangential component of $\gamma''(t)$. It follows easily from the above equation that if $\gamma(t)$ is a geodesic, then $\|\gamma'(t)\|_g$ is constant. On the other hand, if $\|\gamma'(t)\|_g$ is constant, then by the chain rule, $\langle\gamma'(t), \gamma'(t)\rangle$ is also constant. So by the above formula, $\left\langle \frac{D\gamma'(t)}{dt}, \gamma'(t) \right\rangle_g = 0$. Since $T_{\gamma(t)}S^1$ is 1-dimensional, and $\gamma'(t)$ lies in $T_{\gamma(t)}S^1$, a vector being perpendicular to $\gamma'(t)$ implies that its tangential component is 0. Therefore, $\frac{D\gamma'(t)}{dt} = 0$.

Example 2.17. Let $M = S^2$, embedded as the unit sphere in $\mathbb{R}^3$, and ∇^T be inherited from the directional derivative on $\mathbb{R}^3$. As in Example 2.16, let g be the metric induced by $\mathbb{R}^3$. Let $\gamma(t)$ be a great circle on S^2, such that $\gamma(t)$ has an arc length of t. With respect to g, $\|\gamma'(t)\|$ is constant. By the formula in Example 2.16, we have that $\gamma''(t)$ is perpendicular to $\gamma'(t)$. Since γ only lies on the plane of the great circle, $\gamma'(t)$ and $\gamma''(t)$ must also be on the plane of the great circle. We know $\gamma'(t)$ is tangent to the sphere, so this forces $\gamma''(t)$ to point along the radius of the circle. Therefore, since the tangential component of $\gamma''(t)$ is 0, we have that $\frac{D\gamma'(t)}{dt} = 0$.

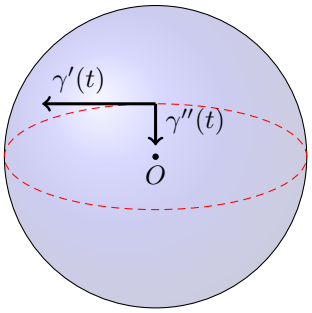

Figure 2: A great circle on S^2. The dashed red line shows the geodesic curve $\gamma(t)$. The velocity of the curve at a point is shown by $\gamma'(t)$, with $\gamma''(t)$ perpendicular to it.

For a manifold M, if we fix the initial conditions, a geodesic always exists and is unique.

Theorem 2.18. *Let $p \in M$ and $v_0 \in T_pM$. Then there exists a unique geodesic $\gamma : (-\epsilon, \epsilon) \to M$ with $\gamma(0) = p$ and $\gamma'(0) = v_0$.*

Proof. Let γ be a curve on M and $(U, \varphi = (x^1, ..., x^n))$ be a chart containing p. Let $y^i(t) = (x^i \circ \gamma)(t)$. We can describe the desired geodesic as the solution to a system of ODEs, and refer to the existence and uniqueness theorem of solutions of ODEs. For convenience, we will simply write functions such as $y^i(t)$ instead as y^i and $\partial_i|_{\gamma(t)}$ simply as ∂_i. The velocity of γ can be expressed in local coordinates:

$$\gamma' = \sum_i (y^i)' \partial_i.$$

For γ to be a geodesic, it must be the case that:

$$\frac{D\gamma'}{dt} = 0.$$

By the properties of covariant differentiation:

$$\begin{aligned}
\frac{D\gamma'}{dt} &= \sum_k (y^k)'' \partial_k + \sum_j (y^j)' \frac{D\partial_j}{dt} \\
&= \sum_k (y^k)'' \partial_k + \sum_j (y^j)' \nabla_{\gamma'} \partial_j \\
&= \sum_k (y^k)'' \partial_k + \sum_{i,j} (y^j)' \nabla_{(y^i)' \partial_i} \partial_j \\
&= \sum_k (y^k)'' \partial_k + \sum_{i,j,k} (y^i)' (y^j)' \Gamma_{ij}^k \partial_k.
\end{aligned}$$

Therefore, γ is a geodesic if and only if for all $k = 1, .., n$:

$$(y^k)'' + \sum_{i,j} (y^i)'(y^j)'\Gamma_{ij}^k = 0.$$

So by the existence and uniqueness theorem of solutions of ODEs, there exists $\gamma : (-\epsilon, \epsilon) \to M$ satisfying this system, such that $\gamma(0) = p$ and $\gamma'(0) = v_0$. □

The system of ODEs in the previous theorem are commonly referred to as the **geodesic equations**.

By their existence and uniqueness, geodesics induce a map $\exp_p : U \to M$, defined by $\exp_p(v) = \gamma(1)$, where U is an open subset of T_pM containing p and $\gamma(t)$ is the unique curve such that $\gamma(0) = p$ and $\gamma'(0) = v$. This is known as the **exponential map**.

2.5 Parallel Translation

Next we generalize the notion of two tangent vectors in $\mathbb{R}^n$ being parallel to each other. Let $\gamma : [a, b] \to M$ be a curve on a manifold M and let $X(t)$ be a vector field defined on $\gamma(t)$, parameterized by $t \in [a, b]$. When we looked at geodesics, we were choosing $X(t) = \gamma'(t)$, but here, we allow $X(t)$ to be any smooth vector field over $\gamma(t)$. An affine connection ∇ allows us to understand how $X(t)$ changes with respect to t as we move along the curve γ. Similarly, we can think about what it means for $X(t)$ to be constant along γ.

Definition 2.19. We say a vector field $X(t)$ along a curve $\gamma : [a, b] \to M$ is **parallel** with respect to an affine connection ∇ if $\frac{DX(t)}{dt} = 0$ for all t.

It is actually the case that if we fix a tangent vector on a curve γ, there exists a unique parallel vector field along γ satisfying that initial condition. This result comes from the fact that a parallel vector field can be described as the solution to a system of ODEs, similar to the case of geodesics.

Theorem 2.20. *Let $\gamma : [a, b] \to M$ be a curve on a Riemannian manifold (M, g) with affine connection ∇. Given a tangent vector v_0 at $\gamma(t_0)$ for some $t_0 \in (a, b)$, there exists a unique parallel vector field $X(t)$ on γ such that $X(t_0) = v_0$.*

Proof. We will describe the parallel vector field by a system of ODEs and appeal to the uniqueness and existence theorem for the solution of ODEs. Assume for now that γ is a curve that lies in a single coordinate chart, $(U, \varphi = (x^1, ..., x^n))$. Let $X = \sum_i X^i \partial_i$, and let

$y^i = x^i \circ \gamma$. Then we have:

$$\gamma' = \sum_i (y^i)' \partial_i.$$

The condition for X to be parallel is:

$$\frac{DX}{dt} = 0.$$

By the properties of covariant differentiation:

$$\begin{aligned}
\frac{DX}{dt} &= \sum_k (X^k)' \partial_k + \sum_j X^j \frac{D\partial_j}{dt} \\
&= \sum_k (X^k)' \partial_k + \sum_j X^j \nabla_{\gamma'(t)} \partial_j \\
&= \sum_k (X^k)' \partial_k + \sum_{i,j} X^j \nabla_{(y^i)' \partial_i} \partial_j \\
&= \sum_k (X^k)' \partial_k + \sum_{i,j,k} X^j (y^i)' \Gamma_{ij}^k \partial_k
\end{aligned}$$

By the parallel condition, this is equivalent to solving the system of n ODEs:

$$(X^k)' = -\sum_{i,j} X^j (y^i)' \Gamma_{ij}^k$$

for $k = 1, ..., n$. If $\gamma(t)$ is contained in a single chart, then since this is a system of linear ODEs, a solution exists for the entirety of $[a, b]$. The solution also guarantees that we obtain a smooth vector field. If $\gamma(t)$ is covered by multiple charts, then given the initial condition $X(t_0) = v_0$, there exists a unique solution in an open interval $(t_0 - \epsilon, t_0 + \epsilon)$. However, we can choose coordinates centered at $\gamma(t)$ for any $t \in [a, b]$ and by uniqueness, solutions will have to agree on the overlap of open sub-intervals of $[a, b]$. Furthermore, since $[a, b]$ is compact, it is covered by finitely many open intervals. Hence there is one solution over the entire interval $[a, b]$. □

Definition 2.21. Given a curve $\gamma : [a, b] \to M$ and $v \in T_{\gamma(t_0)} M$ for some $t_0 \in [a, b]$, we say $u \in T_{\gamma(t_0+h)} M$ is obtained from v by **parallel translation** if the unique parallel vector field $X(t)$ satisfying $X(t_0) = v$ also has $X(t_0 + h) = u$.

Once a curve has been fixed, the parallel translation along the curve induced by an affine connection defines a vector space isomorphism for any pair of points on a curve. This isomorphism is denoted by:

$$P_{\gamma(a)\gamma(b)} : T_{\gamma(a)} M \to T_{\gamma(b)} M.$$

In general, the isomorphism depends on the particular choice of γ.

Example 2.22. Let $M = \mathbb{R}^2$ and $\gamma : [0, \pi] \to \mathbb{R}^2$ be defined to be $\gamma(t) = (\cos t, \sin t)$. Suppose we fix $v = \partial_y|_{\gamma(0)} \in T_{\gamma(0)}\mathbb{R}^2$ and we wish to compute $P_{\gamma(0)\gamma(\pi)}(v)$ where ∇ is the directional derivative on $\mathbb{R}^2$. If we can come up with a vector field $X(t)$ on $\gamma(t)$ which is parallel and satisfies $X(0) = v$, then $X(t)$ will give us the parallel translation of v along the entire curve. If we let $X(t) = \partial_y|_{\gamma(t)}$, then $X(0) = v$, and since $D_{\gamma'(t)}\partial_y|_{\gamma(t)} = 0$, we must have $\frac{DX}{dt} = 0$ for all t. Therefore, $P_{\gamma(0)\gamma(\pi)}(v) = \partial_y|_{\gamma(\pi)}$.

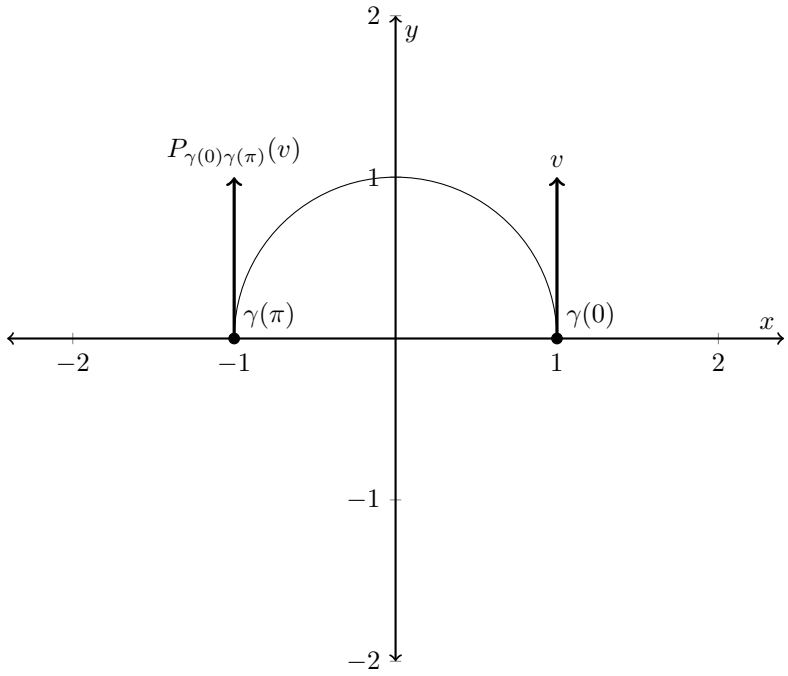

Figure 3: Parallel translation on $\mathbb{R}^2$ along the curve $\gamma(t) = (\cos t, \sin t)$.

Example 2.23. A version of this example can be found at [31]. Consider the unit sphere S^2 embedded in $\mathbb{R}^3$ with affine connection ∇^T. Let $\gamma(t) = (\frac{\sqrt{2}}{2}\cos t, \frac{\sqrt{2}}{2}\sin t, \frac{\sqrt{2}}{2})$, on the domain $[0, 2\pi]$, and let $v = (-\frac{\sqrt{2}}{2}, 0, \frac{\sqrt{2}}{2})$ be centered at $\gamma(0)$. It is tempting to think that $P_{\gamma(0)\gamma(2\pi)}(v) = v$, that is, translating v around the closed loop will result in getting v back. However, this is not the case. To help visualize what the parallel transport will look like on the sphere, picture the cone C, with a vertex lying above the sphere's origin, which is tangent to the sphere at $\gamma(t)$.

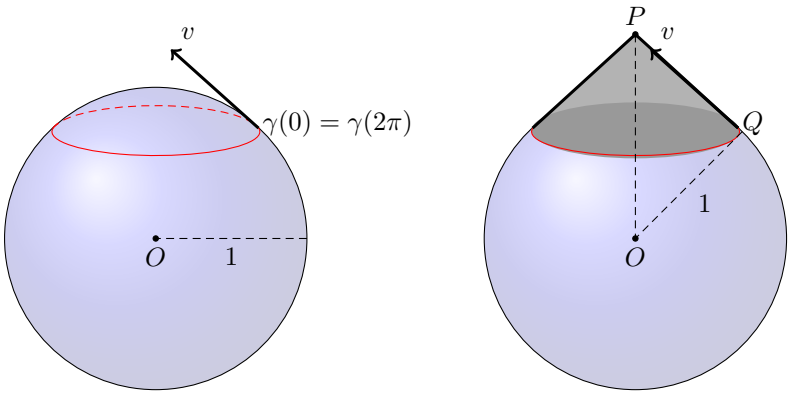

Figure 4: Parallel translation around a closed curve (in red) on S^2. The right image shows the cone tangent to S^2 along $\gamma(t)$, and the triangle formed by P, O, and $Q = \gamma(0)$.

The key thing to note is if two surfaces embedded in $\mathbb{R}^3$ are tangent, then at the point of tangency, their respective tangent spaces coincide. That is, for all t, $p = \gamma(t)$ can be thought of as a point in $\mathbb{R}^3$, in S^2, and in C. T_pS^2 and T_pC can be thought of as subspaces of $T_p\mathbb{R}^3$, but because of the tangency, $T_pS^2 = T_pC$. Since the tangent spaces T_pS^2 and T_pC coincide along all of γ, parallel translation with respect to either surface along γ will agree with the other.

Additionally, we can "flatten" the cone into $\mathbb{R}^2$. If we were to cut the cone along v and lay it flat, we would get:

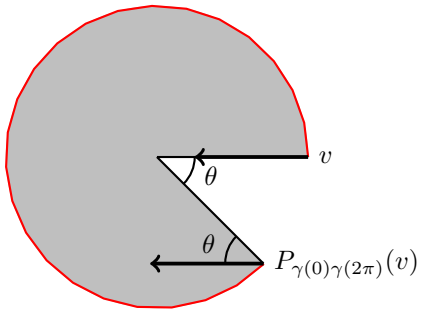

Figure 5: The flattened cone, on which parallel translation mimics parallel translation on $\mathbb{R}^2$.

Even though C is not smooth and the tangent space at the vertex of the cone is not defined, this does not pose a problem, since we are only interested in points along $\gamma(t)$. The "flattening" is a local isometric embedding of $\gamma(t)$ in $\mathbb{R}^2$, by which for all t, $T_{\gamma(t)}C = T_{\gamma(t)}\mathbb{R}^2$. Therefore, parallel translation along γ in C will agree with parallel translation on $\mathbb{R}^2$, using the directional derivative.

To determine how much v will change after translation, we can compute θ: the angle

between v and $P_{\gamma(0)\gamma(2\pi)}(v)$. Note the radius of the curve $\gamma(t)$ as it appears on the sphere is $\frac{\sqrt{2}}{2}$. Consider the triangle formed by P, O, and $Q = \gamma(0)$. Since the cone is tangent to the sphere, $\angle PQO = 90^o$. Furthermore, since the x-coordinate of Q is $\frac{\sqrt{2}}{2}$, and $\cos(45^o) = \frac{\sqrt{2}}{2}$, we have $\angle POQ = 45^o$. Therefore, $\triangle POQ$ is isosceles, which means the radius of the flattened cone is equal to the radius of the sphere, which is 1.

The ratio of the radius of the curve $\gamma(t)$ to the radius of the flattened cone is the same as the ratio of their circumferences. Therefore, $\theta = 2\pi(1 - \frac{\sqrt{2}}{2})$, and we can conclude that $P_{\gamma(0)\gamma(2\pi)}(v)$ will make an angle of $2\pi(1 - \frac{\sqrt{2}}{2})$ with v. This proves that parallel translation along a closed curve does not necessarily give the starting vector back.

Finally, we remark that the covariant derivative along a curve $\gamma(t)$ can also be written in terms of parallel translation.

Theorem 2.24. *Given an affine connection ∇ on a manifold M, a curve $\gamma : [a, b] \to M$, and a vector field $Y(t)$ defined on $\gamma(t)$ we have:*

$$\frac{DY(t_0)}{dt} = \lim_{h\to 0} \frac{P_{\gamma(t_0+h)\gamma(t_0)}(Y(t_0 + h)) - Y(t_0)}{h}.$$

Proof. Let $\{\partial_1, ..., \partial_n\}$ be a basis for $T_{\gamma(t_0)}M$. For some $\epsilon > 0$, we can parallel translate $\partial_i|_{\gamma(t_0)}$ to any point $\gamma(t)$ for $t \in (t_0 - \epsilon, t_0 + \epsilon)$. In other words, in these local coordinates:

$$\frac{D\partial_i|_{\gamma(t_0)}}{dt} = 0.$$

By the Leibniz rule, we have:

$$\begin{aligned}
\frac{DY(t_0)}{dt} &= \sum_i (Y^i)'(t_0)\partial_i|_{\gamma(t_0)} + Y^i(t_0)\frac{D\partial_i|_{\gamma(t_0)}}{dt} \\
&= \sum_i (Y^i)'(t_0)\partial_i|_{\gamma(t_0)} \\
&= \lim_{h\to 0} \frac{\sum_i Y^i(t_0 + h) - \sum_i Y^i(t_0)}{h}\partial_i|_{\gamma(t_0)} \\
&= \lim_{h\to 0} \frac{P_{\gamma(t_0+h)\gamma(t_0)}(Y(t_0 + h)) - Y(t_0)}{h}.
\end{aligned}$$

The last equality is by the choice of a parallel frame along γ. Therefore, we have the desired result. □

2.6 Metric Connections

So far, the Riemannian metric and affine connections on M are independently defined objects. There is no particular reason for an affine connection to be related to the metric, so we will define a class of connections that interact in a desirable way with the metric.

Definition 2.25. We say an affine connection ∇ on (M, g) is a **metric connection** (or compatible with the metric) if for all $X, Y, Z \in \mathfrak{X}(M)$ the following equation is satisfied:

$$Z\langle X, Y\rangle_g = \langle \nabla_Z X, Y\rangle_g + \langle X, \nabla_Z Y\rangle_g.$$

The criterion to be a metric connection can also be interpreted as a requirement on the parallel translation given by a connection.

Theorem 2.26. *An affine connection ∇ is a metric connection if and only if for every curve $\gamma : [a, b] \to M$ the vector space isomorphism $P_{\gamma(a)\gamma(b)}$ induced by the parallel transport of ∇ is an isometry.*

Proof. ($\Longrightarrow$) Suppose ∇ is a metric connection. Let X and Y be parallel vector fields on a curve γ, and set $Z = \gamma'$. Then:

$$\begin{aligned} Z\langle X, Y\rangle_g &= \langle \nabla_Z X, Y\rangle_g + \langle X, \nabla_Z Y\rangle_g \\ &= \langle 0, Y\rangle_g + \langle X, 0\rangle_g \\ &= 0. \end{aligned}$$

Therefore, the inner product along γ is preserved by parallel translation.

($\Longleftarrow$) Let $X, Y, Z \in \mathfrak{X}(M)$. Let $p \in M$ and consider a curve γ such that $\gamma(0) = p$ and $\gamma'(0) = Z(p)$. We will denote $X(\gamma(t))$ by $X(t)$. Therefore, $Z(p)\langle X, Y\rangle_g = \frac{d}{dt}\Big|_{t=0} \langle X(t), Y(t)\rangle_g$, so we will prove that:

$$\frac{d}{dt}\Big|_{t=0} \langle X(t), Y(t)\rangle_g = \langle X(t), \nabla_{\gamma'(t)} Y(t)\rangle_g + \langle \nabla_{\gamma'(t)} X(t), Y(t)\rangle_g.$$

For clarity, we will denote $P_{\gamma(h)\gamma(0)}$ by P_{-h}. By assumption, this map does not change the

inner product, so we have:

$$\begin{aligned}
\left.\frac{d}{dt}\right|_{t=0}\langle X(t),Y(t)\rangle_g &= \lim_{h\to 0}\frac{\langle X(t+h),Y(t+h)\rangle_g - \langle X(t),Y(t)\rangle_g}{h}\\
&= \lim_{h\to 0}\frac{1}{h}\left(\langle P_{-h}(X(t+h)),P_{-h}(Y(t+h))\rangle_g - \langle X(t),Y(t)\rangle_g\right)\\
&= \lim_{h\to 0}\frac{1}{h}\left(\langle P_{-h}(X(t+h)),P_{-h}(Y(t+h))\rangle_g - \langle P_{-h}(X(t+h)),Y(t)\rangle_g\right.\\
&\qquad \left. + \langle P_{-h}(X(t+h)),Y(t)\rangle_g - \langle X(t),Y(t)\rangle_g\right)\\
&= \lim_{h\to 0}\frac{1}{h}\langle P_{-h}(X(t+h)),P_{-h}(Y(t+h))-Y(t)\rangle_g\\
&\qquad + \lim_{h\to 0}\frac{1}{h}\langle P_{-h}(X(t+h))-X(t),Y(t)\rangle_g\\
&= \langle X(t),\nabla_{\gamma'(t)}Y(t)\rangle_g + \langle \nabla_{\gamma'(t)}X(t),Y(t)\rangle_g,
\end{aligned}$$

where the final step is due to Theorem 2.24. □

2.7 Torsion and the Levi-Civita Connection

A Riemannian manifold (M,g) may allow many different metric connections. In this section we will see that if we impose another condition, there is only one connection satisfying these two requirements. To this end, we have the following definition.

Definition 2.27. For a connection ∇ on (M,g), the map $T:\mathfrak{X}(M)\times\mathfrak{X}(M)\to\mathfrak{X}(M)$ given by:

$$T(X,Y)=\nabla_X Y-\nabla_Y X-[X,Y]$$

is called the **torsion** of ∇. If $T=0$, then we say ∇ is **torsion-free**.

The torsion can be shown to be a $\binom{2}{1}$-tensor field on M, meaning it is $C^\infty(M)$-linear in both entries.

Sometimes, instead of calling ∇ torsion-free, we call it symmetric. This name comes from the following theorem.

Theorem 2.28. *An affine connection ∇ is torsion-free if and only if in some local coordinates, $\Gamma^k_{ij}=\Gamma^k_{ji}$ for all i,j,k.*

Proof. It suffices to show that, in some local coordinates, $T(\partial_i,\partial_j)=0$ if and only if $\Gamma^k_{ij}=\Gamma^k_{ji}$

for all k.

$$\begin{aligned} T(\partial_i, \partial_j) = 0 &\iff \nabla_{\partial_i}\partial_j - \nabla_{\partial_j}\partial_i - [\partial_i, \partial_j] = 0 \\ &\iff \nabla_{\partial_i}\partial_j - \nabla_{\partial_j}\partial_i = 0 \\ &\iff \sum_k \Gamma_{ij}^k \partial_k = \sum_k \Gamma_{ji}^k \partial_k \\ &\iff \Gamma_{ij}^k = \Gamma_{ji}^k \text{for all } k \end{aligned}$$

□

Now we see the fundamental theorem of Riemannian geometry.

Theorem 2.29. *There exists a unique connection ∇ on (M, g) such that:*

1. *∇ is a metric connection,*
2. *∇ is torsion-free.*

Furthermore, in local coordinates $(x^1, ..., x^n)$, the connection is written in terms of the metric as:

$$\Gamma_{ij}^k = \frac{1}{2} g^{kl}(\partial_i g_{jl} + \partial_j g_{il} - \partial_l g_{ij})$$

where $G^{-1} = (g^{kl})$ is the inverse matrix of the metric.

Proof. Condition 1) means that ∇ satisfies the formula:

$$\partial_i \langle \partial_j, \partial_k \rangle_g = \langle \nabla_{\partial_i}\partial_j, \partial_k \rangle_g + \langle \partial_j, \nabla_{\partial_i}\partial_k \rangle_g$$

for all i, j, k. By the definition of the metric, we know that:

$$\langle \partial_i, \partial_j \rangle_g = g_{ij}.$$

And by the properties of this inner product, we have:

$$\begin{aligned} \langle \nabla_{\partial_i}\partial_j, \partial_k \rangle_g &= \left\langle \sum_l \Gamma_{ij}^l \partial_l, \partial_k \right\rangle_g \\ &= \sum_l \Gamma_{ij}^l \langle \partial_l, \partial_k \rangle_g \\ &= \sum_l \Gamma_{ij}^l g_{lk}. \end{aligned}$$

Therefore 1) can be rewritten as:

$$\partial_i g_{jk} = \sum_l (\Gamma^l_{ij} g_{lk} + \Gamma^l_{ik} g_{lj}).$$

Since condition 2) allows the bottom indices of the Christoffel symbols to commute, we have:

$$\begin{aligned}
\frac{1}{2} g^{kl}(\partial_i g_{jl} + \partial_j g_{il} - \partial_l g_{ij}) &= \frac{1}{2} g^{kl} \sum_m (\Gamma^m_{ij} g_{ml} + \Gamma^m_{il} g_{mj} + \Gamma^m_{ji} g_{ml} + \Gamma^m_{jl} g_{mi} - \Gamma^m_{li} g_{mj} - \Gamma^m_{lj} g_{mi}) \\
&= \frac{1}{2} g^{kl} \sum_m (\Gamma^m_{ij} g_{ml} + \Gamma^m_{il} g_{mj} + \Gamma^m_{ij} g_{ml} + \Gamma^m_{jl} g_{mi} - \Gamma^m_{il} g_{mj} - \Gamma^m_{jl} g_{mi}) \\
&= \frac{1}{2} g^{kl} \sum_m (2\Gamma^m_{ij} g_{ml}) \\
&= \sum_m \Gamma^m_{ij} g^{kl} g_{ml} \\
&= \sum_m \Gamma^m_{ij} \delta^k_m \\
&= \Gamma^k_{ij}.
\end{aligned}$$

Since the expression on the left-hand side is only in terms of the metric, the connection is unique. □

Definition 2.30. The unique torsion-free metric connection on a Riemannian manifold is known as the **Levi-Civita connection**.

2.8 Curvature and Flat Manifolds

In Example 2.23, it may seem unusual that parallel translation along a closed loop results in a different vector than the starting vector, whereas on $\mathbb{R}^n$, this is not the case. However, this is due to the sphere having an intrinsic curvature that is not present in Euclidean space. To take a closer look at this, we have the following definition:

Definition 2.31. Given a connection ∇ on (M, g), the map $R : \mathfrak{X}(M) \times \mathfrak{X}(M) \times \mathfrak{X}(M) \to \mathfrak{X}(M)$ given by:

$$R(X, Y)Z = \nabla_X \nabla_Y Z - \nabla_Y \nabla_X Z - \nabla_{[X,Y]} Z$$

is called the **curvature endomorphism** of ∇. If $R(X, Y)Z = 0$ for all X, Y, Z, then we say M is **∇-flat**.

It follows from straight-forward calculations that the curvature endomorphism is a $\binom{3}{1}$-tensor field on M.

There are several equivalent ways of formulating flatness on a manifold. For example, as was alluded to at the beginning of the section, M being ∇-flat can be understood in terms of the parallel translation induced by ∇.

Theorem 2.32. *The following are equivalent:*

1. *M is ∇-flat.*
2. *For all $p \in M$, there exists a local parallel frame around p.*
3. *Parallel translation with respect to ∇ is path-independent. That is, if γ and $\tilde{\gamma}$ are two curves $\mathbb{R} \to U \subset M$ with $\gamma(0) = \tilde{\gamma}(0)$ and $\gamma(1) = \tilde{\gamma}(1)$, then $P_{\gamma(0)\gamma(1)}(v) = P_{\tilde{\gamma}(0)\tilde{\gamma}(1)}(v)$ for all $v \in T_{\gamma(0)}M$.*

Proof. ($1 \implies 2$) This proof is due to Lee [27]. Suppose the curvature endomorphism with respect to ∇ is $R = 0$. Let $p \in M$ be centered in the chart $(U, \varphi = (x^1, ..., x^n))$, so that $\varphi(p) = (0, ..., 0)$ and $T_pM = \text{span}\{\partial_1|_p, ..., \partial_n|_p\}$. To avoid confusion, we will denote the basis of T_pM by $\{E_1, ..., E_n\}$. Without loss of generality, we can assume that the image of φ is an open hypercube in $\mathbb{R}^n$: $C = \{x \in \mathbb{R}^n \mid -\epsilon < x^i < \epsilon, \forall i \in \{1, ..., n\}\}$, where $\epsilon > 0$, since C will be contained in the image of the coordinate chart for some ϵ.

Consider fixing $j \in \{1, ..., n\}$ and using ∇ to parallel translate E_j along the x^1 direction. By doing so, we can obtain vectors in C along the entirety of the x^1-axis, which we denote by $E_j(q)$, where $\varphi(q)$ is on the x^1-axis. Then for each $E_j(q)$, we can parallel translate this vector along the x^2-direction and obtain vectors on the entire (x^1, x^2)-plane. If we continue this construction for the remaining x^i-axes, we obtain a vector field $E_j(q)$ over the entirety of U. Theorem 2.20 guarantees that $E_j(q)$ will be smooth.

We now show by induction that if $\varphi(q)$ lies in the span of the first k axes of C, then $\nabla_{\partial_i} E_j(q) = 0$ for all $i \in \{1, ..., k\}$. If $k = 1$, then $\varphi(q)$ is on the x^1-axis, so by construction, $\nabla_{\partial_1} E_j(q) = 0$. Suppose that the inductive hypothesis holds for $k = m$, with $1 \leq m < n$. We must show that if $\varphi(q)$ lies in the span of the first $m + 1$ axes, then $\nabla_{\partial_i} E_j(q) = 0$ for all $i \in \{1, ..., m + 1\}$. We know that $E_j(q)$ was obtained by parallel translating a vector in the span of the first m axes along the x^{m+1}-axis, so by construction, $\nabla_{\partial_{m+1}} E_j(q) = 0$. If $\varphi(q)$ happens to lie only in the span of the first m axes, then $\nabla_{\partial_i} E_j(q) = 0$ for all $i \in \{1, ..., m\}$ by the inductive hypothesis. We will show that $\nabla_{\partial_{m+1}} \nabla_{\partial_i} E_j(q) = 0$ for all $i \in \{1, ..., m\}$.

Note that by our assumption of ∇-flatness, and the fact that $[\partial_{m+1}, \partial_i] = 0$ for $i \in \{1, ..., m\}$,

we have:

$$\begin{aligned} 0 &= \nabla_{\partial_{m+1}} \nabla_{\partial_i} E_j(q) - \nabla_{\partial_i} \nabla_{\partial_{m+1}} E_j(q) - \nabla_{[\partial_{m+1}, \partial_i]} E_j(q) \\ 0 &= \nabla_{\partial_{m+1}} \nabla_{\partial_i} E_j(q) - \nabla_{\partial_i} \nabla_{\partial_{m+1}} E_j(q) \\ \nabla_{\partial_i} \nabla_{\partial_{m+1}} E_j(q) &= \nabla_{\partial_{m+1}} \nabla_{\partial_i} E_j(q). \end{aligned}$$

But $\nabla_{\partial_{m+1}} E_j(q) = 0$, so:

$$\begin{aligned} \nabla_{\partial_i} 0 &= \nabla_{\partial_{i+1}} \nabla_{\partial_i} E_j(q) \\ 0 &= \nabla_{\partial_{i+1}} \nabla_{\partial_i} E_j(q). \end{aligned}$$

Therefore, $\nabla_{\partial_i} E_j(q)$ is constant along the x^{m+1} direction. However, we know that when $x^{m+1}(q) = 0$, $\nabla_{\partial_i} E_j(q) = 0$ by the inductive hypothesis, so by uniqueness of the parallel transport $\nabla_{\partial_i} E_j(q) = 0$ on the entire span of the first $m+1$ axes. Finally, if we set $k = n$, we can conclude that $E_j(q)$ is parallel on all of U. Therefore, following the construction for each j, we obtain a local parallel frame.

($2 \implies 1$) Let $\{s_1, ..., s_n\}$ be a local parallel frame on $U \subset M$. That is, $\nabla_X s_i(p) = 0$ for all $p \in U$ and all $X \in \mathfrak{X}(M)$ for $i = 1, ..., n$. Then for all $X, Y \in \mathfrak{X}(M)$:

$$R(X,Y)(s_i(p)) = \nabla_X \nabla_Y s_i(p) - \nabla_Y \nabla_X s_i(p) - \nabla_{[X,Y]} s_i(p) = 0.$$

Since the curvature endomorphism is a tensor, it is $C^\infty(M)$-linear in each entry. If $Z(p) = \sum_i Z^i(p) s_i(p)$, then:

$$\begin{aligned} R(X,Y)Z(p) &= R(X,Y)\left(\sum_i Z^i(p) s_i(p)\right) \\ &= \sum_i Z^i(p) R(X,Y) s_i(p) \\ &= \sum_i Z^i(p) 0 \\ &= 0. \end{aligned}$$

So the curvature endomorphism vanishes.

($2 \implies 3$) Let $\{s_1, ..., s_n\}$ be a local parallel frame on $U \subset M$. Let $X_p \in T_pM$ for $p \in U$, which we can write in terms of the local frame:

$$X_p = \sum_i x_i s_i(p)$$

where $x_i \in \mathbb{R}$. Let $\gamma : \mathbb{R} \to U$ be such that $\gamma(0) = p$ and $\gamma(1) = q$, where $q \in U$. Note that since s_i is parallel, $\nabla_{\gamma'(t)} s_i(p) = 0$ for all $t \in [0,1]$, so therefore by uniqueness of parallel translation,

$$P_{\gamma(0)\gamma(1)}(s_i(p)) = s_i(q).$$

This result is independent of the choice of γ. Since $P_{\gamma(0)\gamma(1)}$ is an isomorphism from T_pM to T_qM, we have:

$$\begin{aligned} P_{\gamma(0)\gamma(1)}(X_p) &= P_{\gamma(0)\gamma(1)}\left(\sum_i x_i s_i(p)\right) \\ &= \sum_i \left(x_i P_{\gamma(0)\gamma(1)}(s_i(p))\right) \\ &= \sum_i x_i s_i(q). \end{aligned}$$

Hence, parallel transport is path-independent.

(3 $\implies$ 2) Let $p \in M$ be centered in the chart $(U, \varphi = (x^1, ..., x^n))$, with $T_pM = \text{span}\{\partial_1|_p, ..., \partial_n|_p\}$. Suppose that parallel translation with respect to ∇ is path-independent. Denote $\partial_i|_p$ by E_i. We can define a local parallel frame $\{s_1, ..., s_n\}$ by setting

$$s_i(q) = P_{\gamma(0)\gamma(1)}(E_i)$$

where $\gamma : \mathbb{R} \to U$ is any curve with $\gamma(0) = p$ and $\gamma(1) = q$. This is well-defined since parallel translation is path-independent. Furthermore, $s_i(p)$ is parallel on U and smooth by construction. Therefore, $\{s_1, ..., s_n\}$ is a local parallel frame. □

Corollary 2.33. *If M is ∇-flat, and $(U, \varphi = (x^1, ..., x^n))$ is a local chart centered at $p \in M$, then for any curve $\gamma : \mathbb{R} \to U$ with $\gamma(0) = \gamma(1) = p$, the isomorphism $P_{\gamma(0)\gamma(1)}$ is the identity map on T_pM.*

Proof. By the previous theorem, parallel translation with respect to ∇ is path-independent, and clearly when γ is the constant path ($\gamma(t) = p$ for all t), $P_{\gamma(0)\gamma(1)}(v) = v$, for all $v \in T_pM$. □

In Example 2.23, S^2 was not ∇-flat, so based on the results in this section, it is not surprising that the vector obtained by translating around a closed loop was not the same as the starting vector.

2.9 Dual Affine Connections

We have seen that when ∇ is a metric connection, the inner product is preserved by parallel translation. However, even when ∇ is not a metric connection, there exists a so-called dual affine connection which, in conjunction with the original connection, preserves the inner product.

Definition 2.34. Given an affine connection ∇ on (M, g), the **dual affine connection** ∇^* is the connection that satisfies:

$$Z\langle X, Y\rangle_g = \langle \nabla_Z X, Y\rangle_g + \langle X, \nabla^*_Z Y\rangle_g$$

for all $X, Y, Z \in \mathfrak{X}(M)$.

First, we will see that the dual affine connection always exists and is unique, as well as the reason why it can be considered a dual.

Theorem 2.35. *The dual affine connection exists and is unique.*

Proof. Recall from Theorem 2.29 that:

$$\langle \nabla_{\partial_i} \partial_j, \partial_k\rangle_g = \sum_l \Gamma^l_{ij} g_{lk}.$$

Therefore, ∇^* can be written in terms of ∇ as:

$$\begin{aligned}
\partial_i g_{jk} &= \sum_l \Gamma^l_{ij} g_{lk} + \sum_l (\Gamma^*)^l_{ik} g_{lj}, \\
\partial_i g_{jk} - \sum_l \Gamma^l_{ij} g_{lk} &= \sum_l (\Gamma^*)^l_{ik} g_{lj} \\
\sum_j \left(\partial_i g_{jk} - \sum_l \Gamma^l_{ij} g_{lk} \right) g^{jm} &= \sum_{j,l} (\Gamma^*)^l_{ik} g_{lj} g^{jm} \\
\sum_j \left(\partial_i g_{jk} - \sum_l \Gamma^l_{ij} g_{lk} \right) g^{jm} &= \sum_l (\Gamma^*)^l_{ik} \delta^m_l \\
\sum_j \left(\partial_i g_{jk} - \sum_l \Gamma^l_{ij} g_{lk} \right) g^{jm} &= (\Gamma^*)^m_{ik}
\end{aligned}$$

Now suppose ∇^1 and ∇^2 are both dual to ∇. Then for all $X, Y, Z \in \mathfrak{X}(M)$:

$$\begin{aligned}
0 &= \langle \nabla_Z X, Y\rangle_g + \langle X, \nabla^1_Z Y\rangle_g - \langle \nabla_Z X, Y\rangle_g - \langle X, \nabla^2_Z Y\rangle_g \\
0 &= \langle X, \nabla^1_Z Y\rangle_g - \langle X, \nabla^2_Z Y\rangle_g \\
0 &= \langle X, \nabla^1_Z Y - \nabla^2_Z Y\rangle_g 0 && = \nabla^1_Z Y - \nabla^2_Z Y \\
\nabla^1_Z Y &= \nabla^2_Z Y.
\end{aligned}$$

Therefore, the dual affine connection is unique. □

Theorem 2.36. *The connection ∇^* is dual in the sense that $(\nabla^*)^* = \nabla$.*

Proof. Let $X, Y, Z \in \mathfrak{X}(M)$. By symmetry of the metric:

$$\begin{aligned}
Z\langle X, Y\rangle_g &= Z\langle Y, X\rangle_g \\
&= \langle \nabla_Z Y, X\rangle_g + \langle Y, \nabla^*_Z X\rangle_g \\
&= \langle \nabla^*_Z X, Y\rangle_g + \langle X, \nabla_Z Y\rangle_g.
\end{aligned}$$

So $(\nabla^*)^* = \nabla$. □

A natural question to ask is: under what conditions is an affine connection equal to its own dual?

Theorem 2.37. *An affine connection ∇ is self-dual ($\nabla^* = \nabla$) if and only if ∇ is a metric connection.*

Proof. ($\Longrightarrow$) Substituting ∇^* with ∇ in the definition of the dual affine connection yields exactly the requirement for ∇ to be a metric connection.

($\Longleftarrow$) If ∇ satisfies the formula:

$$Z\langle X, Y\rangle_g = \langle \nabla_Z X, Y\rangle_g + \langle X, \nabla_Z Y\rangle_g$$

then by uniqueness of the dual affine connection, the only choice for ∇^* is $\nabla^* = \nabla$. □

An interesting fact about dual connections is that their average is always a metric connection, even if neither of them is a metric connection. Linear combinations of affine connections are, in general, not connections since the Leibniz rule may fail. However, convex combinations are valid, so the average of two connections is a connection.

Lemma 2.38. *A finite convex combination of affine connections is an affine connection.*

Proof. Let $\nabla^1, ..., \nabla^k$ be affine connections on M. Set $\nabla = \sum_i \alpha_i \nabla^i$ where $\sum_i \alpha_i = 1$ and $\alpha_i \geq 0$ for all i. Then ∇ will have linearity in the first component:

$$\begin{aligned}\nabla_{fX} Y &= \sum_i \alpha_i \nabla^i_{fX} Y \\ &= \sum_i \alpha_i f \nabla^i_X Y \\ &= f \sum_i \alpha_i \nabla^i_X Y \\ &= f \nabla_X Y.\end{aligned}$$

And ∇ will satisfy the Leibniz rule:

$$\begin{aligned}\nabla_X fY &= \sum_i \alpha_i \nabla^i_X (fY) \\ &= \sum_i \alpha_i ((Xf)Y + f\nabla^i_X Y) \\ &= \sum_i \alpha_i (Xf)Y + \sum_i \alpha_i f \nabla^i_X Y \\ &= (Xf)Y + f\nabla_X Y.\end{aligned}$$

Hence, ∇ is an affine connection. □

Theorem 2.39. *Given an affine connection ∇ and its dual ∇^*, their average $\frac{1}{2}(\nabla + \nabla^*)$ is a metric connection.*

Proof. By the previous lemma, $\frac{1}{2}(\nabla + \nabla^*)$ is an affine connection. Let $X, Y, Z \in \mathfrak{X}(M)$. Then we have:

$$\begin{aligned}Z\langle X, Y\rangle_g &= \frac{1}{2}(2Z\langle X, Y\rangle_g) \\ &= \frac{1}{2}(\langle \nabla_Z X, Y\rangle_g + \langle X, \nabla^*_Z Y\rangle_g + \langle \nabla^*_Z X, Y\rangle_g + \langle X, \nabla_Z Y\rangle_g) \\ &= \frac{1}{2}(\langle \nabla_Z X, Y\rangle_g + \langle \nabla^*_Z X, Y\rangle_g) + \frac{1}{2}(\langle X, \nabla_Z Y\rangle_g + \langle X, \nabla^*_Z Y\rangle_g) \\ &= \langle \frac{1}{2}(\nabla + \nabla^*)_Z X, Y\rangle_g + \langle X, \frac{1}{2}(\nabla + \nabla^*)_Z Y\rangle_g.\end{aligned}$$

So $\frac{1}{2}(\nabla + \nabla^*)$ is a metric connection. □

With this result, it is tempting to find a relationship between dual affine connections and the Levi-Civita connection. However, it is not the case that their average yields a torsion-free

connection in general. To get this result, we still require that both ∇ and its dual ∇^* are torsion-free.

Theorem 2.40. *Let ∇ be an affine connection on (M, g) and its dual be ∇^*. If ∇ and ∇^* are both torsion-free, then $\frac{1}{2}(\nabla + \nabla^*)$ is the Levi-Civita connection on (M, g).*

Proof. We know that $\frac{1}{2}(\nabla + \nabla^*)$ is a metric connection, so all that remains to show is that it is torsion-free when ∇^* and ∇^* are torsion-free. This can easily be seen by the Christoffel symbols. Let $\overline{\Gamma}^k_{ij}$ denote the Christoffel symbols of the average of the affine connections. Note that:

$$\begin{aligned}\overline{\Gamma}^k_{ij} &= \left(\frac{1}{2}(\nabla + \nabla^*)_{\partial_i}\partial_j\right)^k \\ &= \left(\frac{1}{2}(\nabla_{\partial_i}\partial_j + \nabla^*_{\partial_i}\partial_j)\right)^k \\ &= \frac{1}{2}(\Gamma^k_{ij} + (\Gamma^*)^k_{ij}).\end{aligned}$$

So the Christoffel symbols of the average of the connections is simply the average of the symbols. Then it is clear that:

$$\begin{aligned}\overline{\Gamma}^k_{ij} &= \frac{1}{2}(\Gamma^k_{ij} + (\Gamma^*)^k_{ij}) \\ &= \frac{1}{2}(\Gamma^k_{ji} + (\Gamma^*)^k_{ji}) \\ &= \overline{\Gamma}^k_{ji}.\end{aligned}$$

So $\frac{1}{2}(\nabla + \nabla^*)$ is the Levi-Civita connection on (M, g). □

Remark 2.41. It is not the case that ∇ being torsion-free implies ∇^* is torsion-free. Take $\mathbb{R}^2$ with the Euclidean metric. Define ∇ by setting $\Gamma^2_{11} = 1$, and all else to be the constant function 0. We can see that ∇ is torsion-free, since clearly $\Gamma^k_{ii} = \Gamma^k_{ii}$ for $i = 1, 2$, and $\Gamma^k_{12} = 0 = \Gamma^k_{21}$ for all k. This forces ∇^* to be given by:

$$\begin{aligned}\partial_i\langle\partial_j, \partial_k\rangle_g &= \langle\nabla_{\partial_i}\partial_j, \partial_k\rangle_g + \langle\partial_j, \nabla^*_{\partial_i}\partial_k\rangle_g \\ 0 &= \Gamma^k_{ij} + (\Gamma^*)^j_{ik} \\ -\Gamma^k_{ij} &= (\Gamma^*)^j_{ik}.\end{aligned}$$

Then $(\Gamma^*)^1_{12} = -\Gamma^2_{11} = -1$, but $(\Gamma^*)^1_{21} = -\Gamma^1_{21} = 0$. So ∇^* is not torsion-free, meaning the requirement that both ∇ and ∇^* have to be torsion-free in the previous theorem cannot be relaxed to only ∇ being torsion-free.

3 Geometry of Statistical Manifolds

3.1 Parameter Spaces of Probability Distributions

Information geometry is primarily about spaces of parametric probability distributions. First, we give the definition of a discrete probability distribution.

Definition 3.1. Let $\mathcal{X} = \{X_1, ..., X_n\}$. A **discrete probability distribution** is any non-negative function $P : \mathcal{X} \to \mathbb{R}$, such that $\sum_i P(X_i) = 1$.

The set $\mathcal{X}$ is the domain of the **random variable**. This definition can be extended to any measure space, but aside from finite sets, we will only consider when $\mathcal{X} = \mathbb{R}$.

Definition 3.2. Suppose $\mathcal{X} = \mathbb{R}$. A **continuous probability distribution** is any non-negative, Lebesgue integrable function $P : \mathcal{X} \to \mathbb{R}$ such that $\int_{\mathcal{X}} P(x)dx = 1$.

Remark 3.3. The function $P(x)$ given in Definition 3.2 is generally referred to as the **probability density function**. The probability that the random variable $x \in \mathcal{X}$ lies in $[a, b] \subset \mathbb{R}$ is given by:

$$\int_a^b P(x)dx.$$

In this paper, we simply refer to this function as the probability distribution.

Consider the parametric family of normal distributions, which have two parameters $\theta = (\mu, \sigma)$:

$$P(x; \theta) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

where $\mu \in \mathbb{R}$ and $\sigma > 0$. Suppose we have a set of data $\{X_1, ..., X_n\}$, which we know follows a normal distribution, and we want to estimate the values of θ of that distribution. Since the probability density function of the normal distribution is very well-behaved, one could simply apply the method of maximum-likelihood estimation and determine that the best estimates are

$$\mu = \frac{\sum_i X_i}{n}$$

and

$$\sigma = \sqrt{\frac{\sum_i (X_i - \mu)}{n}}.$$

However, not all families of parametric distributions have a closed-form solution for their maximum-likelihood estimates. If this is the case, we can still estimate θ using a numerical

method. We will continue to use the normal distribution as a simple example, despite the existence of a closed-form solution.

Let us associate every normal distribution with a point in the open upper half-plane of $\mathbb{R}^2$, where the coordinates of a distribution are given by (μ, σ). Then, in order to find an optimal choice for θ that could have produced $\{X_1, ..., X_n\}$, we define the likelihood function:

$$L(\theta) = \prod_i P(X_i; \theta),$$

and maximize $L(\theta)$ over the upper half-plane of $\mathbb{R}^2$ using a numerical method. By doing so, we will approximate the values of θ that were most likely to have produced $\{X_1, ..., X_n\}$, in the sense that the total probability of producing these values has been maximized.

Numerical methods that are used to optimize functions (such as gradient descent or Newton's method) are often iterative: that is, they involve traversing the domain over many steps. We should, therefore, be concerned about whether or not Euclidean distance in the parameter space is a good measurement of "distance" between two normal distributions, as it comes into play in many of these numerical methods. For example, take the following four points in the parameter space:

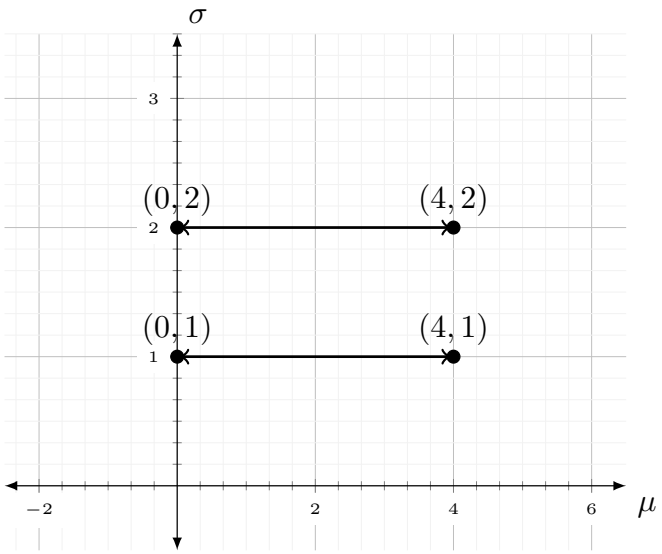

Figure 6: Points in the parameter space of normal distributions.

The upper two points and the lower two points have the same Euclidean distance in the parameter space. However, these are what their probability distributions look like:

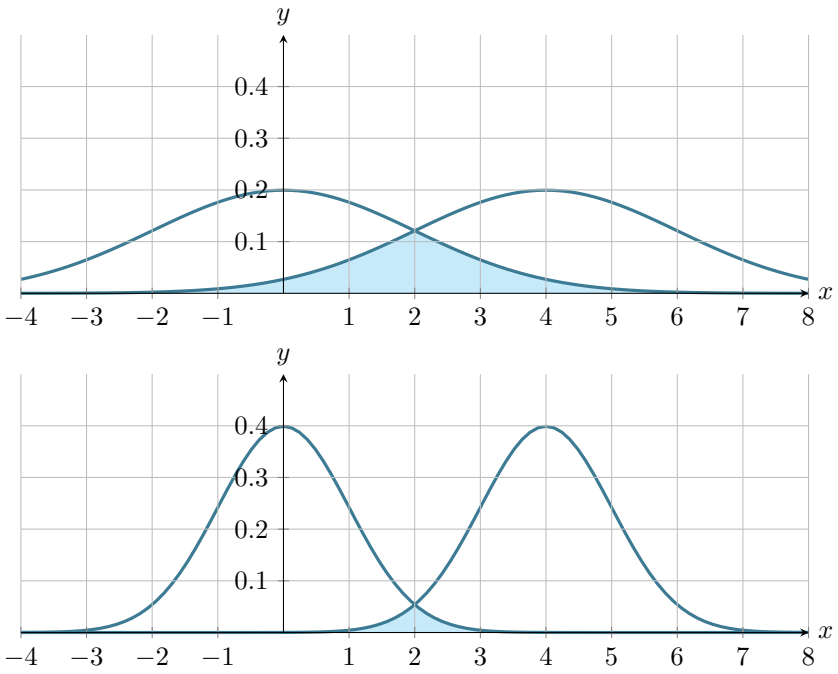

Figure 7: Comparison between two pairs of normal distributions that have the same Euclidean distance in the parameter space. The first plot shows the points $(0, 2)$ and $(4, 2)$, while the second plot shows $(0, 1)$ and $(4, 1)$.

This suggests that the improvements made by an iterative optimization method may be unpredictable, since the Euclidean distance in the parameter space is seemingly unrelated to how similar two distributions appear. The goal of this section is to show that we can interpret spaces of parametric probability distributions as Riemannian manifolds, where the parameters define local coordinates. Later, we will see a precise example of a numerical optimization method, called gradient descent, and how its performance can be improved by leveraging Riemannian geometry.

3.2 Divergences of Probability Distributions

In this section, we will introduce the concept of a divergence, which is a way of comparing two probability distributions, as an alternative to using the Euclidean distance in the parameter space. Shannon and Weaver [40] viewed a probability distribution as an information source. They studied the concept of entropy, which Shannon referred to as the amount of freedom of choice a source of information has. If the result of a random variable is often the same value, then the entropy of that probability distribution is low, and if the random variable is evenly distributed, then the entropy is as high as it can be. Formally, the entropy H of a

discrete distribution $P(x)$ with $x \in \mathcal{X} = \{X_1, ..., X_n\}$ is written as:

$$H(P) = -\sum_{x \in \mathcal{X}} P(x) \log P(x).$$

If $P(x)$ is over a continuous random variable ($\mathcal{X} = \mathbb{R}$), then the entropy (sometimes referred to as differential entropy) is given by:

$$H(P) = -\int_{\mathcal{X}} P(x) \log P(x) dx.$$

It is worth noting that in the continuous case, may not be finite, but we will assume finiteness for this paper. We consider the logarithm here to be with base e, but in many applications in computer science where entropy represents the expected number of bits needed to encode a message, the base would be 2.

In practice, one often approximates a true distribution $P(x)$ using an estimated distribution $Q(x)$. Similar to above, we can then compute the entropy of $Q(x)$, but over the data produced by $P(x)$:

$$H(P, Q) = -\int_{\mathcal{X}} P(x) \log Q(x) dx.$$

This is referred to as the cross-entropy between $P(x)$ and $Q(x)$. In the context of machine learning, cross-entropy is often adapted as a loss function, which is used to optimize the performance of a network, when the output of the classifier can be compared to the true desired output (for example, in image classification tasks [20]).

Kullback and Leibler [23, 13], who studied decision theory, introduced an idea closely related to cross-entropy which they called the Kullback-Leibler risk. Currently, in information geometry and machine learning, this is called the KL-divergence.

Definition 3.4. The **KL-divergence** between two distributions $P(x)$ and $Q(x)$ over the same random variable is:

$$KL(P \parallel Q) = -\int_{\mathcal{X}} P(x) \log \left(\frac{Q(x)}{P(x)} \right) dx.$$

One interpretation of the KL-divergence is that it is the difference between the cross-entropy of an approximating distribution and the entropy of the true distribution. We can

see this by a short calculation:

$$\begin{aligned} H(P,Q) - H(P) &= -\int_{\mathcal{X}} P(x) \log Q(x) dx + \int_{\mathcal{X}} P(x) \log P(x) dx \\ &= \int_{\mathcal{X}} P(x) [\log P(x) - \log Q(x)] dx \\ &= \int_{\mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right) dx \\ &= -\int_{\mathcal{X}} P(x) \log \left(\frac{Q(x)}{P(x)} \right) dx \\ &= KL(P \parallel Q). \end{aligned}$$

It is sometimes called the KL-distance, but this is a misnomer, since it does not satisfy all the requirements of being a distance metric. In particular, the KL-divergence is not symmetric and it does not satisfy the triangle inequality.

The KL-divergence is deeply motivated by information theory, but we can define a more general notion of a divergence on probability distributions.

Definition 3.5. Let S be the set of all probability distributions over $\mathcal{X}$. A **divergence** on S is a function $D : S \times S \to \mathbb{R}$ that satisfies for all $P(x), Q(x) \in S$:

1. $D(P \parallel Q) \geq 0$
2. $D(P \parallel Q) = 0$ if and only if $P = Q$

We will see later that the KL-divergence falls under the more general definition of a divergence. The aim of the remainder of this chapter is to show that even when we allow ourselves to use the more general definition of a divergence as above, the KL-divergence will remain the most natural choice of divergence in information geometry, as it lies in the intersection of two important divergence classes.

3.3 f-Divergences

We start with the definition of an f-divergence.

Definition 3.6. An **f-divergence** is a divergence function of the form:

$$D_f(P \parallel Q) = \int_{\mathcal{X}} P(x) f \left(\frac{Q(x)}{P(x)} \right) dx$$

where $f : [0, +\infty) \to \mathbb{R}$ is a strictly convex, differentiable function such that $f(1) = 0$.

Theorem 3.7. *An f-divergence is a divergence.*

Proof. By Jensen's inequality:

$$\begin{aligned} D_f(P \parallel Q) &= \int_{\mathcal{X}} P(x) f\left(\frac{Q(x)}{P(x)}\right) dx \\ &\geq f\left(\int_{\mathcal{X}} P(x) \frac{Q(x)}{P(x)} dx\right) \\ &= f\left(\int_{\mathcal{X}} Q(x) dx\right) \\ &= f(1) \\ &= 0 \end{aligned}$$

Clearly, $D_f(P \parallel P) = 0$, since $f(1) = 0$. To see that equality holds if and only if $P(x) = Q(x)$, requires strict convexity. We refer the reader to [28]. □

Example 3.8. The KL-divergence is an f-divergence, where $f(u) = -\log u$.

Example 3.9. When $f(u) = \frac{1}{2}(u-1)^2$, the derived f-divergence is the Pearson χ^2-divergence:

$$D_f(P \parallel Q) = \frac{1}{2} \int_{\mathcal{X}} \frac{(Q(x) - P(x))^2}{P(x)} dx.$$

This version of the χ^2-divergence is not symmetric, but there is a symmetric version that replaces the denominator with $P(x) + Q(x)$.

Example 3.10. When $f(u) = (\sqrt{u}-1)^2$, the f-divergence is known as the squared Hellinger distance:

$$D_f(P \parallel Q) = \int_{\mathcal{X}} \left(\sqrt{P(x)} - \sqrt{Q(x)}\right)^2 dx.$$

Clearly the above definition is symmetric, however, it does not satisfy the triangle inequality.

The class of f-divergences satisfies a few properties. If we let f be a strictly convex, differentiable function with $f(1) = 0$, then for any $c > 0$, cf is also strictly convex and differentiable. Furthermore:

$$\begin{aligned} D_{cf}(P \parallel Q) &= \int_{\mathcal{X}} P(x) \cdot cf\left(\frac{Q(x)}{P(x)}\right) dx \\ &= c \cdot \int_{\mathcal{X}} P(x) f\left(\frac{Q(x)}{P(x)}\right) dx \\ &= c \cdot D_f(P \parallel Q). \end{aligned}$$

Since f is strictly convex, we can standardize the convexity by requiring that $f''(1) = 1$. Furthermore, if $g(u) = f(u) - c(u-1)$, then g is also convex and differentiable, with $g(1) = 0$ and $g'(1) = f'(1) - c$. Then we note that:

$$\begin{aligned} D_g(P \parallel Q) &= \int_{\mathcal{X}} P(x) \left[f\left(\frac{Q(x)}{P(x)}\right) - c\left(\frac{Q(x)}{P(x)} - 1\right)\right] dx \\ &= \int_{\mathcal{X}} P(x) f\left(\frac{Q(x)}{P(x)}\right) dx - c\left(\int_{\mathcal{X}} (Q(x) - P(x))dx\right) \\ &= D_f(P \parallel Q) - c\int_{\mathcal{X}} Q(x)dx + c\int_{\mathcal{X}} P(x)dx \\ &= D_f(P \parallel Q) - c + c \\ &= D_f(P \parallel Q). \end{aligned}$$

So without loss of generality, we can also assume $f'(1) = 0$ in addition to $f''(1) = 1$, since adding this linear term does not affect the second derivative.

Definition 3.11. An f-divergence that additionally satisfies $f'(1) = 0$ and $f''(1) = 1$ is called a **standard f-divergence**.

The purpose that f-divergences serve in information geometry is that they are exactly the divergences that satisfy a certain invariance criterion, which we now discuss. Csiszár first studied the concept of information monotonicity in the context of discrete distributions, which is summarized in [14]. Let $\mathcal{X}$ be the domain of a discrete random variable, and $P(x)$ and $Q(x)$ be probability distributions with $x \in \mathcal{X}$. Consider a map $k : \mathcal{X} \to \mathcal{Y}$ which transforms the random variable. Define:

$$\overline{P}(y) := \sum_{x:k(x)=y} P(x).$$

To avoid complications, we will always assume k is surjective. If k is also injective, then $D(P \parallel Q) = D(\overline{P} \parallel \overline{Q})$, since k is simply a relabelling of the random events.

The more interesting case is when k is not injective, in which case information may be lost. First we will see an example where such a map makes it impossible to tell two distributions apart by the KL-divergence even though they were originally distinct.

Example 3.12. Let $\mathcal{X} = \{1, 2, 3, 4, 5, 6\}$ be the outcome of a die roll. Let $k : \mathcal{X} \to \mathcal{Y} = \{0, 1\}$ be such that $k(x) = 0$ is x is even and $k(x) = 1$ when x is odd. Consider the following probability distributions:

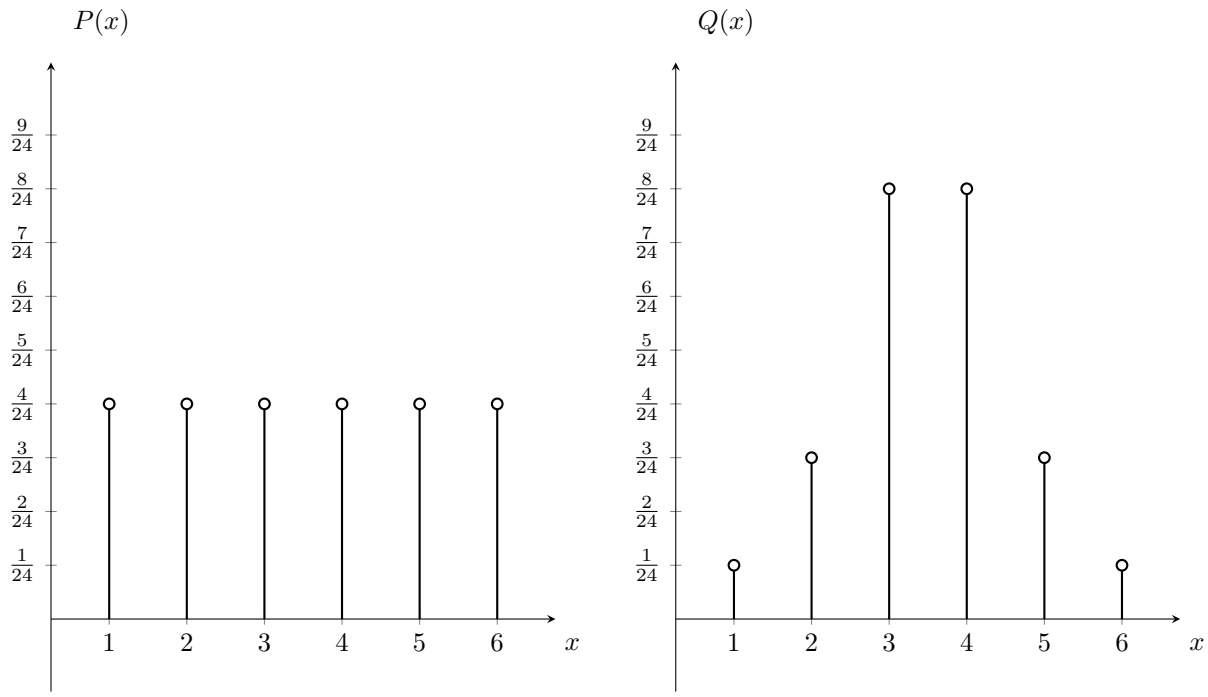

Figure 8: Two distributions over $\mathcal{X} = \{1, 2, 3, 4, 5, 6\}$.

One can compute the KL-divergence between $P(x)$ and $Q(x)$:

$$\begin{aligned} KL(P \parallel Q) &= \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right) \\ &= \frac{1}{6}\left[\log 4 + \log\left(\frac{4}{3}\right) + \log\left(\frac{1}{2}\right) + \log\left(\frac{1}{2}\right) + \log\left(\frac{4}{3}\right) + \log 4\right] \\ &= 0.3252. \end{aligned}$$

However, under the transformation k, the two distributions $\overline{P}(y)$ and $\overline{Q}(y)$ can no longer be told apart.

$$\begin{aligned} KL(\overline{P} \parallel \overline{Q}) &= \sum_{y \in \mathcal{Y}} \overline{P}(y) \log\left(\frac{\overline{P}(y)}{\overline{Q}(y)}\right) \\ &= (P(2) + P(4) + P(6)) \log\left(\frac{P(2) + P(4) + P(6)}{Q(2) + Q(4) + Q(6)}\right) \\ &\quad + (P(1) + P(3) + P(5)) \log\left(\frac{P(1) + P(3) + P(5)}{Q(1) + Q(3) + Q(5)}\right) \\ &= \frac{1}{2}(\log(1) + \log(1)) \\ &= 0. \end{aligned}$$

The purpose of this example is to demonstrate that, at least with the KL-divergence, we expect that $KL(P \parallel Q) \geq KL(\overline{P} \parallel \overline{Q})$. Intuitively, this means that if a transformation of the random variable discards information, then distributions will appear closer together. This turns out to be a property of any f-divergence.

Definition 3.13. Let D be a divergence and $k : \mathcal{X} \to \mathcal{Y}$ be a transformation of a random variable. We say D satisfies **information monotonicity** if for any pair of distributions $P(x)$ and $Q(x)$ over $\mathcal{X}$, we have:

$$D(P \parallel Q) \geq D(\overline{P} \parallel \overline{Q}),$$

where

$$\overline{P}(y) = \sum_{x:k(x)=y} P(x)$$

and

$$\overline{Q}(y) = \sum_{x:k(x)=y} Q(x).$$

Theorem 3.14. *Every f-divergence satisfies information monotonicity.*

Proof. This proof is due to Amari [3, 4]. Let $k : \mathcal{X} \to \mathcal{Y}$, let $P(x)$ and $Q(x)$ be distributions over $\mathcal{X}$, and let $\overline{P}(y)$ and $\overline{Q}(y)$ be the distributions under the transformation k. Suppose D_f is an f-divergence. As previously noted, if k is injective, then $D_f(P \parallel Q) = D_f(\overline{P} \parallel \overline{Q})$.

Suppose k is not injective. That is, there exists distinct $x_0, x_1 \in \mathcal{X}$ and $y \in \mathcal{Y}$ with $k(x_0) = k(x_1) = y$. Then by convexity of f:

$$\begin{aligned}
\overline{P}(y) f\left(\frac{\overline{Q}(y)}{\overline{P}(y)}\right) &= (P(x_0) + P(x_1)) f\left(\frac{Q(x_0) + Q(x_1)}{P(x_0) + P(x_1)}\right) \\
&= (P(x_0) + P(x_1)) f\left(\frac{Q(x_0)}{P(x_0) + P(x_1)} + \frac{Q(x_1)}{P(x_0) + P(x_1)}\right) \\
&= (P(x_0) + P(x_1)) f\left(\frac{P(x_0)}{P(x_0) + P(x_1)} \cdot \frac{Q(x_0)}{P(x_0)} + \frac{P(x_1)}{P(x_0) + P(x_1)} \cdot \frac{Q(x_1)}{P(x_1)}\right) \\
&\leq \frac{(P(x_0) + P(x_1))P(x_0)}{P(x_0) + P(x_1)} f\left(\frac{Q(x_0)}{P(x_0)}\right) + \frac{(P(x_0) + P(x_1))P(x_1)}{P(x_0) + P(x_1)} f\left(\frac{Q(x_1)}{P(x_1)}\right) \\
&= P(x_0) f\left(\frac{Q(x_0)}{P(x_0)}\right) + P(x_1) f\left(\frac{Q(x_1)}{P(x_1)}\right).
\end{aligned}$$

For all other $x \in \mathcal{X}$, we simply have:

$$\overline{P}(k(x)) f\left(\frac{\overline{Q}(k(x))}{\overline{P}(k(x))}\right) = P(x) f\left(\frac{Q(x)}{P(x)}\right).$$

Therefore, we satisfy:

$$D_f(\overline{P} \parallel \overline{Q}) \leq D_f(P \parallel Q).$$

This proves the case when there is a two-to-one mapping. A similar proof works for many-to-one mappings. □

In some cases, equality is achieved even when k is not injective. To look at this, we have a definition.

Definition 3.15. Let $P(x;\theta)$ be a probability distribution with n parameters, $\theta = (\theta_1, ..., \theta_n)$, and random variable $x \in \mathcal{X}$. A transformation of the random variable $k : \mathcal{X} \to \mathcal{Y}$ is called a **sufficient statistic for** θ_i if the conditional probability $P(x;\theta \mid k(x) = y)$ does not depend on θ_i.

Example 3.16. Consider a coin being tossed n times, where the probability of it landing heads is p. Let $\mathcal{X} = \{x = (x_1, ..., x_n) \mid x_i = 0, 1\}$ where 1 indicates heads and 0 indicates tails. The probability of a particular event occurring is:

$$P(x = (x_1, ..., x_n); p) = \prod_i p^{x_i}(1-p)^{(1-x_i)}.$$

For example, if $p = 0.6$, then the probability of seeing $x = (1, 0, 0)$ is:

$$(0.6 \cdot 1)(1 \cdot 0.4)(1 \cdot 0.4) = 0.096.$$

Let $k(x) = \sum_i x_i$. We show that this is a sufficient statistic for p. The probability that we see an event where $k(x) = y$ (that is, we see y heads) is:

$$\binom{n}{y} p^y (1-p)^{(n-y)}.$$

So, the probability that we see a particular event, given that we see y heads is given by:

$$P(x = (x_1, ..., x_n); p \mid k(x) = y) = \frac{p^y(1-p)^{(n-y)}}{\binom{n}{y} p^y (1-p)^{(n-y)}} = \frac{1}{\binom{n}{y}},$$

which is independent of p. So k is a sufficient statistic for p.

One way of thinking about sufficient statistics is in terms of estimating a parameter. If $k(x)$ is sufficient for θ, then knowing $k(x)$ gives the same amount of information about θ as knowing x. To use the example above, if we had a unfair coin and wanted to guess the probability p of flipping heads, then having a list of n trials is as helpful as knowing the number of heads in n trials.

We now relate this back to f-divergences. We give two more definitions involving divergences.

Definition 3.17. A divergence D over a parametric family of probability distributions is **invariant** if it satisfies information monotonicity and equality holds if and only if k is a sufficient statistic for each parameter.

Definition 3.18. A divergence D is **decomposable** if it is of the form:

$$D(P \parallel Q) = \sum_{x \in \mathcal{X}} d(P(x), Q(x)),$$

for some function $d : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$.

We give the following theorem, without proof. This theorem appears in [4] and [3].

Theorem 3.19. *An f-divergence is invariant and decomposable. Furthermore, an invariant, decomposable divergence is an f-divergence, unless it is over $P(x)$ for $x \in \mathcal{X}$ where $|\mathcal{X}| = 1$.*

Therefore, except in the case $|\mathcal{X}| = 1$, f-divergences are exactly the class of invariant, decomposable divergences.

3.4 Fisher Information as a Riemannian Metric

In this section we show that a Riemannian metric can be derived from a divergence. By doing so, we can interpret a parameter space of probability distributions as a Riemannian manifold. For an explicit example, we show the entire derivation for the KL-divergence, where the metric obtained is the Fisher information matrix. However, due to a theorem by Chentsov [11, 16], every f-divergence gives the Fisher information matrix up to a scalar factor, making it the only metric invariant under sufficient statistics. In this section, we use ∂_i to denote $\frac{\partial}{\partial \theta^i}$ where $\theta = (\theta^1, ..., \theta^n)^T$ is a vector of parameters.

Definition 3.20. Given a parametric family of probability distributions, for a particular value of the parameter $\theta = (\theta^1, ..., \theta^n)^T$, the Fisher information matrix of $P(x; \theta)$ is (g_{ij}) where:

$$g_{ij} = \int_{\mathcal{X}} P(x; \theta)(\partial_i \log P(x; \theta))(\partial_j \log P(x; \theta))dx.$$

We will denote this matrix by $G(\theta)$.

Theorem 3.21. *The Fisher information matrix $G(\theta)$ is always symmetric and positive semidefinite.*

Proof. It is clear from the definition that $g_{ij} = g_{ji}$. Now let $\xi = (\xi_1, ..., \xi_n) \in \mathbb{R}^n$. Then:

$$\begin{aligned}
\xi^T G(\theta)\xi &= \sum_{i,j} \xi_i g_{ij} \xi_j \\
&= \sum_{i,j} \xi_i \left[\int_{\mathcal{X}} P(x;\theta)(\partial_i \log P(x;\theta))(\partial_j \log P(x;\theta))dx \right] \xi_j \\
&= \sum_{i,j} \int_{\mathcal{X}} P(x;\theta)(\xi_i \partial_i \log P(x;\theta))(\xi_j \partial_j \log P(x;\theta))dx \\
&= \int_{\mathcal{X}} \sum_{i,j} P(x;\theta)(\xi_i \partial_i \log P(x;\theta))(\xi_j \partial_j \log P(x;\theta))dx \\
&= \int_{\mathcal{X}} P(x;\theta) \left(\sum_i \xi_i \partial_i \log P(x;\theta) \right) \left(\sum_j \xi_j \partial_j \log P(x;\theta) \right) dx \\
&= \int_{\mathcal{X}} P(x;\theta) \left(\sum_i \xi_i \partial_i \log P(x;\theta) \right)^2 dx \\
&\geq 0.
\end{aligned}$$

So $G(\theta)$ is positive semidefinite. □

In order for the Fisher information matrix to be a true Riemannian metric, it must be positive definite. We can see from the previous theorem that this may fail, for example, if the support of $P(x;\theta)$ is strictly contained in the domain. That is, if there exist random events that have a probability of zero. There are many papers in information geometry that study manifolds where the requirements of the Riemannian metric are weakened [44, 46, 29], for example, not requiring the metric to positive definite. On the other hand, many standard distribution families have a positive definite Fisher information matrix, so for the remainder of this paper, we will assume positive definiteness.

Now we show the relationship between the Fisher information matrix and the KL-divergence. The following theorem depends on $P(x;\theta)$ meeting certain regularity conditions which guarantee that:

$$\frac{\partial}{\partial \theta^i} \int_{\mathcal{X}} P(x;\theta) = \int_{\mathcal{X}} \frac{\partial}{\partial \theta^i} P(x;\theta).$$

An overview of these conditions is given in [26]. We assume this holds in the proof of the theorem.

Theorem 3.22. *Let $P_\theta = P(x;\theta)$ and $P_{\theta+\delta\theta} = P(x;\theta+\delta\theta)$. The second order Taylor expansion of $KL(P_\theta \parallel P_{\theta+\delta\theta})$ yields the quadratic form $\frac{1}{2}\delta\theta^T G(\theta)\delta\theta$, where $G(\theta)$ is the Fisher information matrix.*

Proof. The Taylor expansion gives:

$$\begin{aligned}
KL(P_\theta \parallel P_{\theta+\delta\theta}) &= \int_{\mathcal{X}} P_\theta \log\left(\frac{P_\theta}{P_{\theta+\delta\theta}}\right) dx \\
&= \int_{\mathcal{X}} P_\theta \log P_\theta dx - \int_{\mathcal{X}} P_\theta \log P_{\theta+\delta\theta} dx \\
&\approx \int_{\mathcal{X}} P_\theta \log P_\theta dx - \int_{\mathcal{X}} P_\theta \left(\log P_\theta + \left(\frac{\nabla P_\theta}{P_\theta}\right)^T \delta\theta + \frac{1}{2}\delta\theta^T(\nabla^2 \log P_\theta)\delta\theta\right) dx \\
&= \left(-\int_{\mathcal{X}} \nabla P_\theta dx\right)^T \delta\theta - \frac{1}{2}\delta\theta^T \left(\int_{\mathcal{X}} P_\theta(\nabla^2 \log P_\theta) dx\right) \delta\theta \\
&= \left(-\nabla \int_{\mathcal{X}} P_\theta dx\right)^T \delta\theta - \frac{1}{2}\delta\theta^T \left(\int_{\mathcal{X}} P_\theta(\nabla^2 \log P_\theta) dx\right) \delta\theta \\
&= (-\nabla 1)^T \delta\theta - \frac{1}{2}\delta\theta^T \left(\int_{\mathcal{X}} P_\theta(\nabla^2 \log P_\theta) dx\right) \delta\theta \\
&= -\frac{1}{2}\delta\theta^T \left(\int_{\mathcal{X}} P_\theta(\nabla^2 \log P_\theta) dx\right) \delta\theta.
\end{aligned}$$

Then we can compute the Hessian of $\log P_\theta$ by applying the quotient rule:

$$\begin{aligned}
\partial_i \partial_j \log P_\theta &= \partial_i \left(\frac{\partial_j P_\theta}{P_\theta}\right) \\
&= \frac{(\partial_i \partial_j P_\theta) P_\theta - (\partial_i P_\theta)(\partial_j P_\theta)}{P_\theta P_\theta} \\
&= \frac{\partial_i \partial_j P_\theta}{P_\theta} - (\partial_i \log P_\theta)(\partial_j \log P_\theta).
\end{aligned}$$

In matrix form, this is:

$$\nabla^2 \log P_\theta = \frac{\nabla^2 P_\theta}{P_\theta} - (\nabla \log P_\theta)(\nabla \log P_\theta)^T.$$

Therefore:

$$\begin{aligned}
-\frac{1}{2}\delta\theta^T\left(\int_{\mathcal{X}} P_\theta(\nabla^2 \log P_\theta)dx\right)\delta\theta &= -\frac{1}{2}\delta\theta^T\left(\int_{\mathcal{X}} \nabla^2 P_\theta - P_\theta(\nabla \log P_\theta)(\nabla \log P_\theta)^T dx\right)\delta\theta \\
&= -\frac{1}{2}\delta\theta^T\left(\int_{\mathcal{X}} \nabla^2 P_\theta dx - \int_{\mathcal{X}} P_\theta(\nabla \log P_\theta)(\nabla \log P_\theta)^T dx\right)\delta\theta \\
&= -\frac{1}{2}\delta\theta^T\left(\nabla^2 \int_{\mathcal{X}} P_\theta dx - \int_{\mathcal{X}} P_\theta(\nabla \log P_\theta)(\nabla \log P_\theta)^T dx\right)\delta\theta \\
&= -\frac{1}{2}\delta\theta^T\left(\nabla^2 1 - \int_{\mathcal{X}} P_\theta(\nabla \log P_\theta)(\nabla \log P_\theta)^T dx\right)\delta\theta \\
&= \frac{1}{2}\delta\theta^T\left(\int_{\mathcal{X}} P_\theta(\nabla \log P_\theta)(\nabla \log P_\theta)^T dx\right)\delta\theta \\
&= \frac{1}{2}\delta\theta^T G(\theta)\delta\theta.
\end{aligned}$$

□

In light of this, we can interpret the space $\{P(x;\theta) \mid \theta \in \mathbb{R}^n\}$ as a Riemannian manifold. This manifold is covered by a single coordinate patch, so we treat the manifold as $\mathbb{R}^n$. Since the points on the manifold are associated with probability distributions, we call the manifold a **statistical manifold**.

By starting with a different divergence function, one could try to obtain a Riemannian manifold with a different metric, but when the divergence is an f-divergence, the metric obtained is always the Fisher information matrix (up to re-scaling). Chentsov first proved this in the framework of category theory.

Theorem 3.23. *(Chentsov, [11]) The Fisher information matrix is the only invariant Riemannian metric up to a scalar factor, in the sense that it is induced by an invariant divergence.*

3.5 Bregman Divergences and Dually Flat Structures

The class of f-divergences allow us to interpret the parameter space of a probability distribution as a Riemannian manifold, where the Riemannian metric has the desirable property that it is invariant under transformations of the random variable. This does not tell us much about the affine connections associated with the manifold. In this section we explore another class of divergences known as Bregman divergences, which are in bijection with so-called dually flat structures.

Definition 3.24. Let $\{P(x;\theta) \mid \theta \in \mathbb{R}^n\}$ be a family of probability distributions. Let

$P_\theta = P(x;\theta)$ and $P_{\theta'} = P(x;\theta')$. A **Bregman divergence** is a divergence of the form:

$$D_\psi(P_\theta \parallel P_{\theta'}) = \psi(\theta) - \psi(\theta') - \nabla\psi(\theta') \cdot (\theta - \theta'),$$

where $\psi : \mathbb{R}^n \to \mathbb{R}$ is a strictly convex, differentiable function.

Theorem 3.25. *A Bregman divergence is a divergence.*

Proof. The supporting hyperplane of ψ at θ' is given by:

$$f(\theta) = \psi(\theta') + \nabla\psi(\theta') \cdot (\theta - \theta').$$

Therefore, the Bregman divergence is the difference between $\psi(\theta)$ and the hyperplane supporting ψ at θ', evaluated at θ. By convexity of ψ, $D_\psi(P_\theta \parallel P_{\theta'}) \geq 0$, and by strictness, $D_\psi(P_\theta \parallel P_{\theta'}) = 0$ if and only if $\theta = \theta'$. □

However, Bregman divergences are not symmetric in general.

Example 3.26. Let θ be a single parameter ($\theta \in \mathbb{R}$) and $\psi(\theta) = e^\theta$. Then $\psi'(\theta) = e^\theta$. If we consider $\theta = 0$ and $\theta' = 1$, we have:

$$D_\psi(P_\theta \parallel P_{\theta'}) = \psi(0) - \psi(1) - \psi'(1)(0 - 1) = 1,$$

and:

$$D_\psi(P_{\theta'} \parallel P_\theta) = \psi(1) - \psi(0) - \psi'(0)(1 - 0) = e - 2.$$

Therefore, the Bregman divergence given by ψ is not symmetric.

Since the function $\psi(\theta)$ is strictly convex, its Hessian is positive semi-definite. Positive definiteness is not guaranteed, since for example, if $\theta \in \mathbb{R}$, then $\psi(\theta) = \theta^4$ is strictly convex, but $\psi''(0) = 0$. If the Hessian is positive definite, a Bregman divergence induces a Riemannian metric, where g_{ij} at the point θ is given by:

$$g_\theta(\partial_i, \partial_j) = \frac{\partial^2}{\partial\theta_i \partial\theta_j}\bigg|_{\theta'=\theta} D_\psi(P_\theta \parallel P_{\theta'}).$$

Furthermore, a Bregman divergence induces a pair of flat, dual affine connections [2], where the Christoffel symbols are given by:

$$\Gamma^k_{ij}(\theta) = -\frac{\partial^3}{\partial\theta_i \partial\theta_j \partial\theta'_k}\bigg|_{\theta'=\theta} D_\psi(P_\theta \parallel P_{\theta'})$$
$$(\Gamma^*)^k_{ij}(\theta) = -\frac{\partial^3}{\partial\theta_i \partial\theta_j \partial\theta'_k}\bigg|_{\theta'=\theta} D^*_\psi(P_\theta \parallel P_{\theta'}),$$

where

$$D^*_\psi(P_\theta \parallel P_{\theta'}) = D_\psi(P_{\theta'} \parallel P_\theta)$$

The notation D^* is called the **dual divergence**.

Now suppose we are given a statistical manifold (M, g), with a flat affine connection ∇. By Proposition 2.1 in [29], the dual affine connection ∇^* must also be flat. The quadruple (M, g, ∇, ∇^*) is called a **dually flat structure**. We have the following remarkable theorem.

Theorem 3.27. *(Proposition 3.1 [29]) Let $\theta = (\theta^1, ..., \theta^n)$ be coordinates with respect to a parallel frame (that is, the Christoffel symbols of ∇ vanish) and let $\eta = (\eta_1, ..., \eta_n)$ be the dual coordinates for which the Christoffel symbols of the dual affine connection ∇^* vanish. Then there exist functions ψ and ϕ on M such that:*

$$\frac{\partial \psi}{\partial \theta^i} = \eta_i, \quad \frac{\partial \phi}{\partial \eta_i} = \theta^i,$$

and:

$$\psi(P) + \phi(P) - \sum_i \theta^i(P)\eta_i(P) = 0,$$

for all $P \in M$.

From this pair of functions, one can obtain what is called the **canonical divergence**, with respect to the dually flat structure (M, g, ∇, ∇^*). For $P, Q \in M$, it is defined by:

$$D(P \parallel Q) = \psi(P) + \phi(Q) - \sum_i \theta^i(P)\eta_i(Q).$$

Theorem 3.28. *Let $\theta = (\theta^1, ..., \theta^n)$ and $\eta = (\eta_1, ..., \eta_n)$ be the dual coordinates associated with the dual affine connections of a dually flat structure (M, g, ∇, ∇^*). Let ψ and ϕ be given by the previous theorem. The canonical divergence of (M, g, ∇, ∇^*) is a Bregman divergence, associated with the strictly convex function ψ.*

Proof. Let $P, Q \in M$. By the previous theorem:

$$\psi(Q) + \phi(Q) - \sum_i \theta^i(Q)\eta_i(Q) = 0.$$

Rearranging this, we get:

$$-\psi(Q) = \phi(Q) - \sum_i \theta^i(Q)\eta_i(Q).$$

If we substitute this into the expression for the Bregman divergence given by ψ, we see:

$$\begin{aligned} D_\psi(P \parallel Q) &= \psi(P) - \psi(Q) - \nabla\psi(Q) \cdot (\theta(P) - \theta(Q)) \\ &= \psi(P) + \phi(Q) - \left(\sum_i \theta^i(Q)\eta_i(Q)\right) - \left(\sum_i \frac{\partial\psi}{\partial\theta_i}(Q)(\theta^i(P) - \theta^i(Q))\right) \\ &= \psi(P) + \phi(Q) - \left(\sum_i \theta^i(Q)\eta_i(Q)\right) - \left(\sum_i \eta_i(Q)(\theta^i(P) - \theta^i(Q))\right) \\ &= \psi(P) + \phi(Q) - \sum_i \theta^i(P)\eta_i(Q). \end{aligned}$$

So the canonical divergence of (M, g, ∇, ∇^*) is the Bregman divergence $D_\psi(P \parallel Q)$. □

Therefore, there is a one-to-one correspondence between Bregman divergences and dually flat structures. This duality has been studied extensively in information geometry [29, 41, 4]. Finally, we note a key result about the KL-divergence.

Theorem 3.29. *(Amari, [3]) The KL-divergence and its dual are the only divergences that lie in the intersection of the f-divergence and Bregman divergence classes.*

The main goal of this chapter was to motivate the use of the KL-divergence and the Fisher information matrix to induce a Riemannian structure on parameter spaces of probability distributions. In the next chapter, we show how information geometry can be applied to machine learning.

4 Applications to Machine Learning

4.1 Background on Neural Networks

In this chapter we see more specific applications of Riemannian geometry and information geometry in machine learning. The online books [18] and [30] are excellent resources on introductory machine learning topics. In this section, we introduce the concept of a neural network.

In general, an artificial neural network (ANN) is a function with a large number of parameters, mapping input vectors to output vectors. Their use has been studied extensively in the areas of regression, computer vision, and speech processing [8, 18].

The building block of a neural network is the **neuron**, which is loosely related to a biological neuron. It consists of several components:

- **Input**: $x = (x_1, ..., x_n) \in \mathbb{R}^n$
- **Weight**: $w = (w_1, ..., w_n) \in \mathbb{R}^n$
- **Bias**: $b \in \mathbb{R}$
- **Activation function**: $f : \mathbb{R} \to \mathbb{R}$

A neuron maps the input vector x to a real-valued output $y = f(w \cdot x + b)$. This is typically visualized as:

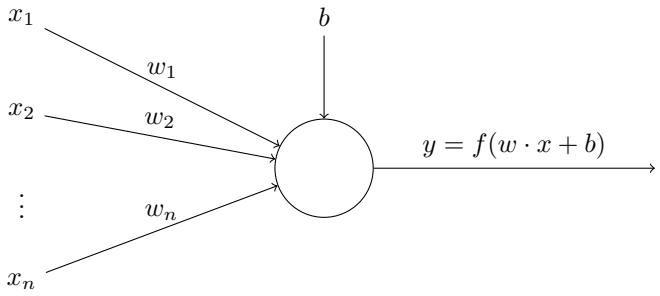

Figure 9: A single neuron.

Intuitively, one can think of the input x as a stimulus. If the input to the neural network is an image, then each x_i might be the intensity of a particular pixel. The vector w amplifies, dampens, or negates specific inputs, based on their importance to this particular neuron. The activation determines how the neuron "fires", in relation to its weighted input. Some typical choices of f are the **step-function**, the **sigmoid function**, and the **rectified linear unit** (ReLU):

$$f(a) = \begin{cases} 0 & a \leq 0 \\ 1 & a > 0 \end{cases} \qquad f(a) = \frac{1}{1 + e^{-a}} \qquad f(a) = \begin{cases} 0 & a \leq 0 \\ a & a > 0 \end{cases}$$

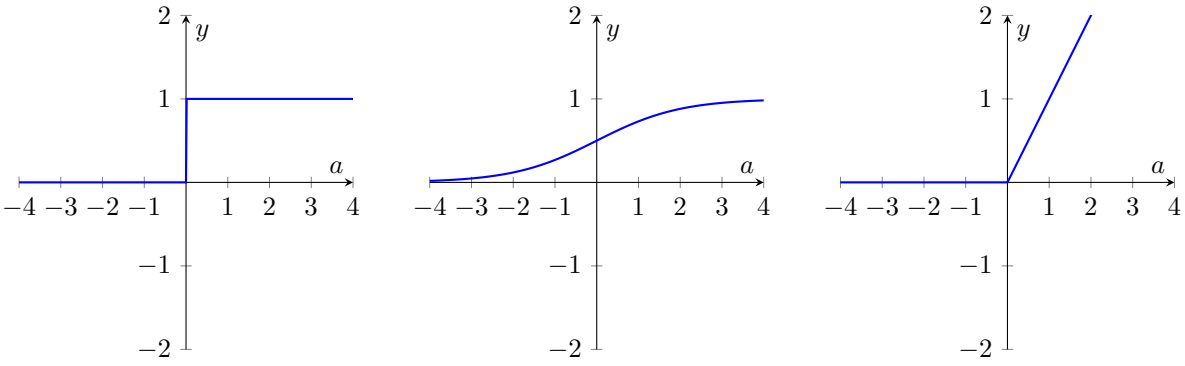

Figure 10: Various activation functions used in neurons.

The bias effectively translates the graphs of these functions horizontally, requiring the summed input to be lower or higher for the neuron to "fire".

The simplest kind of ANN is called a **multilayer perceptron**. It consists of several layers of neurons such that the output of a neuron in one layer is used as input to neurons in the next layer. Below is a figure showing a multilayer perceptron, where each unlabelled circle is a neuron. Every arrow has a weight parameter associated with it and every neuron has its own bias parameter.

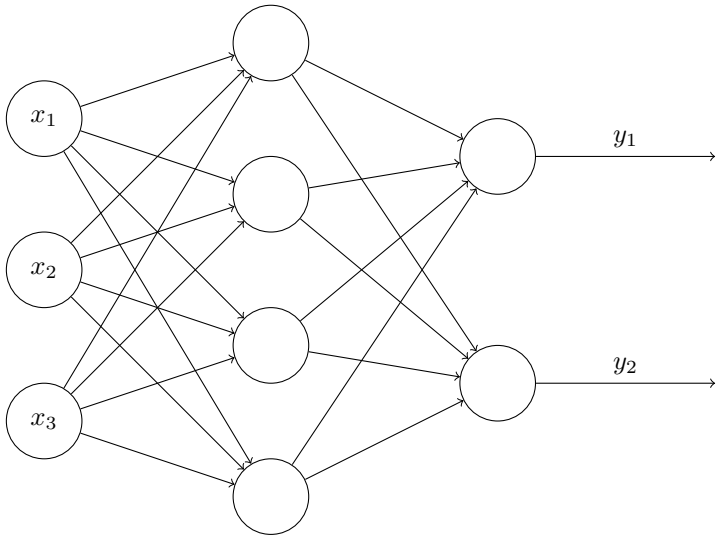

Figure 11: A multilayer perceptron. Each unlabelled node is a neuron, with its own bias parameter, and each arrow has a weight associated with it.

The weights and biases are the parameters that can be adjusted to change how the multilayer perceptron functions. Features such as the number of layers and the number of neurons in each layer are referred to as the **architecture** of the ANN, and are usually fixed

before optimizing the parameters.

A large part of machine learning research is studying how to adjust the parameters of an ANN to make it perform well at a certain task. These processes of learning the parameters are generally separated into two major categories: **supervised learning** and **unsupervised learning**. In supervised learning, we use a sample of desired input-output pairs, called a training set. An example of this is an ANN that learns to recognize stop signs in images, by using a dataset of images that have stop signs and a dataset of images that do not. In this case, we define a way to measure the performance of the ANN based on how close its outputs are to the expected outputs in the training set. Unsupervised learning covers techniques that do not use a training set. For example, data compression algorithms could use an ANN, by learning to transform an input x into a lower-dimensional representation such that little to no information about x is lost.

We will focus on supervised learning in this paper. A well-known dataset used to benchmark learning algorithms is the MNIST dataset. It has a training set with 60,000 images of handwritten digits, each 28 pixels in width and 28 pixels in height. It also has a test set of 10,000 images, which are used to evaluate the overall performance of the ANN after the parameters are adjusted, to see how well the network is able to classify images it has never seen before. The dataset was compiled by Yann LeCun, Corinna Cortes, and Christopher Burges, and it was used to train ANNs to recognize handwritten digits in the seminal paper [25].

Now we present an overview of how supervised learning works, using the MNIST dataset to optimize the parameters of an ANN. To begin, we fix the architecture of the network and initialize the parameters. Since the images are 28x28 in size, the input will be a 784-dimensional vector. The output will be a 10-dimensional vector such that the entries are between 0 and 1 and add to 1. We can force this by taking the exponentiation of the output vector component-wise and normalizing it (this is referred to as a **softmax** layer). The ith entry of the vector can be thought of as the probability that the ANN thinks the input is the digit $i-1$. For example, if the input is an image of a 8, the expected output will be $(0,0,0,0,0,0,0,0,1,0)^T$.

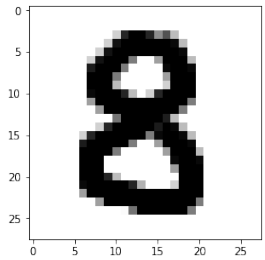

Figure 12: An example of a handwritten digit from the MNIST dataset. To input the image to an ANN, the array of pixels is converted into a vector, where the entries are values between 0 and 1, based on the intensity of the pixel.

We also define a real-valued **loss function**, which measures the overall error of the network's classifications of the training set, as a function of the parameters. The most important feature of a loss function is that the lowest possible value is achieved exactly where the ANN's output matches the expected output. Mean squared error (MSE) is commonly used. We denote the vector containing all the weight and bias parameters by θ, the input by $x = (x_1, ..., x_n)$, the expected output by $\overline{y} = (\overline{y_1}, ..., \overline{y_m})$, and the output of the ANN by $y = (y_1, ..., y_m)$. The MSE associated with the training example $(x, \overline{y})$ is:

$$C(\theta, x, \overline{y}) = \frac{1}{2} \sum_i (y_i - \overline{y_i})^2.$$

The MSE associated with a subset of the training set $S = \{(x^1, \overline{y^1}), ..., (x^k, \overline{y^k})\}$ is the average of the MSEs for each pair:

$$C(\theta) = \frac{\sum_i C(\theta, x^i, \overline{y^i})}{k}.$$

This function satisfies the property that $C(\theta) = 0$ if and only if the ANN produces the expected output on every training example.

To learn the parameters of the ANN, the **gradient descent** algorithm is typically employed to minimize the loss function on the training set. Gradient descent is a first-order optimization algorithm. To use it, we estimate $\nabla C(\theta)$ by estimating the partial derivatives of $C(\theta)$ with respect to each weight and bias parameter. Then we simply update θ to θ' by the rule:

$$\theta' = \theta - \eta \frac{\nabla C(\theta)}{\|\nabla C(\theta)\|_2},$$

where the scalar $\eta > 0$ is referred to as the **learning rate** and $\|\cdot\|_2$ is the Euclidean norm. By Taylor's theorem, $\nabla C(\theta)$ is in the direction of steepest ascent of the linearization of C at θ. By adjusting the parameters by a small negative amount in the direction of the gradient, we expect the error to decrease. The choice of η is important: if it is small, improvements

will be small, but the linearization of C becomes less trustworthy further away from θ. After many iterations, we expect (and we may observe in practice) that the value of the loss function will tend toward a local minimum, since every step aims to reduce the value of the loss function.

The above approach is somewhat effective, but there are several problems with it that have been addressed in machine learning research. The biggest issue is that when the network has a large number of parameters, estimating the partial derivatives can be time-inefficient. The naïve way to estimate one partial derivative is to use a finite difference method, which involves computing the difference between the error of the network and the error when the parameter has been perturbed by a small amount. To estimate all the parameters would require passing inputs through the network at least the same number of times as there are parameters in the network. The **back-propagation algorithm** [37, 15] was conceived as a way to compute $\nabla C(\theta)$ more efficiently, using only one forward pass of the input and one backward pass that determines how much contribution each parameter had in the error.

Another issue is that gradient descent tends toward a local minimum, where ideally we want the global minimum. This is highly dependent on the how the network was initialized. We can modify the gradient descent algorithm to improve this: by adding a momentum term to the update [37] or using an adaptive learning rate [21]. These methods also tend to have faster convergence compared to unmodified gradient descent. In practice, convergence speed is also improved by estimating the cost of the entire training set by only using a fraction of the set per update. The sample of the training data used for an update is called a **mini-batch**. Many studies have shown that performing an update after every example (i.e. using a mini-batch size of 1) is effective [9]. When this technique is used, gradient descent is referred to as **stochastic gradient descent**.

Finally, there is the issue of **overfitting**. We are measuring the performance of the ANN based on how well it can classify data in the training set. However, the real goal is to create an ANN that can classify data well, even if it has never been seen before. Since ANNs have a large number of parameters, they are prone to overfitting: that is, they have low error on training data, but high error on data outside of the training set. A study showed that ANNs with a large number of neurons trained using back-propagation yielded similar results to ANNs with less neurons [10]. A common way of reducing overfitting is to use a **validation set** and **early-stopping** [10]. At the beginning of training, a portion of the training data is set aside as a validation set, which is never used to update the parameters. Periodically during training, the ANNs performance is measured on the validation set. When the error on validation set starts to increase, it indicates that overfitting is occurring and that the

training should stop. Overfitting is also reduced by artificially augmenting the training data with random distortions, and by a technique called **Dropout**, where inputs to individual neurons are randomly zeroed out during the training process [22].

4.2 The Natural Gradient

In the previous section, we saw that the parameters of an ANN are optimized to minimize a loss function $C(\theta)$ using the gradient descent algorithm. When updates are performed after each training example, it is referred to as stochastic gradient descent. Let us only consider computing the loss with respect to a single training input-output example $(x, \overline{y})$.

Another way of thinking of an update in the gradient descent algorithm is that the learning rate implicitly defines an optimization sub-problem in the region around the current parameter θ_0:

$$\begin{aligned} &\min_{\theta} C(\theta) \\ &\text{s.t. } \|\theta - \theta_0\|_2 = \eta \\ &\qquad \theta \in \mathbb{R}^m. \end{aligned}$$

Gradient descent solves a simplified version of this sub-problem, by using a first-order approximation of $C(\theta)$ around θ_0 (by Taylor's theorem). This is the linearized sub-problem:

$$\begin{aligned} &\min_{\theta} \left[C(\theta_0) + \nabla C(\theta_0)^T(\theta - \theta_0)\right] \\ &\text{s.t. } \|\theta - \theta_0\|_2 = \eta \\ &\qquad \theta \in \mathbb{R}^m. \end{aligned}$$

It is clear that the solution to the linearized sub-problem is $\theta = \theta_0 - \eta \frac{\nabla C(\theta_0)}{\|\nabla C(\theta_0)\|_2}$.

However, it is not clear whether or not the constraint $\|\theta - \theta_0\|_2 = \eta$ is the best search space for optimizing the ANN overall. Let us denote the output vector of the ANN with parameters θ by y_θ. As we have seen in previous sections, if y_θ is a probability distribution, then there are more natural ways of measuring the distance between two distributions y_θ and $y_{\theta'}$ than simply using the Euclidean distance of their parameters θ and θ'. Let us consider the parameter space of the ANN as defining a global coordinate system on a Riemannian manifold (M, g), where $M = \{y_\theta \mid \theta \in \mathbb{R}^n\}$ and g is given by the Fisher information matrix $G(\theta)$ at each point. We can rewrite the linearized optimization sub-problem above

to incorporate the Riemannian metric.

$$\begin{aligned} &\min_{\theta} \left[C(\theta_0) + \nabla C(\theta_0)^T(\theta - \theta_0)\right] \\ &\text{s.t. } \|y_\theta - y_{\theta_0}\|_g = \eta \\ &\quad \theta \in \mathbb{R}^m. \end{aligned}$$

To solve this sub-problem, we have the following theorem, which is generally credited to Amari in machine learning literature:

Theorem 4.1 (Amari, [5])**.** *The direction of steepest descent of $C(\theta_0)$ over a Riemannian manifold (M, g) with global coordinates $\theta = (\theta_1, ..., \theta_m)$ is $-G(\theta_0)^{-1}\nabla C(\theta_0)$.*

Proof. Let $x = \theta - \theta_0$ and let $\eta = 1$. The linearized sub-problem over the Riemannian manifold can then be written as:

$$\begin{aligned} &\min_{x} \left[C(\theta_0) + \nabla C(\theta_0)^T x\right] \\ &\text{s.t. } \|y_x\|_g^2 = 1 \\ &\quad x \in \mathbb{R}^m. \end{aligned}$$

The squared norm is to simplify calculations. By the method of Lagrange multipliers, this is equivalent to solving the system for $i = 1, ..., m$:

$$\begin{aligned} 0 &= \frac{\partial}{\partial x_i}\left(C(\theta_0) + \nabla C(\theta_0)^T x - \lambda(x^T G(\theta_0)x - 1)\right) \\ 0 &= \frac{\partial}{\partial x_i} C(\theta_0) + \frac{\partial}{\partial x_i}\nabla C(\theta_0)^T x - \frac{\partial}{\partial x_i}\lambda(x^T G(\theta_0)x - 1). \end{aligned}$$

The first term vanishes. For the second term, we have:

$$\begin{aligned} \frac{\partial}{\partial x_i}\nabla C(\theta_0)^T x &= \frac{\partial}{\partial x_i}\sum_j \nabla C(\theta_0)_j x_j \\ &= \nabla C(\theta_0)_j. \end{aligned}$$

For the third term, we get:

$$\frac{\partial}{\partial x_i}\lambda(x^T G(\theta_0)x - 1) = \lambda\frac{\partial}{\partial x_i}\left(\sum_{i,j} G(\theta_0)_{jk} x_j x_k\right).$$

When neither j nor k is equal to i, the derivative under the sum is 0. When only $j = i$, the derivative is $G(\theta_0)_{ik}x_k$, and $k = i$ gives $G(\theta_0)_{ji}x_j$. When $j = k = i$, we get $2G(\theta_0)_{ii}x_i$.

Therefore, using the fact that $G(\theta_0)_{jk} = G(\theta_0)_{kj}$:

$$\lambda \frac{\partial}{\partial x_i} \left(\sum_{i,j} G(\theta_0)_{jk} x_j x_k \right) = 2\lambda G(\theta_0) x.$$

So the system given by the Lagrange multipliers is equivalent to:

$$0 = \nabla C(\theta_0) - 2\lambda G(\theta_0) x.$$

If $\lambda = 0$, then θ_0 is already a critical point, since $0 = \nabla C(\theta_0)$. Otherwise, solving for x gives:

$$x = \frac{1}{2\lambda} G(\theta_0)^{-1} \nabla C(\theta_0).$$

Since $G(\theta)$ is positive definite, $G(\theta)^{-1}$ is also positive definite. Therefore:

$$\nabla C(\theta_0) \cdot G(\theta_0)^{-1} \nabla C(\theta_0) > 0,$$

so x is the solution that maximizes the objective function, making the direction of steepest descent $-G(\theta_0)^{-1} \nabla C(\theta_0)$. □

The vector $G(\theta_0)^{-1} \nabla C(\theta_0)$ is referred to in machine learning as the **natural gradient**. The above theorem suggests that the update rule used in gradient descent:

$$\theta' = \theta - \eta \frac{\nabla C(\theta)}{\|\nabla C(\theta)\|_2},$$

should be replaced with:

$$\theta' = \theta - \eta \frac{G(\theta)^{-1} \nabla C(\theta)}{\|G(\theta)^{-1} \nabla C(\theta)\|_g}.$$

This method was used by Amari in [6], for the study of blind separation of mixed signals, but the theory of Riemannian metrics in statistical settings already existed well before this (see [1] and [34]). Computing the inverse of the Fisher information matrix can be costly, but there are many ways of approximating it [38]. In a statistical framework, this algorithm was shown to be theoretically more efficient for estimating parameters than stochastic gradient descent [5].

4.3 Batch Normalization

When ANNs have many layers, they are difficult to train. By changing the parameters on neurons in the early layers, the possible inputs that are seen in later layers are also changed.

This effect is sometimes referred to as **internal covariate shift**. This effect is especially a problem when the neurons have activation functions that map $\mathbb{R}$ to a bounded interval (such a function is called a **saturating non-linearity**), as in the case of the sigmoid function [19, 42]. A method known as **batch normalization** was conceived by Ioffe and Szegedy [19] to reduce the effect of internal covariate shift. This stabilizes the learning process, allowing for larger learning rates to be used.

We now give an overview of batch normalization. In order to stabilize the distributions of inputs to a layer, the real-valued inputs to the activation functions can be individually normalized over the training set. In a typical ANN, for a neuron in any given layer, we take the dot product of the input vector x from the previous layer with the weight vector of the neuron w and add a bias scalar b, before using it as input to an activation function f (see Figure 9).

We will focus on a single neuron in the ANN. The exact value of $w \cdot x + b$ will depend on the initial input to the ANN. To normalize this value, we would want to compute the mean and standard deviation of this expression over the entire training set, but doing so is time-consuming. Instead, we estimate the mean and standard deviation by using a mini-batch as a sample. Also, note that the bias parameter can effectively be removed, since normalization cancels the addition of the bias.

Suppose we have a mini-batch of size k. Let $\mathcal{B} = \{x^1, ..., x^k\}$ be the set of input vectors to a particular layer over the mini-batch. We also write $a^i = w \cdot x^i$ to denote the input to a specific neuron for the ith example in the batch. We compute the mean and standard deviation of the a^i by:

$$\mu_{\mathcal{B}} = \frac{1}{k} \sum_{i=1}^{k} a^i$$

$$\sigma_{\mathcal{B}} = \sqrt{\frac{1}{k} \sum_{i=1}^{k} (a^i - \mu_{\mathcal{B}})^2}.$$

Then we can normalize the neuron inputs across the mini-batch by substituting each a^i with:

$$\widehat{a^i} = \frac{a^i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$

where ϵ is a small positive constant for numerical stability. This is used in practice, but for theoretical results, we will ignore it.

However, as it is, this will reduce the layers' ability to transform the input. In order

to restore the representational power of the ANN, we add additional learnable parameters, $\gamma \in \mathbb{R}$ and $\beta \in \mathbb{R}$, which can scale and shift the normalized neuron input $\widehat{a^i}$. We define the batch normalization transformation by the function $BN : \mathbb{R} \to \mathbb{R}$ where:

$$BN(a^i) = \gamma \widehat{a^i} + \beta.$$

To summarize, the output from a neuron in a typical ANN is:

$$f(w \cdot x + b),$$

but batch normalization replaces this with:

$$f(BN(w \cdot x)).$$

The two parameters, γ and β, are added to every neuron to which we wish to apply batch normalization. It should be noted that the effect of batch normalization, given a particular input, depends on the other examples in the mini-batch. Therefore, once the ANN parameters have been tuned and the training set is finished being used, the population mean and standard deviation over the entire training set may be computed, in order for the ANN's output to be deterministic.

An important property of applying batch normalization to a neuron is that the weights to that neuron become invariant to scaling by a positive constant.

Theorem 4.2. *For a fixed mini-batch, denote by $\mathcal{B} = \{x^1, ..., x^n\}$ the input vectors to a particular layer of an ANN. The function BN described above satisfies the property:*

$$BN(w \cdot x^i) = BN((\alpha w) \cdot x^i),$$

for all $\alpha > 0$.

Proof. As above, let:

$$\begin{aligned} a^i &= w \cdot x^i \\ \mu_{\mathcal{B}} &= \frac{1}{k}\sum_{i=1}^{k} a^i \\ \sigma_{\mathcal{B}} &= \sqrt{\frac{1}{k}\sum_{i=1}^{k}(a^i - \mu_{\mathcal{B}})^2} \\ \widehat{a^i} &= \frac{a^i - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}}. \end{aligned}$$

Now we substitute w with αw for some $\alpha > 0$. The formulas above then become:

$$\begin{aligned} (\alpha w) \cdot x^i &= \alpha(w \cdot x^i) \\ &= \alpha a^i \end{aligned}$$

$$\begin{aligned} \frac{1}{k}\sum_{i=1}^{k} \alpha a^i &= \alpha\left(\frac{1}{k}\sum_{i=1}^{k} a^i\right) \\ &= \alpha\mu_{\mathcal{B}} \end{aligned}$$

$$\begin{aligned} \sqrt{\frac{1}{k}\sum_{i=1}^{k}(\alpha a^i - \alpha\mu_{\mathcal{B}})^2} &= \sqrt{\alpha^2\left(\frac{1}{k}\sum_{i=1}^{k}(a^i - \mu_{\mathcal{B}})^2\right)} \\ &= \alpha\sqrt{\frac{1}{k}\sum_{i=1}^{k}(a^i - \mu_{\mathcal{B}})^2} \\ &= \alpha\sigma_{\mathcal{B}} \end{aligned}$$

$$\begin{aligned} \frac{\alpha a^i - \alpha\mu_{\mathcal{B}}}{\alpha\sigma_{\mathcal{B}}} &= \left(\frac{\alpha}{\alpha}\right)\left(\frac{a^i - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}}\right) \\ &= \widehat{a^i}. \end{aligned}$$

Therefore, the normalized input to the neuron is unchanged. □

Information geometry can be applied to understand how much a gradient update will change the output distribution of the ANN. In the case of batch normalization, it becomes even more obvious that distances in the parameter space do not correspond with the mag-

nitude of change in the output, since re-scaling the weights does not change the output at all. We can use the Riemannian metric derived in Section 3.4. If we compute the Fisher information matrix $G(\theta)$ with respect to the neuron model, then a change in the parameter vector by $\delta\theta$ will result in a change in the output distribution by $\frac{1}{2}\delta\theta^T G(\theta)\delta\theta$. For an ANN with multiple neurons and layers, the impact of changing the parameters can be analyzed by using a block-diagonal approximation of the Fisher information matrix, where each block corresponds to the parameters of a single neuron. A detailed overview of this analysis can be found in [7].

4.4 Momentum-based Optimization

A large drawback of gradient descent is that when optimizing for non-convex loss functions, we can get stuck in a poor local minimum. Additionally, there are versions of gradient descent that do not use a normalized gradient in the update rule. That is, for a parameter vector $\theta \in \mathbb{R}^n$, a learning rate $\eta \in \mathbb{R}$, and a loss function $C : \mathbb{R}^n \to \mathbb{R}$, the update rule is:

$$\theta' = \theta - \eta \nabla C(\theta)$$

so that optimization is sped up in areas where C is steep (that is, ∇C is large). We expect improvements to slow near local minima, as the algorithm tends toward a solution, but this also results in slow optimization around other critical points, including saddle points and local maxima, since the gradient is close to 0.

These problems are partly caused by the fact that the direction of the gradient update can change instantaneously. To deal with these challenges, it is useful to incorporate some form of momentum into the gradient update rule, to avoid the optimization from slowing down when the recent changes were large. Intuitively, it is similar to how a ball gains speed rolling down a hill and is able to climb back up over small bumps.

We now outline gradient descent with momentum, which achieves a faster, more stable convergence than the update rule above [36]. It involves keeping track of a momentum vector $v \in \mathbb{R}^n$, and selecting a momentum coefficient $\alpha \in \mathbb{R}$ before training begins. At each step, the momentum vector is updated by:

$$v' = \alpha v + \eta \nabla C(\theta).$$

A typical value for α is 0.9. The vector of parameters is then updated by the rule:

$$\theta' = \theta - v'.$$

Recall that for $\alpha > 0$, $BN((\alpha w) \cdot x) = BN(w \cdot x)$. In this context, the problem of optimizing the weight parameters over $\mathbb{R}^n$ can be rephrased as a constrained optimization problem, where the vector of weights is required to be normalized. However, many constrained optimization problems in $\mathbb{R}^n$ can be thought of as unconstrained optimization problems over a Riemannian manifold embedded in $\mathbb{R}^n$ [32].

Cho and Lee [12] showed that neurons with n weight parameters and batch normalization can be identified with the set of orthonormal vectors in $\mathbb{R}^n$ (the Stiefel manifold $\mathcal{V}(1, n)$), or alternatively, they can (almost) be identified with the set of 1-dimensional subspaces of $\mathbb{R}^n$ (the Grassmannian manifold $\mathcal{G}(1, n)$). The issue with the second identification is that batch normalization is not invariant under negative scalar multiplication:

$$BN((-\alpha w) \cdot x) = -BN(w \cdot x).$$

However, this is generally not a problem, if the gradient updates never allow the weight vector to change by more than an angle of 90^o.

Therefore, optimizing the weights of a neuron with batch normalization can be phrased as an unconstrained problem over $\mathcal{G}(1, n)$:

$$\begin{aligned} &\min C(p) \\ &\text{s.t. } p \in \mathcal{G}(1, n), \end{aligned}$$

where $C : \mathcal{G}(1, n) \to \mathbb{R}$ is the restriction of the original loss function $C(\theta)$ on $\mathbb{R}^n$. There are a few changes that need to be made to gradient descent for it to work over a Riemannian manifold embedded in $\mathbb{R}^n$. First, if the gradient of the loss function $C(p)$ is computed with respect to the coordinates θ in $\mathbb{R}^n$, it needs to be projected onto the tangent space at the point p. The gradient update rule also has to be changed, since we would be subtracting a vector from a point on the manifold. Instead, we follow the geodesic given by the projected gradient vector in the tangent space. Finally, if we were to use gradient descent with momentum, since the momentum vector would lie in the tangent space at p, we can use parallel translation to carry the momentum vector to the new point on $\mathcal{G}(1, n)$.

We will now summarize gradient descent with momentum on $\mathcal{G}(1, n)$, embedded in $\mathbb{R}^n$. We denote by θ_p the coordinates in $\mathbb{R}^n$ of a point $p \in \mathcal{G}(1, n)$. We also use $\nabla C(\theta)$ to denote the gradient of C computed with respect the coordinates in $\mathbb{R}^n$. The gradient can be thought of as lying in $T_p\mathbb{R}^n$, so we denote the projection from $T_p\mathbb{R}^n$ to $T_p\mathcal{G}(1, n)$ by π. Given:

- $p \in \mathcal{G}(1, n)$ the current point

- $v \in T_p\mathcal{G}(1,n)$ the current momentum vector
- $\alpha \in \mathbb{R}$ the momentum coefficient
- $\eta \in \mathbb{R}$ the learning rate

Then the update rule is as follows:

- $h = \pi(\nabla C(\theta))$ project the gradient onto $\mathcal{G}(1,n)$
- $d = \alpha v - \eta h$ compute the direction of the update
- $p' = \exp_p(d)$ use the exponential map to update the current point
- $v' = P_{pp'}(d)$ use parallel translation to update the current momentum

In practice, the norm of h can be forced to stay below some threshold, since we do not want to allow the exponential map to change the angle of the weight vector by more than 90^o. Furthermore, the parameters of an ANN with m neurons can be interpreted as lying on the manifold:

$$\mathcal{G}(1,n_1) \times \cdots \times \mathcal{G}(1,n_m) \times \mathbb{R}^l.$$

The $\mathbb{R}^l$ component accounts for additional parameters, such as the parameters introduced by batch normalization. Cho and Lee [12] showed that stochastic gradient descent with momentum over this Riemannian manifold was able to outperform momentum-based methods in Euclidean space on the standard ANN architectures at the time.

5 Conclusion

Despite the large amount of theory that has been developed in information geometry, second-order methods, such as the natural gradient and Newton's method, are not widely used in machine learning. The machine learning community is moving toward using larger, deeper networks, and using rapid optimization methods. There is evidence to show that the number of gradient updates required to train networks using second-order methods is less than stochastic gradient descent [17]. However, the gains made by these techniques are outweighed by their computational cost [33]. There exists promising work, showing that the Fisher information matrix can be approximated using a low-rank, block-diagonal matrix [35], but in order for Fisher information and the natural gradient to be feasible in machine learning, further work must be done towards approximating these efficiently.

References

[1] Shun-ichi Amari. *Differential-Geometrical Methods in Statistics*. Springer-Verlag, 1985.

[2] Shun-ichi Amari. "Divergence Function, Information Monotonicity and Information Geometry". *Proceedings of the 2nd Workshop on Information Theoretic Methods in Science and Engineering* (2009).

[3] Shun-ichi Amari. "α-Divergence is Unique, Belonging to Both f-Divergence and Bregman Divergence Classes". *IEEE Transactions on Information Theory* 55.11 (2009), pp. 4925–4931. ISSN: 0018-9448. DOI: 10.1109/TIT.2009.2030485.

[4] Shun-ichi Amari. *Information Geometry and Its Applications*. Springer, 2016.

[5] Shun-ichi Amari. "Natural Gradient Works Efficiently in Learning". *Neural Comput.* 10.2 (Feb. 1998), pp. 251–276. ISSN: 0899-7667. DOI: 10.1162/089976698300017746. URL: http://dx.doi.org/10.1162/089976698300017746.

[6] Shun-ichi Amari, A. Cichocki, and H. H. Yang. "A New Learning Algorithm for Blind Signal Separation". *Proceedings of the 8th International Conference on Neural Information Processing Systems* (1995), pp. 757–763. URL: http://dl.acm.org/citation.cfm?id=2998828.2998935.

[7] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. 2016. eprint: arXiv:1607.06450.

[8] Christopher M Bishop. *Pattern recognition and machine learning*. Information science and statistics. Softcover published in 2016. New York, NY: Springer, 2006. URL: http://cds.cern.ch/record/998831.

[9] Léon Bottou. "Large-scale machine learning with stochastic gradient descent". *Proceedings of the 19th International Conference on Computational Statistics*. 2010, pp. 177–186.

[10] Rich Caruana, Steve Lawrence, and C. Lee Giles. "Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping". *Advances in Neural Information Processing Systems 13*. Ed. by T. K. Leen, T. G. Dietterich, and V. Tresp. MIT Press, 2001, pp. 402–408. URL: http://papers.nips.cc/paper/1895-overfitting-in-neural-nets-backpropagation-conjugate-gradient-and-early-stopping.pdf.

[11] N. N. Chentsov. *Statistical Decision Rules and Optimal Inference*. English translation (1984), Providence. Nauka, Moscow, 1972.

[12] Minhyung Cho and Jaehyung Lee. *Riemannian approach to batch normalization*. 2017. eprint: arXiv:1709.09603.

[13] Daniel Commenges. *Information Theory and Statistics: an overview*. 2015. eprint: arXiv:1511.00860.

[14] Imre Csiszár. "Axiomatic Characterizations of Information Measures". *Entropy* 10 (2008), pp. 261–273.

[15] Yann Le Cun. *A Theoretical Framework for Back-Propagation*. 1988.

[16] James G. Dowty. *Chentsov's theorem for exponential families*. 2017. eprint: arXiv:1701.08895.

[17] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks". *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*. 2010.

[18] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[19] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. eprint: arXiv:1502.03167.

[20] Katarzyna Janocha and Wojciech Marian Czarnecki. *On Loss Functions for Deep Neural Networks in Classification*. 2017. eprint: arXiv:1702.05659.

[21] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2015. URL: http://arxiv.org/abs/1412.6980.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., 2012, pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

[23] S. Kullback. *Information Theory*. Wiley, 1959.

[24] S. Kullback and R. A. Leibler. "On Information and Sufficiency". *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.

[25] Yann Lecun et al. "Gradient-based learning applied to document recognition". *Proceedings of the IEEE*. 1998, pp. 2278–2324.

[26] Pierre L'Ecuyer. "On the Interchange of Derivative and Expectation for Likelihood Ratio Derivative Estimators". *Management Science* 41.4 (1995), pp. 738–748.

[27] John M. Lee. *Riemannian Manifolds: an Introduction to Curvature*. Springer, 1997.

[28] Sabir Hussain Matloob Anwar and J. Pecarić. "Some Inequalities for Csiszár-Divergence Measures". *International Journal of Mathematical Analysis* 3.26 (2009), pp. 1295–1304.

[29] Hiroshi Matsuzoe. "Statistical manifolds and affine differential geometry". *Probabilistic Approach to Geometry*. Tokyo, Japan: Mathematical Society of Japan, 2010, pp. 303–321. DOI: 10.2969/aspm/05710303.

[30] Michael A. Nielsen. *Neural Networks and Deep Learning*. neuralnetworksanddeeplearning.com. Determination Press, 2015.

[31] N. P. Ong. *A Simple Discussion of the Berry Phase*. http://physics.princeton.edu/~npo/SurveyTopics/Berryphase.html. Accessed: 2019-08-24.

[32] Rodolphe Sepulchre P-A Absil Robert Mahony. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2009.

[33] Tapani Raiko, Harri Valpola, and Yann Lecun. "Deep Learning Made Easier by Linear Transformations in Perceptrons". *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Neil D. Lawrence and Mark Girolami. Vol. 22. Proceedings of Machine Learning Research. PMLR, 2012, pp. 924–932.

[34] C. R. Rao. "Information and the Accuracy Attainable in the Estimation of Statistical Parameters". *Bulletin of the Calcutta Mathematical Society* 37 (1945), pp. 81–91.

[35] Nicolas L. Roux, Pierre-antoine Manzagol, and Yoshua Bengio. "Topmoumoute Online Natural Gradient Algorithm". *Advances in Neural Information Processing Systems 20*. Ed. by J. C. Platt et al. Curran Associates, Inc., 2008, pp. 849–856.

[36] Sebastian Ruder. *An overview of gradient descent optimization algorithms*. 2016. eprint: arXiv:1609.04747.

[37] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Neurocomputing: Foundations of Research". Ed. by James A. Anderson and Edward Rosenfeld. Cambridge, MA, USA: MIT Press, 1988. Chap. Learning Representations by Back-propagating Errors, pp. 696–699. ISBN: 0-262-01097-6. URL: http://dl.acm.org/citation.cfm?id=65669.104451.

[38] Tim Salimans and Diederik P. Kingma. "Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks". *Proceedings of the 30th International Conference on Neural Information Processing Systems* (2016), pp. 901–909.

[39] Claude Shannon. "A Mathematical Theory of Communication". *The Bell System Technical Journal* 27.3 (1948), pp. 379–423. URL: https://ieeexplore.ieee.org/document/6773024/.

[40] Claude E. Shannon and Warren Weaver. *The Mathematical Theory of Communication*. The University of Illinois Press, 1964.

[41] Hirohiko Shima. "Geometry of Hessian Structures". *Geometric Science of Information*. Ed. by Frank Nielsen and Frédéric Barbaresco. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 37–55. ISBN: 978-3-642-40020-9.

[42] Hidetoshi Shimodaira. "Improving predictive inference under covariate shift by weighting the log-likelihood function". *Journal of Statistical Planning and Inference* 90.2 (2000), pp. 227–244.

[43] Stephen M. Stigler. "The Epic Story of Maximum Likelihood". *Statist. Sci.* 22.4 (Nov. 2007), pp. 598–620. DOI: 10.1214/07-STS249. URL: https://doi.org/10.1214/07-STS249.

[44] Ke Sun and Stéphane Marchand-Maillet. "An Information Geometry of Statistical Manifold Learning". *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 2014, pp. 1–9.

[45] Loring W. Tu. *Differential Geometry: Connections, Curvature, and Characteristic Classes*. Springer, 2017.

[46] Ting-Kam Leonard Wong and Jiaowen Yang. *Optimal transport and information geometry*. 2019. eprint: arXiv:1906.00030.