TECHNICAL REPORT

Newton Algorithms for Riemannian Distance Related Problems on Connected Locally Symmetric Manifolds

Ricardo Ferreira, João Xavier, João Costeira, and Victor Barroso

Institute for Systems and Robotics (ISR)
Signal and Image Processing Group (SPIG)
Instituto Superior Técnico (IST)

November 2008

Abstract

The squared distance function is one of the standard functions on which an optimization algorithm is commonly run, whether it is used directly or chained with other functions. Illustrative examples include center of mass computation, implementation of k-means algorithm and robot positioning. This function can have a simple expression (as in the Euclidean case), or it might not even have a closed form expression. Nonetheless, when used in an optimization problem formulated on non-Euclidean manifolds, the appropriate (intrinsic) version must be used and depending on the algorithm, its gradient and/or Hessian must be computed. For many commonly used manifolds a way to compute the intrinsic distance is available as well as its gradient, the Hessian however is usually a much more involved process, rendering Newton methods unusable on many standard manifolds. This article presents a way of computing the Hessian on connected locally-symmetric spaces on which standard Riemannian operations are known (exponential map, logarithm map and curvature). Although not a requirement for the result, describing the manifold as naturally reductive homogeneous spaces, a special class of manifolds, provides a way of computing these functions. The main example focused in this article is centroid computation of a finite constellation of points on connected locally symmetric manifolds since it is directly formulated as an intrinsic squared distance optimization problem. Simulation results shown here confirm the quadratic convergence rate of a Newton algorithm on commonly used manifolds such as the sphere, special orthogonal group, special Euclidean group, symmetric positive definite matrices, Grassmann manifold and projective space.

Chapter 1

Introduction and Motivation

The motivation behind the computation of the Hessian of the squared distance function is usually to use this important function in intrinsic Newton-like optimization methods. The advantages of these methods when compared to gradient methods are well known, especially when high precision is required since its quadratic convergence rate is guaranteed to outperform any gradient-based algorithm when enough iterations are run. Several authors have approached the problem of implementing intrinsic Newton algorithms on smooth manifolds. For example [1], [2] and [3] discuss several implementations of Newton algorithms on manifolds and applications can be found in robotics [4], signal processing [5], image processing [6], etc.

In this article, special emphasis is given to the problem of centroid computation since it is readily formulated on manifolds as an optimization problem using little more than the squared distance function to a point. Several authors have tackled the problem of centroid computation on manifolds: Moakher [7] uses centroid computation on $\mathbb{SO}(3)$ for smoothing experimental observations obtained with a significant amount of noise in the study of plate tectonics and sequence-dependent continuum modeling of DNA; Manton [8] confirms the need of centroid computation algorithms on manifolds (particularly compact Lie groups); Pennec [9] uses positive definite symmetric matrices as covariance matrices for statistical characterization, also subject to smoothing; Fletcher [10] uses the computation of centroids for analyzing shapes in medical imaging. Other applications of squared distance cost functions include MAP estimators, where the probability functions depend on the distance to some nominal point, and the classic k-means algorithm.

Several approaches to the optimization of cost functions involving intrinsic squared distance exist, most of them relying on gradient methods, although there are a few exceptions where a Newton method is used. Hüper and Manton [11] have developed a Newton method for this cost function on the special

orthogonal group and in [12] a Newton method which operates on an approximation of the intrinsic distance function on the Grassmann manifold.

It is important to note that all of the manifolds mentioned, and many other commonly used in engineering are a subset of naturally reductive homogeneous spaces (see for example [13] and [14] for an introduction). These objects are interesting since all the required operations considered in this article for computing the Hessian are easily obtained. Please note though that there is no relation between connected locally symmetric spaces (required for this article) and naturally reductive homogeneous spaces (a worthy example is the Stiefel manifold which is not locally symmetric). All the examples presented in this article: the Grassman manifold ($\mathbb{G}(n,k)$), projective space ($\mathbb{P}^n$), sphere ($\mathbb{S}^n$), positive definite matrices ($Sym^+(n)$), the special orthogonal group ($\mathbb{SO}(n)$) and the special Euclidean group ($\mathbb{SE}(n)$) are examples of (and shall later be described as) naturally reductive homogeneous spaces.

1.1 Contribution

This article follows from two conference papers [15] and [16] where the results were presented without proof. The present article is entirely self-contained with respect to the previous two and solidifies them in both clarity and detail. Furthermore, we expand the applications of our result to computing k-means on manifolds. These papers present a method for computing the Hessian of the intrinsic Riemannian squared distance function on a connected locally-symmetric manifold. The result is presented concisely as a matrix (once a base for the tangent space is fixed) as expected in most engineering applications.

The main application focused is to solve optimization problems where the cost function depends on the squared distance function using a Newton method on manifolds. In particular the problem of computing the centroid of a constellation of points is discussed in depth (on the manifolds mentioned above), but an example of MAP estimation is included, as well as an example of clustering using the k-means algorithm.

1.2 Article Organization

The paper is structured as follows:

Section 2 introduces the notation used and defines the necessary differential geometric concepts.

Section 3 provides a brief review of Newton's optimization method both in the $\mathbb{R}^n$ case and the appropriate changes needed to implement it on a manifold. It is by no means a thorough review, serving mainly to refresh the reader. This section provides a skeleton of the algorithm to implement the Newton method.

Section 4 provides the main content of the article, showing how to obtain the Hessian of the squared distance function. As section 2 shows, this is a requirement to implement Newton's method whenever the cost function depends

on this widely used function. Although the main results are shown here, for readability the proof is deferred for an appendix. An algorithm skeleton is also provided for easy implementation.

Section 5 does a slight detour from the main line of the article introducing, almost in tutorial style, how to characterize a certain class of widely used manifolds (known as natural reductive homogeneous spaces) so that the needed operations necessary to implement the results in the previous sections can be obtained. The examples provided were chosen based on eligibility (they needed to be locally symmetric for the main theorem to be applicable) and importance in engineering.

Section 6 describes centroid computation, the main application focused in this article. Results obtained are shown in section 7. Conclusions are drawn in section 7.5.

As stated, there is also an important appendix proving the results shown in section 4.

Chapter 2

Riemannian Manifolds and Notation

Although this article assumes some familiarity with Riemannian manifolds, this section provides a brief introduction, reviewing the necessary definitions and notation for understanding the main theorem. Note that this section is not enough to understand the proof. The books [13,17] should provide the necessary theory for completely understanding the proof. Other texts introducing the field include [18,19] and [14].

For a given smooth n dimensional manifold M, denote its tangent space at a point $p \in M$ by T_pM. The disjoint union of all these tangent spaces is called the tangent bundle of M and is denoted as TM. The set of real valued functions on M is $C^\infty(M)$. If M and N are smooth manifolds, given $f : M \to N$ a smooth map between them, its push-forward is defined as the application $f_* : TM \to TN$ such that for any tangent vector $X_p \in T_pM$ and any function $g \in C^\infty(N)$ the equality $f_*(X_p) \cdot g = X_p \cdot (g \circ f)$ holds. If $f \in C^\infty(M)$, exterior differentiation is denoted by df (here f is seen as a degree 0 differential form).

Additionally, the manifold M might be equipped with a non-degenerate, positive and symmetric 2-tensor field g, called a Riemannian metric, providing the tangent space at each point p with an inner product $g_p : T_pM \times T_pM \to \mathbb{R}$. The notation $\langle X_p, Y_p \rangle = g_p(X_p, Y_p)$ for $X_p, Y_p \in T_pM$ will be used extensively.

The Riemannian exponential map is defined on any Riemannian manifold and sends a vector $X_q \in T_qM$ to a point on the manifold. If γ is the unique unit speed geodesic such that $\gamma(0) = q$ and $\dot{\gamma}(0) = X_q$, then $exp_q(X_q) = \gamma(1)$. In general exp_q is only defined on a neighborhood of the origin in T_qM. However, complete spaces, defined as those where the exp_p map has domain T_pM are very interesting in view of manifold optimization. On a sufficiently small open neighborhood, this map is a diffeomorphism. Its inverse function known as the logarithm, when defined, returns $X_q = log_q(p)$ such that, $\gamma(0) = q$, $\gamma(1) = p$ and $\dot{\gamma}(0) = X_q$. Although computation of these maps may be very involved, many manifolds used in engineering have already been widely studied and these

maps are usually available in the literature (see section 5 for a simple way to compute them for the special class of naturally reductive homogeneous spaces).

The length of a smooth curve $\gamma : [a, b] \to M$ is defined as

$$l(\gamma) = \int_a^b \sqrt{\langle \dot{\gamma}(t), \dot{\gamma}(t) \rangle} dt \ .$$

The intrinsic distance between two points p, q belonging to the same connected component of M is defined as the infimum of the length of all smooth curves joining p and q.

On a Riemannian manifold there is a canonical way of identifying nearby tangent spaces called the Levi-Civita connection, here denoted by ∇. Once a connection is established, the curvature endomorphism is defined as $R(X_q, Y_q) \cdot Z_q = \nabla_X \nabla_Y Z - \nabla_Y \nabla_X Z - \nabla_{[X,Y]} Z$. Here X, Y, Z are any vector fields extending $X_q, Y_q, Z_q \in T_pM$ and $[\cdot, \cdot]$ is the Lie bracket. The operator is independent of the extension chosen.

The gradient vector $\operatorname{grad} f(p) \in T_pM$ is then defined as the unique tangent vector that satisfies $(df)_p X_p = \langle \operatorname{grad} f(p), X_p \rangle$ for any $X_p \in T_pM$. The Hessian is defined as the symmetric 2-form such that $\operatorname{Hess} f(q)(X_q, Y_q) = \left\langle \nabla_{X_q} \operatorname{grad} f, Y_q \right\rangle$ for any $X_p, Y_p \in T_pM$. Note that once an orthonormal basis $\left\{ F_{ip} \right\} \subset T_pM$ for the tangent space is fixed, any tangent vector has a canonical expansion with respect to this basis and inner product given by $X_p = \sum_{i=1}^n x_i F_{ip}$, where $x_i = \left\langle X_{p_i}, F_{ip} \right\rangle$. These coefficients can be collected in a column matrix $\hat{X} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^T$, easily inputed to a computer. The hat will denote a coordinate representation for a given object on the manifold. Similarly the Hessian with respect to this basis can be described as the matrix $\hat{\mathbf{H}}$ such that $\operatorname{Hess} f(q)(X_q, Y_q) = \hat{X}^T \hat{\mathbf{H}} \hat{Y}$ for any $X_p, Y_p \in T_pM$ with coordinate representation in this basis $\hat{X}, \hat{Y} \in \mathbb{R}^n$.

Chapter 3

Newton's Method

3.1 Unconstrained Optimization

Gradient descent methods (familiarity with basic optimization techniques is assumed, see for example [20] or [21] for detailed reference) are undoubtedly among the easiest to implement on smooth cost functions (as is the case of the squared distance function on $\mathbb{R}^n$). Unfortunately it has linear convergence rate, which might be prohibitively expensive on applications where precision is required. Newton's method, when applicable, trades a little in implementation simplicity to gain greatly in convergence speed, guaranteeing quadratic convergence rate when close enough to the optimum.

Suppose a function $f : \mathbb{R}^n \to \mathbb{R}$ is to be minimized (assume f is convex for simplicity). One way of interpreting Newton's method is to describe it as a sequence of second order Taylor expansions and minimizations. Let

$$f(x+d) \approx \hat{f}(x+d) = f(x) + \langle \operatorname{grad} f(x), d \rangle + \frac{1}{2} d^T \mathbf{H} d$$

where $\mathbf{H}$ is a matrix representation of the Hessian function. In $\mathbb{R}^N$ the gradient vector field is easily computed as $\operatorname{grad} f(x) = \left[\frac{\partial}{\partial x_i} f(x)\right]$ and the Hessian matrix has the familiar form $\mathbf{H} = \left[h_{ij}\right]$ where $h_{ij} = \frac{\partial}{\partial x_j} \operatorname{grad} f_i(x)$. Here $\hat{f}$ is a second order polynomial in d attaining a minimum when $d = -\mathbf{H}^{-1} \operatorname{grad} f(x)$. The idea is that $x + d$ will be closer to the point which minimizes f.

Please note that when minimizing non-convex functions, some safeguards must be implemented. One way of guaranteeing convergence to a local minimum is to make sure a descent direction is chosen at each step (its dot product with the gradient should be negative, if not, a standard negative gradient direction should be used) and to use Armijo's rule for step size selection (see the previous references for details). The complete algorithm can now be described:

$\mathbb{R}^n$ **Newton Algorithm**

Input: function $f : \mathbb{R}^N \to \mathbb{R}$ to be minimized.
Output: $\bar{x}$ which minimizes f within tolerances.

choose $x_0 \in \mathbb{R}^N$ and tolerance $\epsilon > 0$. Set $k = 0$.
loop
    $g_k = \operatorname{grad} f(x_k)$.
    if $|g_k| < \epsilon$ set $\bar{x} = x_k$ and return.
    $H_k = \operatorname{Hess} f(x_k)$.
    $d_k = -\mathbf{H}_k^{-1} g_k$ (Newton direction).
    if $\langle d_k, g_k \rangle \geq 0$ set $d_k = -g_k$.
    $\alpha_k \underset{Armijo}{\approx} \operatorname{argmin}_{\alpha \geq 0} x_k + \alpha d_k$.
    $x_{k+1} = x_k + \alpha_k d_k$.
    $k \leftarrow k + 1$.
end loop

3.2 Manifold Optimization

When dealing with constrained optimization the classic solution becomes more involved. When the constraint set is a known manifold M though, the previous description still applies with only slight re-interpretations (see [2], [3] and [22] for some generalizations). A search direction is generated $d_q \in T_q M$ as the solution of the system

$$\nabla_{d_q} \operatorname{grad} f = -\operatorname{grad} f$$

If a basis for the tangent space is chosen, then the former is written as

$$\hat{\mathbf{H}} \cdot \hat{d}_q = -\hat{g} \tag{3.1}$$

where $\hat{\mathbf{H}}$ is a matrix representation of the Hessian of the cost function (with respect to the chosen basis for the tangent space) and $\hat{g}$ is the representation of the gradient $\operatorname{grad} f(q) \in T_q M$ also in the chosen basis. See section 2 for a description of these intrinsic objects and section 5 for a way of computing them in certain spaces.

As stated in the previous section, once a Newton direction has been obtained, it should be checked if it's a descent direction (its dot product with the gradient vector should be negative). If this is not verified, a fallback to the gradient descent direction should be used. Once a direction has been obtained a step in that direction must be taken. Although on a manifold it is not possible to add a vector to a point directly, a canonical way of doing it is available through the Riemannian exponential map which sends a vector $X_q \in T_q M$ to a point on the manifold as described in section 2. So the update equation $q_{k+1} = exp_{q_k}(\alpha_k d_k)$, can be used to obtain a better estimate. Here α_k is again a step size given by Armijo's rule.

The complete algorithm is now described, with only slight modifications relative to the $\mathbb{R}^N$ case:

Manifold Newton Algorithm

Input: function $f : M \to \mathbb{R}$ to be minimized.
Output: $\bar{x}$ which minimizes f within tolerances.

choose $q_0 \in M$ and tolerance $\epsilon > 0$. Set $k = 0$.
loop
    $g_k = \operatorname{grad} f(q_k) \in T_{q_k} M$.
    if $|g_k| < \epsilon$ set $\bar{q} = q_k$ and return.
    compute Newton direction d_k as the solution of (3.1).
    if $\langle d_k, g_k \rangle \geq 0$ set $d_k = -g_k$.
    $\alpha_k \underset{Armijo}{\approx} \operatorname{argmin}_{\alpha \geq 0} exp_{q_k}(\alpha d_k)$.
    $q_{k+1} = exp_{q_k}(\alpha_k d_k)$. Please note that due to finite precision limitations, after a few iterations the result should be enforced to lie on the the manifold.
    $k \leftarrow k + 1$ and re-run the loop.
end loop

Chapter 4

Hessian of the Riemannian Squared Distance Function

In [15] the following theorem was introduced without proof, and later updated in [16] still without proof. The proof is presented as an appendix to this article.

Theorem 4.0.1. *Consider M to be a connected locally-symmetric n-dimensional Riemannian manifold with curvature endomorphism R. Let $B_\epsilon(p)$ be a geodesic ball centered at $p \in M$ and $r_p : B_\epsilon(p) \to \mathbb{R}$ the function returning the intrinsic (geodesic) distance to p. Let $\gamma : [0, r] \to B_\epsilon(p)$ denote the unit speed geodesic connecting p to a point $q \in B_\epsilon(p)$, where $r = r_p(q)$, and let $\dot{\gamma}_q \equiv \dot{\gamma}(r)$ be its velocity vector at q. Define the function $k_p : B_\epsilon(p) \to \mathbb{R}$, $k_p(x) = \frac{1}{2} r_p(x)^2$ and consider any $X_q, Y_q \in T_qM$. Then*

$$\mathrm{Hess}(k_p)_q(X_q, Y_q) \;=\; \left\langle {X_q}^{\parallel}, Y_q \right\rangle \;+\; \sum_{i=1}^{n} \mathrm{ctg}_{\lambda_i}(r) \left\langle {X_q}^{\perp}, E_{iq} \right\rangle \langle Y_q, E_{iq} \rangle \quad .$$

where $\{E_{iq}\} \subset T_qM$ is an orthonormal basis which diagonalizes the linear operator $\mathcal{R} : T_qM \to T_qM$, $\mathcal{R}(X_q) = R(X_q, \dot{\gamma}_q)\dot{\gamma}_q$ with eigenvalues λ_i, this means $\mathcal{R}(E_{iq}) = \lambda_i E_{iq}$. Also,

$$\mathrm{ctg}_\lambda(t) = \begin{cases} \sqrt{\lambda}\, t / \tan(\sqrt{\lambda}\, t) & \lambda > 0 \\ 1 & \lambda = 0 \\ \sqrt{-\lambda}\, t / \tanh(\sqrt{-\lambda}\, t) & \lambda < 0 \end{cases} .$$

Here the $\parallel$ and $\perp$ signs denote parallel and orthogonal components of the vector with respect to the velocity vector of γ, i.e. $X_q = {X_q}^{\parallel} + {X_q}^{\perp}$, $\left\langle {X_q}^{\perp}, {X_q}^{\parallel} \right\rangle = 0$, and $\left\langle {X_q}^{\perp}, \dot{\gamma}(r) \right\rangle = 0$.

For practical applications though, presenting the Hessian in matrix notation greatly improves its readability and comprehension. Hence in [16] a second

theorem was presented also without proof which is included in the appendix as well.

Corollary 4.0.2. *Under the same assumptions as above, consider* $\left\{F_{iq}\right\}_{i=1}^{n} \subset T_qM$ *an orthonormal basis. If* $X_q \in T_qM$ *is a vector, let the notation* $\hat{X}$ *denote the column vector describing the decomposition of* X_q *with respect to the basis* $\left\{F_{iq}\right\}$*, i.e.* $[\hat{X}]_i = \left\langle X_q, F_{iq}\right\rangle$*, let* $\mathbf{R}_k$ *be the matrix with entries* $[\mathbf{R}_k]_{ij} = \left\langle F_{iq}, \mathcal{R}(F_{j_q})\right\rangle$ *and consider the eigenvalue decomposition* $\mathbf{R}_k = E\Lambda E^T$*. Here* λ_i *will be used to describe the i'th diagonal element of* Λ*. Then the Hessian matrix (a representation for the bilinear Hessian tensor on the finite dimensional tangent space with respect to the fixed basis) is given by:*

$$\mathbf{H}_{k_p} = \mathbf{E}\Sigma\mathbf{E}^T$$

where Σ *is diagonal with elements* σ_i *given by* $\sigma_i = \operatorname{ctg}_{\lambda_i}(r)$*. Hence*

$$\operatorname{Hess}(k_p)_q(X_q, Y_q) = \hat{X}^T\hat{\mathbf{H}}_{k_p}\hat{Y}\ .$$

This corollary is only a useful rewrite of the first, from an implementation perspective.

4.1 Spaces of Constant Curvature

In spaces of constant curvature (such as the sphere and $\mathbb{SO}(3)$) with sectional curvature λ, computing the Hessian has almost zero cost. Due to the symmetries of the curvature tensor, $\langle X_q, \mathcal{R}(Y_q)\rangle = 0$ whenever X_q or Y_q are parallel to $\dot{\gamma}_q$. Hence, matrix $\mathbf{R}_k$, which is the matrix representation on the given basis for the bilinear operator $\langle X_p, \mathcal{R}(Y_p)\rangle$, has a null eigenvalue with eigenvector $\dot{\gamma}_q$. Since the sectional curvature is by definition equal to $\langle X_q, \mathcal{R}(X_q)\rangle$ and is constant, equal to λ whenever X_q is not parallel to $\dot{\gamma}_q$, then

$$\left\{\begin{array}{ll} \max & \hat{X}^T\mathbf{R}_k\hat{X} \\ & \left\langle \hat{X}, \hat{X}\right\rangle = 1 \\ \text{s.t.} & \left\langle \hat{X}, \dot{\gamma}_q\right\rangle = 0 \end{array}\right\} = \left\{\begin{array}{ll} \min & \hat{X}^T\mathbf{R}_k\hat{X} \\ & \left\langle \hat{X}, \hat{X}\right\rangle = 1 \\ \text{s.t.} & \left\langle \hat{X}, \dot{\gamma}_q\right\rangle = 0 \end{array}\right\} = \lambda$$

hence using the Rayleigh quotient, the eigenvalues of $\mathbf{R}_k$ are constant and equal to λ. So an eigenvalue decomposition for $\mathbf{R}_k$ is

$$\mathbf{R}_k = \begin{bmatrix}\dot{\gamma}_q & \dot{\gamma}_q^{\perp}\end{bmatrix}\begin{bmatrix}0 & 0\\ 0 & \lambda\mathbf{I}\end{bmatrix}\begin{bmatrix}\dot{\gamma}_q^T \\ \dot{\gamma}_q^{\perp T}\end{bmatrix}$$

where $\dot{\gamma}_q^{\perp}$ is any orthonormal complement of $\dot{\gamma}_q$. It follows then from the last theorem that the Hessian is given by

$$\begin{aligned}\mathbf{H}_k &= \dot{\gamma}_q\dot{\gamma}_q^T + \mathrm{ctg}_\lambda(r)\dot{\gamma}_q^{\perp}\dot{\gamma}_q^{\perp^T} \\ &= \dot{\gamma}_q\dot{\gamma}_q^T + \mathrm{ctg}_\lambda(r)(\mathbf{I} - \dot{\gamma}_q\dot{\gamma}_q^T) \\ &= \mathrm{ctg}_\lambda(r)\mathbf{I} + (1 - \mathrm{ctg}_\lambda(r))\dot{\gamma}_q\dot{\gamma}_q^T\end{aligned}$$

This removes the need for the numerical computation of matrix $\mathbf{R}_k$ and its eigenvalue decomposition, significantly speeding the computation of the Hessian matrix.

4.2 Algorithm

Note that when inserted in a Newton optimization algorithm, it is usually better to have the inverse of the Hessian available. This is obtained at almost no cost since the eigenvalue decomposition is available (just substitute every occurrence of $\mathrm{ctg}_\lambda(r)$ with its reciprocal in the expressions above).

The complete algorithm is presented in both situations, when the space is not known to be of constant curvature:

Hessian of Riemannian squared distance function

Input: an orthonormal base $\{F_{iq}\} \subset T_qM$, $\dot{\gamma}_q = log_q(p)$ and the Riemannian curvature tensor.

Output: $\hat{\mathbf{H}}$ the Hessian matrix of the Riemannian squared distance function $\frac{1}{2}r_p(q)^2$.

1: Build matrix $[\mathbf{R}]_{ij} = \left\langle F_{iq}, \mathcal{R}(F_{jq})\right\rangle$.
2: Compute its eigenvalue decomposition $\mathbf{R} = \mathbf{E}\Lambda\mathbf{E}^T$.
3: Assemble diagonal matrix Σ with elements $\sigma_i = \mathrm{ctg}_{\lambda_i}(r)$.
4: $\hat{\mathbf{H}} = \mathbf{E}\Sigma\mathbf{E}^T$.

or when it is known to be of constant curvature:

Hessian of Riemannian squared distance function (spaces of constant curvature)

Input: an orthonormal base $\{F_{iq}\} \subset T_qM$, $\dot{\gamma}_q = log_q(p)$ and the Riemannian curvature tensor.

Output: $\hat{\mathbf{H}}$ the Hessian matrix of the Riemannian squared distance function $\frac{1}{2}r_p(q)^2$.

1: Represent $\dot{\gamma}_q$ in the given basis.
2: $\hat{\mathbf{H}} = \mathrm{ctg}_\lambda(r)\mathbf{I} + (1 - \mathrm{ctg}_\lambda(r))\dot{\gamma}_q\dot{\gamma}_q^T$.

Chapter 5

Naturally Reductive Homogeneous Spaces

Although this section is not critical for presenting the main result in this article, it does show that the method presented is viable by providing a recipe for obtaining the required data in a vast class of manifolds used in engineering. Note that a basic understanding of Lie group theory is assumed. Naturally Reductive Homogeneous Spaces, henceforth denoted by NRHS, (see for example [13] and [14], are important since they can lead to closed formula solutions for the Riemannian exponential maps, logarithm maps and curvature endomorphisms, exactly what's needed to implement the Hessian algorithm presented. A space with this property is defined as a coset manifold $M = G/H$, where G is a Lie group (with Lie algebra $\mathfrak{g}$) and H a closed subgroup (with Lie sub-algebra $\mathfrak{h} \subset \mathfrak{g}$), furnished with a G-invariant metric such that there exists an $Ad_H(\cdot)$ invariant subspace $\mathfrak{m} \subset \mathfrak{g}$ that is complementary to $\mathfrak{h} \subset \mathfrak{g}$. Note that $\mathfrak{g} = \mathfrak{h} \oplus \mathfrak{m}$ but $\mathfrak{m}$ is usually not a Lie sub-algebra since it is usually not closed under the Lie bracket operation. Furthermore, the property

$$\langle [X, Y]_{\mathfrak{m}}, Z \rangle = \langle X, [Y, Z]_{\mathfrak{m}} \rangle \quad \text{for } X, Y, Z \in \mathfrak{m}$$

needs to hold. Here the subscript $\mathfrak{m}$ denotes projection on this subspace. Henceforth, for spaces with this property, $\mathfrak{m}$ will be called a Lie subspace for G/H.

5.1 NRHS Construction For a Particular Riemannian Manifold

When faced with an optimization problem on a particular Riemannian manifold M, it is not usually known whether or not it admits an NRHS structure. Since many usefull manifolds in engineering admit such structures, the process of identifying it will be described here with the construction of a few manifolds.

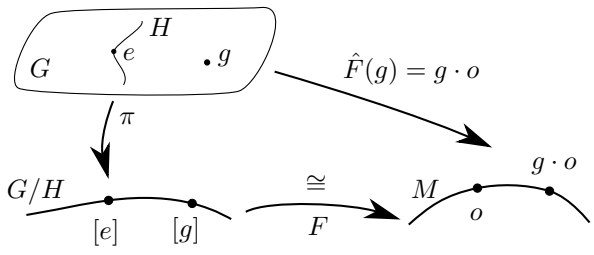


Figure 5.1: Figure illustrating the process of NRHS construction.

First it is necessary to describe M as a coset manifold $M \cong G/H$ were the symbol $\cong$ states that the two sides are diffeomorphic (see figure 5.1 for an illustration). Here, a proposition stated in [13] solves the problem, stating that all that needs to be done is to find a Lie group G which acts transitively on M (see figure 5.1):

Theorem 5.1.1. *Let $G \times M \to M$ be a transitive action and let H be its isotropy subgroup at a point $o \in M$. Then there is a natural map $F : G/H \to M$ which is a diffeomorphism. In particular, the projection $\hat{F} : G \to M,\ g \mapsto go$ is a submersion.*

Furthermore, this action must be an isometry as stated in the definition of an NRHS space, which means that for any $p \in M$, $X_p, Y_p \in T_pM$ and $g \in G$ the following must hold:

$$\langle X_p, Y_p \rangle_p = \langle g_* X_p, g_* Y_p \rangle_{gp} ,$$

where g_* denotes the push forward of the translation by g.

Examples

1. $\mathbb{SO}(n+1) \subset \mathbb{GL}(n+1)$ acts on the unit sphere $\mathbb{S}^n \subset \mathbb{R}^{n+1}$ (seen as a Riemannian subspace) as the restriction of the usual action of $\mathbb{GL}(n+1)$ on $\mathbb{R}^{n+1}$. This action is transitive. The isotropy subgroup of $o = (1, 0, \ldots, 0) \in \mathbb{S}^n$ consists of the subgroup

$$H = \left\{ \begin{bmatrix} 1 & 0 \\ 0 & q \end{bmatrix} \in \mathbb{SO}(n+1) : q \in \mathbb{SO}(n) \right\} \cong \mathbb{SO}(n)$$

   Hence, ignoring the natural diffeomorphism yields $\mathbb{S}^n \cong \mathbb{SO}(n+1)/\mathbb{SO}(n)$. To verify that this action is G-invariant let $X_p, Y_p \in T_p\mathbb{S}^n$ and $g \in \mathbb{SO}(n+1)$. Then:

$$\langle g_* X_p, g_* Y_p \rangle_{gp} = {X_p}^T g^T g Y_p = {X_p}^T Y_p = \langle X_p, Y_p \rangle_p .$$

2. As a trivial example, $\mathbb{SO}(n)$ acts transitively on itself (seen as a Riemannian submanifold of $\mathbb{GL}(n+1)$ with the Euclidean inner product) through group multiplication. The isotropy subgroup at any point is the

trivial subgroup $H = \{e\}$ (where e is the group identity), hence trivially $\mathbb{SO}(n) \cong \mathbb{SO}(n)/\{e\}$. As before, to verify that this action preserves the inner product let $X_p, Y_p \in T_p\mathbb{SO}(n)$ and $g \in \mathbb{SO}(n)$. Then:

$$\langle g_* X_p, g_* Y_p \rangle_{gp} = \operatorname{tr}\left\{ {X_p}^T g^T g Y_p \right\} = \langle X_p, Y_p \rangle_p .$$

3. Expanding the previous example, the Lie group product $G = \mathbb{SO}(n) \times \mathbb{R}^n$ acts transitively on the Special Euclidean group $M = \mathbb{SE}(n)$ (seen as a Riemannian submanifold of $\mathbb{GL}(n+1)$) as

$$\begin{aligned} (\mathbb{SO}(n) \times \mathbb{R}^n) \times \mathbb{SE}(n) &\longrightarrow \mathbb{SE}(n) \\ \left((R, v), \begin{bmatrix} Q & t \\ 0 & 1 \end{bmatrix}\right) &\longmapsto \begin{bmatrix} RQ & t+v \\ 0 & 1 \end{bmatrix} \end{aligned}$$

Once again the isotropy subgroup is trivial, hence $\mathbb{SE}(n) \cong \mathbb{SO}(n) \times \mathbb{R}^n$. If $X_p = \begin{bmatrix} K_X & \Delta_X \\ 0 & 0 \end{bmatrix} \in T_p M$ then for a given element $g = (R_g, \Delta_g) \in G$ the push forward of the action of g is given by:

$$g_* X_p = g_* \begin{bmatrix} K_X & \Delta_X \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} R_g K_X & t_X \\ 0 & 0 \end{bmatrix}$$

Hence the action preserves the inner product since:

$$\begin{aligned} \langle g_* X_p, g_* Y_p \rangle_{gp} &= \operatorname{tr}\left\{ \begin{bmatrix} R_g K_X & \Delta_X \\ 0 & 0 \end{bmatrix}^T \begin{bmatrix} R_g K_Y & \Delta_Y \\ 0 & 0 \end{bmatrix} \right\} \\ &= \operatorname{tr}\left\{ \begin{bmatrix} K_X^T K_Y & - \\ - & \Delta_X^T \Delta_Y \end{bmatrix} \right\} \\ &= \langle X_p, Y_p \rangle_p . \end{aligned}$$

4. $\mathbb{GL}(n)$ (the set of $n \times n$ invertible matrices with real entries) acts transitively on $Sym^+(n)$ (the set of $n \times n$ positive definite symmetric matrices with the inner product described below) by conjugation, that is $(g, p) \mapsto g^T p g$. The isotropy subgroup of the identity matrix seen as an element of $Sym^+(n)$ is the set $H = \left\{ g : g^T g = e \right\} = \mathbb{O}(n)$. So, $Sym^+(n) = \mathbb{GL}(n)/\mathbb{O}(()n)$. Letting $X_p, Y_p \in T_p Sym^+(n)$ and $g \in \mathbb{GL}(n)$ and assuming the inner product is given by $\langle X_p, Y_p \rangle_p = \operatorname{tr}\left\{ {X_p}^T p^{-1} Y_p p^{-1} \right\}$, then:

$$\begin{aligned} &\langle g_* X_p, g_* Y_p \rangle_{gp} \\ &\quad = \operatorname{tr}\left\{ g^T {X_p}^T g g^{-1} p^{-1} g^{-T} g^T Y_p g g^{-1} p^{-1} g^{-T} \right\} \\ &\quad = \operatorname{tr}\left\{ {X_p}^T p^{-1} Y_p p^{-1} \right\} = \langle X_p, Y_p \rangle_p \end{aligned}$$

5. If $S \in M(n,r)$ with $n > r$ let the notation $[S]$ denote the subspace generated by the column vectors of S. The Grassman manifold consists of the set of all such subspaces, i.e. $\mathbb{G}(n,r) = \{[S] : S \in M(n,r)\}$. Please note that the elements of the Grassman manifold are equivalence classes, where $[S_1] = [S_2] \Leftrightarrow$ the columns of S_1 and the columns of S_2 span the same subspace.

   Consider the transitive action of $G = \mathbb{SO}(n)$ on $M = \mathbb{G}(n,r)$ defined as

$$\mathbb{SO}(n) \times \mathbb{G}(n,r) \longrightarrow \mathbb{G}(n,r), \qquad (g,[S]) \longmapsto [gS]$$

   the isotropy subgroup of $o = \left[\begin{bmatrix} I \\ 0 \end{bmatrix}\right] \in \mathbb{G}(n,r)$ is the set

$$H = \left\{ \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix} : (Q_1,\ Q_2) \in S(\mathbb{O}(r) \times \mathbb{O}(n-r)) \right\}$$

Let $\mathfrak{g}$ be the Lie algebra of G and $\mathfrak{h}$ be the Lie sub-algebra of H. The next step consists of finding a Lie subspace $\mathfrak{m}$ such that $\mathfrak{m} + \mathfrak{h} = \mathfrak{g}$ and $Ad_H(\mathfrak{m}) = \mathfrak{m}$. This step must be done by inspection but it is usually not hard to accomplish.

Examples

1. Let $Skew(n)$ denote the set of $n \times n$ skew symmetric matrices with real entries. For the coset space $\mathbb{S}^n = \mathbb{SO}(n+1)/\mathbb{SO}(n)$, $\mathfrak{g} = Skew(n+1)$ and $\mathfrak{h} = \left\{ \begin{bmatrix} 0 & 0 \\ 0 & k \end{bmatrix} : k \in Skew(n) \right\}$. By inspection (due to the requirement that $\mathfrak{m} + \mathfrak{h} = \mathfrak{g}$) a logical candidate for $\mathfrak{m}$ is the set $\mathfrak{m} = \left\{ \begin{bmatrix} 0 & -x^T \\ x & 0 \end{bmatrix} : x \in \mathbb{R}^n \right\}$. Since

$$Ad_H(\mathfrak{m}) = \left\{ \begin{bmatrix} 1 & 0 \\ 0 & q \end{bmatrix} \begin{bmatrix} 0 & -x^T \\ x & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & q^T \end{bmatrix} = \right.$$
$$\left. = \begin{bmatrix} 0 & -(qx)^T \\ qx & 0 \end{bmatrix} : x \in \mathbb{R}^n, q \in \mathbb{SO}(n) \right\} = \mathfrak{m} \quad (5.1)$$

   results that $\mathfrak{m}$ is indeed a Lie subspace.

2. When $G = \mathbb{SO}(n)$ and $H = \{e\}$, $\mathfrak{g} = Skew(n)$ and $\mathfrak{h} = 0$ (the trivial vector space). Hence, the obvious choice is $\mathfrak{m} = \mathfrak{g}$, which is obviously invariant under $Ad_H(\cdot)$.

3. The same happens when considering $G = \mathbb{SO}(n) \times \mathbb{R}^n$ and $H = \{(e,0)\}$. Hence, for $\mathbb{SE}(n)$, $\mathfrak{m} = \mathfrak{g} = Skew(n) \times \mathbb{R}^n$.

4. If $G = \mathbb{GL}(n)$ and $H = \mathbb{O}(n)$ as is the case for $M = Sym^+(n)$, the corresponding Lie algebras are $\mathfrak{g} = M(n)$ ($n \times n$ real matrices) and $\mathfrak{h} = Skew(n)$. A natural candidate for the Lie subspace is the set of symmetric matrices $\mathfrak{m} = Sym(n)$, and indeed if $q \in \mathbb{O}(n)$ and $X_e \in Sym(n)$ then $qX_eq^T = qX_e{}^Tq^T = (qX_eq^T)^T$ hence it is $Ad_H(\cdot)$ invariant.

5. For the case of the Grassmann, $G = \mathbb{SO}(n)$ and $H = S(\mathbb{O}(r) \times \mathbb{O}(n-r))$ (as seen previously). The corresponding Lie algebras are $\mathfrak{g} = Skew(n)$ and $\mathfrak{h} = \left\{ \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} : (K_1,\ K_2) \in Skew(r) \times Skew(n-r) \right\}$. Then, by inspection, the obvious choice for the Lie subspace is

$$\mathfrak{m} = \left\{ \begin{bmatrix} 0 & M \\ -M^T & 0 \end{bmatrix} : M \in \mathbb{R}^{r \times n-r} \right\} .$$

It is easily checked that this choice is $Ad_H(\cdot)$ invariant.

All that remains to be done is to verify if the construction verifies the property

$$\langle [X,Y]_{\mathfrak{m}}, Z \rangle = \langle X, [Y,Z]_{\mathfrak{m}} \rangle \quad \text{for } X, Y, Z \in \mathfrak{m}$$

Since $\mathfrak{m}$ is identified with T_eM, the dot product is the pull-back by $\hat{F}$ of the dot product on M. If the property is not satisfied, another construction with another Lie group G acting on M might be tried, or it is possible that M does not admit an NRHS structure.

Examples

1. Continuing the previous examples consider the sphere, where $X_e, Y_e, Z_e \in \mathfrak{m} \subset Skew(n+1)$. Let $X_e = \begin{bmatrix} 0 & -x^T \\ x & 0 \end{bmatrix}$ and $Y_e = \begin{bmatrix} 0 & -y^T \\ y & 0 \end{bmatrix}$ where $x, y \in \mathbb{R}^n$. Then it results that

$$\begin{aligned} [X_e, Y_e]_{\mathfrak{m}} &= (X_eY_e - Y_eX_e)|_{\mathfrak{m}} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & y^Tx - xy^T \end{bmatrix}\Big|_{\mathfrak{m}} \\ &= 0 \end{aligned}$$

Thus the required result is trivially verified . If needed, the corresponding dot product on $\mathfrak{m}$ can be found by noting that $F_*X_e = \begin{bmatrix} 0 \\ x \end{bmatrix}$. Hence

$$\begin{aligned} \langle X_e, Y_e \rangle_{\mathfrak{m}} &= \left\langle \hat{F}_*X_e, \hat{F}_*Y_e \right\rangle_M \\ &= x^Ty \\ &= \frac{1}{2} \operatorname{tr}\{{X_e}^T Y_e\} \end{aligned}$$

2. For $M = G = \mathbb{SO}(n)$, the tangent vectors in both manifolds are canonically identified, hence if $X_e, Y_e, Z_e \in \mathfrak{m}$ the inner product on $\mathfrak{m}$ is given in the usual way:

$$\langle X_e, Y_e \rangle_{\mathfrak{m}} = \operatorname{tr}\{{X_e}^T Y_e\}$$

So the required property is once again satisfied

$$\begin{aligned}\langle [X_e, Y_e]_{\mathfrak{m}}, Z\rangle &= \operatorname{tr}\left\{ (X_e Y_e - Y_e X_e)^T Z_e \right\} \\ &= \operatorname{tr}\left\{ Y_e X_e Z_e - X_e Y_e Z_e \right\} \\ &= \operatorname{tr}\left\{ -{X_e}^T Z_e Y_e + {X_e}^T Y_e Z_e \right\} \\ &= \langle X_e, [Y_e, Z_e]_{\mathfrak{m}} \rangle\end{aligned}$$

3. When $M = \mathbb{SE}(n)$ and $G = \mathbb{SO}(n) \times \mathbb{R}^n$, the inner product on $\mathfrak{m}$ is found in the same manner as in the first example. Thus if $X_e = (K_X, \Delta_X), Y_e = (K_Y, \Delta_Y), Z_e = (K_Z, \Delta_Z) \in \mathfrak{m}$

$$\begin{aligned}\langle X_e, Y_e \rangle_{\mathfrak{m}} &= \left\langle \hat{F}_* X_e, \hat{F}_* Y_e \right\rangle_M \\ &= \operatorname{tr}\left\{ \begin{bmatrix} K_X & \Delta_X \\ 0 & 0 \end{bmatrix}^T \begin{bmatrix} K_Y & \Delta_Y \\ 0 & 0 \end{bmatrix} \right\} \\ &= \operatorname{tr}\{K_X^T K_Y\} + \Delta_X^T \Delta_Y \\ &= \langle K_X, K_Y \rangle_{\mathbb{SO}(n)} + \langle \Delta_X, \Delta_Y \rangle_{\mathbb{R}^n}\end{aligned}$$

The Lie bracket on the product group is given by the product of the Lie brackets. Hence

$$\begin{aligned}[X_e, Y_e]_{\mathbb{SO}(n))\times\mathbb{R}^n} &= \left([K_X, K_Y]_{\mathbb{SO}(n)}, [\Delta_X, \Delta_Y]_{\mathbb{R}^n} \right) \\ &= (K_X K_Y - K_Y K_X, 0)\end{aligned}$$

Then to check the required property:

$$\begin{aligned}\langle [X_e, Y_e]_{\mathfrak{m}}, Z\rangle &= \operatorname{tr}\left\{ (K_X K_Y - K_Y K_X)^T K_Z \right\} + 0 \\ &= \langle X_e, [Y_e, Z_e]_{\mathfrak{m}} \rangle\end{aligned}$$

(check the example for the $\mathbb{SO}(n)$ case for details of the last step).

4. In the case of the symmetric positive definite matrices where $\mathfrak{m} = Sym(n)$ the Lie bracket results in a skew symmetric matrix, hence the projection back to $\mathfrak{m}$ results in a null vector. Hence the requirement is trivially satisfied.

5. Noting that for the Grassmann manifold $X_e, Y_e \in \mathfrak{m} \subset Skew(n)$

$$\begin{aligned}[X_e, Y_e]_{\mathfrak{m}} &= (X_e Y_e - Y_e X_e)|_{\mathfrak{m}} \\ &= 0\end{aligned}$$

The needed property is once again trivially verified.

Hence all five manifolds have been described as naturally reductive homogeneous spaces.

5.2 Operations on NRHS manifolds

Now is the time to unveil why this structure is important in the Riemannian optimization process. A preposition in [13] states:

Theorem 5.2.1. *If $M = G/H$ is a naturally reductive homogeneous space, its geodesic starting at o with tangent vector $X_o \in T_oM$ are given by $\gamma(t) = \hat{F} \circ \alpha(t)$ for all $t \in \mathbb{R}$, where $\alpha(t)$ is the one parameter subgroup of X_o identified as an element of $\mathfrak{m}$.*

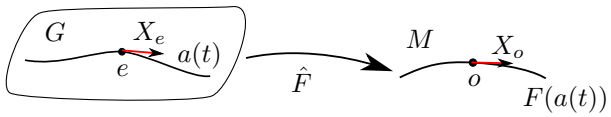


Figure 5.2: How to compute geodesics in NRHS constructions.

Hence the Riemannian exponential map follows directly from the Lie group's exponential map which in our examples is the standard matrix exponential (since G is either $\mathbb{GL}(n)$ or $\mathbb{SO}(n)$). Geodesics starting at any other point of M can be found by translation of γ since G acts transitively as an isometry. The Riemannian logarithm map follows from inversion.

On a manifold with NRHS structure, the curvature endomorphism is also computable as seen for example in [23]:

$$R(X_p, Y_p) \cdot Z_p = \left[Z_p, [X_p, Y_p]_{\mathfrak{h}}\right] + \frac{1}{2}\left[Z_p, [X_p, Y_p]_{\mathfrak{m}}\right]_{\mathfrak{m}} + \frac{1}{4}\left[[X_p, Z_p]_{\mathfrak{m}}, Y_p\right]_{\mathfrak{m}} + \frac{1}{4}\left[X_p, [Y_p, Z_p]_{\mathfrak{m}}\right]_{\mathfrak{m}} \quad (5.2)$$

Examples To finish the examples, a summary of the functions needed for each of the considered manifolds is provided:

1. The Sphere $\mathbb{S}^n$:
   This n-dimensional manifold is described as the set $\mathbb{S}^n = \{x \in \mathbb{R}^{n+1} : \|x\| = 1\}$ whose tangent space at a point $p \in \mathbb{S}^n$ is $T_p\mathbb{S}^n \cong \{x \in \mathbb{R}^{n+1} : p^T x = 0\}$. Let $p, q \in \mathbb{S}^n$, $X_p, Y_p, Z_p \in T_p\mathbb{S}^n$ and s is the norm of X_p. It can be shown that for the ambient metric $\langle X_p, Y_p \rangle = {X_p}^T Y_p$:
   - $exp_p(X_p) = p\cos(s) + \frac{X_p}{s}\sin(s)$.
   - $log_p(q) = (q - p(p^T q))\frac{a}{\sin(a)}$ where $a = \arccos(p^T q)$.
   - $R(X_p, Y_p) \cdot Z_p = \langle Y_p, Z_p \rangle X_p - \langle X_p, Z_p \rangle Y_p$.
2. Special Orthogonal Group $\mathbb{SO}(n)$
   This $n(n-1)/2$ dimensional manifold represents the set of rotations of $\mathbb{R}^n$ and is described as $\mathbb{SO}(n) = \{x \in M(n,n) : x^T x = \mathbf{I}\}$ whose tangent space at a point $p \in \mathbb{SO}(n)$ is $T_p\mathbb{SO}(n) \cong \{pk : k \in Skew(n)\}$, where $Skew(n)$ denotes the set of $n \times n$ skew-symmetric matrices. The metric comes naturally from the Riemannian embedding as $\langle X_p, Y_p \rangle = \mathrm{tr}\{{X_p}^T Y_p\}$

- $exp_p(X_p) = p\exp(p^T X_p)$, where exp denotes the matrix exponential function.
- $log_p(q) = p\log(p^T q)$ where log denotes the matrix logarithm function.
- $R(X_p, Y_p) \cdot Z_p = -\frac{1}{4}\left[\left[X_p, Y_p\right], Z_p\right]$.

3. Special Euclidean Group $\mathbb{SE}(n)$:
The Euclidean group, characterized as the product manifold $\mathbb{SO}(n) \times \mathbb{R}^n$, inherits these manifolds' properties. Hence, at a point $p = (R, t)$, the tangent space $T_p\mathbb{SE}(n) = T_R\mathbb{SO}(n) \times T_t\mathbb{R}^n$ with dot product given by $\langle (X_R, v_t), (Y_R, u_t)\rangle_{\mathbb{SE}(n)} = \langle X_R, Y_R\rangle_{\mathbb{SO}(n)} + \langle v_t, u_t\rangle_{\mathbb{R}^n}$.

   - $exp_{(R,t)}\left((X, v)_{(R,t)}\right) = (R\exp(R^T X_R), t + v_t)$, where exp denotes the matrix exponential function.
   - $log_{(R,t)}\left((Q, s)\right) = (R\log(R^T Q), s - t)$, where log denotes the matrix logarithm function.
   - $R\left((X, v)_{(R,t)}, (Y, u)_{(R,t)}\right)\cdot(Z, w)_{(R,t)} = \left(-\frac{1}{4}\left[\left[X_R, Y_R\right], Z_R\right], 0\right)$ where the brackets denote the Lie bracket only on $\mathfrak{so}(n)$ since $\mathbb{R}^n$ is flat.

4. Symmetric positive definite matrices $Sym^+(n)$:
This $n(n+1)/2$ dimensional manifold is described as the set $Sym^+(n) = \{x \in M(n, n) : x = x^T, \text{with positive eigenvalues}\}$ whose tangent space at a point $p \in Sym^+(n)$ is $T_pSym^+(n) \cong \{x : x \in Sym(n)\}$, where $Sym(n)$ denotes the set of $n \times n$ symmetric matrices. Let $p, q \in Sym^+(n)$, $X_p, Y_p, Z_p \in T_pSym^+(n)$. When considering the metric

$$\langle X_p, Y_p\rangle = \mathrm{tr}\{{X_p}^T p^{-1} Y_p p^{-1}\}$$

   the following expressions hold

   - $exp_p(X_p) = p^{1/2}\exp(p^{-1/2} X_p p^{-1/2})p^{1/2}$.
   - $log_p(q) = p^{1/2}\log(p^{-1/2} q p^{-1/2})p^{1/2}$.
   - $R(X_p, Y_p) \cdot Z_p = 1/4(Z_p p^{-1} O - O p^{-1} Z_p)$, where $O = X_p p^{-1} Y_p - Y_p p^{-1} X_p$.

5. The Grassmann manifold $\mathbb{G}(n, r)$:
The Grassmann is an $r(n-r)$ dimensional manifold of r dimensional linear subspaces in $\mathbb{R}^n$. It is naturally described as a quotient manifold, hence a point $p = [P]$ is described by a representative $P \in M(n, r)$. Note that no canonical embeding in $\mathbb{R}^{\cdot}$ exists, unlike the previous manifolds.

Chapter 6

Centroid Computation on Manifolds

Let M be a connected manifold and $\mathcal{X} = \{p_1, \ldots, p_L\} \subset M$ a constellation of L points. Let $r : M \times M \to \mathbb{R}$ be the function that returns the intrinsic distance of any two points on the manifold and define a cost function $C_{\mathcal{X}} : M \to \mathbb{R}$ as

$$C_{\mathcal{X}}(q) = \frac{1}{2} \sum_{l=1}^{L} r(p_l, q)^2 = \sum_{l=1}^{L} k_{p_l}(q) \ , \tag{6.1}$$

The set of solutions to the optimization problem $m_f(\mathcal{X}) = \operatorname{argmin}_{q \in M} C_{\mathcal{X}}(q)$ is defined as the Frï¿½chet mean set of the constellation and each member will be called a centroid of $\mathcal{X}$. Depending on the manifold M, the centroid might not be unique, for example if the sphere is considered with a constellation consisting of two antipodal points, all the equator points are centroids. The set of points at which the function (6.1) attains a local minimum is called the Karcher mean set and is denoted as $m_k(\mathcal{X})$. The objective is to find a centroid for the given constellation (which in the applications of interest should be unique), but the possibility of convergence to a local minimum is not dealt with. If the points on the constellation are close enough to each other, it is known that the global set $m_f(\mathcal{X})$ has a single element and so the centroid is unique as stated in [8] and [24].

Using linearity of the gradient and the Hessian operators (meaning in particular that if $f, g : M \to \mathbb{R}$ then $\operatorname{Hess}(f+g) = \operatorname{Hess} f + \operatorname{Hess} g$ and $\operatorname{grad}(f+g) = \operatorname{grad} f + \operatorname{grad} g$), the cost function in equation (6.1) allows for the decomposition

$$\begin{aligned} \operatorname{grad} C_{\mathcal{X}}(q) &= \sum_{l=1}^{L} \operatorname{grad} k_{p_l}(q) = -\sum_{l=1}^{L} log_q(p_n) \\ \operatorname{Hess} C_{\mathcal{X}}(q) &= \sum_{l=1}^{L} \operatorname{Hess} k_{p_l}(q) \ , \end{aligned} \tag{6.2}$$

where the fact that the gradient of the squared Riemannian distance function is the symmetric of the Riemannian *log* map is used (as stated in [17] as a corollary to Gauss's lemma).

The algorithm for centroid computation is then

Centroid computation

Input: A constellation $\mathcal{X} = \{p_1, \ldots, p_L\} \subset M$ with L points and an initial estimate q_0

Output: An element q of the Karcher mean set of the constellation

1: Apply Newton's algorithm as described in section 3 to function $C_{\mathcal{X}}(q)$ where at each step the Hessian and gradient is computed as follows:
2: **for** each point p_l in the constellation **do**
3: $\quad \operatorname{grad} k_{p_l}(q) = -log_q(p_n)$
4: $\quad \operatorname{Hess} k_{p_l}(q)$ (as described in section 4)
5: **end for**
6: $\operatorname{grad} C_{\mathcal{X}}(q) = \sum_{l=1}^{L} \operatorname{grad} k_{p_l}(q)$
7: $\operatorname{Hess} C_{\mathcal{X}}(q) = \sum_{l=1}^{L} \operatorname{Hess} k_{p_l}(q)$

Chapter 7

Results

This section holds experimental results for 3 applications: centroid computation, k-means algorithm and an MAP estimator. Special emphasis is given to the problem of centroid computation exactly because it is the simplest to formulate and test. The natural extension is an implementation of the k-means algorithm on a manifold. The example of MAP estimation applied to robot navigation is provided as a slightly different application where these results might prove usefull as well.

7.1 Point Cloud Generation

The examples in this section need a constellation of points to be available on some locally symmetric manifold so first an explanation on how these constellations are generated is given. The idea is that for the centroid to be well defined the points should be close together and for an error measure to be available the true centroid of the constellation must be known. Hence points on the cloud are generated with the following algorithm:

Point cloud generation

Input: An n dimensional manifold M with the corresponding exponential map,
L the number of desired points on the manifold and a radius ϵ.
Output: A constellation $\mathcal{X}$ of points and the known centroid p.

1: Generate a random point p on the manifold. This will be the true centroid and the rest of the constellation will be built around this point.
2: Generate $\{F_{p_i}\}_{i=1}^{n} \subset T_pM$ an orthonormal base.
3: Randomly generate $\{\hat{v_i}\}_{i=1}^{L}$ vectors in $\mathbb{R}^n$ with norm less than ϵ.
4: Remove the mean such that the vectors are centered around 0.
5: Build $v_{p_j} = \sum_{i=1}^{n} \hat{v_j}^i F_{p_i}$, the corresponding tangent vectors.
6: Use the exponential map to generate the constellation $\mathcal{X} = \{q \in M : q = exp_p(v_{p_j})\}$

Note that the cloud will not necessarily be contained in an ϵ-ball due to the recentering of the tangent vectors, but the result should still be "almost" within it. If needed it can be enforced by scaling each $\hat{v_i}$ after the mean is removed. The recentering step guarantees that p is the centroid of the constellation (compare with expression 6.2 and the fact that a (Karcher) centroid was defined as the points where the gradient is zero).

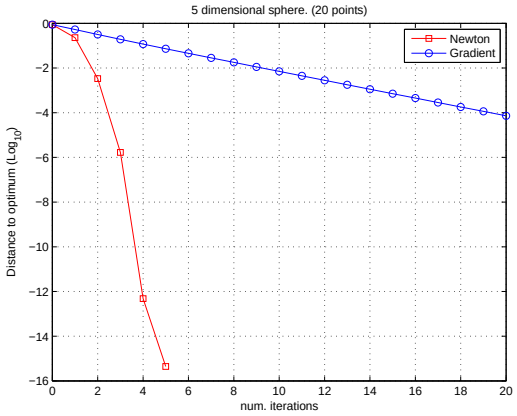
7.2 Centroid Computation

Figure 7.1 compares the results of applying a Newton algorithm and a standard gradient algorithm when computing the centroid of a constellation on 6 different manifolds. The 20-point constellations were generated using the algorithm just described, using a radius of $\pi/3$ except for the grassman where the radius used was $\pi/6$. The results presented in logarithmic scale clearly show the quadratic convergence rate of Newton's method and the linear convergence rate of the gradient method. Notice the finite precision plateau at 10^{-15}.

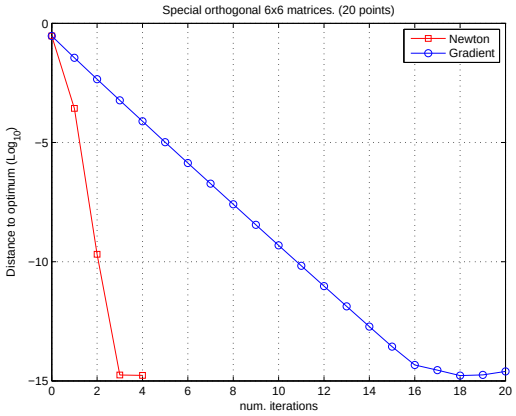
Note that the projective space manifold $\mathbb{P}^n = \mathbb{G}(n+1, 1)$ is a special case of the Grassman, hence the previous expressions are applicable.

7.3 K-means Algorithm

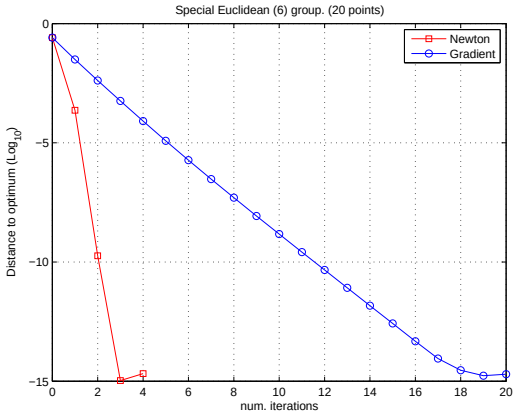
The implementation of a K-means algorithm is straightforward once a working centroid computing algorithm is available. The algorithm is as follows:



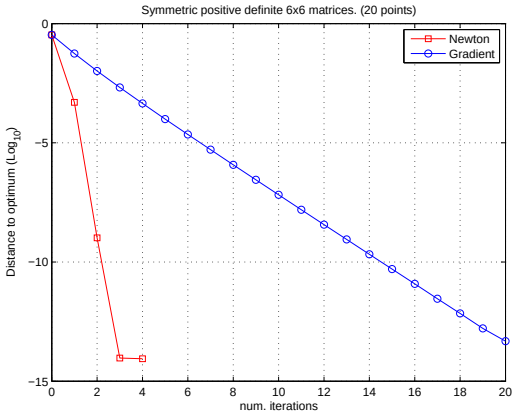
(a)



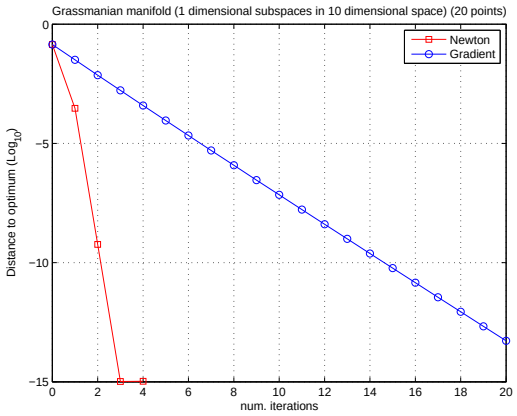
(b)



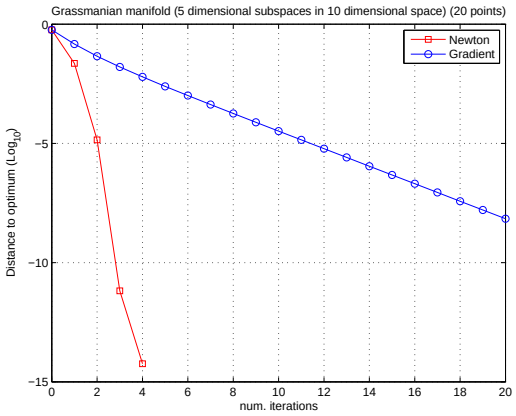
(c)



(d)



(e)



(f)

Figure 7.1: Simulation results for centroid computation. Except for the Grassman manifold, whose constellations were built with a radius of $\pi/6$, all constellations were built with a radius of $\pi/3$.

k-means algorithm

Input: An n dimensional manifold M where the centroid is computable, $\mathcal{X}$ a cloud of points and l the number of desired classes.
Output: $\{p_1...p_l\}$ centroids of each class.

1: Choose $\{p_1...p_l\} \subset \mathcal{X}$ randomly as initial estimate for the centroids.
2: **for** each point q in $\mathcal{X}$ **do**
3: Compute distance to each centroid: $r_i = r_q^2(p_i)$.
4: Label point as belonging to set $\mathcal{X}_j$, where $j = \text{argmin}_i\{r_i\}$.
5: **end for**
6: Recompute the centroids $p_i \leftarrow \mathit{centroid}(\mathcal{X}_i)$.
7: If the centroids did not change position (or a maximum number of iteration reached), return.

As shown in figure 7.2, the algorithm works as expected, not necessarily well, due to known limitations of the algorithm itself. All standard modifications available from the $\mathbb{R}^n$ case are still applicable (such as incrementally introducing new classes where the variance of the data is high). The 2 dimensional sphere and the 3 dimensional symmetric positive definite matrixes were chosen because they can be visualized in low dimensions. Tests on the other manifolds indicate that it works as expected as well.

7.4 MAP Estimator

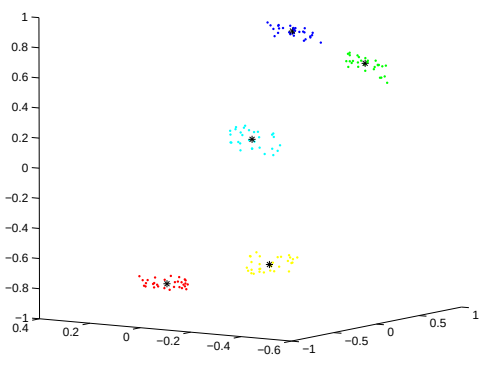
A slightly different application, albeit a simple one, comes from the field of robot navigation. Simple as it may be it is still the application where Newton's convergence rate is better evidenced when compared with the gradient method.

Consider a freely moving robot in $\mathbb{R}^n$ whose position is represented as a point $T \in \mathbb{SE}(n)$, seen as the rigid transformation that transforms points in the world referential, taking them to the local (robot) referential. Keeping the experiment simple, consider that the robot observes several known landmarks in the world $\{x_1, ..., x_k\} \in \mathbb{R}^n$. Hence, in the local referential, the robot observes the points $T\,x_i$. If the robot is considered to be at T_0 with a certain uncertainty, it is possible to build a prior knowledge probability density function as:

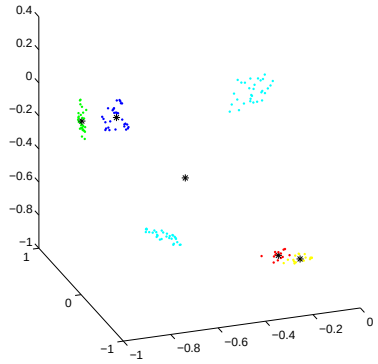
$$p(T) = k_1 \exp^{-\frac{1}{2}r^2(T,T_0)/\sigma^2}$$

where k_1 is a normalizing constant and σ^2 describes an isotropic level of uncertainty. Notice that all directions are treated equally which is usually not the case. Please note that by the identity $\mathbb{SE}(n) = \mathbb{SO}(n) \times \mathbb{R}^n$ a slightly more usefull prior may be built weighting differently translations from rotations. With simplicity in mind, assume that this description is usefull. Assume also that the robot's sensor is not perfect and the observations obey the following Gaussian probability distribution:

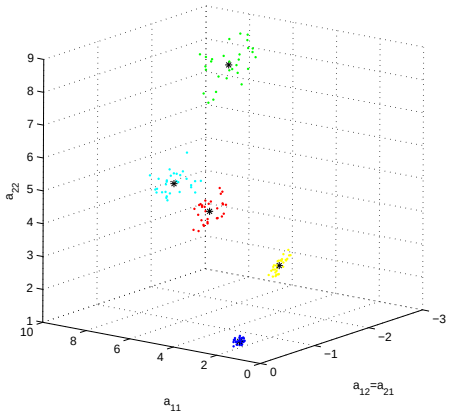
$$p(y_i|T) = k_2 \exp^{-(y_i - T\,x_i)^T R^{-1} (y_i - T\,x_i)}$$



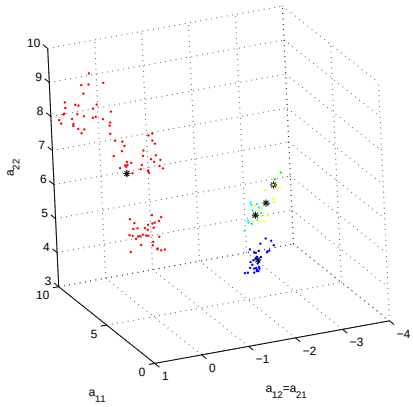
(a) $\mathbb{S}^2$



(b) $\mathbb{S}^2$



(c) $Sym^+(2)$



(d) $Sym^+(2)$

Figure 7.2: Results of running the k-means algorithm on $\mathbb{S}^2$ ((a) and (b)) and on $Sym^+(2)$ ((c) and (d)). There were 5 clouds each with 30 points generated with $\epsilon = 0.1$. In the results on the left ((a) and (c)), the algorithm succeeded in classifying the classes, but the k-means has many known limitations even in $\mathbb{R}^n$ and on the right ((b) and (d)) failed to correctly classify the classes. The sphere is represented as its usual embedding in $\mathbb{R}^3$ while the symmetric matrixes are represented by the three independent entries (the axis are labeled accordingly).

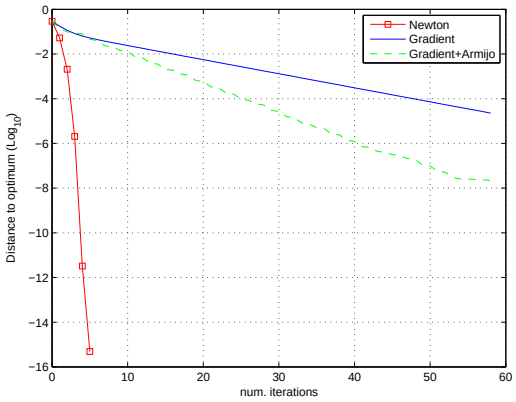


Figure 7.3: MAP estimate results using 5 observations $\sigma = 1$ and $R = I$.

where, again k_2 is a normalizing constant and R is a matrix encoding the uncertainty of the sensor. If the observations are considered to be independent, the MAP estimator of the robot's position is given by

$$\begin{aligned} T^* &= \arg \max_{T \in \mathbb{SE}(n)} p(T|y_1, y_2, ..., y_k) \\ &= \arg \max_{T \in \mathbb{SE}(n)} \left(\prod_{i=1}^{k} p(y_i|T) \right) p(T) \end{aligned}$$

Using the usual trick of applying the logarithm and discarding constants, the former problem is equivalent to

$$\begin{aligned} T^* = \arg \max_{T \in \mathbb{SE}(n)} \sum_{i=1}^{k} & -(y_i - Tx_i)^T R^{-1} (y_i - Tx_i) \\ & - \frac{1}{2} d(T, T_0)^2 / \sigma^2 \end{aligned}$$

This is formulated as an optimization problem on $\mathbb{SE}(n)$. The gradient of each term is readily available and the Hessian of the first terms can be obtained using standard techniques (see the chapter of Riemannian Embeddings on any Riemannian geometry book, specifically the part about the second fundamental form). The result presented in this paper allows for the Hessian of the last term to be obtained as well, thus allowing for a Newton algorithm to be implemented. Figure 7.3 shows the results obtained in an experiment using 5 observations. The gradient method is clearly outperformed by the 5 iterations taken by the Newton method to attain the required precision.

7.5 Conclusions

This article describes a simple algorithm to obtain the Hessian of the intrinsic squared distance function on connected locally-symmetric manifolds on which it is known how to compute basic Riemannian differential operations. Results are presented for centroid computation on the commonly used manifolds $\mathbb{SO}(n)$, $Sym^+(n)$, $\mathbb{S}^n$, $\mathbb{SE}(n)$, $\mathbb{G}(n,p)$, and $\mathbb{P}^n$. This is by no means an exhaustive list, and the result is valid for other manifolds fitting the requisites (fer example the hyperbolic plane). Besides the main application, simple examples of MAP estimation and k-means clustering are also provided, extending the range of applications besides centroid computation.

Appendix A

Proof of the theorem

Theorem A.0.1. *Consider M to be a connected locally-symmetric n-dimensional Riemannian manifold with curvature endomorphism R. Let $B_\epsilon(p)$ be a geodesic ball centered at $p \in M$ and $r_p : B_\epsilon(p) \to \mathbb{R}$ the function returning the intrinsic (geodesic) distance to p. Let $\gamma : [0, r] \to B_\epsilon(p)$ denote the unit speed geodesic connecting p to a point $q \in B_\epsilon(p)$, where $r = r_p(q)$, and let $\dot\gamma_q \equiv \dot\gamma(r)$ be its velocity vector at q. Define the function $k_p : B_\epsilon(p) \to \mathbb{R}$, $k_p(x) = \frac{1}{2} r_p(x)^2$ and consider any $X_q, Y_q \in T_qM$. Then*

$$\mathrm{Hess}(k_p)_q(X_q, Y_q) = \left\langle {X_q}^{\parallel}, Y_q \right\rangle + \sum_{i=1}^{n} \mathrm{ctg}_{\lambda_i}(r) \left\langle {X_q}^{\perp}, E_{iq} \right\rangle \langle Y_q, E_{iq} \rangle \ . \quad \text{(A.1)}$$

where $\{E_{iq}\} \subset T_qM$ is an orthonormal basis which diagonalizes the linear operator $\mathcal{R} : T_qM \to T_qM$, $\mathcal{R}(X_q) = R(X_q, \dot\gamma_q)\dot\gamma_q$ with eigenvalues λ_i, this means $\mathcal{R}(E_{iq}) = \lambda_i E_{iq}$. Also,

$$\mathrm{ctg}_\lambda(t) = \begin{cases} \sqrt{\lambda}\, t / \tan(\sqrt{\lambda}\, t) & \lambda > 0 \\ 1 & \lambda = 0 \\ \sqrt{-\lambda}\, t / \tanh(\sqrt{-\lambda}\, t) & \lambda < 0 \end{cases} .$$

Here the $\parallel$ and $\perp$ signs denote parallel and orthogonal components of the vector with respect to the velocity vector of γ, i.e. $X_q = {X_q}^{\parallel} + {X_q}^{\perp}$, $\left\langle {X_q}^{\perp}, {X_q}^{\parallel} \right\rangle = 0$, and $\left\langle {X_q}^{\perp}, \dot\gamma(r) \right\rangle = 0$.

This is the main result presented and the rest of this section is devoted to its proof. With the intent of finding the Hessian of the function k_p, recall that (see for example [13])

$$\mathrm{Hess}\, k_p(q)(X_q, Y_q) = \left\langle (\nabla_X \,\mathrm{grad}\, k_p)_q \,, Y_q \right\rangle \quad \text{(A.2)}$$

where X is any local extension of X_q. Note that from the properties of a connection, its value depends only of X at q, but for the expression to be

formally correct the extension X must be considered. Knowing that the gradient operator is linear and that for any two smooth functions $f,\ g : U \to \mathbb{R}$ defined on an open set $U \subset M$ satisfies the point-wise multiplication property

$$\operatorname{grad}(f\ g) = f \operatorname{grad} g + g \operatorname{grad} f\ ,$$

allows for the simplification

$$\operatorname{grad} k_p = \frac{1}{2} \operatorname{grad}(r_p\ r_p) = r_p \operatorname{grad} r_p\ .$$

Defining $\frac{\partial}{\partial\ r_p}$ as the unit normed radial vector field when written in normal coordinates centered at p, a corollary to Gauss's Lemma [17] states that $\operatorname{grad} r_p = \frac{\partial}{\partial\ r_p}$. Hence the former expression is written as

$$\operatorname{grad} k_p = r_p \frac{\partial}{\partial\ r_p}\ .$$

Gauss's Lemma also allows for the decomposition of any vector field $X \in TU$ as $X = X^{\perp} + X^{\parallel}$, where $X^{\parallel} \in TU$ is a vector field parallel to $\frac{\partial}{\partial\ r_p}$ and $X^{\perp} \in TU$ is orthogonal to it. These statements, along with the properties of a connection, are used to write

$$\begin{aligned}
\nabla_X \operatorname{grad} k_p &= \nabla_X \left(r_p \frac{\partial}{\partial\ r_p} \right) \\
&= X\,(r_p) \frac{\partial}{\partial\ r_p} + r_p \nabla_{(X^{\parallel} + X^{\perp})} \left(\frac{\partial}{\partial\ r_p} \right) \\
&= X\,(r_p) \frac{\partial}{\partial\ r_p} + r_p \nabla_{X^{\parallel}} \left(\frac{\partial}{\partial\ r_p} \right) \\
&\qquad + r_p \nabla_{X^{\perp}} \left(\frac{\partial}{\partial\ r_p} \right)\ .
\end{aligned}$$

Noting that for any vector field X:

$$X(r_p) = \mathrm{d}\, r_p(X) = \langle \operatorname{grad} r_p, X \rangle = \left\langle \frac{\partial}{\partial\ r_p}, X \right\rangle$$

and since $X^{\parallel}$ is parallel to $\frac{\partial}{\partial\ r_p}$, there is a smooth function $f : U \to \mathbb{R}$ such that $X^{\parallel} = f \frac{\partial}{\partial\ r_p}$. Since $\frac{\partial}{\partial\ r_p}$ is tangent to unit speed geodesics emanating from p, $\nabla_{\frac{\partial}{\partial\ r_p}} \frac{\partial}{\partial\ r_p} = 0$. Hence

$$\begin{aligned}
\nabla_X \operatorname{grad} k_p &= \left\langle \frac{\partial}{\partial\ r_p}, X \right\rangle \frac{\partial}{\partial\ r_p} + r_p f\ \nabla_{\frac{\partial}{\partial\ r_p}} \left(\frac{\partial}{\partial\ r_p} \right) \\
&\qquad + r_p \nabla_{X^{\perp}} \left(\frac{\partial}{\partial\ r_p} \right) \\
&= X^{\parallel} + r_p \nabla_{X^{\perp}} \left(\frac{\partial}{\partial\ r_p} \right) \qquad \text{(A.3)}
\end{aligned}$$

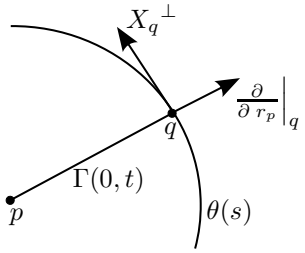


Figure A.1: Construction of the Jacobi field.

Now let $\theta : \mathbb{R} \to S_r \subset M$ be a curve in the geodesic sphere $S_r = \{s \in M : r_p(s) = r = r_p(q)\}$ with $\theta(0) = q$ and $\dot{\theta}(0) = X_q{}^{\perp}$. In normal coordinates centered at p, consider the geodesic variation $\hat{\Gamma} :]-\delta, \delta[\times[0, r]$ for some δ, given by $\hat{\Gamma}(s,t) = \frac{t}{r}\hat{\theta}(s)$. Here the hat notation denotes a coordinate representation, hence if $\phi : M \to \mathbb{R}^n$ is the normal coordinate function:

$$\begin{aligned}\hat{p} &= \phi(p)\\ \hat{\theta} &= \phi \circ \theta\\ \hat{r}_{\hat{p}} &= r_p \circ \phi^{-1}\end{aligned}$$

Defining $\hat{T}(s,t) = \frac{\partial\hat{\Gamma}}{\partial t}(s,t)$ and $\hat{S}(s,t) = \frac{\partial\hat{\Gamma}}{\partial s}(s,t)$, the corresponding Jacobi field $J : M \to TM$ (see [17] for an introduction to Jacobi fields) is given in coordinates by $\hat{J}(t) = \hat{S}(0,t) = \frac{t}{r}\hat{X}^{\perp}_{\hat{q}}$. Note that $\hat{J}$ is normal to the unit-speed geodesic $\hat{\gamma}(t) = \hat{\Gamma}(0,t) = \frac{t}{r}\hat{q}$ and that $\hat{J}(r) = \hat{X}^{\perp}_{\hat{q}}$. Also, notice that $\hat{T}(s,t) = \phi_*\left(\frac{\partial}{\partial\, r_p}\right)\Big|_{\hat{\Gamma}(s,t)}$ since $\hat{\Gamma}(0,t)$ is a geodesic. In order to ease notation the coordinate representation for these objects will be hidden, although not forgotten. Hence, at q:

$$\nabla_{X^{\perp}} \frac{\partial}{\partial\, r_p}\Big|_q = D_s T(0,r) = D_t S(0,r) = D_t J(r)$$

where the fact that $D_s T = D_t S$ is used (see [17], specifically Lemma 6.3). Substituting in equation A.3, again at q, results in

$$\begin{aligned}(\nabla_X \operatorname{grad} k_p)|_q &= X_q{}^{\parallel} + r\, \nabla_{X^{\perp}} \left(\frac{\partial}{\partial\, r_p}\right)\Big|_q\\ &= X_q{}^{\parallel} + r D_t J(r)\end{aligned}$$

Substituting back into equation A.2, yields

$$\begin{aligned}\operatorname{Hess} k_p(q)(X_q, Y_q) &= \left\langle X_q{}^{\parallel} + r D_t J(r), Y_q \right\rangle\\ &= \left\langle X_q{}^{\parallel}, Y_q \right\rangle + r \left\langle D_t J(r), Y_q \right\rangle \qquad \text{(A.4)}\end{aligned}$$

All that remains to do is to find an expression for the Jacobi field and take its covariant derivative. This leads to a rather lengthy discussion so it is stated here as a couple of lemmas:

Lemma A.0.2. *The solution of the ODE*

$$\ddot{u} + ku = 0, \quad u(0) = a, \quad \dot{u}(0) = b$$

is given by

$$u(t) = a\,\mathrm{c}_k(t) + b\,\mathrm{s}_k(t)$$

where

$$\mathrm{c}_k(t) = \begin{cases} \cos(\sqrt{k}\,t) & k > 0 \\ 1 & k = 0 \\ \cosh(\sqrt{-k}\,t) & k < 0 \end{cases}, \; \mathrm{s}_k(t) = \begin{cases} \frac{1}{\sqrt{k}}\sin(\sqrt{k}\,t) & k > 0 \\ t & k = 0 \\ \frac{1}{\sqrt{-k}}\sinh(\sqrt{-k}\,t) & k < 0 \end{cases}$$

Proof. Picard's existence theorem guarantees uniqueness and direct substitution of the result in the differential equation proves the result. □

Lemma A.0.3. *Let M be a locally-symmetric Riemannian manifold and $B_\epsilon(p)$ be a geodesic ball centered at $p \in M$. If $r = r_p(q)$ is the intrinsic distance of a point $q \in B_\epsilon(p)$ to p and $\gamma : [0, r] \to B_\epsilon(p)$ is the unit speed radial geodesic running from p to q, given a tangent vector $V_q \in T_qM$ orthogonal to $\dot{\gamma}(r)$, then the normal Jacobi field $J : [0, r] \to TM$ along γ which satisfies $J(0) = 0$ and $J(r) = V_q$ is given by*

$$J(t) = \sum_{i=1}^{n} \left[\frac{\langle V_q, E_i(r) \rangle}{\mathrm{s}_{\lambda_i}(r)}\, \mathrm{s}_{\lambda_i}(t) \right] E_i(t) \tag{A.5}$$

where $E_i(t) \in TM$ is the parallel transport along γ of the tangent space's orthonormal basis $\{E_{iq}\} \subset T_qM$ which diagonalizes the linear operator $\mathcal{R} : T_qM \to T_qM$ defined as $\mathcal{R}(X_q) = R(X_q, \dot{\gamma}(0)) \cdot \dot{\gamma}(0)$, i.e., $\mathcal{R}(E_{iq}) = \lambda_i E_{iq}$. Note that R denotes the curvature tensor of M and s_{λ_i} is defined as in the previous lemma.

Proof. As stated in [17] the Jacobi field J specified by its two endpoints $J(0) = 0$ and $J(r) = V_q$ exists and is unique as long as q is not conjugate to p along γ. Consider the Jacobi equation

$$\ddot{J}(t) + R(J(t), \dot{\gamma}(t)) \cdot \dot{\gamma}(t) = 0 \tag{A.6}$$

Choosing an orthonormal basis for the tangent space $\{F_{iq}\} \subset T_qM$ and creating the vector fields $F_i \in TM$ along γ by parallel translation of F_{iq}, it is possible to write $J(t)$ with respect to this basis (note that the set $\{F_{i\gamma(t)}\} \subset T_{\gamma(t)}M$ is a basis for $T_{\gamma(t)}M$) as:

$$J(t) = \sum_{i=1}^{n} J^i(t) F_i(t)$$

where $J^i : [0, r] \to \mathbb{R}$. Hence the left hand side of equation A.6 can be written as

$$\begin{aligned}
&\ddot{J}(t) + R(J(t), \dot{\gamma}(t)) \cdot \dot{\gamma}(t) \\
&\quad = \sum_{i=1}^{n} \ddot{J}^i(t) F_i(t) + R(J^i(t) F_i(t), \dot{\gamma}(t)) \cdot \dot{\gamma}(t) \\
&\quad = \sum_{i=1}^{n} \ddot{J}^i(t) F_i(t) + J^i(t) R(F_i(t), \dot{\gamma}(t)) \cdot \dot{\gamma}(t) \qquad \text{(A.7)}
\end{aligned}$$

where the identity $\ddot{J}(t) = \sum_{i=1}^{n} \ddot{J}^i(t) F_i(t)$ is used, which follows from the fact that the vector fields $F_i(t)$ are parallel. The goal is now to solve this ordinary differential equation. Start by defining the linear operator $\mathcal{R} : T_q M \to T_q M$ as $\mathcal{R}(X_q) = R(X_q, \dot{\gamma}(0)) \cdot \dot{\gamma}(0)$ and note that due to the symmetries of the Riemannian curvature tensor, this operator is self-adjoint, i.e. $\langle \mathcal{R}(X_q), Y_q \rangle = \langle X_q, \mathcal{R}(Y_q) \rangle$. This guarantees that there is an orthonormal basis $E_{iq} \in T_q M$ such that $\mathcal{R}(E_{iq}) = \lambda_i E_{iq}$ for some $\lambda_i \in \mathbb{R}$ as described next. Write $\mathcal{R}(F_{iq})$ as a linear combination of the basis $\{F_{iq}\}$ as follows:

$$\mathcal{R}(F_{iq}) = \sum_{j=1}^{n} \left\langle F_{jq}, \mathcal{R}(F_{iq}) \right\rangle F_{jq}$$

Since any two vector spaces with the same dimension are isomorphic there is an isomorphism, $\phi : T_p M \to \mathbb{R}^n$ taking F_{iq} to $\hat{F}_i \in \mathbb{R}^n$. In this vector space the operator $\mathcal{R}$ may be written as $\hat{\mathcal{R}} = \phi \circ \mathcal{R} \circ \phi^{-1}$. Hence, writing in matrix notation by defining $\hat{F} = \begin{bmatrix} \hat{F}_1 & \hat{F}_2 & \dots & \hat{F}_n \end{bmatrix}$ and since $\mathcal{R}$ is linear in a finite dimensional vector field, $\hat{\mathcal{R}}$ can be described as a matrix. Hence,

$$\hat{\mathcal{R}}\hat{F} = \hat{F} \underbrace{\begin{bmatrix} \left\langle F_{1q}, \mathcal{R}(F_{1q}) \right\rangle & \left\langle F_{1q}, \mathcal{R}(F_{2q}) \right\rangle & \dots & \left\langle F_{1q}, \mathcal{R}(F_{nq}) \right\rangle \\ \left\langle F_{2q}, \mathcal{R}(F_{1q}) \right\rangle & \left\langle F_{2q}, \mathcal{R}(F_{2q}) \right\rangle & \dots & \left\langle F_{2q}, \mathcal{R}(F_{nq}) \right\rangle \\ \vdots & \vdots & \ddots & \vdots \\ \left\langle F_{nq}, \mathcal{R}(F_{1q}) \right\rangle & \left\langle F_{nq}, \mathcal{R}(F_{2q}) \right\rangle & \dots & \left\langle F_{nq}, \mathcal{R}(F_{nq}) \right\rangle \end{bmatrix}}_{A}.$$

The fact that the operator is self-adjoint makes A a symmetric matrix and as such, it admits an eigenvalue decomposition

$$A = UDU^T$$

where $D = \operatorname{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ is a diagonal matrix such that λ_i are the eigenvalues of A and U is an orthogonal matrix with the normalized eigenvectors of A as its columns. Hence

$$\begin{aligned}
\hat{\mathcal{R}}\hat{F} &= \hat{F}UDU^T \\
\iff \hat{\mathcal{R}} \underbrace{\hat{F}U}_{\hat{E}} &= \underbrace{\hat{F}U}_{\hat{E}} D
\end{aligned}$$

hence, if $U = [u_{ij}]$ are the entries of the matrix, $\hat{E}_j = \sum_{i=1}^{n} u_{ij}\hat{F}_i$ is the orthonormal basis such that

$$\hat{\mathcal{R}}\hat{E}_j = \lambda_j \hat{E}_j$$

Using the isomorphism once again, this means that $E_{j_q} = \sum_{i=1}^{n} u_{ij} F_{i_q}$ is the basis that diagonalizes the operator $\mathcal{R}$:

$$\mathcal{R}(E_{i_q}) = \lambda_i E_{i_q}$$

Define $E_i(t)$ as the parallel transport of E_{i_q} along γ. Define as well the vector fields

$$R_i(t) = R(E_i(t), \dot{\gamma}(t)) \cdot \dot{\gamma}(t)$$

which, since M is locally symmetric, are parallel along $\gamma(t)$ [13]. It follows then, using the fact that parallel transport preserves inner products and $\{E_{i_q}\}$ is an orthonormal basis

$$\begin{aligned}
R_i(t) &= \sum_{k=1}^{n} \langle R_i(t), E_k(t)\rangle \, E_k(t) \\
&= \sum_{k=1}^{n} \langle R_i(r), E_k(r)\rangle \, E_k(t) \\
&= \sum_{k=1}^{n} \left\langle \mathcal{R}(E_{i_q}), E_{k_q} \right\rangle E_k(t) \\
&= \lambda_i \sum_{k=1}^{n} \left\langle E_{i_q}, E_{k_q} \right\rangle E_k(t) \\
&= \lambda_i E_i(t)
\end{aligned}$$

Writing equation A.7 in terms of the new basis:

$$\begin{aligned}
&\ddot{J}(t) + R(J(t), \dot{\gamma}(t)) \cdot \dot{\gamma}(t) \\
&\quad = \sum_{i=1}^{n} \ddot{J}^i(t) E_i(t) + J^i(t) \underbrace{R(E_i(t), \dot{\gamma}(t)) \cdot \dot{\gamma}(t)}_{R_i(t)} \\
&\quad = \sum_{i=1}^{n} \left(\ddot{J}^i(t) + \lambda_i J^i(t) \right) \, E_i(t)
\end{aligned}$$

Hence, the Jacobi equation decouples into n scalar differential equations with the corresponding two-point boundary conditions as stated below:

$$\begin{cases}
\ddot{J}^1(t) + \lambda_1 J^1(t) = 0, & J^1(0) = 0 \text{ and } J^1(r) = \langle V_q, E_1(r)\rangle \\
\ddot{J}^2(t) + \lambda_1 J^2(t) = 0, & J^2(0) = 0 \text{ and } J^2(r) = \langle V_q, E_2(r)\rangle \\
& \vdots \\
\ddot{J}^n(t) + \lambda_1 J^n(t) = 0, & J^n(0) = 0 \text{ and } J^n(r) = \langle V_q, E_n(r)\rangle
\end{cases}$$

Invoking lemma A.0.2, the solution of the ith equation is given by

$$J^i(t) = \frac{\langle V_q, E_i(r)\rangle}{\mathrm{s}_{\lambda_i}(r)}\,\mathrm{s}_{\lambda_i}(t)$$

Hence

$$J(t) = \sum_{i=1}^{n}\left[\frac{\langle V_q, E_i(r)\rangle}{\mathrm{s}_{\lambda_i}(r)}\,\mathrm{s}_{\lambda_i}(t)\right]E_i(t)$$

□

Now that an expression for $J(t)$ has been found, it can be substituted in equation A.4, by making $V_q = {X_q}^{\perp}$. Taking the covariant derivative of $J(t)$ evaluated at $t = r$, considering that $D_t E_i(t) = 0$ since $E_i(t)$ is parallel along the geodesic, results in

$$D_t J(r) = \sum_{i=1}^{n}\left[\frac{\left\langle {X_q}^{\perp}, E_i(r)\right\rangle}{\mathrm{s}_{\lambda_i}(r)}\,\mathrm{c}_{\lambda_i}(r)\right]E_i(r)$$

Hence, defining by pointwise division $\mathrm{ctg}_{\lambda_i}(r) = r\frac{\mathrm{c}_{\lambda_i}(r)}{\mathrm{s}_{\lambda_i}(r)}$,

$$\begin{aligned}\operatorname{Hess} k_p(q)(X_q, Y_q) &= \left\langle {X_q}^{\parallel}, Y_q\right\rangle + r\,\langle D_t J(r), Y_q\rangle \\ &= \left\langle {X_q}^{\parallel}, Y_q\right\rangle \\ &\quad + \sum_{i=1}^{n}\mathrm{ctg}_{\lambda_i}(r)\left\langle {X_q}^{\perp}, E_i(r)\right\rangle\langle Y_q, E_i(r)\rangle\end{aligned}$$

A.1 Matrix Form

Although equation A.1 provides a way to compute the Hessian, it is not very implementation-friendly. This section re-writes the equation in matrix form once a tangent basis $\{F_{iq}\} \subset T_qM$ is fixed.

Theorem A.1.1. *Under the same assumptions as above, consider $\{F_{iq}\} \subset T_qM$ an orthonormal basis. If $X_q \in T_qM$ is a vector, let the notation $\hat{X}$ denote the column vector describing the decomposition of X_q with respect to the basis $\{F_{iq}\}$, i.e. $[\hat{X}]_i = \langle X_q, F_{iq}\rangle$, let R_k be the matrix with entries $[R_k]_{ij} = \left\langle F_{iq}, \mathcal{R}(F_{jq})\right\rangle$ and consider the eigenvalue decomposition $R_k = E\Lambda E^T$. Here λ_i will be used to describe the i'th diagonal element of Λ. Then the Hessian matrix (a representation for the bilinear Hessian tensor on the finite dimensional tangent space with respect to the fixed basis) is given by:*

$$H_{k_p} = E\Sigma E^T \tag{A.8}$$

where Σ is diagonal with elements σ_i given by $\sigma_i = \mathrm{ctg}_{\lambda_i}(r)$. Hence

$$\mathrm{Hess}(k_p)_q(X_q, Y_q) = \hat{X}^T H_{k_p}\hat{Y}\ .$$

From the symmetries of the curvature tensor follows that any vector parallel to $\dot{\gamma}(r)$ (for example ${X_q}^{\parallel}$) belongs to the kernel of the operator $\mathcal{R}$. Without loss of generality assume that $E_1(r)$ is parallel to $\dot{\gamma}(r)$ (hence λ_1=0, ${X_q}^{\parallel} = \langle X_q, E_1(r)\rangle E_1(r)$ and ${X_q}^{\perp} = \sum_{j=2}^{n} \langle X_q, E_j(r)\rangle E_j(r)$. Then, since $\mathrm{ctg}_0(r) = 1$, equation A.1 can be re-written:

$$
\begin{aligned}
&\operatorname{Hess} k_p(q)(X_q, Y_q) \\
&\quad = \mathrm{ctg}_{\lambda_1}(r) \left\langle \langle X_q, E_1(r)\rangle E_1(r), \sum_{j=1}^{n} \langle Y_q, E_j(r)\rangle E_j(r) \right\rangle \\
&\qquad + \sum_{i=1}^{n} \mathrm{ctg}_{\lambda_i}(r) \left\langle \sum_{j=2}^{n} \langle X_q, E_j(r)\rangle E_j(r), E_i(r) \right\rangle \langle Y_q, E_i(r)\rangle \\
&\quad = \mathrm{ctg}_{\lambda_1}(r) \langle X_q, E_1(r)\rangle \langle Y_q, E_1(r)\rangle \\
&\qquad + \sum_{i=2}^{n} \mathrm{ctg}_{\lambda_i}(r) \langle X_q, E_i(r)\rangle \langle Y_q, E_i(r)\rangle \\
&\quad = \sum_{i=1}^{n} \mathrm{ctg}_{\lambda_i}(r) \langle X_q, E_i(r)\rangle \langle Y_q, E_i(r)\rangle \\
&\quad = \hat{X}^T H_{k_p} \hat{Y}
\end{aligned}
$$

where the matrices are defined in the theorem statement.

Bibliography

[1] D. Gabay, "Minimizing a differentiable function over a differential manifold," *J. Optim. Theory Appl.*, vol. 37(2), pp. 177–219, 1982.

[2] Alan Edelman, Tomas A. Arias, and Steven T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 20, no. 2, pp. 303–353, 1998.

[3] Jonathan H. Manton, "Optimisation algorithms exploiting unitary constraints," *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 635–650, March 2002.

[4] Calin Belta and Vijay Kumar, "An svd-based projection method for interpolation on SE(3)," *IEEE transactions on Robotics and Automation*, vol. 18(3), pp. 334–345, June 2002.

[5] Jonathan H. Manton, "A centroid (Karcher mean) approach to the joint approximate diagonalization problem: The real symmetric case," *Digital Signal Processing*, 2005.

[6] Uwe Helmke, Knut Hüper, Pei Lee, and John Moore, "Essential matrix estimation via Newton-type methods," *16th International Symposium on Mathematical Theory of Network and System (MTNS), Leuven*, 2004.

[7] Maher Moakher, "Means and averaging in the group of rotations," *SIAM J. Matrix Anal. Appl.*, vol. 24, no. 1, pp. 1–16, 2002.

[8] Jonathan H. Manton, "A globally convergent numerical algorithm for computing the centre of mass on compact Lie groups," in *Eighth International Conference on Control, Automation, Robotics and Vision*, Kunming, China, December 2004.

[9] Xavier Pennec, Pierre Fillard, and Nicholas Ayache, "A Riemannian framework for tensor computing," Research Report 5255, INRIA, July 2004, To appear to the Int. Journal of Computer Vision.

[10] P. Fletcher, C., Lu S.Pizer, and S. Joshi, "Principal geodesic analysis for the study of nonlinear statistics of shape," *IEEE Transactions on Medical Imaging*, vol. 23(8), pp. 995–1005, Aug 2004.

[11] K. Hüper and J. Manton, "The Karcher mean of points on the orthogonal group," *Preprint*, 2005.

[12] P. Absil, R. Mahony, and R. Sepulchre, "Riemannian geometry of grassmann manifolds with a view on algorithmic computation," *Acta Appl. Math*, vol. 80(2), pp. 199–220, Jan 2004.

[13] Barret O'Neil, *Semi-Riemannian Geometry*, Academic Press, 1983.

[14] Andreas Arvanitoyeorgos, *An Introduction to Lie Groups and the Geometry of Homogeneous Spaces*, American Mathematical Society, 1999.

[15] R. Ferreira, J. Xavier, J. Costeira, and V. Barroso, "Newton method for Riemannian centroid computation in naturally reductive homogeneous spaces," *International Conference on Acoustics, Speech, and Signal Processing ICASSP'06*, May 2006.

[16] Ricardo Ferreira and João Xavier, "Hessian of the Riemannian squared distance function on connected locally symmetric spaces with applications," *Controlo 2006*, 2006.

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

[18] W. M. Boothby, *An Introduction to Differentiable Manifolds and Riemannian Geometry*, Academic Press, 1975.

[19] Manfredo P. do Carmo, *Riemannian Geometry*, Birkhäuser, Boston, MA, 1992.

[20] Stephen Boyd and Lieven Vandenberghe, *Convex Optimization*, Cambridge University Press, New York, NY, USA, 2004.

[21] Dimitri P. Bertsekas, *Nonlinear Programming: 2nd Edition*, Athena Scientific, 1999.

[22] Knut Hüper and Jochen Trumpf, "Newton-like methods for numerical optimization on manifolds," *Asilomar Conference on Signals, Systems and Computers*, pp. 136–139, 2004.

[23] Ilka Agricola, "Connections on naturally reductive spaces, their dirac operator and homogeneous models in string theory," *Communications in Mathematical Physics*, , no. 232, pp. 535–563, 2003.

[24] H. Karcher, "Riemannian center of mass and mollifier smoothing," *Comm. Pure Appl. Math.*, vol. 30, pp. 509–541, 1977.