Properties of the BFGS method on Riemannian manifolds

M. Seibert
Technische Universität München
München, Germany
m.seibert@tum.de

M. Kleinsteuber
Technische Universität München
München, Germany
kleinsteuber@tum.de

K. Hüper
Julius-Maximilians-Universität
Würzburg, Germany
hueper@mathematik.uni-wuerzburg.de

Abstract. We discuss the BFGS method on Riemannian manifolds and put a special focus on the degrees of freedom which are offered by this generalization. Furthermore, we give an analysis of the BFGS method on Riemannian manifolds that are isometric to $\mathbb{R}^n$.

1 Introduction

Optimization problems can be found in a variety of forms, and there are countless different optimization algorithms that attempt to solve these problems. Newton's method represents one of the most famous of these algorithms, though it poses some computational bottlenecks. Quasi-Newton methods are variations developed to avoid these intricacies. The most successful of them turned out to be the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method. It has many favorable properties and is thereforen often in the main focus. The classical, unconstrained BFGS method on Euclidean spaces has been discussed extensively, see for instance the monographs [12, 14].

However, Euclidean spaces are not the only spaces in which optimization algorithms are employed. There are many applications of optimization on Riemannian manifolds, especially in the field of engineering. As long as the considered manifolds are embedded in $\mathbb{R}^n$, the powerful tools of constrained optimization, that are, for example, examined in [14], can be applied. Still, in many cases such an embedding is *a priori* not at hand, and optimization methods explicitly designed for Riemannian manifolds have to be utilized. These concepts were to the authors' knowledge first introduced by Gabay in [7, 8] where he developed a steepest decent, a Newton, and a quasi-Newton algorithm. Some of these were further expanded by Udrişte in [17] where he discussed steepest descent and generalized Newton methods. In [6] the authors developed conjugate gradient and Newton algorithms for Stiefel and Graßmann manifolds. The common denominator of these approaches is that instead of conducting a linear step during the line search procedure, they define the step along a geodesic via the use of the exponential mapping. An alternative approach was presented in [2] using the concept of retractions, oftentimes a computationally cheaper way of mapping a tangent vector onto the manifold. It was used to formulate a Newton's method that maintains the convergence properties while being significantly cheaper to compute. This idea was picked up again in [1] and expanded with the notion of vector transport which no longer restricts us to the use of parallel transport to connect different tangent spaces. The purpose of vector transport is similar to that of retractions, i.e., less computational cost while maintaining the convergence properties. The authors used the concepts of retraction and vector transport to develop several optimization algorithms, namely a Newton's and a conjugate gradient descent algorithm as well as a quasi-Newton algorithm. This quasi-Newton algorithm was discussed further in [15]. In [3] a similar quasi-Newton algorithm, defined to work on a Graßmann manifold, was presented which also applies the notion of vector transport. In their recent work [16] Ring and Wirth proved the superlinear convergence of BFGS methods on Riemannian manifolds, however, under strong conditions to the manifold structure.

In all the works mentioned above, the implementation of the quasi-Newton algorithm on Riemannian manifolds is based on the method Gabay first introduced. However, the process of defining the algorithm on manifolds already offers a whole bunch of degrees of freedom. In this paper we will propose several different approaches to define the algorithm by diversifying the application of the vector transport. The other focus of this paper is to proof that the BFGS method on Riemannian manifolds that are isometric to $\mathbb{R}^n$ are equivalent to a classical BFGS method on $\mathbb{R}^n$.

2 Notation and basic concepts

In the following we introduce several important concepts, which will come of use later on. First, we recall the algorithm for the classical local BFGS method. Then we introduce some differential geometric concepts necessary to generalize the BFGS method on Riemannian manifolds based on [1]. Exponential mapping and retractions are used to associate tangent vectors with points on the manifold, while parallel and vector transport are used to identify different tangent spaces.

Algorithm 1: Local BFGS method

1: Given starting point x_0, convergence tolerance $\varepsilon > 0$, and an inverse Hessian approximation H_0. Set $k = 0$.
2: **repeat**
3: Solve the equation
$$H_k p_k = -\nabla f(x_k) \tag{1}$$
for p_k, and define the new iterate as $x_{k+1} := x_k + p_k$.
4: Set $s_k := x_{k+1} - x_k,\ y_k := \nabla f(x_{k+1}) - \nabla f(x_k)$, and compute a new approximation of the inverse Hessian matrix according to the update rule
$$H_{k+1} := H_k + \frac{y_k y_k^\top}{y_k^\top s_k} - \frac{H_k s_k s_k^\top H_k}{s_k^\top H_k s_k}. \tag{2}$$
5: Set $k \leftarrow k+1$.
6: **until** $\|\nabla f(x_k)\| < \varepsilon$

2.1 The BFGS method

As mentioned before, Quasi-Newton (QN) methods were developed to circumvent problems that arise with Newton's method, such as the necessity to evaluate the Hessian matrix in each iteration. There certainly exist several prominent QN methods which all possess their advantages and disadvantages. In general, the BFGS method is proved to be reliable and robust. It is therefore recalled here. It should also be mentioned that instead of approximating the Hessian one can also approximate the inverse of the Hessian. The update for the inverse Hessian approximation has the form

$$B_{k+1} := B_k + \frac{(s_k - B_k y_k)s_k^\top + s_k(s_k - B_k y_k)^\top}{y_k^\top s_k} - \frac{(s_k - B_k y_k)^\top y_k}{(y_k^\top s_k)^2} s_k s_k^\top. \tag{3}$$

This yields the advantage that it is no longer necessary to solve a system of equations. Instead, only a matrix vector product has to be calculated.

The BFGS method maintains the excellent convergence properties of Newton's method. It is shown *e.g.* in [13] that it converges at a superlinear rate under certain conditions.

2.2 Affine connections

Affine connections are an important concept in differential geometry. They identify nearby tangent spaces with each other and thereby offer a possibility to differentiate tangent vector fields. Affine connections allow us to infinitesimally view a manifold as an Euclidean space. While any manifold admits an infinite amount of affine connections, there are some that offer unique and therefore often more attractive properties.

Definition 1 (Affine connection)**.** An *affine connection* ∇ on a manifold M is a mapping

$$\begin{aligned} \nabla : \Gamma(TM) \times \Gamma(TM) &\to \Gamma(TM), \\ (\eta, \xi) &\mapsto \nabla_\eta \xi \end{aligned} \tag{4}$$

that satisfies the following properties.

1. $C(M)$-linearity in the first variable: $\nabla_{f\eta + g\chi}\xi = f\nabla_\eta \xi + g\nabla_\chi \xi$.
2. $\mathbb{R}$-linearity in the second variable: $\nabla_\eta(a\xi + b\zeta) = a\nabla_\eta \xi + b\nabla_\eta \zeta$.
3. Product rule (Leibniz' law): $\nabla_\eta(f\xi) = (\eta f)\xi + f\nabla_\eta \xi$,

in which $\eta, \chi, \xi, \zeta \in \Gamma(TM)$, $f, g \in C(M)$, and $a, b \in \mathbb{R}$. $\nabla_\eta \xi$ is also called the *covariant derivative of* ξ *in the direction of* η.

Here, $\Gamma(TM)$ denotes the set of smooth vector fields on M.

The notation ηf that appears here is to be understood as the application of a vector field to a function which is defined as

$$\eta f(x) := \eta_x(f) = \dot{\gamma}(0)f := \left.\frac{d(f(\gamma(t)))}{dt}\right|_{t=0}. \tag{5}$$

An important affine connection with additional properties is the

Definition 2 (Levi-Civita connection)**.** On a Riemannian manifold (M,g), there exists a unique affine connection ∇ that satisfies the following two conditions.

1. $\nabla_\eta \xi - \nabla_\xi \eta = [\eta,\xi]$, i. e., it is *torsion free*.
2. $\chi\langle\eta,\xi\rangle = \langle\nabla_\chi \eta,\xi\rangle + \langle\eta,\nabla_\chi \xi\rangle$, i. e., it is *compatible with the Riemannian metric*

for all $\chi, \eta, \xi \in \Gamma(TM)$. This affine connection ∇ is called the *Levi-Civita connection* or the *Riemannian connection* of M.

Assuming this connection exists, it is uniquely defined.

2.3 Exponential mapping and retractions

A retraction R is a mapping from the tangent bundle TM onto the manifold M. At a specific point x the retraction is denoted by R_x and it is a mapping from T_xM onto M.

Definition 3 (Retraction)**.** A *retraction* on a manifold M is a smooth mapping R from the tangent bundle TM onto M with the following properties. Let R_x denote the restriction of R to T_xM.

1. $R_x(0_x) = x$ where 0_x denotes the zero element of T_xM.
2. With the canonical identification $T_{0_x}T_xM \simeq T_xM$, R_x satisfies $\mathrm{D}R_x(0_x) = \mathrm{id}_{T_xM}$ where id_{T_xM} denotes the identity mapping on T_xM.

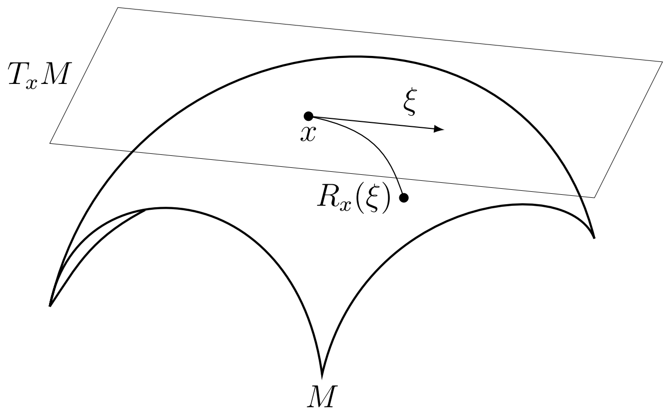

Figure 1: Illustration of a retraction.

In [1] it has been shown that the classical exponential map fulfills the requirements of a retraction. And while, in a geometric sense, the exponential mapping is the most natural retraction, it faces the problem that in order to find a solution, one is asked to solve a nonlinear ordinary differential equation. Retractions offer a way to approximate the exponential map at less computational cost while not adversely influencing the behavior of an optimization algorithm.

2.4 Parallel transport and vector transport

Given a Riemannian manifold (M,g) endowed with an affine connection and a smooth curve $\gamma\colon I \to M$. A vector field ξ on γ is called *parallel* if the equation

$$\nabla_{\dot{\gamma}(t)}\xi = 0 \quad \text{for all } t \in I \tag{6}$$

is satisfied. Now, let η_x be a tangent vector at x, and let γ be a smooth curve in M with $\gamma(a) = x$ and $\gamma(b) = y$. Then the parallel transport (or parallel translation) of η_x along γ is given by the vector field in the tangent bundle TM that fulfills the initial value problem

$$\begin{aligned} \nabla_{\dot{\gamma}(t)}\xi &= 0, \\ \xi_{\gamma(a)} &= \eta_x. \end{aligned} \tag{7}$$

The concept of parallel transport is used to identify tangent spaces.

In the same way that retractions avoid the high cost of evaluating exponential maps, there is a concept to avoid the necessity of solving a differential equation in order to identify tangent spaces. It is introduced in the next definition.

Definition 4 (Vector transport)**.** A *vector transport* on a manifold M is defined as a smooth mapping $\Gamma\colon M \times M \times TM \to TM,\ (x,y,\eta_x) \mapsto \Gamma_x^y(\eta_x),\ x,y \in M$ that fulfills the properties:

1. There exists an associated retraction R and a tangent vector ξ_x satisfying $\Gamma_x^y(\eta_x) \in T_{R_x(\xi_x)}M$ for all $\eta_x \in T_xM$.
2. $\Gamma_x^x(\eta_x) = \eta_x$ for all $\eta_x \in T_xM$.
3. The mapping $\Gamma_x^y\colon T_xM \to T_yM$ is linear.

3 The Riemannian BFGS algorithm

The classical BFGS algorithm is defined on Euclidean spaces and is easily extended to submanifolds of $\mathbb{R}^n$ that are given by equality constraints, i. e., to the case where the regular value theorem applies. In order to define a BFGS algorithm on generic manifolds, further structure is necessary. The manifold has to be equipped with a Riemannian structure, so that we are able to calculate gradients and inner products necessary to define a quasi-Newton algorithm. The following algorithm introduces the seemingly most natural way to define a Riemannian BFGS (in short: RBFGS) algorithm. The changes to the original algorithm and possible variations will be discussed in detail later on. In [15] the following algorithm is presented:

Algorithm 2: BFGS on Riemannian manifolds

1: Given a Riemannian manifold M with Riemannian metric $\langle\cdot,\cdot\rangle$, a vector transport Γ with associated retraction R, the real valued function $f\colon M\to\mathbb{R}$, an initial iterate $x_0\in M$, and an initial approximation to the Hessian H_0. Set $k:=0$.
2: **repeat**
3: Solve $H_k p_k = -\operatorname{grad} f(x_k)$ for $p_k\in T_{x_k}M$.
4: Obtain the step length α through an appropriate line search algorithm.
5: Define

$$
\begin{aligned}
s_k &:= \Gamma_{x_k}^{x_{k+1}}(\alpha p_k),\\
y_k &:= \operatorname{grad} f(x_{k+1}) - \Gamma_{x_k}^{x_{k+1}}(\operatorname{grad} f(x_k)),\\
H_{k+1}\eta &:= \tilde{H}_k\eta + \frac{\langle y_k,\eta\rangle}{\langle y_k,s_k\rangle}y_k - \frac{\langle s_k,\tilde{H}_k\eta\rangle}{\langle s_k,\tilde{H}_k s_k\rangle}\tilde{H}_k s_k \\
\text{with } \tilde{H}_k &:= \Gamma_{x_k}^{x_{k+1}}\circ H_k\circ\Gamma_{x_{k+1}}^{x_k}
\end{aligned}
\tag{8}
$$

where $H_{k+1}\colon T_{x_{k+1}}M\to T_{x_{k+1}}M$ is a linear operator.
6: Set $k\leftarrow k+1$.
7: **until** $\|\operatorname{grad} f(x_{k+1})\| < \varepsilon$

The first thing to note in Algorithm 2 is the change in the update formula for the approximated Hessian H_k. The classical update formulas for the approximation of the Hessian which are used in Euclidean space have no meaning in a Riemannian manifold setting. First, at instances where the transpose of a column vector is multiplied by another vector the standard inner product of vectors in Euclidean space is meant. When operating on Riemannian manifolds as above, s_k and y_k are vectors in the tangent space $T_{x_{k+1}}M$. The inner product on tangent spaces is then given by the chosen Riemannian metric. Furthermore, the dyadic product of a vector with the transpose of another vector, which results in a matrix in the Euclidean space, is not a naturally defined operation on a Riemannian manifold. And finally, while in Euclidean space the Hessian can be expressed as a symmetric matrix, on Riemannian manifolds it can be defined as a symmetric, bilinear form. However, due to the Lax-Milgram Lemma, [18], there exists a linear function $H\colon T_xM\to T_xM$ with

$$
\mathrm{D}^2 f(x)(\eta,\xi) = \langle\eta, H\xi\rangle,\quad \eta,\xi\in T_xM. \tag{9}
$$

This Lemma can be applied since the Hessian at x is a bilinear form on the tangent space T_xM which is actually a Hilbert space. It is this linear function H that will be updated during the BFGS algorithm instead of the Hessian matrix. Together with the use of the Riemannian metric, this leads to the update

$$
H_{k+1}\eta = \tilde{H}_k\eta + \frac{\langle y_k,\eta\rangle}{\langle y_k,s_k\rangle}y_k - \frac{\langle s_k,\tilde{H}_k\eta\rangle}{\langle s_k,\tilde{H}_k s_k\rangle}\tilde{H}_k s_k \tag{10}
$$

which was used in the algorithm above. As a result the search direction p_k is the vector in the tangent space $T_{x_{k+1}}M$ that satisfies $H_{k+1}p_k = -\operatorname{grad} f(x_{k+1})$. Instead of approximating the Hessian it is also possible to define the BFGS update for the inverse of the Hessian. This approach offers the advantage that it is not necessary to solve a system of equations. On Riemannian manifolds this update approach has the form

$$\begin{aligned} B_{k+1}\eta = \tilde{B}_k\eta + \frac{\langle s_k,\eta\rangle}{\langle y_k,s_k\rangle}s_k - \frac{\langle s_k,\eta\rangle}{\langle y_k,s_k\rangle}\tilde{B}_k y_k \\ - \frac{\langle y_k,\tilde{B}_k\eta\rangle}{\langle y_k,s_k\rangle}s_k + \frac{\langle y_k,\tilde{B}_k y_k\rangle\langle s_k,\eta\rangle}{\langle y_k,s_k\rangle^2}s_k. \end{aligned} \tag{11}$$

As before, it has the advantage that there is no need to solve a linear equation. Finding the solution to a linear equation has the additional difficulty that the solution has to be an element of a certain tangent space. Instead, this update only requires the evaluation of $B_{k+1}\operatorname{grad} f(x_{k+1})$.

Remember the notation $\tilde{H}_k$ and $\tilde{B}_k$ that has been introduced in these update formulas. The operators H_k and B_k can only be applied to elements of $T_{x_k}M$ by definition, whereas the search direction p_k is an element of $T_{x_{k+1}}M$. Thus, the tangent vector p_k has to be transported to $T_{x_k}M$. Consequently, one of the operators H_k or B_k can be applied, and the resulting tangent vector in x_k is moved back to $T_{x_{k+1}}M$ via vector transport in order for the final result to be a tangent vector at x_{k+1}. Succinctly, this means we have $\tilde{H}_k := \Gamma_{x_k}^{x_{k+1}} \circ H_k \circ (\Gamma_{x_k}^{x_{k+1}})^{-1}$ and $\tilde{B}_k = \Gamma_{x_k}^{x_{k+1}} \circ B_k \circ (\Gamma_{x_k}^{x_{k+1}})^{-1}$, respectively.

The definition of the values s_k and y_k has also been remodelled. In Algorithm 2 we chose to compute the new search direction as an element of $T_{x_{k+1}}M$. Thus, the natural choice to define s_k is αp_k, which is the closest relation we have to the difference between the last two iteration points, and transport it to $T_{x_{k+1}}M$ via vector transport. Furthermore, we define $y_k := \operatorname{grad} f(x_{k+1}) - \Gamma_{x_k}^{x_{k+1}}(\operatorname{grad} f(x_k))$. This concludes the definition of a BFGS algorithm on Riemannian manifolds.

However, this is not the only possible way to transfer the BFGS algorithm to Riemannian manifolds. There are several further degrees of freedom, which are as follows.

- The choice of the Riemannian metric.
- The choice of retraction and parallel transport.
- The choice in which tangent space the update is calculated.

For the second point, the natural choice is to use the parallel transport that is induced by the Levi-Civita connection as the vector transport and exponential mapping as the retraction. These two choices describe the “exact” operations and are the ones classically used in differential geometry. They have the drawback of high computational burden, as we already mentioned in Section 2.

An alternative way to address the third point is to move all the required variables to $T_{x_k}M$ and formulate the update there. Then the update step in Algorithm 2 has the following form:

Algorithm 3: Modified update step

5: Define

$$
\begin{aligned}
s_k &:= \alpha p_k,\\
y_k &:= (\Gamma_{x_k}^{x_{k+1}})^{-1}(\operatorname{grad} f(x_{k+1})) - \operatorname{grad} f(x_k),\\
\tilde{H}_{k+1}\tilde{\eta} &:= H_k\tilde{\eta} + \frac{\langle y_k, \tilde{\eta}\rangle}{\langle y_k, s_k\rangle} y_k - \frac{\langle s_k, H_k\tilde{\eta}\rangle}{\langle s_k, H_k s_k\rangle} H_k s_k, \quad \tilde{\eta} \in T_{x_k}M,\\
H_{k+1}\eta &:= \left(\Gamma_{x_k}^{x_{k+1}} \circ \tilde{H}_{k+1} \circ (\Gamma_{x_k}^{x_{k+1}})^{-1}\right)(\eta), \quad \eta \in T_{x_{k+1}}M
\end{aligned}
\tag{12}
$$

where $\tilde{H}_{k+1} : T_{x_k}M \to T_{x_k}M$ is a linear operator.

It is a well known fact that the parallel transport induced by the Levi-Civita connection along geodesics leaves the inner product on the two connected tangent spaces invariant. So if the vector transport is chosen to be this parallel transport, it can easily be shown that the algorithm with update rule (12) is identical to the first algorithm in the sense that the point sequences they produce coincide. However, any other choice of vector transport will lead to a different point sequence, and it is not obvious whether there is an algorithm with superior convergence properties. Another fact to note is that while update rule (8) requires five instances of vector transport per iteration, the algorithm defined with (12) only takes three and is therefore computationally cheaper if only a single iteration is considered.

We now have considered moving all required variables either to $T_{x_k}M$ or to $T_{x_{k+1}}M$, and subsequently calculating the formula for the updated, approximated Hessian in the respective tangent space. While these are the options that first come to mind, it is also possible to move all variables to a third, completely unrelated tangent space. For the arbitrary but fixed point z on the manifold M the update step then assumes the following form:

Algorithm 4: Modified update step

5: Define

$$
\begin{aligned}
s_k &:= \Gamma_{x_k}^{z}(\alpha p_k),\\
y_k &:= \Gamma_{x_{k+1}}^{z}(\operatorname{grad} f(x_{k+1})) - \Gamma_{x_k}^{z}(\operatorname{grad} f(x_k)),\\
\hat{H}_{k+1}\hat{\eta} &:= \hat{H}_k\hat{\eta} + \frac{\langle y_k, \hat{\eta}\rangle}{\langle y_k, s_k\rangle} y_k - \frac{\langle s_k, \hat{H}_k\hat{\eta}\rangle}{\langle s_k, \hat{H}_k s_k\rangle} \hat{H}_k s_k, \quad \hat{\eta} \in T_zM,\\
H_{k+1}\eta &:= \left(\Gamma_{z}^{x_{k+1}} \circ \hat{H}_{k+1} \circ \Gamma_{x_{k+1}}^{z}\right)(\eta), \quad \eta \in T_{x_{k+1}}M
\end{aligned}
\tag{13}
$$

where $\hat{H}_{k+1} : T_zM \to T_zM$ is a linear operator.

While this will most likely not have any computational advantages, it is still a possible way to define a working BFGS algorithm.

The diagrams in Figures 2 and 3 visualize the three different methods to employ the vector transport that were discussed.

$$\begin{tikzcd}
 & \tilde{p}_k \arrow[r, "\Gamma_{x_k}^{x_{k+1}}"] & p_k \\
\tilde{g}_{k+1} \arrow[ur, "\tilde{H}_{k+1}"] & & \hat{p}_k \arrow[u, "\Gamma_z^{x_{k+1}}"'] \\
g_{k+1} \arrow[u, "\Gamma_{x_{k+1}}^{x_k}"] \arrow[uurr, "H_{k+1}"] \arrow[r, "\Gamma_{x_{k+1}}^{z}"'] & \hat{g}_{k+1} \arrow[ur, "\hat{H}_{k+1}"] &
\end{tikzcd}$$

Figure 2: Obtaining the search direction.

$$\begin{tikzcd}
T_{x_k}M \arrow[rr, "\Gamma_{x_k}^{x_{k+1}}"] & & T_{x_{k+1}}M \\
 & T_{x_{k+1}}M \arrow[ur, "\Gamma_{x_{k+1}}^{x_{k+1}}"] & \\
T_{x_{k+1}}M \arrow[uu, "\Gamma_{x_{k+1}}^{x_k}"] \arrow[ur, "\Gamma_{x_{k+1}}^{x_{k+1}}"] \arrow[rr, "\Gamma_{x_{k+1}}^{z}"] & & T_zM \arrow[uu, "\Gamma_z^{x_{k+1}}"']
\end{tikzcd}$$

Figure 3: Relations between the involved tangent spaces.

The diagram in Figure 2 describes the process of obtaining the search direction p_k starting from $g_{k+1} := \operatorname{grad} f(x_{k+1})$ while the diagram in Figure 3 pictures the tangent spaces that are passed through in the course of this operation. It is important to keep in mind that in general each method of obtaining the search direction produces a different result. Hence, the diagrams are not commutative.

While these alternative ways to incorporate the vector transport were only discussed for the direct BFGS update, it is obvious that the inverse update can be adjusted in the same way. The respective update formulas B_k are obtained analogously to the ones for the direct update.

Another aspect to mention is that both the retractions and vector transports obviously are specific to the manifold on which the algorithm operates. This means that for each new manifold that is considered, these operations have to be adjusted individually.

Finally, in the case that the manifold M is simply $\mathbb{R}^n$, it is obvious that Algorithm 2, as well as its alterations Algorithms 3 and 4 reduce to the classical BFGS method in Euclidean space.

3.1 Line search along geodesics

Although this will not be the main focus of this paper, we will give a brief introduction to line search procedures on Riemannian manifolds that are required for globalizing Algorithm 2.

There are several changes that have to be made to the well known line search algorithms that are used in Euclidean space in order to use them for optimization algorithms on Riemannian manifolds. The first method that is introduced is called a backtracking procedure and represents a very basic form of line search. The implementation used here is similar to the one in [3] and is defined as follows.

Algorithm 5: Step length calculation via backtracking

1: Set $\alpha = 1$ and define $c := \langle \operatorname{grad} f(x_k), p_k \rangle$.
2: While $f(R_{x_k}(2\alpha p_k)) - f(x_k) < \alpha c$, set $\alpha := 2\alpha$.
While $f(R_{x_k}(\alpha p_k)) - f(x_k) \geq 0.5\alpha c$, set $\alpha := 0.5\alpha$.

Another possibility is to customize classical line search conditions which are used for optimization problems in $\mathbb{R}^n$. For a Riemannian BFGS problem, the *Armijo condition* assumes the form

$$f(R_x(\alpha p_k)) \leq f(x_k) + c_1 \alpha \langle \operatorname{grad} f(x_k), p_k \rangle. \tag{14}$$

As in Euclidean space we can add a curvature condition, and obtain

$$\begin{aligned} f(R_x(\alpha p_k)) &\leq f(x_k) + c_1 \alpha \langle \operatorname{grad} f(x_k), p_k \rangle, \\ \langle \operatorname{grad} f(R_x(\alpha p_k), p_k \rangle &\geq c_2 \langle \operatorname{grad} f(x_k), p_k \rangle \end{aligned} \tag{15}$$

with $0 < c_1 < c_2 < 1$. This is the *Wolfe-Powell condition* in analogy to the original Wolfe-Powell condition found in [13]. Just as in $\mathbb{R}^n$ an even more constrictive condition can be derived from this by tightening the second condition to

$$|\langle \operatorname{grad} f(R_x(\alpha p_k), p_k \rangle| \geq c_2 |\langle \operatorname{grad} f(x_k), p_k \rangle| \tag{16}$$

which results in the *strong Wolfe-Powell condition* for Riemannian manifolds.

4 Relation of BFGS algorithms on isometric manifolds

In this section, we will analyze whether RBFGS algorithms that operate on manifolds which are linked by an isometry can be related to one another.

In the following let (M, g) and (N, h) be two Riemannian manifolds, and let the function $\Phi\colon M \to N$ be a smooth map between these two smooth manifolds.

Definition 5 (Pushforward)**.** The differential $\mathrm{D}\Phi_x$ of Φ at $x \in M$ is a linear mapping

$$\begin{aligned} \mathrm{D}\Phi_x : T_x M &\to T_{\Phi(x)} M \\ \xi &\mapsto \mathrm{D}\Phi_x[\xi]. \end{aligned} \tag{17}$$

It is called the *pushforward by* Φ. In future, we will often denote the linear operator that represents this mapping by $\Phi_*[x]$, i. e., $\mathrm{D}\Phi_x[\xi] = \Phi_*[x]\xi$. If the point at which Φ is evaluated is evident from the context, it is omitted to simplify notation.

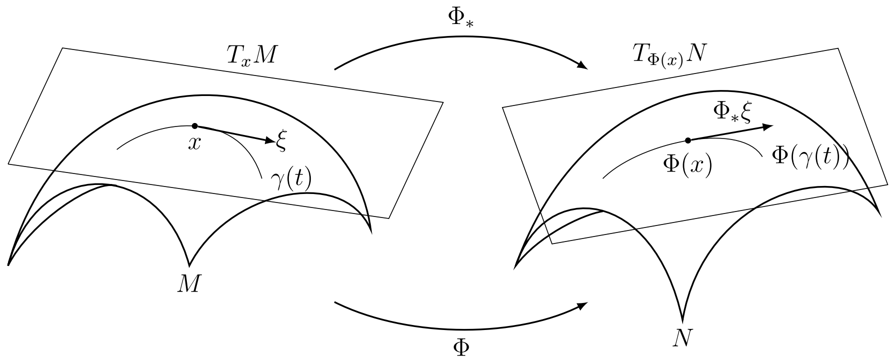

Figure 4: Illustration of the Pushforward operation.

One thing to note is that Φ is an immersion if and only if Φ_* is injective for all $x \in M$ or a submersion if and only if Φ_* is surjective for all $x \in M$.

Definition 6 (Pullback)**.** There is a linear map from the space of 1-forms on M to the space of 1-forms on N. This map is called the *pullback by* Φ and is denoted by

$$\Phi^*: T^*_{\Phi(x)}N \to T^*_x M. \tag{18}$$

Under the condition that Φ is a diffeomorphism, the pullback and the pushforward can be used to map any vector (and even tensor) field from M to N and vice versa. Of interest to us is the application of the pullback to two kinds of functions.

First, we consider the pullback of smooth cost functions. Let $f: N \to \mathbb{R}$ be a smooth function. Then the pullback of f by Φ is defined as

$$(\Phi^* f)(x) = f(\Phi(x)). \tag{19}$$

Secondly, the pullback of covariant tensor fields is examined. Let S be a $(0,s)$-tensor field on N of the form $S: T_yN \times T_yN \times \ldots \times T_yN \to \mathbb{R}$. Then the pullback of S by Φ to M is defined as

$$(\Phi^* S)_x(\xi_1, \ldots, \xi_s) := S_{\Phi(x)}(\Phi_*\xi_1, \ldots, \Phi_*\xi_s). \tag{20}$$

Note that a Riemannian metric represents a $(0,2)$-tensor field with certain additional properties.

Definition 7 (Isometry)**.** Let $\Phi: M \to N$ be a smooth map. Then Φ is called an *isometry* if Φ is a diffeomorphism with

$$g = \Phi^* h \tag{21}$$

where $\Phi^* h$ is the pullback of the Riemannian metric by Φ. For two arbitrary tangent vectors $\xi, \zeta \in T_xM$ the pullback is defined as

$$g(\xi, \zeta) = \Phi^* h(\xi, \zeta) = h(\Phi_*\xi, \Phi_*\zeta) \tag{22}$$

which means that Φ_* is an isometry of the Euclidean vector spaces T_xM and $T_{\Phi(x)}N$.

One important aspect of isometries is given in the following proposition.

Proposition 8. *Given the isometry $\Phi : M \to N$ and the geodesic γ on M. Then $\Phi \circ \gamma$ is a geodesic on N.*

Proof. Let x_1 be an arbitrary point in M with a neighborhood $U_r(x_1)$ such that for any $x_2 \in U_r(x_1)$ there exists a unique geodesic γ in $U_r(x_1)$ that connects x_1 and x_2. Without loss of generality we can assume that γ is parametrized by unit length. Since γ is a geodesic, we have equality in the triangle inequality, namely

$$d(x_1,x_2) = d(x_1,\gamma(t)) + d(\gamma(t),x_2) \tag{23}$$

where d is the distance function on M induced by the Riemannian metric g, i. e., the length of the shortest curve connecting the two points. Due to the fact that g is the pullback of h by Φ, we have $d(x_1,x_2) = d'(\Phi(x_1),\Phi(x_2))$ for the distance function d' on N which is induced by h. Hence, the equation

$$d'(\Phi(x_1),\Phi(x_2)) = d'(\Phi(x_1),\Phi(\gamma(t))) + d'(\Phi(\gamma(t)),\Phi(x_2)) \tag{24}$$

holds which means that we again have equality in the triangle inequality, and thus $\Phi \circ \gamma$ is a geodesic as well. □

For more information on pushforward, pullback, and isometries see [11] and [10].

Proposition 9. *Given two Riemannian manifolds (M,g), (N,h), the isometry $\Phi : M \to N$ with $v, w \in M$, $x := \Phi(v)$, $y := \Phi(w) \in N$, and the tangent vectors $p \in T_vM$, $q := \Phi_* p \in T_xN$. Let $\gamma : [a,b] \to M$ be the geodesic connecting v and w with $\gamma(a) = v$, $\gamma(b) = w$, and ∇ the Levi-Civita connection on N. Then the operations of parallel transport and pushforward commute, i. e.,*

$$\Phi_* \Gamma_v^w(p) = \Gamma_x^y(\Phi_* p). \tag{25}$$

The notation used for the parallel transport which we used here describes parallel transport along the geodesics γ and $\Phi \circ \gamma$ connecting the two respective points. An alternative notation that involves the path along which the parallel transport occurs is $\Gamma(\gamma)_a^b(p) = \Gamma_v^w(p)$.

Proof. The parallel transport of p along γ is defined as the solution the the initial value problem

$$\begin{aligned} \tilde{\nabla}_{\dot{\gamma}(t)} \xi &= 0, \\ \xi_{\gamma(a)} &= p \end{aligned} \tag{26}$$

with $\tilde{\nabla}$ being a connection on M. Note that in a local trivialization this is a system of linear differential equations. Since the connection on N is required to be the Levi-Civita connection, the connection on M has to be chosen to suit our purposes. The pullback connection $\Phi^* \nabla$ defined by $(\Phi^* \nabla)_X Y := \nabla_{\Phi_* X} \Phi_* Y$ appears to be an adequate choice for $\tilde{\nabla}$ because it involves the isometry that connects the two manifolds. With this connection the first equation in (26) can be written and transformed as follows

$$\begin{aligned} & \tilde{\nabla}_{\dot{\gamma}(t)}\xi = 0 \\ \Leftrightarrow \quad & (\Phi^*\nabla)_{\dot{\gamma}(t)}\xi = 0 \\ \Leftrightarrow \quad & \nabla_{\Phi_*\dot{\gamma}(t)}\Phi_*\xi = 0 \\ \Leftrightarrow \quad & \nabla_{\dot{\tilde{\gamma}}(t)}\zeta = 0. \end{aligned} \tag{27}$$

In the last equivalence we introduced substitutions for the smooth curve $\tilde{\gamma} := \Phi \circ \gamma$ and the vector field $\zeta := \Phi_*\xi$. Note that $\tilde{\gamma}$ is the geodesic connecting x and y in N since isometries preserve geodesics as seen in Proposition 8, and that the pushforward of a vector field is only defined because Φ is an isometry and therefore, by definition, a diffeomorphism.
This shows that the system of differential equations (26) is equivalent to the problem

$$\begin{aligned} \nabla_{\dot{\tilde{\gamma}}(t)}\zeta &= 0, \\ \zeta_{\tilde{\gamma}(a)} &= q \end{aligned} \tag{28}$$

in the sense that if ξ is a solution to (26), then $\zeta := \Phi_*\xi$ is a solution to (28). With these two solutions of the differential equations this, together with the definition of the parallel transport of a tangent vector from one tangent space to another in Section 2.4, leads to

$$\Phi_*\Gamma_v^w(p) = \Phi_*\Gamma(\gamma)_a^b(p) = \Phi_*\xi_{\gamma(b)} = \zeta_{\tilde{\gamma}(b)} = \Gamma(\tilde{\gamma})_a^b(\Phi_* p) = \Gamma_x^y(q) \tag{29}$$

which proves the proposition. □

Proposition 10. *Let (M,g), (N,h) be two Riemannian, isometric manifolds, i. e., there exists an isometry $\Phi : M \to N$. Let f be a smooth cost function defined on N and $\tilde{f} := f \circ \Phi$ the composition of the cost function and the isometry. Then the equation*

$$\Phi_* \operatorname{grad}_M \tilde{f} = \operatorname{grad}_N f \tag{30}$$

holds.

Proof. To define the gradient of f at $y := \Phi(x)$ for an $x \in M$ the directional derivative of f at y in direction $\xi \in T_yN$ is needed. To obtain it, a smooth curve with the properties $\alpha(t) \in N$, $\alpha(0) = y$, $\dot{\alpha}(0) = \xi$ is required. The directional derivative then is

$$\mathrm{D}f(y)[\xi] = \tfrac{\mathrm{d}}{\mathrm{d}t} f(\alpha(t))|_{t=0} = \mathrm{D}f(y)[\dot{\alpha}(0)] = \mathrm{D}f(y)[\xi]. \tag{31}$$

The directional derivative of the function $\tilde{f}$ can be obtained by using the chain rule

$$\begin{aligned} \mathrm{D}\tilde{f}(x)[\zeta] = \mathrm{D}f(\Phi(x))[\zeta] &= \tfrac{\mathrm{d}}{\mathrm{d}t} f(\Phi(\beta(t)))|_{t=0} \\ &= \mathrm{D}f(\Phi(\beta(0)))[\mathrm{D}\Phi(\beta(0))[\dot{\beta}(0)]] \\ &= \mathrm{D}f(\Phi(x))[\mathrm{D}\Phi(x)[\zeta]] = \mathrm{D}f(y)[\mathrm{D}\Phi(x)[\zeta]]. \end{aligned} \tag{32}$$

The function $\beta(t)$ is a smooth curve in M with $\beta(0)=x$, $\dot{\beta}(0)=\zeta$. According to the Riesz representation theorem, the gradient of the function f at y is the unique element $\mathrm{grad}_N f(y)$ in T_yN that satisfies

$$\langle \xi, \mathrm{grad}_N f(y)\rangle_N = \mathrm{D}f(y)[\xi] \quad \forall \xi \in T_yN. \tag{33}$$

Analogously, for $\tilde{f}$ the gradient at x is defined as the unique element $\mathrm{grad}_M \tilde{f}(x)$ in T_xM with

$$\langle \zeta, \mathrm{grad}_M \tilde{f}(x)\rangle_M = \mathrm{D}f(y)[\mathrm{D}\Phi(x)[\zeta]] \quad \forall \zeta \in T_xM. \tag{34}$$

As we know from Definition 5, $\mathrm{D}\Phi(x)[\zeta]=\Phi_*\zeta$ is the pushforward of ζ by Φ. Thus, it is an element of T_yN. Since both manifolds have the same dimension and Φ is an isometry, Φ_* describes a vector space homomorphism from T_xM to $T_{\Phi(x)}N$. Furthermore, because we have $g=\Phi^*h$, the expression on the right side of the equation can be written as

$$\langle \zeta, \mathrm{grad}_M \tilde{f}(x)\rangle_M = \langle \Phi_*\zeta, \Phi_* \,\mathrm{grad}_M \tilde{f}(x)\rangle_N. \tag{35}$$

In combination this yields

$$\mathrm{D}f(y)[\xi] = \langle \xi, \Phi_* \,\mathrm{grad}_M \tilde{f}(x)\rangle_N \quad \forall \xi \in T_yN, \tag{36}$$

and as a result we have $\mathrm{grad}_N f(y) = \Phi_* \,\mathrm{grad}_M \tilde{f}(x)$ which concludes the proof. □

In the following, the relation of the BFGS algorithm on two isometric manifolds is analyzed. Therefore, we need to introduce the necessary parameters. Let (M,g) and (N,h) be Riemannian submanifolds related by the isometry $\Phi : M \to N$ with the associated pushforward operation Φ_*. Furthermore, the parallel transport along the shortest geodesic connecting the two points x and y in either manifold is denoted by Γ_x^y. The considered cost functions are $f : N \to \mathbb{R}$ and $\tilde{f} : M \to \mathbb{R}$.

Given $x_k \in M$, $\alpha_M \in \mathbb{R}$, and $\tilde{\eta} \in T_{x_k}M$ where $\tilde{\eta}$ is the search direction established in the previous iteration. The point $x_{k+1} \in M$ is then defined as $x_{k+1} = \exp_M(\alpha_M \tilde{\eta}_k)$. Furthermore, we have $p \in T_{x_{k+1}}M$, $\tilde{B}_k{:}T_{x_k}M \to T_{x_k}M$, $\hat{\tilde{B}}_k := \Gamma_{x_k}^{x_{k+1}} \circ \tilde{B}_k \circ \Gamma_{x_{k+1}}^{x_k}$. The BFGS update on M is then defined as

$$\begin{aligned}\tilde{B}_{k+1}p = \hat{\tilde{B}}_k p + \frac{\langle s_k, p\rangle_M}{\langle s_k, v_k\rangle_M} s_k + \frac{\langle v_k, \hat{\tilde{B}}_k v_k\rangle_M \langle s_k, p\rangle_M}{\langle s_k, v_k\rangle_M^2} s_k \\ - \frac{\langle s_k, p\rangle_M}{\langle s_k, v_k\rangle_M} \hat{\tilde{B}}_k v_k - \frac{\langle v_k, \hat{\tilde{B}}_k p\rangle_M}{\langle s_k, v_k\rangle_M} s_k\end{aligned} \tag{37}$$

with $s_k = \Gamma_{x_k}^{x_{k+1}}(\alpha_M \tilde{\eta}_k)$ and $v_k = \mathrm{grad}_M \tilde{f}(x_{k+1}) - \Gamma_{x_k}^{x_{k+1}} \,\mathrm{grad}_M \tilde{f}(x_k)$.

To define the analogous algorithm on N, all the variables need to be transported to N and its tangent bundle TN via the isometry. Doing this leads to $y_k := \Phi(x_k) \in N$, $\alpha_N := \alpha_M$, and $\eta := \Phi_* \tilde{\eta}$ which is assumed for now but will be shown as a byproduct of the following proof. Due to the fact that Φ is an isometry the point that is defined as

$y_{k+1} = \Phi(x_{k+1})$ is the same point as $y_{k+1} := \exp(\alpha_N \eta)$ produces. Given in addition are $q \in T_{y_{k+1}}N$, $B_k : T_{y_k}N \to T_{y_k}N$, and $\hat{B}_k := \Gamma_{y_k}^{y_{k+1}} \circ B_k \circ \Gamma_{y_{k+1}}^{y_k}$. To simplify the notation we define $\Phi_* := \Phi_*[x_{k+1}]$. Then the BFGS update on N has the form

$$\begin{aligned} B_{k+1}q = \hat{B}_k q + \frac{\langle \sigma_k, q\rangle_N}{\langle \sigma_k, \gamma_k\rangle_N}\sigma_k + \frac{\langle \gamma_k, \hat{B}_k\gamma_k\rangle_N \langle \sigma_k, q\rangle_N}{\langle \sigma_k, \gamma_k\rangle_N^2}\sigma_k \\ - \frac{\langle \sigma_k, q\rangle_N}{\langle \sigma_k, \gamma_k\rangle_N}\hat{B}_k\gamma_k - \frac{\langle \gamma_k, \hat{B}_k q\rangle_N}{\langle \sigma_k, \gamma_k\rangle_N}\sigma_k \end{aligned} \tag{38}$$

with $\sigma_k := \Gamma_{y_k}^{y_{k+1}}(\alpha_N \eta)$ and $\gamma_k := \operatorname{grad}_N f(y_{k+1}) - \Gamma_{y_k}^{y_{k+1}} \operatorname{grad}_N f(y_k)$. Mind that the equation

$$\begin{aligned} \gamma_k &= \operatorname{grad}_N f(y_{k+1}) - \Gamma_{y_k}^{y_{k+1}} \operatorname{grad}_N f(y_k) \\ &= \Phi_*(\operatorname{grad}_M \tilde{f}(x_{k+1}) - \Gamma_{x_k}^{x_{k+1}} \operatorname{grad}_M \tilde{f}(x_k)) \\ &= \Phi_* v_k \end{aligned} \tag{39}$$

follows from Proposition 9 and 10.

These two updates already look pretty similar, especially if $\sigma_k = \Phi_* s_k$ holds. However, this is not immediately obvious, but will be shown in the course of the proof of the following proposition.

Proposition 11. *Given the two isometric Riemannian submanifolds (M,g), (N,h) which are connected by the isometry $\Phi : M \to N$ with the associated pushforward operation $\Phi_* : TM \to TN$. Let $\tilde{B}_k$ and B_k be the approximations to the inverse Hessians on the two manifolds with the update rule defined as above and $p \in T_{x_k}M$, $q := \Phi_* p \in T_{y_k}N$. Then the equation*

$$B_k q = B_k \Phi_* p = \Phi_* \tilde{B}_k p \tag{40}$$

holds. That is, the two BFGS updates are related via the pushforward operation.

Proof. By induction:
Before we start, we have to show that $\sigma_0 = \Phi_* s_0$. Since $\tilde{B}_0$ and B_0 are both the identity on the respective tangent space they are defined on, the first search direction on M is $\tilde{\eta}_0 = -\operatorname{grad}_M \tilde{f}(x_0)$ while on N it is $\eta_0 = -\operatorname{grad}_M f(y_0)$. Because of Proposition 10 and the linearity of the pushforward, this means we have $\eta_0 = \Phi_* \tilde{\eta}_0$. It can be assumed that the line search produces the same step length α for both algorithms due to the way the functions are defined. Then the equation

$$\sigma_0 = \Gamma_{y_0}^{y_1}(\alpha\eta_0) = \Gamma_{y_0}^{y_1}(\alpha\Phi_*\tilde{\eta}_0) = \Phi_*\Gamma_{x_0}^{x_1}(\alpha\tilde{\eta}) = \Phi_* s_0 \tag{41}$$

holds per definition of σ_k, s_k and because of Proposition 9.

Now the induction can be started. For $k = 1$ we have $\hat{\tilde{B}}_1 p = p$ and $\hat{B}_1 q = q$ with $p \in T_{x_1}M$ and $q := \Phi_* p \in T_{y_1}N$ since $B_0 = id_{T_{x_0}M}$, $\tilde{B}_0 = id_{T_{y_0}N}$. In combination with the fact that we are using a pullback metric we get

$$B_1 q = \Phi_*(\tilde{B}_1 p). \tag{42}$$

The notation $\Phi_* := \Phi_*[x_1]$ is used to simplify the expression. This means that the proposition holds for $k = 1$.

Assume that the proposition is true for an arbitrary $k \in \mathbb{N},\ k \geq 1$. Then for the search directions

$$\eta_k = -B_k \operatorname{grad}_N f(y_k) = -B_k \Phi_* \operatorname{grad}_M \tilde{f}(x_k) = -\Phi_* \tilde{B}_k \operatorname{grad}_M \tilde{f}(x_k) = \Phi_* \tilde{\eta}_k \tag{43}$$

holds. With identical step lengths this leads to $\sigma_k = \Phi_* s_k$ in the same way as for the case $k = 0$. Hence, for $k+1$ we get the following equation

$$B_{k+1} q = \Phi_*(\tilde{B}_{k+1} p). \tag{44}$$

with $\Phi_* := \Phi_*[x_{k+1}]$, $p \in T_{x_{k+1}}$, and $q := \Phi_* p \in T_{y_{k+1}}$
In order for this equation to hold, the equation

$$\hat{B}_k q = \Phi_*[x_{k+1}](\hat{\tilde{B}}_k p) \tag{45}$$

has to hold. To show this, let us examine what is done in detail:

$$\begin{aligned}
\hat{B}_k q &= \left(\Gamma_{y_k}^{y_{k+1}} \circ B_k \circ \Gamma_{y_{k+1}}^{y_k} \circ \Phi_*[x_{k+1}]\right)(p) \\
&= \left(\Gamma_{y_k}^{y_{k+1}} \circ B_k \circ \Phi_*[x_k] \circ \Gamma_{x_{k+1}}^{x_k}\right)(p) \\
&= \left(\Gamma_{y_k}^{y_{k+1}} \circ \Phi_*[x_k] \circ \tilde{B}_k \circ \Gamma_{x_{k+1}}^{x_k}\right)(p) \\
&= \left(\Phi_*[x_{k+1}] \circ \Gamma_{x_k}^{x_{k+1}} \circ \tilde{B}_k \circ \Gamma_{x_{k+1}}^{x_k}\right)(p) \\
&= \Phi_*[x_{k+1}](\hat{\tilde{B}}_k p)
\end{aligned} \tag{46}$$

The other transformations hold because of Proposition 9 and the assumption made for the induction. This concludes the proof. □

We can now show that the BFGS method is invariant under isometries, i. e., given two isometric Riemannian manifolds (M,g), (N,h) with the isometry $\Phi : M \to N$ and the cost functions $f : N \to \mathbb{R}$, $\tilde{f} : M \to \mathbb{R}$ with $\tilde{f} := \Phi^* f$. Starting from the two points $x_0 \in M$, $y_0 := \Phi(x_0) \in N$ the BFGS methods on the two manifolds converge to $x_*, y_* = \Phi(x_*)$, respectively.

To show this, we consider the BFGS step from the starting point $x_0 \in M$ and, respectively, the BFGS step on N starting from $y_0 := \Phi(x_0)$. As seen in Proposition 10 we have

$$\Phi_* \operatorname{grad}_M \tilde{f}(x_0) = \operatorname{grad}_N f(y_0). \tag{47}$$

Furthermore, we know from Proposition 11 that the equation

$$\begin{gathered}
B_0 \operatorname{grad}_N f(y_0) = B_0 \Phi_* \operatorname{grad}_M \tilde{f}(x_0) = \Phi_* \tilde{B}_0 \operatorname{grad}_M \tilde{f}(x_0) \\
\text{with } B_0 = id_{T_{y_0}N} \text{ and } \tilde{B}_0 = id_{T_{x_0}M}
\end{gathered} \tag{48}$$

holds. This means that the search directions are equivalent under pushforward by Φ. According to the RBFGS algorithm the next iteration point in M then lies on the geodesic that emanates from x_0 in direction $\eta := -\tilde{B}\,\mathrm{grad}_M \tilde{f}$ while in N it is placed on the one geodesic starting in y_0 in direction $\Phi_* \eta$. Its exact location is determined by using a line search algorithm which yields the step lengths α_M and α_N. Since the function $\tilde{f}$ is defined as $\Phi^* f$ and with the backtracking algorithm 5 chosen as the line search, the two step lengths are equal to one another. Therefore, we will omit the subscript indices and simply write α for the step length. Since Φ is an isometry, it maps geodesics on M onto geodesics on N, and in conclusion we have $y_1 = \Phi(x_1)$. Thus, the first iterate is invariant.

Now, assume that $y_k = \Phi(x_k)$ for $k \geq 1 \in \mathbb{N}$. Then, analogous to the case where $k = 0$, we have

$$\begin{aligned} \Phi_*\,\mathrm{grad}_M \tilde{f}(x_k) &= \mathrm{grad}_N f(y_k), \\ B_k\,\mathrm{grad}_N f(y_k) &= B_k \Phi_*\,\mathrm{grad}_M \tilde{f}(x_k) = \Phi_* \tilde{B}_k\,\mathrm{grad}_M \tilde{f}(x_k) \end{aligned} \tag{49}$$

according to Propositions 10 and 11. As before, this means that the search direction for the BFGS algorithm on N is the pushforward by Φ of the search direction for the BFGS algorithm on M. Hence, the geodesic describing the search direction on N is the image of Φ of the geodesic on M, and they lead (with step length $\alpha_M = \alpha_N$) to the iteration points $x_{k+1} \in M$ and $y_{k+1} \in N$ with $y_{k+1} = \Phi(x_{k+1})$. By induction this means that the BFGS algorithms on the two isometric Riemannian manifolds converge to the points x^*, y^* with $y^* = \Phi(x^*)$. Thus, the BFGS algorithm is invariant under isometries. Figure 5 illustrates this.

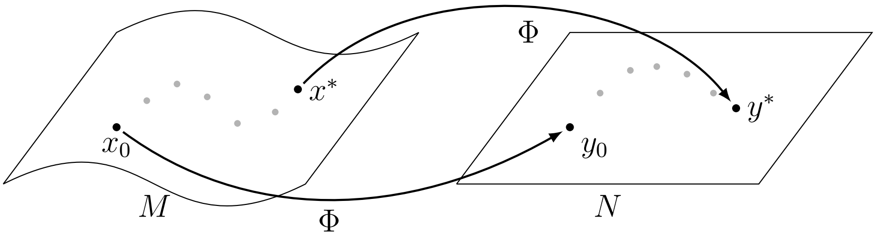

Figure 5: The RBFGS algorithm on isometric manifolds. The gray points represent the iteration points of the RBFGS algorithm on the respective manifolds.

In summary these propositions yield the following result:

Corollary 12. *For every BFGS algorithm on Riemannian manifolds that are isometric to $\mathbb{R}^n$ there is an equivalent BFGS algorithm on $\mathbb{R}^n$. This means that the convergence proof that is given in [5] for the unrestricted BFGS method in $\mathbb{R}^n$ can be applied to these algorithms, and the global superlinear convergence rate carries over to these Riemannian manifolds.*

Bibliography

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008. Cited pp. 2 and 4.

[2] R. L. Adler, J.-P. Dedieu, J. Y. Margulies, M. Martens, and M. Shub. Newton's method on Riemannian manifolds and a geometric model for the human spine. *IMA J. of Numerical Analysis*, 22:359–390, 2002. Cited p. 1.

[3] I. Brace and J. H. Manton. An improved BFGS-on-manifold algorithm for computing weighted low rank approximations. In *Proceedings of the MTNS*, pages 1735–1738, 2006. Cited pp. 2 and 10.

[4] I. M. Brace. *Optimization on manifolds with applications*. PhD thesis, The University of Melbourne, 2006. Not cited.

[5] J. E. Dennis and J. J. Moré. A characterization of superlinear convergence and its application to quasi-Newton methods. *Math. Comp.*, 28:549–560, 1974. Cited p. 17.

[6] A. Edelmann, T. Arias, and S. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix. Anal. Appl.*, 20(2):303–353, 1998. Cited p. 1.

[7] D. Gabay. Minimizing a differentiable function over a differential manifold. Research Report RR-0009, INRIA, 1980. Cited p. 1.

[8] D. Gabay. Minimizing a differentiable function over a differential manifold. *Journal of Optimization Theory and Applications*, 37:177–219, 1982. Cited p. 1.

[9] P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization*. Academic Press, 1981. Not cited.

[10] J. Jost. *Riemannian geometry and geometric analysis*. Springer, 5th edition, 2008. Cited p. 12.

[11] J. M. Lee. *Introduction to Smooth Manifolds*. Springer, 2002. Cited p. 12.

[12] D. G. Luenberger and Y. Ye. *Linear and nonlinear programming*. Springer, 3rd edition, 2008. Cited p. 1.

[13] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999. Cited pp. 3 and 10.

[14] E. Polak. *Optimization. Algorithms and consistent approximations.* Springer, 1997. Cited p. 1.

[15] C. Qi, K. A. Gallivan, and P.-A. Absil. Riemannian BFGS algorithm with applications. In M. Diehl, F. Glineur, E. Jarlebring, and W. Michiels, editors, *Recent Advances in Optimization and its Applications in Engineering*, pages 183–192. Springer, 2010. Cited pp. 2 and 5.

[16] W. Ring and B. Wirth. Optimization methods on Riemannian manifolds and their application to shape space. *SIAM Journal on Optimization*, 2012. To appear. Cited p. 2.

[17] C. Udrişte. *Convex Functions and Optimization Methods on Riemannian Manifolds*. Kluwer Academic Publishers, 1994. Cited p. 1.

[18] K. Yosida. *Functional analysis*. Springer, 6th edition, 1980. Cited p. 6.