1-1-2022

Finding geodesics joining given points

Lyle Noakes

Erchuan Zhang
Edith Cowan University, erchuan.zhang@ecu.edu.au

10.1007/s10444-022-09966-y
Noakes, L., & Zhang, E. (2022). Finding geodesics joining given points. Advances in Computational Mathematics, 48, 50, 1-27. https://doi.org/10.1007/s10444-022-09966-y

Advances in Computational Mathematics (2022) 48: 50
https://doi.org/10.1007/s10444-022-09966-y

Finding geodesics joining given points

Lyle Noakes[1] · **Erchuan Zhang**[2]

Received: 30 July 2021 / Accepted: 17 June 2022 / Published online: 27 July 2022

Abstract
Finding a geodesic joining two given points in a complete path-connected Riemannian manifold requires much more effort than determining a geodesic from initial data. This is because it is much harder to solve boundary value problems than initial value problems. Shooting methods attempt to solve boundary value problems by solving a sequence of initial value problems, and usually need a good initial guess to succeed. The present paper finds a geodesic $\gamma : [0,1] \to M$ on the Riemannian manifold M with $\gamma(0) = x_0$ and $\gamma(1) = x_1$ by dividing the interval [0,1] into several sub-intervals, preferably just enough to enable a good initial guess for the boundary value problem on each subinterval. Then a geodesic joining consecutive endpoints (local junctions) is found by single shooting. Our algorithm then adjusts the junctions, either (1) by minimizing the total squared norm of the differences between associated geodesic velocities using Riemannian gradient descent, or (2) by solving a nonlinear system of equations using Newton's method. Our algorithm is compared with the known *leapfrog algorithm* by numerical experiments on a 2-dimensional ellipsoid Ell(2) and on a left-invariant 3-dimensional special orthogonal group $SO(3)$. We find Newton's method (2) converges much faster than leapfrog when more junctions are needed, and that a good initial guess can be found for (2) by starting with Riemannian gradient descent method (1).

Keywords Geodesic · Shooting · Leapfrog · Riemannian gradient descent · Newton's method · Jacobi equation

Mathematics Subject Classification (2010) MSC2020: 34B60 · 49M15

Communicated by: Thanh Tran

This article belongs to the Topical Collection: *Mathematics of Computation and Optimisation* Guest Editors: Jerome Droniou, Andrew Eberhard, Guoyin Li, Russell Luke, Thanh Tran

Lyle Noakes and Erchuan Zhang contributed equally to this work.

✉ Erchuan Zhang
erchuan.zhang@ecu.edu.au

Extended author information available on the last page of the article

1 Introduction

Geodesics on Riemannian manifolds are generalizations of straight lines in Euclidean space, and are fundamental in many areas of mathematics, engineering, and computer science, to name a few. For instance, *geodesic regression* is used to relate a real-valued independent variable and a manifold-valued dependent data variable [1, 2]. Another example is an extension of principal component analysis namely *principal geodesic analysis*, which is used to study variability of data on a Riemannian manifold [3, 4]. A third example is the detection of object boundaries, where *geodesic active contours* are used to explore relationships between active contours and geodesics via curve evolution theory [5].

Let M be a p-dimensional ($1 \leqslant p < \infty$) smooth path-connected manifold with a Riemannian metric g. When M is complete with respect to the Riemannian metric, the Hopf-Rinow theorem says that any two points x_0,$x_1 \in M$ can be joined by a minimal geodesic, namely, a constant-speed curve $\gamma : [0,1] \to M$ of minimal length with $\gamma(0) = x_0$ and $\gamma(1) = x_1$. For some well-studied examples of Riemannian manifolds, geodesics can be found in closed form, but such cases are rare and in practice numerical methods are usually needed. Even in cases where closed form expressions for geodesics are known, it may be difficult to use these expressions to find geodesics joining two given points. We refer to [31–33] for background in differential geometry.

By the fundamental theorem of Riemannian geometry, every Riemannian manifold (M,g) admits a unique Levi-Civita connection ∇, which relates to g by the Koszul formula [6],

$$\begin{aligned} 2g(\nabla_X Y, Z) =& X(g(Y,Z)) + Y(g(Z,X)) - Z(g(X,Y)) \\ &+ g([X,Y],Z) - g([Y,Z],X) - g([X,Z],Y), \end{aligned} \tag{1}$$

where X,Y,Z are vector fields on M and $[\cdot,\cdot]$ is the Lie bracket of vector fields. It turns out that a geodesic joining two given endpoints is precisely a zero-acceleration curve with respect to the Levi-Civita connection ∇, namely the solution of the following 2-point boundary value problem for a system of ODEs

$$\nabla_{\dot\gamma(t)}\dot\gamma(t) = \mathbf{0}, \quad s.t.\ \gamma(0) = x_0, \gamma(1) = x_1. \tag{2}$$

For any system of ODEs it is usually much harder to solve a 2-point boundary value problem than an initial value problem. Solutions (often non-unique) to boundary value problems can sometimes be found using *shooting methods* [7]. In *single shooting* an unknown initial quantity is guessed, then improved using the terminal error of a solution to an initial value problem. In *multiple shooting*, multiple initial data are estimated on some range, and then improved. The quality of an initial guess is crucial to the performance of both single and multiple shooting.

In our Riemannian situation a good initial guess can be made when the endpoints x_0 and x_1 are reasonably close. In such cases, the 2-point boundary value problem (2) is solved efficiently using single shooting. In more general nonlocal cases, where x_0 and x_1 are distant, one possible strategy is to divide the interval [0,1] into small subintervals [t_{i-1},t_i] and update on each subinterval. In [8] the so-called *leapfrog*

algorithm[1] is used to find geodesics joining given points (the method has also been used for optimal control problems [9, 10]). The basic idea is to treat the local problem as effectively solved, then update the junctions $y_i := \gamma(t_i)$ using a minimal geodesic from y_{i-1} to y_{i+1}. Leapfrog always converges to a geodesic (Theorem 5.1 in [8]) and generally performs well in the first few iterations, but convergence can be slow if many iterations are needed. The present paper achieves fast convergence by using single shooting to update junctions in a different way, as follows.

Recall that for the first variation of the energy $\mathcal{E} := \frac{1}{2}\int_0^1 g(\dot\gamma,\dot\gamma)dt$, we have

$$\begin{aligned}\delta\mathcal{E} &= \sum_{i=0}^{n-1}\left(g(\dot\gamma,\delta\gamma)|_{t_i}^{t_{i+1}} - \int_{t_i}^{t_{i+1}} g(\nabla_t\dot\gamma,\delta\gamma)dt\right)\\ &= \sum_{i=0}^{n-1} g(\dot\gamma,\delta\gamma)|_{t_i}^{t_{i+1}}\end{aligned} \tag{3}$$

assuming that (for the second equality) each $\gamma|_{[t_i,t_{i+1}]}$ is a geodesic. So, for piecewise geodesic γ, $\delta\mathcal{E}$ vanishes for all variations precisely when all differences $\dot\gamma(t_i^+) - \dot\gamma(t_i^-)$ are zero for $1 \le i \le n-1$. This reduces our infinite-dimensional boundary value problem to the following finite dimensional problem.

Setting $t_i := i/n$, we denote by v_i^+ the *right-side velocity* tangent to M at y_i determined by the geodesic $\gamma_i : [0,1] \to M$ from y_i to y_{i+1}. Similarly v_i^- is the *left-side velocity* determined by the geodesic from y_{i-1} to y_i. We may write $v_i^+ = \log_{y_i} y_{i+1}$ and $v_i^- = -\log_{y_i} y_{i-1}$, found by solving (2) with boundary conditions $\gamma(0) = y_i, \gamma(1) = y_{i+1}$ and $\gamma(0) = y_{i-1}, \gamma(1) = y_i$ respectively. The numerical values of these Riemannian logarithms are usually not available from closed form expressions.

Then, for the corresponding piecewise geodesic γ we have $\dot\gamma(t_i^+) = nv_i^+$, $\dot\gamma(t_i^-) = nv_i^-$, and $\delta\mathcal{E}$ vanishes precisely when $(y_1, y_2, \ldots, y_{n-1})$ is a singularity of the vector field F on $M \times M \times \ldots \times M$ given locally by

$$F(y_1, y_2, \cdots, y_{n-1}) := \left(v_1^+ - v_1^-, v_2^+ - v_2^-, \cdots, v_{n-1}^+ - v_{n-1}^-\right). \tag{4}$$

Note that $(y_1, y_2, \ldots, y_{n-1})$ is also the global minimizer of $f(y_1, y_2, \ldots, y_n)$ for

$$f(y_1, y_2, \cdots, y_{n-1}) := \frac{1}{2}\sum_{i=1}^{n-1} \|v_i^+ - v_i^-\|^2 \tag{5}$$

whose minimum value is 0. In the present paper we exploit the fast convergence of Newton's method to find the singularity of the vector field (4). When a good initialization for Newton's method is needed, we use Riemannian gradient descent to minimize the cost function (5). Other variants of gradient descent method, such as Nesterov's accelerated gradient descent [11, 12] and accelerated higher-order gradient methods [13] might also be used to minimize (5).

The paper is organized as follows. In Section 2 we review the general framework, of finding a singularity of (4) by Newton's method and minimizing (5) by

[1] Refer to I for the algorithm.

Riemannian gradient descent. In Section 3 the derivatives of the Riemannian logarithms are related to solutions of the Jacobi equation. For a general Riemannian manifold M, the Jacobi equation is rewritten in terms of Christoffel symbols and local coordinates. Alternatively, when M is isometrically embedded in a Euclidean space of one dimension higher, the Jacobi equation is rewritten in terms of the embedding. If instead M is a left-invariant Lie group, we rewrite the Jacobi equation in terms of bilinear transformations of the Lie algebra. Numerical calculations of derivatives of Riemannian logarithms are given by Algorithms 4, 7 and 10. To demonstrate the effectiveness of our method, we describe numerical experiments on the 2-dimensional ellipsoid $Ell(2)$ using the first fundamental form of the standard Euclidean metric, and on the special orthogonal group $SO(3)$ with a left-invariant metric.

2 General framework for finding geodesics

There may be multiple geodesics joining points x_0,x_1 in a general Riemannian manifold M, and the geodesic found by our method may depend on an initial choice for the junctions y_1,y_2,…,$y_{n-1} \in M$. Consecutive junctions should be reasonably close, namely y_i,y_{i+1} should be joined by a unique minimal geodesic with an easy initial guess for single shooting. Because single shooting performs well with a good initial guess, we should not take n to be extremely large: consecutive junctions should be close enough to enable a good initial guess, but no closer.

2.1 Riemannian gradient descent for finding geodesics

To use Riemannian gradient descent we need to evaluate the Riemanian gradient of the cost function (5) with respect to all y_i's. Throughout this paper, we denote[2] $\partial_{y_i} f$, $\partial^R_{y_i} f$ the standard Euclidean gradient and the Riemannian gradient of f with respect to y_i respectively, which are related by

$$\langle \partial_{y_i} f, v\rangle_{\mathbb{E}} = \langle \partial^R_{y_i} f, v\rangle_{y_i}, \tag{6}$$

where $\langle\cdot,\cdot\rangle_{\mathbb{E}}$ is the Euclidean metric for a coordinate chart containing y_i, $\langle\cdot,\cdot\rangle_{y_i}$ is the Riemannian metric at y_i, and v is an arbitrary tangent vector at y_i. As usual when $g : M \to N$ is a smooth map, we denote its derivative by $dg : TM \to TN$, where TM and TN are the tangent bundles. The derivative at $x \in M$ is denoted by $d_x g : T_x M \to T_{g(x)} N$.

Given junctions y_1,y_2,…,y_{n-1} we set $y_0 := x_0$, $y_n := x_1$, $\mathbf{y} := (y_1,y_2,\cdots,y_{n-1})$, $v_0^- := \mathbf{0}$, $v_n^+ := \mathbf{0}$ and $v_i^+ := \log_{y_i} y_{i+1}$, $v_i^- := -\log_{y_i} y_{i-1}$, $1 \leqslant i \leqslant n-1$. We have

$$\partial_{\mathbf{y}} f = (\partial_{y_1} f, \partial_{y_2} f, \cdots, \partial_{y_{n-1}} f) \tag{7}$$

[2] We avoid the notation $\nabla_y f$, reserving ∇ for the Levi-Civita connection.

with

$$\begin{aligned}\partial_{y_i} f =&(d_{y_i}(v_i^+ - v_i^-))^\dagger(v_i^+ - v_i^-) + (d_{y_i} v_{i-1}^+)^\dagger(v_{i-1}^+ - v_{i-1}^-)\\ &- (d_{y_i} v_{i+1}^-)^\dagger(v_{i+1}^+ - v_{i+1}^-),\end{aligned} \tag{8}$$

where the Riemannian adjoint of the logarithm map derivative is defined by

$$\langle (d_x \log_x y)(u), w\rangle = \langle u, (d_x \log_x y)^\dagger(w)\rangle. \tag{9}$$

Since only v_{i-1}^+, v_i^+, v_i^- and v_{i+1}^- contain y_i, (8) follows from the fact that

$$\begin{aligned}\langle \partial_{y_i} f, w\rangle =&\langle v_i^+ - v_i^-, d_{y_i}(v_i^+ - v_i^-)(w)\rangle + \langle v_{i-1}^+ - v_{i-1}^-, d_{y_i}(v_{i-1}^+ - v_{i-1}^-)(w)\rangle\\ &+ \langle v_{i+1}^+ - v_{i+1}^-, d_{y_i}(v_{i+1}^+ - v_{i+1}^-)(w)\rangle\\ =&\langle (d_{y_i}(v_i^+ - v_i^-))^\dagger(v_i^+ - v_i^-) + (d_{y_i} v_{i-1}^+)^\dagger(v_{i-1}^+ - v_{i-1}^-)\\ &- (d_{y_i} v_{i+1}^-)^\dagger(v_{i+1}^+ - v_{i+1}^-), w\rangle.\end{aligned}$$

In the next section we develop strategies to calculate the derivatives and adjoints of derivatives of logarithm maps in the setting of general Riemannian manifolds, codimension-one embedded submanifolds and Lie groups.

The Riemannian gradient method for finding a geodesic joining given points x_0,x_1 is summarized as Algorithm 1.

Algorithm 1 Riemannian Gradient Descent For Finding Geodesics (RGD-G)

Require: Two points $x_0, x_1 \in U \subset M$, number of sub-intervals n, maximum number of iterations N, step size $\eta > 0$, stop error $\epsilon > 0$.

Ensure: Geodesic $\gamma : [0,1] \to M$ with $\gamma(0) = x_0$ and $\gamma(1) = x_1$.

1: Choose an arbitrary curve $\tilde{\gamma} : [0,1] \to U \subset M$ joining x_0 and x_1, and $y_i := \tilde{\gamma}(i/n)$, $i = 1, 2, \cdots, n-1$.
2: Replace $\tilde{\gamma}$ by a piecewise geodesic γ joining all y_i's via (single) shooting.
3: **for** $k \leftarrow 1$ to N **do**
4: **for** $i \leftarrow 1$ to $n-1$ **do**
5: Calculate the Euclidean gradient of (5) via (7).
6: Calculate the Riemannian gradient of (5) by the relationship (6). ▷ Details will be provided in Algorithm 3, 4, 6, 7, 9, 10.
7: Update y_i via Algorithm 5 ,8 or 11 with input y_i and $-\eta\partial_{y_i}^R f$.
8: **end for**
9: **if** $\|\partial_{\mathbf{y}}^R f\| \leqslant \epsilon$ or $f \leqslant \epsilon$ **then**
10: Update the piecewise geodesic γ based on new y_i's.
11: Return γ.
12: **end if**
13: **end for**

In general, the step size η is chosen to make the algorithm converge at a suitable rate. There is usually some trial and error. Many iterations are needed when η is too small,

but when η is large there can be problems with overshooting. For a general Riemannian manifold where x_0,x_1 are in the same coordinate chart, a reasonable choice for the initial curve $\tilde{\gamma}$ would be a line segment in chart coordinates. Alternatively, when M is embedded in Euclidean space, $\tilde{\gamma}$ could be the orthogonal projection of the Euclidean line segment from x_0 to x_1.

To implement the shooting method we use either MATLAB's function *bvp4c/bvp5c* or Mathematica's function *NDSolve* with method *shooting* in Lines 2 and 10 of Algorithm 1.

In Line 7, y_i is updated via the exponential map exp as follows

$$y_i \leftarrow \exp_{y_i}(-\eta \partial^R_{y_i} f),$$

where different forms of geodesics equations are needed to solve when it comes to Riemannian manifolds with local coordinates available, or of codimension one in Euclidean space, or left-invariant Lie groups.

2.2 Newton's method for finding geodesics

Newton's method is a powerful tool for finding the zeros of nonlinear functions in Euclidean spaces, which has a wide range of applications in mathematics and other subjects. This has motivated studies of the generalization of Newton's method from the Euclidean setting to Riemannian manifolds [23–26]. For a smooth vector field defined on a Riemannian manifold, Fernandes et al. [19] showed that in a suitable neighborhood of the singularity of the vector field, the sequence generated by Newton's method converges quadratically when the covariant derivative of the vector field is invertible in a convex neighborhood containing the intial guess. However, Newton's method may diverge if the initial guess is not sufficiently close to the solution. Many strategies have been proposed to overcome this drawback, such as incorporating the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm, the Levenberg-Marquardt algorithm, the trust region algorithm [27, 28], by introducing a merit function, which is the squared norm of the vector field, to replace a divergent iteration generated by Newton's method or make up a failed iteration in Newton's method [20, 21]. For simplicity, throughout this paper, we consider the basic Newton's method and leave other variants and improvements for future study. In our experience, the Riemannian gradient descent of Section 2.1 gives a suitable initial guess for Newton's method.

We calculate the derivative $\partial F(\mathbf{y})$ of the vector field F, in terms of derivatives of logarithm maps in Section 2.1 as follows, $\partial F(\mathbf{y}) =$

$$\begin{bmatrix} d_{y_1}(v_1^+ - v_1^-) & d_{y_2} v_1^+ & & & \\ -d_{y_1} v_2^- & d_{y_2}(v_2^+ - v_2^-) & d_{y_3} v_2^+ & & \\ & \ddots & \ddots & \ddots & \\ & -d_{y_{n-3}} v_{n-2}^- & d_{y_{n-2}}(v_{n-2}^+ - v_{n-2}^-) & d_{y_{n-1}} v_{n-2}^+ & \\ & & -d_{y_{n-2}} v_{n-1}^- & d_{y_{n-1}}(v_{n-1}^+ - v_{n-1}^-) & \end{bmatrix}, \tag{10}$$

where we have arranged F as a column vector. Then the covariant derivative $\nabla F(\mathbf{y})$ is related to $\partial F(\mathbf{y})$ by (6).

Newton's method for finding the singularity of the vector field F is to update y_i by

$$y_i \leftarrow \exp_{y_i}(w_i), \tag{11}$$

where exp is the exponential map on manifold M and $w = (w_1, w_2, \cdots, w_{n-1})$ is the solution of

$$F(\mathbf{y}) + \nabla F(\mathbf{y})w = \mathbf{0}. \tag{12}$$

Then Newton's method for finding a geodesic joining given points x_0 and x_1 by looking for a singularity of (4) is summarized as Algorithm 2.

Algorithm 2 Newton's Method For Finding Geodesics (NM-G)

Require: Two points $x_0, x_1 \in U \subset M$, number of sub-intervals n, maximum number of iterations N, step size $\eta > 0$, stop error $\epsilon > 0$.

Ensure: Geodesic $\gamma : [0,1] \to M$ with $\gamma(0) = x_0$ and $\gamma(1) = x_1$.

1: Choose an arbitrary curve $\tilde{\gamma} : [0,1] \to U \subset M$ joining x_0 and x_1, and $y_i := \tilde{\gamma}(i/n)$, $i = 1, 2, \cdots, n-1$.
2: **for** $k \leftarrow 1$ to N **do**
3: **for** $i \leftarrow 1$ to $n-1$ **do**
4: Calculate the velocities v_i^+ and v_i^-.
5: Calculate the derivatives $d_{y_i} v_i^+$ and $d_{y_i} v_i^-$. ▷ Details will be provided in Algorithm 3, 4, 6, 7, 9, 10.
6: **end for**
7: **if** $f \leqslant \epsilon$ **then**
8: Update the piecewise geodesic γ based on new y_i's.
9: Return γ.
10: **end if**
11: Construct $\partial F(\mathbf{y})$ by (10) and calculate $\nabla F(\mathbf{y})$.
12: Solving the linear equation (12) and update y_i according to (11).
13: **end for**

3 The derivative of the logarithm map

In Algorithms 1 and 2 a key task is to find the derivatives of v_i^+ and v_i^-, namely derivatives of Riemannian logarithms. We write $\mathrm{Log}(x, y) := \log_x y$. Let $v := \mathrm{Log}(x,y)$, then $\mathrm{Exp}(x, v) := \exp_x(v) = y$ is the inverse of Log.

Differentiating $y = \mathrm{Exp}(x,v)$ with respect to y along the direction w yields

$$w = (d_x \mathrm{Exp})_{x,v}(0) + (d_v \mathrm{Exp})_{x,v}(d_y v(w)),$$

then

$$(d_y \mathrm{Log})_{x,y}(w) = (d_v \mathrm{Exp})^{-1}_{x,v}(w). \tag{13}$$

Differentiating $y = \mathrm{Exp}(x,v)$ with respect to x along the direction w gives

$$\mathbf{0} = (d_x\mathrm{Exp})_{x,v}(w) + (d_v\mathrm{Exp})_{x,v}(d_x v(w)),$$

then

$$(d_x\mathrm{Log})_{x,y}(w) = -(d_v\mathrm{Exp})^{-1}_{x,v}(d_x\mathrm{Exp})_{x,v}(w). \tag{14}$$

So derivatives of the logarithm map are found from derivatives of the exponential, using (13) and (14). When $x = y$,

$$(d_y\mathrm{Log})_{x,y}(w) = w, \quad (d_x\mathrm{Log})_{x,y}(w) = -w. \tag{15}$$

In practice, when x is extremely close to y (though not exactly the same), using (13) and (14) may cause some numerical issues, and we use (15) instead.

To calculate derivatives of the exponential map, we first consider a variation of the geodesic $\gamma(t) = \mathrm{Exp}(x,v)$ given by $\gamma_1(t,s) := \mathrm{Exp}(\mathrm{Exp}(x,su_1),tv(s))$, where $u_1 \in T_xM$ defines a variation of the initial point along the geodesic $\mathrm{Exp}(x,su_1)$. We extend $v \in T_xM$ as a vector field $v(s)$ along $\mathrm{Exp}(x,su_1)$ via parallel translation. We also consider a variation of $\gamma(t)$ given by $\gamma_2(t,s) := \mathrm{Exp}(x,tv + su_2)$, where $u_2 \in T_v(T_xM) = T_xM$. These variations determine Jacobi fields, which are solutions J of the second-order linear dynamical system [2, 6, 22]

$$\nabla^2_{\dot\gamma(t)} J(t) + R(J(t), \dot\gamma(t))\dot\gamma(t) = \mathbf{0}, \tag{16}$$

where ∇ is the Levi-Civita connection for the Riemannian metric g on M, and R is the Riemannian curvature tensor. Then

$$(d_x\mathrm{Exp})_{x,v}(u_1) = J_1(1) \tag{17}$$

where the field J_1 is found by numerically solving (16) with the initial conditions $J_1(0) = u_1$, $\dot J_1(0) = \mathbf{0}$. We also have

$$(d_v\mathrm{Exp})_{x,v}(u_2) = J_2(1) \tag{18}$$

where J_2 is found by numerically solving (16) with the initial conditions $J_2(0) = \mathbf{0}$, $\dot J_2(0) = u_2$.

Next we consider in more detail how to solve the Jacobi equation (16) where M is a general Riemannian manifold, and in the important special case when M is a left-invariant Lie group (a specialized technique can then be applied).

3.1 *M* is a Riemannian manifold with local coordinates available

Let M be a p-dimensional ($1 \leqslant p < \infty$) Riemannian manifold with a Riemannian metric g, then g induces a Levi-Civita connection ∇ by the Koszul formula (1). Given a local coordinate system $\{x^i\}$ on an open set $U \subset M$, $\{\partial_i := \frac{\partial}{\partial x^i}\}$ is a basis for vector fields defined on U. Suppose we have two vector fields $X = X^i\partial_i$ and $Y = Y^j\partial_j$, then the expression for the connection ∇ in local coordinates $\{x^i\}$ on U is given by

$$\nabla_X Y = \nabla_{X^i\partial_i} Y^j \partial_j = X^i \frac{\partial Y^j}{\partial x^i}\partial_j + X^i Y^j \nabla_{\partial_i}\partial_j. \tag{19}$$

The *Christoffel symbols* are defined as the functions $\Gamma_{ij}^k : U \to \mathbb{R}$ given by

$$\nabla_{\partial_i}\partial_j = \Gamma_{ij}^k \partial_k, \tag{20}$$

where $1 \leqslant i,j,k \leqslant p$. By the fundamental theorem of Riemannian geometry, Christoffel symbols have the following form with respect to the metric $g_{ij} = g(\partial_i,\partial_j)$ [6],

$$\Gamma_{ij}^k = \frac{1}{2} g^{kl}(\partial_i g_{jl} + \partial_j g_{il} - \partial_l g_{ij}), \tag{21}$$

where the matrix $\left(g^{kl}\right)$ is the inverse of the symmetric $p \times p$ matrix $\left(g_{kl}\right)$.

We define the symmetric bilinear Christoffel form $\Gamma_x : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}^p$ by $\Gamma_x(v,w)_k := \sum_{i,j=1}^p \Gamma_{ij}^k v_i w_j$.

Example 1 Given $a,b,c > 0$, we define $Ell(2) \subset \mathbb{R}^3$ to be the 2-dimensional ellipsoid given by

$$\frac{x_1^2}{a^2} + \frac{x_2^2}{b^2} + \frac{x_3^2}{c^2} = 1$$

with Riemannian metric defined as the restriction of the standard Euclidean inner product. This manifold admits the following parameterization

$$Ell(\theta,\varphi) = (a\sin\varphi\cos\theta, b\sin\varphi\sin\theta, c\cos\varphi), \tag{22}$$

where $\theta \in [0,2\pi)$, $\varphi \in [0,\pi]$ act as global coordinates of $Ell(2)$. We can take advantage of the coordinate system $\{\theta,\varphi\}$ to choose the initial junctions y_i in Algorithms 1 and 2.

With respect to the restriction of the Euclidean metric, we have

$$\begin{aligned} g &= \begin{pmatrix} \langle \partial_\theta Ell, \partial_\theta Ell\rangle & \langle \partial_\theta Ell, \partial_\varphi Ell\rangle \\ \langle \partial_\varphi Ell, \partial_\theta Ell\rangle & \langle \partial_\varphi Ell, \partial_\varphi Ell\rangle \end{pmatrix} \\ &= \begin{pmatrix} \sin^2\varphi(a^2\sin^2\theta + b^2\cos^2\theta) & (b^2-a^2)\sin\varphi\cos\varphi\sin\theta\cos\theta \\ (b^2-a^2)\sin\varphi\cos\varphi\sin\theta\cos\theta & \cos^2\varphi(a^2\cos^2\theta + b^2\sin^2\theta) + c^2\sin^2\varphi \end{pmatrix}. \end{aligned} \tag{23}$$

If $a = b$, then by (21), non-vanishing Christoffel symbols are given by

$$\Gamma_{\theta\varphi}^\theta = \Gamma_{\varphi\theta}^\theta = \cot\varphi, \quad \Gamma_{\theta\theta}^\varphi = -\frac{a^2\tan\varphi}{a^2+c^2\tan^2\varphi}, \quad \Gamma_{\varphi\varphi}^\varphi = \frac{(c^2-a^2)\tan\varphi}{a^2+c^2\tan^2\varphi}.$$

Example 2 Let $SO(3)$ be the special orthogonal group composed by all positively-oriented 3×3 orthogonal matrices, endowed with a bi-invariant metric

$$\langle V_1, V_2\rangle_X := -\frac{1}{2}\mathrm{tr}(X^{-1}V_1X^{-1}V_2),$$

where $V_1,V_2 \in T_X SO(3)$ and $X \in SO(3)$.

By considering the following variation

$$\delta\tfrac{1}{2}\int_0^1\langle\dot X,\dot X\rangle dt = \mathrm{tr}(X^{-1}\dot X X^{-1}\delta X)|_{t=0}^1 - \int_0^1 \mathrm{tr}(X^{-1}(\ddot X - \dot X X^{-1}\dot X)X^{-1}\delta X)dt,$$

we find $\Gamma_X(V,V) = -V\,X^{-1}V$. Then, because the Christoffel form is symmetric and bilinear,

$$\Gamma_X(U,V) = -\tfrac{1}{2}(UX^{-1}V + VX^{-1}U)$$

for any $U,V \in T_X SO(3)$.

Now we rewrite the Jacobi equation (16) in terms of Christoffel symbols and their derivatives. Then

$$\begin{aligned}\nabla_{\dot\gamma(t)}J(t) &= \dot J(t) + \Gamma_{\gamma(t)}(J(t),\dot\gamma(t)),\\ \nabla^2_{\dot\gamma(t)}J(t) &= \tfrac{d}{dt}\big(\dot J(t) + \Gamma_{\gamma(t)}(J(t),\dot\gamma(t))\big) + \Gamma_{\gamma(t)}(\dot J(t) + \Gamma_{\gamma(t)}(J(t),\dot\gamma(t)),\dot\gamma(t))\\ &= \ddot J(t) + d\Gamma_{\gamma(t)}(\dot\gamma(t))(J(t),\dot\gamma(t)) + 2\Gamma_{\gamma(t)}(\dot J(t),\dot\gamma(t))\\ &\quad -\Gamma_{\gamma(t)}(J(t),\Gamma_{\gamma(t)}(\dot\gamma(t),\dot\gamma(t))) + \Gamma_{\gamma(t)}(\dot\gamma(t),\Gamma_{\gamma(t)}(J(t),\dot\gamma(t))),\end{aligned}$$

where we used the fact that $\nabla_{\dot\gamma(t)}\dot\gamma(t) = \ddot\gamma(t) + \Gamma_{\gamma(t)}(\dot\gamma(t),\dot\gamma(t)) = \mathbf{0}$. Combining with the result

$$\begin{aligned}R(J(t),\dot\gamma(t))\dot\gamma(t) = \quad & d\Gamma_{\gamma(t)}(J(t))(\dot\gamma(t),\dot\gamma(t)) - d\Gamma_{\gamma(t)}(\dot\gamma(t))(\dot\gamma(t),J(t))\\ &+\Gamma_{\gamma(t)}(J(t),\Gamma_{\gamma(t)}(\dot\gamma(t),\dot\gamma(t))) - \Gamma_{\gamma(t)}(\dot\gamma(t),\Gamma_{\gamma(t)}(\dot\gamma(t),J(t))),\end{aligned}$$

we have

Proposition 1 The Jacobi equation (16) is equivalent to

$$\ddot J(t) + 2\Gamma_{\gamma(t)}(\dot J(t),\dot\gamma(t)) + d\Gamma_{\gamma(t)}(J(t))(\dot\gamma(t),\dot\gamma(t)) = \mathbf{0}. \tag{24}$$

Algorithm 3 Finding geodesics joining two points and the logarithm map on Riemannian manifold M (Geo-Log-M)

Require: Two points $y_i, y_{i+1} \in U \subset M$, Christoffel form Γ, small positive number $\epsilon_0 > 0$.

Ensure: Geodesic (denoted by some discrete points on M) $\gamma : [0,1] \to M$ with $\gamma(0) = y_i$ and $\gamma(1) = y_{i+1}$, velocity $\mathrm{Log}(y_i, y_{i+1})$.

1: **if** $\|y_i - y_{i+1}\| \leqslant \epsilon_0$ **then**
2: $\gamma = y_i$.
3: $\mathrm{Log}(y_i, y_{i+1}) = \mathrm{TProj}(\mathrm{y_{i+1}} - \mathrm{y_i})$. ▷ TProj is the projection from the embedded Euclidean space to the tangent space $T_{y_i}M$.
4: **else**
5: Solving the system

$$\begin{cases} \ddot{\gamma}(t) + \Gamma_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t)) = \mathbf{0}, \\ \gamma(0) = y_i, \quad \gamma(1) = y_{i+1} \end{cases}$$

using single shooting with the initial guess $\dot{\gamma}(0) = \mathrm{TProj}(\mathrm{y_{i+1}} - \mathrm{y_i})$ gives γ, $\mathrm{Log}(y_i, y_{i+1}) = \dot{\gamma}(0)$.
6: **end if**

Algorithm 4 Finding the derivatives of the logarithm map on Riemannian manifold M (DLog-M)

Require: Two points $y_i, y_{i+1} \in U \subset M$ and the geodesic γ joining them, Christoffel form Γ, basis vectors $\{e_\alpha\}_{\alpha=1}^p$ of the tangent space $T_{y_i}M$, small positive number $\epsilon_0 > 0$.

Ensure: The derivatives $(d_{y_i}\mathrm{Log})_{y_i,y_{i+1}}(w)$ and $(d_{y_{i+1}}\mathrm{Log})_{y_i,y_{i+1}}(w)$.

1: **if** $\|y_i - y_{i+1}\| \leqslant \epsilon_0$ **then**
2: $(d_{y_i}\mathrm{Log})_{y_i,y_{i+1}}(w) = -w$.
3: $(d_{y_{i+1}}\mathrm{Log})_{y_i,y_{i+1}}(w) = w$.
4: **else**
5: Solving the system (24) with the initial conditions $J(0) = e_\alpha$ and $\dot{J}(0) = 0$ returns $J_1^\alpha(1)$. Combine all vectors $J_1^\alpha(1)$ as a matrix $J_1(1)$.
6: Solving the system (24) with the initial conditions $J(0) = 0$ and $\dot{J}(0) = e_\alpha$ returns $J_2^\alpha(1)$. Combine all vectors $J_2^\alpha(1)$ as a matrix $J_2(1)$.
7: $(d_{y_i}\mathrm{Log})_{y_i,y_{i+1}}(w) = -J_2(1)^{-1}J_1(1)w$.
8: $(d_{y_{i+1}}\mathrm{Log})_{y_i,y_{i+1}}(w) = J_2(1)^{-1}w$.
9: **end if**

Algorithm 5 Finding the exponential map on Riemannian manifold M (Exp-M)

Require: Point $x \in U \subset M$, velocity v at x, Christoffel form Γ, small positive number $\epsilon_0 > 0$.

Ensure: The end point $\mathrm{Exp}(x, v)$.

1: **if** $\|v\| \leqslant \epsilon_0$ **then**

2: $\mathrm{Exp}(x, v) = x$.

3: **else**

4: Solving the system

$$\begin{cases} \ddot{\gamma}(t) + \Gamma_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t)) = \mathbf{0}, \\ \gamma(0) = y, \quad \dot{\gamma}(0) = v \end{cases}$$

gives $\mathrm{Exp}(x, v) = \gamma(1)$.

5: **end if**

In summary, Algorithm 3 finds the geodesic joining two consecutive points y_i,y_{i+1} and the logarithm map v_i^+. Algorithm 4 finds the derivatives of the logarithm maps $d_{y_i} v_i^+, d_{y_{i+1}} v_i^+$. Algorithm 5 finds the exponential map on M.

Note that in Algorithm 4 (similar for Algorithms 7 and 10), since the p-dimensional tangent space $T_{y_i}M = \mathbb{R}^p$, we need to solve the (24) $2p$ times with the standard Euclidean basis vectors $\{e_\alpha\}_{\alpha=1}^p$, where e_α is a zero column except 1 in α-th position. All solutions for the type of initial conditions $J(0) = e_\alpha$, $\dot{J} = 0$ is arranged as a matrix $J_1(1)$, and those for the other type is arranged as a matrix $J_2(1)$. Then, the derivative $(d_{y_i}\mathrm{Log})_{y_i,y_{i+1}}(w) = -J_2(1)^{-1}J_1(1)w$ and $(d_{y_{i+1}}\mathrm{Log})_{y_i,y_{i+1}}(w) = J_2(1)^{-1}w$.

The relationship between Algorithm 1 and Algorithm 4 is that Algorithm 4 is a significant sub-algorithm of Algorithm 1 that calculate the derivatives $d_{y_i} v_i^+, d_{y_{i+1}} v_i^+$, which will be used in calculating the Riemannian gradient of the cost function (see (8)).

3.2 Riemannian manifolds of codimension one in Euclidean space

Let M be a p-dimensional Riemannian manifold isometrically embedded in $(p+1)$-dimensional Euclidean space $\mathbb{E}^{p+1}$. Then M is a level set of some smooth function $h : M \subset \mathbb{E}^{p+1} \to \mathbb{R}$, that is, $M = h^{-1}(c)$ for some constant $c \in \mathbb{R}$. The benefit of writing geodesic and Jacobi equations in terms of h is to represent the geodesic as a curve in Euclidean space and use standard Euclidean derivatives as illustrated below.

Let $\nu_\gamma := \dfrac{\mathrm{grad}\, h(\gamma)}{\|\mathrm{grad}\, h(\gamma)\|}$ be the unit normal to the manifold M, where grad h is the Euclidean gradient of h and $\|\cdot\|$ is the norm induced by standard Euclidean metric. Then, the covariant derivative $\nabla_{\dot{\gamma}(t)}\dot{\gamma}(t)$ is nothing more than the Euclidean projection of the vector $\ddot{\gamma}(t)$ onto the tangent space $T_{\gamma(t)}M$, namely

$$\nabla_{\dot\gamma(t)}\dot\gamma(t) = \ddot\gamma(t) - \langle \ddot\gamma(t), \nu_\gamma(t)\rangle \nu_\gamma(t). \tag{25}$$

Since $\nu_\gamma(t)$ is normal to M at $\gamma(t)$, $\langle \dot\gamma(t), \nu_\gamma(t)\rangle = 0$, by differentiating, we have $\langle \ddot\gamma(t), \nu_\gamma(t)\rangle = -\langle \dot\gamma(t), \dot\nu_\gamma(t)\rangle$, where $\dot\nu_\gamma(t) = d\nu_\gamma(\dot\gamma(t))$. Therefore, the geodesic equation $\nabla_{\dot\gamma(t)}\dot\gamma(t) = 0$ can now be rewritten as

$$\ddot\gamma(t) + \langle \dot\gamma(t), \dot\nu_\gamma(t)\rangle \nu_\gamma(t) = \mathbf{0}. \tag{26}$$

Although now $\gamma : [0,1] \to \mathbb{E}^{p+1}$ is viewed as a curve in Euclidean space, the boundary conditions $\gamma(0)$,$\gamma(1) \in M$ will force the solution of (26) to stay on the manifold M.

In order to represent the Jacobi equation (16) in terms of the function h or the unit normal ν, we consider a variation $\gamma_s : (-\varepsilon, \varepsilon) \times [0,1] \to \mathbb{E}^{p+1}$ of $\gamma : [0,1] \to \mathbb{E}^{p+1}$ for a small $\varepsilon > 0$. Then,

$$\begin{aligned}
&\frac{d}{ds}|_{s=0}\left(\ddot\gamma_s(t) + \langle \dot\gamma_s(t), \dot\nu_{\gamma_s}(t)\rangle \nu_{\gamma_s}(t)\right) \\
=&\ddot\xi(t) + \langle \dot\xi(t), \dot\nu_\gamma(t)\rangle \nu_\gamma(t) + \langle \dot\gamma(t), d^2\nu_\gamma(\dot\gamma(t), \xi(t))\rangle \nu_\gamma(t) + \langle \dot\gamma(t), \dot\nu_\gamma(t)\rangle d\nu_\gamma(\xi(t)),
\end{aligned}$$

where $\xi(t) = \frac{d}{ds}|_{s=0}\gamma_s(t)$ is the variational vector field along $\gamma(t)$. Therefore, we have

Proposition 2 The Jacobi equation (16) is equivalent to

$$\begin{aligned}
\ddot\xi(t) + \langle \dot\xi(t), \dot\nu_\gamma(t)\rangle \nu_\gamma(t) + \langle \dot\gamma(t), d^2\nu_\gamma(\dot\gamma(t), \xi(t))\rangle \nu_\gamma(t) \\
+\langle \dot\gamma(t), \dot\nu_\gamma(t)\rangle d\nu_\gamma(\xi(t)) = \mathbf{0}.
\end{aligned} \tag{27}$$

As with the geodesic equation, the initial conditions $\xi(0), \dot\xi(0) \in T_{\gamma(0)}M$ force the solution of (27) to stay on the tangent space of M, although $\xi : [0,1] \to \mathbb{E}^{p+1}$ is viewed as a curve in Euclidean space.

In summary, Algorithm 6[3] finds the geodesic joining two consecutive junctions y_i,y_{i+1} and the logarithm map v_i^+. Algorithm 7 finds the derivatives of the logarithm maps $d_{y_i}v_i^+, d_{y_{i+1}}v_i^+$. Algorithm 8 finds the exponential map on M.

[3] The geodesic is denoted by some discrete points on M.

Algorithm 6 Finding geodesics joining two points and the logarithm map on Riemannian manifold M (Geo-Log-M1)

Require: Two points $y_i, y_{i+1} \in U \subset M$, embedding function g, small positive number $\epsilon_0 > 0$.

Ensure: Geodesic[3] $\gamma : [0,1] \to M$ with $\gamma(0) = y_i$ and $\gamma(1) = y_{i+1}$, velocity $\mathrm{Log}(y_i, y_{i+1})$.

1: **if** $\|y_i - y_{i+1}\| \leqslant \epsilon_0$ **then**

2: $\gamma = y_i$.

3: $\mathrm{Log}(y_i, y_{i+1}) = \mathrm{TProj}(\mathrm{y_{i+1}} - \mathrm{y_i})$. ▷ TProj is the projection from the embedded Euclidean space to the tangent space $T_{y_i}M$.

4: **else**

5: Solving the system

$$\begin{cases} \ddot{\gamma}(t) + \langle \dot{\gamma}(t), \dot{\nu}_\gamma(t) \rangle \nu_\gamma(t) = \mathbf{0}, \\ \gamma(0) = y_i, \quad \gamma(1) = y_{i+1} \end{cases}$$

using single shooting with the initial guess $\dot{\gamma}(0) = \mathrm{TProj}(\mathrm{y_{i+1}} - \mathrm{y_i})$ gives γ, $\mathrm{Log}(y_i, y_{i+1}) = \dot{\gamma}(0)$.

6: **end if**

Algorithm 7 Finding the derivatives of the logarithm map on Riemannian manifold M (DLog-M)

Require: Two points $y_i, y_{i+1} \in U \subset M$ and the geodesic γ joining them, Christoffel form Γ, basis vectors $\{e_\alpha\}_{\alpha=1}^p$ of the tangent space $T_{y_i}M$, small positive number $\epsilon_0 > 0$.

Ensure: The derivatives $(d_{y_i}\mathrm{Log})_{y_i, y_{i+1}}(w)$ and $(d_{y_{i+1}}\mathrm{Log})_{y_i, y_{i+1}}(w)$.

1: **if** $\|y_i - y_{i+1}\| \leqslant \epsilon_0$ **then**

2: $(d_{y_i}\mathrm{Log})_{y_i, y_{i+1}}(w) = -w$.

3: $(d_{y_{i+1}}\mathrm{Log})_{y_i, y_{i+1}}(w) = w$.

4: **else**

5: Solving the system (27) with the initial conditions $\xi(0) = e_\alpha$ and $\dot{\xi}(0) = 0$ returns $\xi_1^\alpha(1)$. Combine all vectors $\xi_1^\alpha(1)$ as a matrix $\xi_1(1)$.

6: Solving the system (27) with the initial conditions $\xi(0) = \mathbf{0}$ and $\dot{\xi}(0) = e_\alpha$ returns $\xi_2^\alpha(1)$. Combine all vectors $\xi_2^\alpha(1)$ as a matrix $\xi_2(1)$.

7: $(d_{y_i}\mathrm{Log})_{y_i, y_{i+1}}(w) = -\xi_2(1)^{-1}\xi_1(1)w$.

8: $(d_{y_{i+1}}\mathrm{Log})_{y_i, y_{i+1}}(w) = \xi_2(1)^{-1}w$.

9: **end if**

Algorithm 8 Finding the exponential map on Riemannian manifold M (Exp-M1)

Require: Point $x \in U \subset M$, velocity v at x, embedding function h, small positive number $\epsilon_0 > 0$.

Ensure: The end point $\mathrm{Exp}(x, v)$.

1: **if** $\|v\| \leqslant \epsilon_0$ **then**

2: $\mathrm{Exp}(x, v) = x$.

3: **else**

4: Solving the system

$$\begin{cases} \ddot{\gamma}(t) + \langle \dot{\gamma}(t), \dot{\nu}_\gamma(t) \rangle \nu_\gamma(t) = \mathbf{0}, \\ \gamma(0) = y, \quad \dot{\gamma}(0) = v \end{cases}$$

gives $\mathrm{Exp}(x, v) = \gamma(1)$.

5: **end if**

3.3 *M* is a left-invariant Lie group

Given an inner product on the Lie algebra of a Lie group G, a left-invariant Riemannian metric on G is defined by left-translation:

$$\langle w_1, w_2 \rangle_g = \langle h \cdot w_1, h \cdot w_2 \rangle_{h \cdot g} = \langle (dL_{g^{-1}})_g(w_1), (dL_{g^{-1}})_g(w_2) \rangle_{\mathfrak{g}}, \tag{28}$$

where $(dL_g)_h : T_hG \to T_{gh}G$ is the derivative of L_g at $h \in G$, $w_1, w_2 \in T_gG$. To shorten notation we may drop subscripts in the metric $\langle \cdot, \cdot \rangle$ if there is no ambiguity. The Lie group G equipped with this Riemannian metric is said to be *left-invariant*.

To simplify the second-order system (16), we use the technique of left Lie reduction [15]. Given a C^∞ curve $x : [0, 1] \to G$, let F be any C^∞ vector field along x. Then the *left Lie reduction* $\hat{F}$ of F is defined as

$$\hat{F}(t) := (dL_{x(t)^{-1}})_{x(t)} F(t), \tag{29}$$

where $t \in [0,1]$. We refer to [16–18, 29, 30] and references for applications of left Lie reduction in reducing high order nonlinear dynamical systems on Lie groups and Riemannian homogeneous spaces.

Now introduce a bilinear operator $B : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$ defined by

$$\langle B(w_1, w_2), w_3 \rangle := \langle [w_2, w_3], w_1 \rangle, \tag{30}$$

where $\mathfrak{g}$ is the Lie algebra TG_e, $w_1, w_2, w_3 \in \mathfrak{g}$, and $[\cdot, \cdot]$ is the Lie bracket. If G is a bi-invariant Lie group, then $\langle [X,Y], Z \rangle = \langle Y, [Z,X] \rangle$, which gives $B(X,Y) = [X,Y]$.

Let $\{E_i\}$ be an orthonormal basis of $\mathfrak{g}$ at the identity e, then $\{\tilde{E}_i := (dL_g)_e(E_i)\}$ forms an orthonormal basis at $g \in G$. By the Koszul formula (1), the constant $\langle \tilde{E}_i, \tilde{E}_j \rangle$ and the left invariance of the metric and vectors, we have (see page 118 in [14])

$$\begin{aligned} 2\langle \nabla_{\tilde{E}_i} \tilde{E}_j, \tilde{E}_k \rangle =& \langle \tilde{E}_i, [\tilde{E}_k, \tilde{E}_j] \rangle + \langle \tilde{E}_j, [\tilde{E}_k, \tilde{E}_i] \rangle + \langle \tilde{E}_k, [\tilde{E}_i, \tilde{E}_j] \rangle \\ =& \langle E_i, [E_k, E_j] \rangle + \langle E_j, [E_k, E_i] \rangle + \langle E_k, [E_i, E_j] \rangle \\ =& \langle [E_i, E_j] - B(E_i, E_j) - B(E_j, E_i), E_k \rangle, \end{aligned}$$

from which

$$(dL_{g^{-1}})_g \nabla_{\tilde{E}_i} \tilde{E}_j = \tfrac{1}{2}\big([E_i, E_j] - B(E_i, E_j) - B(E_j, E_i)\big). \tag{31}$$

Further, we find $(dL_{g^{-1}})_g R(\tilde{E}_i, \tilde{E}_j)\tilde{E}_k =$

$$\begin{aligned} & \nabla_{E_i} \nabla_{E_j} E_k - \nabla_{E_j} \nabla_{E_i} E_k - \nabla_{[E_i, E_j]} E_k \\ =& \frac{1}{2} \nabla_{E_i} \big([E_j, E_k] - B(E_j, E_k) - B(E_k, E_j)\big) \\ & - \frac{1}{2} \nabla_{E_j} \big([E_i, E_k] - B(E_i, E_k) - B(E_k, E_i)\big) \\ & - \frac{1}{2} \big([[E_i, E_j], E_k] - B([E_i, E_j], E_k) - B(E_k, [E_i, E_j])\big) \\ =& \frac{1}{4}[E_k, [E_i, E_j]] + \frac{1}{2} B([E_i, E_j], E_k) + \frac{1}{2} B(E_k, [E_i, E_j]) \\ & - \frac{1}{4}[E_i, B(E_j, E_k) + B(E_k, E_j)] + \frac{1}{4}[E_j, B(E_i, E_k) + B(E_k, E_i)] \\ & - \frac{1}{4} B(E_i, [E_j, E_k] - B(E_j, E_k) - B(E_k, E_j)) \\ & - \frac{1}{4} B([E_j, E_k] - B(E_j, E_k) - B(E_k, E_j), E_i) \\ & + \frac{1}{4} B(E_j, [E_i, E_k] - B(E_i, E_k) - B(E_k, E_i)) \\ & + \frac{1}{4} B([E_i, E_k] - B(E_i, E_k) - B(E_k, E_i), E_j), \end{aligned} \tag{32}$$

where we have used the Jacobi identity in the last equality.

Lemma 3 Let $x : [0, 1] \to G$ be a C^∞ curve, V the left Lie reduction of the velocity vector field $\dot{x}$ of x, W_0 any C^∞ vector field along x and W its left Lie reduction. Then

$$(dL_{x^{-1}})_x \nabla_{\dot{x}} W_0 = \dot{W} + \tfrac{1}{2}([V, W] - B(V, W) - B(W, V)), \tag{33}$$

$$\begin{aligned}(dL_{x^{-1}})_x\nabla^2_{\dot{x}}W_0 =&\ddot{W}+[V,\dot{W}]+\frac{1}{2}[\dot{V},W]+\frac{1}{4}[V,[V,W]]-\frac{1}{4}[V,B(V,W)]\\&-\frac{1}{4}[V,B(W,V)]-\frac{1}{2}B(\dot{V},W)-\frac{1}{2}B(W,\dot{V})-B(V,\dot{W})-B(\dot{W},V)\\&-\frac{1}{4}B(V,[V,W])+\frac{1}{4}B(V,B(V,W))+\frac{1}{4}B(V,B(W,V))\\&-\frac{1}{4}B([V,W],V)+\frac{1}{4}B(B(V,W),V)+\frac{1}{4}B(B(W,V),V),\end{aligned}\tag{34}$$

$$\begin{aligned}(dL_{x^{-1}})_xR(W_0,\dot{x})\dot{x} =&\frac{1}{4}[V,[W,V]]-\frac{1}{2}[W,B(V,V)]+\frac{1}{4}[V,B(W,V)]+\frac{1}{4}[V,B(V,W)]\\&+\frac{3}{4}B([W,V],V)+\frac{3}{4}B(V,[W,V])+\frac{1}{2}B(W,B(V,V))\\&+\frac{1}{2}B(B(V,V),W)-\frac{1}{4}B(V,B(W,V))-\frac{1}{4}B(V,B(V,W))\\&-\frac{1}{4}B(B(W,V),V)-\frac{1}{4}B(B(V,W),V).\end{aligned}\tag{35}$$

Proof Denote $V(t)=(dL_{x(t)^{-1}})_{x(t)}\dot{x}(t)=V^i(t)E_i$, $W(t)=(dL_{x(t)^{-1}})_{x(t)}W_0(t)=W^i(t)E_i$, where $\{E_i\}$ is orthonormal basis of $\mathfrak{g}$. Then, $\dot{x}(t)=V^i(t)(dL_{x(t)})_eE_i$ and $W_0(t)=W^i(t)(dL_{x(t)})_eE_i$. By (31) and the properties of ∇, we have

$$\begin{aligned}(dL_{x^{-1}})_x\nabla_{\dot{x}}W_0 =&\dot{W}^iE_i+W^iV^j\nabla_{E_j}E_i\\=&\dot{W}^iE_i+\frac{1}{2}W^iV^j([E_j,E_i]-B(E_j,E_i)-B(E_i,E_j))\\=&\dot{W}+\frac{1}{2}([V,W]-B(V,W)-B(W,V))\end{aligned}$$

and

$$\begin{aligned}(dL_{x^{-1}})_x\nabla^2_{\dot{x}}W_0 =&\frac{d}{dt}\Big(\dot{W}+\frac{1}{2}([V,W]-B(V,W)-B(W,V))\Big)\\&+\frac{1}{2}([V,\dot{W}+\frac{1}{2}([V,W]-B(V,W)-B(W,V))]\\&-B(V,\dot{W}+\frac{1}{2}([V,W]-B(V,W)-B(W,V)))\\&-B(\dot{W}+\frac{1}{2}([V,W]-B(V,W)-B(W,V)),V)),\end{aligned}$$

which results in (34). Finally, (35) is directly from (32). □

Applying Lemma 3 to the system (16) and using $\nabla_{\dot{\gamma}}\dot{\gamma}=\mathbf{0}$, we derive

Proposition 4 The Jacobi equation (16) is equivalent to

$$\begin{aligned}\ddot{W}+[V,\dot{W}]-B(V,\dot{W})-B(\dot{W},V)-[W,B(V,V)]\\+B([W,V],V)+B(V,[W,V])=\mathbf{0},\end{aligned}\tag{36}$$

where $W=(dL_{\gamma^{-1}})_\gamma J$ and $V=(dL_{\gamma^{-1}})_\gamma\dot{\gamma}$.

By (33), the geodesic equation $\nabla_{\dot\gamma}\dot\gamma = \mathbf{0}$ reduces to $\dot V = B(V,V)$ with $\dot\gamma = \gamma V$. Now we summarize the procedure of finding the geodesic joining two consecutive y_i, the velocity v_i^+ and the derivatives of the logarithm map, the exponential map on G as Algorithms 9, 10, and 11.

Algorithm 9 Finding geodesics joining two points and the logarithm map on Lie group G (Geo-Log-G)

Require: Two points $y_i, y_{i+1} \in U \subset G$, bilinear form $B : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$, small positive number $\epsilon_0 > 0$.

Ensure: Geodesic $\gamma : [0,1] \to G$ with $\gamma(0) = y_i$ and $\gamma(1) = y_{i+1}$, velocity $\mathrm{Log}(y_i, y_{i+1})$.

1: **if** $\|y_i - y_{i+1}\| \leqslant \epsilon_0$ **then**

2: $\gamma = y_i$.

3: $\mathrm{Log}(y_i, y_{i+1}) = \mathrm{TProj}(\mathrm{y_{i+1}} - \mathrm{y_i})$. ▷ TProj is the projection from the embedded Euclidean space to the tangent space $T_{y_i}G$.

4: **else**

5: Solving the system

$$\begin{cases} \dot\gamma(t) = \gamma(t)V(t), \\ \dot V(t) = B(V(t), V(t)), \\ \gamma(0) = y_i, \quad \gamma(1) = y_{i+1} \end{cases}$$

using single shooting with the initial guess $\dot\gamma(0) = \mathrm{TProj}(\mathrm{y_{i+1}} - \mathrm{y_i})$ gives γ, $\mathrm{Log}(y_i, y_{i+1}) = \dot\gamma(0)$.

6: **end if**

Algorithm 10 Finding the derivatives of the logarithm map on Lie group G (DLog-G)

Require: Two points $y_i, y_{i+1} \in U \subset M$ and the geodesic γ joining them, bilinear form $B : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$, basis vectors $\{e_\alpha\}_{\alpha=1}^{p}$ of the tangent space $T_{y_i}G$, small positive number $\epsilon_0 > 0$.

Ensure: The derivatives $(d_{y_i}\mathrm{Log})_{y_i,y_{i+1}}(w)$ and $(d_{y_{i+1}}\mathrm{Log})_{y_i,y_{i+1}}(w)$.

if $\|y_i - y_{i+1}\| \leqslant \epsilon_0$ **then**
    $(d_{y_i}\mathrm{Log})_{y_i,y_{i+1}}(w) = -w$.
    $(d_{y_{i+1}}\mathrm{Log})_{y_i,y_{i+1}}(w) = w$.
else
    Solving the system (36) with the initial conditions $J(0) = e_\alpha$ and $\dot{J}(0) = \mathbf{0}$ returns $J_1^\alpha(1)$. Combine all vectors $J_1^\alpha(1)$ as a matrix $J_1(1)$.
    Solving the system (36) with the initial conditions $J(0) = \mathbf{0}$ and $\dot{J}(0) = e_\alpha$ returns $J_2^\alpha(1)$. Combine all vectors $J_2^\alpha(1)$ as a matrix $J_2(1)$.
    $(d_{y_i}\mathrm{Log})_{y_i,y_{i+1}}(w) = -J_2(1)^{-1}J_1(1)w$.
    $(d_{y_{i+1}}\mathrm{Log})_{y_i,y_{i+1}}(w) = J_2(1)^{-1}w$.
end if

Algorithm 11 Finding the exponential map on Lie group G (Exp-G)

Require: Point $y \in U \subset G$, velocity v at y, bilinear form $B : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$, small positive number $\epsilon_0 > 0$.

Ensure: The end point $\mathrm{Exp}(y, v)$.

if $\|v\| \leqslant \epsilon_0$ **then**
    $\mathrm{Exp}(x, v) = x$.
else
    Solving the system

$$\begin{cases} \dot{\gamma}(t) = \gamma(t)V(t), \\ \dot{V}(t) = B(V(t), V(t)), \\ \gamma(0) = y, \quad \dot{\gamma}(0) = v \end{cases}$$

    gives $\mathrm{Exp}(x, v) = \gamma(1)$.
end if

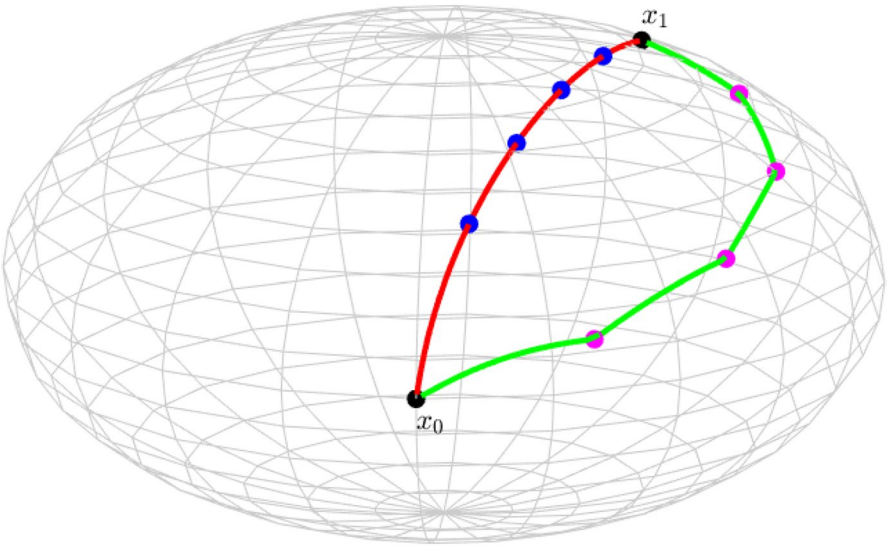

Fig. 1 Two given points x_0 and x_1 are denoted by black points. The magenta points represent the initial junctions and the green curve represents the initial piecewise geodesic joining x_0 and x_1. The blue points represent the final junctions and the red curve represents the final geodesic generated by Algorithm 2

4 Numerical experiments

Next we illustrate our algorithms by numerical experiments on the ellipsoid $Ell(2)$ and on a left-invariant special orthogonal group $SO(3)$. All experiments were executed in MATLAB 2020b on a computer with Intel Core i7-8700K CPU, 32 GB RAM and Windows 10 Enterprise. MATLAB codes are available on github https://github.com/beiliren/Endpoint-geodesic-problems.

4.1 The ellipsoid

Let $Ell(2)$ be the 2-dimensional ellipsoid defined in Example 1. Define the diagonal matrix $A := \text{diag}\{\frac{1}{a^2}, \frac{1}{b^2}, \frac{1}{c^2}\}$, then $Ell(2) = \{x \in \mathbb{R}^3 | x^T A x = 1\}$, whose tangent space is given by $T_x Ell(2) = \{v \in \mathbb{R}^3 | v^T A x = 0\}$. The normal space

Table 1 Comparison of runtime of different methods for finding geodesics

n	Algorithm 1	Leapfrog	Algorithm 2
4	31.8635	0.6673	0.9469
5	> 100	1.1787	1.1267
6	> 100	1.8055	1.3032
7	> 100	2.7095	1.4747
8	> 100	3.8182	1.6749
9	> 100	5.4586	2.0041

$N_x Ell(2) = \{r_0 Ax \in \mathbb{R}^3 | r_0 \in \mathbb{R}\}$ is the orthogonal complement of the tangent space with respect to the Euclidean metric.

According to Section 3.2, $Ell(2)$ is equivalent to $g(1)^{-1}$, where $g(x) = x^T Ax$. Then, straightforward calculations give $v_x = \frac{Ax}{\|Ax\|}$ and $\dot{v}_x = \frac{A\dot{x}}{\|Ax\|} - \frac{\langle Ax, A\dot{x}\rangle}{\|Ax\|^3} Ax$, by (26), the geodesic equation is

$$\ddot{x} + \frac{\langle \dot{x}, A\dot{x}\rangle}{\|Ax\|^2} Ax = 0. \tag{37}$$

Further, we have $dv_x(\xi) = \frac{A\xi}{\|Ax\|} - \frac{\langle Ax, A\xi\rangle}{\|Ax\|^3} Ax$ and

$$\begin{aligned} d^2 v_x(\dot{x}, \xi) = & -\frac{\langle Ax, A\xi\rangle}{\|Ax\|^3} A\dot{x} - \frac{\langle A\dot{x}, A\xi\rangle}{\|Ax\|^3} Ax - \frac{\langle Ax, A\dot{x}\rangle}{\|Ax\|^3} A\xi \\ & + 3\frac{\langle Ax, A\dot{x}\rangle \langle Ax, A\xi\rangle}{\|Ax\|^5} Ax. \end{aligned}$$

By (27), the Jacobi equation is

$$\begin{aligned} & \ddot{\xi} + \langle \dot{\xi}, A\dot{x}\rangle \frac{Ax}{\|Ax\|^2} + \langle \dot{x}, A\dot{x}\rangle \frac{A\xi}{\|Ax\|^2} \\ & - 2(\langle Ax, A\dot{x}\rangle \langle \dot{\xi}, Ax\rangle + \langle \dot{x}, A\dot{x}\rangle \langle Ax, A\xi\rangle) \frac{Ax}{\|Ax\|^4} = \mathbf{0}, \end{aligned} \tag{38}$$

where x is a solution of (37).

For two points x,y on $Ell(2)$, let $\mathrm{TProj}(y - x)$ be the projection of the vector $y - x$ to the tangent space $T_x Ell(2)$. Then, $\mathrm{TProj}(y - x) = y - x - r_0 Ax$ for some $r_0 \in \mathbb{R}$. By requiring that $\|\mathrm{TProj}(y - x)\|^2 + \|r_0 Ax\|^2 = \|y - x\|^2$, we have $r_0 = \frac{y^T Ax - 1}{\|Ax\|^2}$, which implies

$$\mathrm{TProj}(y - x) = y - x - \frac{y^T Ax - 1}{\|Ax\|^2} Ax. \tag{39}$$

Example 3 On a ellipsoid $Ell(2)$ with semi-axes $a = 6, b = 2, c = 1$, we choose two points $x_0 = (0,0,1)^T$ and $x_1 = (\frac{9}{2}, \frac{\sqrt{3}}{2}, -\frac{1}{2})^T$, which correspond to the coordinates $(\frac{\pi}{2}, 0)$ and $(\frac{\pi}{6}, \frac{2\pi}{3})$ in (22), respectively. The initial curve $\tilde{\gamma}$ in Algorithms 1 and 2 corresponds to the line segment in local coordinates.

Solving the geodesic (37) with linear initial guess by shooting is unsuccessful (as warned by MATLAB that the maximum residual at endpoint is very large with affordable mesh points), which indicates the necessity of dividing the interval [0,1] into several sub-intervals. In this experiment, we find when the number of sub-intervals $n \geq 4$, shooting with the initial guess (39) is able to find the geodesic on each sub-interval. Figure 1 shows the geodesic joining x_0 and x_1 generated by Newton's method (Algorithm 2) with $n = 5$ and $\epsilon = 10^{-4}$, where magenta and blue points represent initial and final junctions respectively, green and red curves represent initial and final piecewise geodesics respectively.

To compare Riemannian gradient descent (Algorithm 1) and Newton's method with the leapfrog algorithm (Algorithm 12), we vary the number of sub-intervals n from 4 to 9, set the step size $\eta = 0.001$ in Algorithm 1 and the time limitation for all

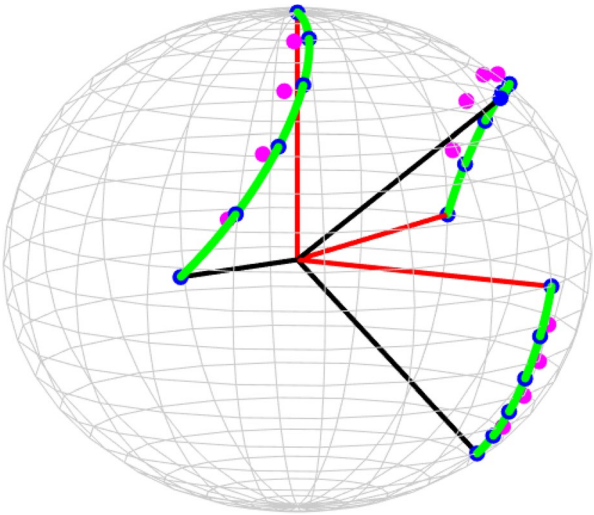

Fig. 2 The red and black frame represent rotation matrix X_0 and X_1 respectively. Geodesic or junctions on the manifold $SO(3)$ are represented by endpoints of frames rotated by elements in $SO(3)$. Magenta and blue points denote initial and final junctions respectively and green curve denotes the final piecewise geodesic

algorithms is 100 seconds. The runtime for different methods is reported in Table 1, from which we find Riemannian gradient descent is the slowest, leapfrog method is getting slower and slower as n increases, and Newton's method is the most efficient one in general.

Note that for a function f defined on $Ell(2)$, its Riemannian gradient $\partial_x^R f$ is related to its Euclidean gradient $\partial_x f$ by $\partial_x^R f = (I_3 - xx^T A)\partial_x f$. If we denote the operator $(d_x\text{Log})_{x,y}$ and $(d_y\text{Log})_{x,y}$ on $Ell(2)$ by matrices, then their adjoints are given by matrix transposes.

4.2 The special orthogonal group

Let $SO(3) := \{X \in \mathbb{R}^{3\times 3} | X^T X = I_3, \det(X) = 1\}$ be the 3-dimensional Lie group of positively-oriented 3×3 orthogonal matrices, with left-invariant metric given by

$$\langle V_1, V_2 \rangle_X := -\frac{1}{2}\text{tr}(X^T V_1 A X^T V_2). \tag{40}$$

Table 2 Comparison of runtime of different methods for finding geodesics

n	Algorithm 1	Leapfrog	Algorithm 2
4	42.0618	5.3832	1.2795
5	> 100	7.3309	1.3795
6	> 100	11.4557	1.5282
7	> 100	40.3261	1.8242
8	> 100	92.5328	1.9546

Here $V_1, V_2 \in T_X SO(3)$, $X \in SO(3)$, and A is a fixed symmetric positive-definite matrix. The tangent space of $SO(3)$ at X is given by $T_X SO(3) = \{V \in \mathbb{R}^3 | V^T X + X^T V = \mathbf{0}\}$. The Lie algebra is the tangent space at the identity, namely the space $\mathfrak{so}(3) = \{V \in \mathbb{R}^{3\times 3} | V^T = -V\}$ of skew-symmetric 3×3 matrices.

Note that Euclidean 3-space $\mathbb{E}^3$ is also a 3-dimensional Lie algebra, with respect to the cross-product $\times$, and there is a Lie isomorphism $w \in \mathbb{E}^3 \mapsto \hat{w} \in \mathfrak{so}(3)$ given by $\hat{w}v := w \times v$ for $w, v \in \mathbb{E}^3$. Here $\hat{w} = \begin{pmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{pmatrix}$, where $w = (w_1, w_2, w_3)^T \in \mathbb{E}^3$.

The normal space is given by $N_X SO(3) = \{XS | S \in \mathbb{R}^{3\times 3}, S^T = S\}$, which can be verified by the fact

$$\langle V, XS \rangle = \mathrm{tr}(V^T XS) = -\mathrm{tr}(X^T VS) = -\mathrm{tr}(SV^T X) = -\mathrm{tr}(V^T XS) = 0$$

for any $V \in T_X SO(3)$ and $XS \in N_X SO(3)$.

By the Lie isomorphism between $\mathbb{E}^3$ and $\mathfrak{so}(3)$, the bilinear form B defined in (30) with respect to the metric (40) can now be rewritten as

$$B(\hat{v}, \hat{w}) = A^{-1}((Av) \times w), \quad \forall \ v, w \in \mathbb{E}^3. \tag{41}$$

Then the geodesic equation is

$$\begin{cases} \dot{x} = x\hat{v}, \\ \dot{v} = A^{-1}((Av) \times v), \end{cases} \tag{42}$$

and the Jacobi equation is

$$\begin{aligned} \ddot{w} \ &+ \ v \times \dot{w} - A^{-1}((Av) \times \dot{w} + (A\dot{w}) \times v) - \tfrac{1}{2} w \times (A^{-1}((Av) \times v)) \\ &+ \ \tfrac{1}{2} A^{-1}((Aw) \times (A^{-1}((Av) \times v)) + ((Av) \times v) \times w) \\ &+ \ A^{-1}((Av) \times (w \times v) + (A(w \times v)) \times v) = \mathbf{0}, \end{aligned}$$

where $J = x\hat{w}$ and v is from the geodesic equation.

Given two nearby points X,$Y \in SO(3)$, the initial guess of the velocity of the geodesic from X to Y can be chosen as $\mathbf{Log}(X^T Y)$, where $\mathbf{Log}$ is the matrix logarithm.

For a function f defined on $SO(3)$, by the relationship between Riemannian gradient and Euclidean gradient (6), we find $\mathrm{tr}((XAX^T \partial_X f - \partial_X f)^T V) = 0$ for any $V \in T_X SO(3)$, which means $XAX^T \partial_X^R f - \partial_X f \in N_X SO(3)$. Suppose

$$XAX^T \partial_X^R f - \partial_X f = XS, \tag{43}$$

then left multiplying (43) by X^T gives $S = AX^T \partial_X^R f - X^T \partial_X f$. Since S is symmetric, we have

$$S = \tfrac{1}{2}(AX^T \partial_X^R f + (\partial_X^R f)^T XA - X^T \partial_X f - (\partial_X f)^T X). \tag{44}$$

Substituting (44) into (43) returns

$$AX^T\partial_X^R f + X^T\partial_X^R fA = X^T\partial_X f - (\partial_X f)^T X, \tag{45}$$

from which

$$\partial_X^R f = X\mathrm{Sy}(A, A, X^T\partial_X f - (\partial_X f)^T X), \tag{46}$$

where Sy(A,B,C) is the solution of the Sylvester equation $AX + XB = C$. Note that MATLAB's function *sylvester* or Mathematica's function *LyapunovSolve* can solve this algebraic equation efficiently.

Example 4 On a left-invariant special orthogonal group $SO(3)$ with $A = \mathrm{diag}\{1,2,4\}$, we choose X_0 as the identity I_3 and X_1 an orthogonal matrix $R_y(45)R_x(80)$, where $R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$, $R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$ are counterclockwise rotations around x-axis and y-axis by α and β degrees, respectively. As seen by experiment, solving the geodesic (42) with linear initial guess without choosing any junctions is unsuccessful. Then, 4 initial junctions are chosen on $SO(3)$, where magenta points in Fig. 2 represent endpoints of frame rotated by orthogonal matrices. After 3 iterations in Algorithm 2, the cost function (5) is less than 10^{-4}.

Similar to the experiment on ellipsoid, we compare the efficiency of three different methods, where runtime is shown in Table 2. The step size in Algorithm 1 is chosen to be 0.01,0.006,0.002,0.002,0.002 for $n = 4,\cdots,8$. The stop criteria for all algorithms is set to $\epsilon = 10^{-4}$. As seen from Table 2, Newton's method is very efficient when initial guess is sufficiently close to the optimal solution and the efficiency of leapfrog method drops quickly as number of sub-intervals increases.

5 Conclusions

Geodesics are fundamental in mathematics and in many applications, but finding a geodesic joining two given points is not easy, except in some very special cases. This paper proposes two new methods to tackle the problem using numerical methods. To find a geodesic $\gamma : [0,1] \to M$ on a finite dimensional path-connected Riemannian manifold M joining x_0 and x_1, we divide the interval [0,1] into subintervals and choose a small number of junctions in M, each moderately close to the next. On each subinterval we solve the geodesic equation between junctions via the shooting method. The purpose of having moderately close junctions is to provide good initial guesses for shooting. By (1) minimizing the difference between geodesic velocities associated with the junctions using the method of Riemannian gradient descent or (2) finding the singularity of the vector field associated with the junctions using Newton's method, we update all junctions until the total difference is relatively small. To find the Riemannian gradient of our cost function f or the covariant derivative of our vector field F,

it is necessary to solve the Jacobi equation. On a general Riemannian manifold: this equation can be rewritten in terms of Christoffel symbols if local coordinates are available or a embedding function if the codimension of the manifold is one. On a Lie group, the Jacobi equation can be reduced to a simpler differential equation in terms of a bilinear transformation of the Lie algebra. Finally, we test the effectiveness of the proposed methods by finding geodesics joining two given points in the 2-dimensional ellipsoid *Ell*(2) with Euclidean metric and in the special orthogonal group *SO*(3) with a left-invariant metric.

Appendix: Leapfrog algorithm for finding geodesics

The following algorithm is adapted from [8].

Algorithm 12 Leapfrog Algorithm For Finding Geodesics (LP-G)

Require: Two points $x_0, x_1 \in U \subset M$, number of sub-intervals n, maximum number of iterations N, stop error $\epsilon > 0$.

Ensure: Geodesic $\gamma : [0,1] \to M$ with $\gamma(0) = x_0$ and $\gamma(1) = x_1$.

Choose an arbitrary curve $\tilde{\gamma} : [0,1] \to U \subset M$ joining x_0 and x_1, and $y_i := \tilde{\gamma}(i/n)$, $i = 1, 2, \cdots, n-1$.
Replace $\tilde{\gamma}$ by a piecewise geodesic γ joining all y_i's via (single) shooting.
for $k \leftarrow 1$ to N **do**
    for $i \leftarrow 1$ to $n-1$ **do**
        Update y_i as the midpoint of the geodesic joining y_{i-1} and y_{i+1} obtained by shooting method.
    end for
    Calculate the value of the cost function (5).
    if $f(\mathbf{y}) \leqslant \epsilon$ **then**
        Update the piecewise geodesic γ based on current y_i's. Stop.
    end if
end for

Acknowledgements The authors would like to thank editors and two anonymous referees for their helpful suggestions and comments, which greatly improved the quality of the present work.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Fletcher, T.: Geodesic regression on Riemannian manifolds. In: Proceedings of the Third International Workshop on Mathematical Foundations of Computational Anatomy-Geometrical and Statistical Methods for Modelling Biological Shape Variability, pp. 75–86 (2011)
2. Fletcher, P.T.: Geodesic regression and the theory of least squares on Riemannian manifolds. Int. J. Comput. Vis. **105**(2), 171–185 (2013)
3. Fletcher, P.T., Lu, C., Pizer, S.M., Joshi, S.: Principal geodesic analysis for the study of nonlinear statistics of shape. IEEE Transactions on Medical Imaging **23**(8), 995–1005 (2004)
4. Fletcher, P.T., Lu, C., Joshi, S.: Statistics of shape via principal geodesic analysis on Lie groups. In: 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings, vol. 1, pp. I–I, IEEE (2003)
5. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. Int. J. Comput. Vis. **22**(1), 61–79 (1997)
6. Carmo, M.P.D.: Riemannian geometry. Birkhäuser (1992)
7. Keller, H.B.: Numerical methods for two-point boundary-value problems. Courier Dover Publications (2018)
8. Noakes, L.: A global algorithm for geodesics. J. Aust. Math. Soc. **65**(1), 37–50 (1998)
9. Kaya, C.Y., Noakes, J.L.: Geodesics and an optimal control algorithm. In: Proceedings of the 36th IEEE Conference on Decision and Control, vol. 5, pp. 4918–4919, IEEE (1997)
10. Kaya, C.Y., Noakes, J.L.: Leapfrog for optimal control. SIAM Journal on Numerical Analysis **46**(6), 2795–2817 (2008)
11. Nesterov, Y.: Smooth minimization of non-smooth functions. Mathematical Programming **103**(1), 127–152 (2005)
12. Nesterov, Y.: Accelerating the cubic regularization of Newton's method on convex problems. Math. Program. **112**(1), 159–181 (2008)
13. Wibisono, A., Wilson, A.C., Jordan, M.I.: A variational perspective on accelerated methods in optimization. Proc. Natl. Acad. Sci. **113**(47), E7351–E7358 (2016)
14. Krakowski, K.A., Noakes, L.: Geometrical methods of inference. University of Western Australia (2002)
15. Noakes, L.: Duality and Riemannian cubics. Adv. Comput. Math. **25**(1), 195–209 (2006)
16. Popiel, T., Noakes, L.: Elastica in SO (3). J. Aust. Math. Soc. **83**(1), 105–124 (2007)
17. Zhang, E., Noakes, L.: Left Lie reduction for curves in homogeneous spaces. Adv. Comput. Math. **44**(5), 1673–1686 (2018)
18. Zhang, E., Noakes, L.: Riemannian cubics and elastica in the manifold SPD (n) of all n× n symmetric positive-definite matrices. Journal of Geometric Mechanics, 11(2) (2019)
19. Fernandes, T.A., Ferreira, O.P., Yuan, J.: On the superlinear convergence of Newton's method on Riemannian manifolds. J. Optim. Theory Appl. **173**(3), 828–843 (2017)
20. Bortoloti, M.A.D.A., Fernandes, T.A., Ferreira, O.P., Yuan, J.: Damped Newton's method on Riemannian manifolds. J. Glob. Optim., pp. 1–18 (2020)
21. Bortoloti, M.A.D.A., Fernandes, T.A., Ferreira, O.P.: On the globalization of Riemannian Newton method. arXiv:2008.06557 (2020)
22. Rentmeesters, Q.: A gradient method for geodesic data fitting on some symmetric Riemannian manifolds. In: 2011 50th IEEE Conference on Decision and Control and European Control Conference, pp. 7141–7146, IEEE (2011)
23. Argyros, I.K., Hilout, S.: Newton's method for approximating zeros of vector fields on Riemannian manifolds. J. Appl. Math. Comput. **29**(1), 417–427 (2009)
24. Li, C., Wang, J.: Newton's method on Riemannian manifolds: Smale's point estimate theory under the γ-condition. IMA Journal of Numerical Analysis **26**(2), 228–251 (2006)
25. Li, C., Wang, J.H., Dedieu, J.P.: Smale's point estimate theory for Newton's method on Lie groups. J. Complex. **25**(2), 128–151 (2009)

26. Absil, P.A., Mahony, R., Sepulchre, R.: Optimization algorithms on matrix manifolds. Princeton University Press (2009)
27. Bertsekas, D.P.: Constrained optimization and Lagrange multiplier methods. Academic Press (2014)
28. Dennis, J.E. Jr, Schnabel, R.B.: Numerical methods for unconstrained optimization and nonlinear equations. Society for Industrial and Applied Mathematics (1996)
29. Zhang, E.: Magnetic cubics in Riemannian manifolds associated with different magnetic fields. Journal of Mathematical Physics **60**(12), 122703 (2019)
30. Zhang, E., Noakes, L.: Riemannian cubics in quadratic matrix Lie groups. Applied Mathematics and Computation **375**, 125082 (2020)
31. Gallier, J., Quaintance, J.: Differential geometry and Lie groups: a computational perspective. vol. 12, Springer Nature (2020)
32. Gallier, J., Quaintance, J.: Differential Geometry and Lie Groups A Second Course. Springer, Cham (2020)
33. Lee, J.M.: Introduction to Riemannian manifolds. Springer International Publishing, Cham (2018)

Authors and Affiliations

Lyle Noakes[1] · Erchuan Zhang[2]

Lyle Noakes
lyle.noakes@uwa.edu.au

[1] Department of Mathematics and Statistics, The University of Western Australia, Crawley WA 6009, Australia

[2] School of Science, Edith Cowan University, Joondalup WA 6027, Australia