Noname manuscript No.
(will be inserted by the editor)

Data fitting on manifolds with composite Bézier-like curves and blended cubic splines

Preprint: November 14, 2018

Pierre-Yves Gousenbourger · Estelle Massart · P.-A. Absil

Received: date / Accepted: date

Abstract We propose several methods that address the problem of fitting a C^1 curve γ to time-labeled data points on a manifold. The methods have a parameter, λ, to adjust the relative importance of the two goals that the curve should meet: being "straight enough" while fitting the data "closely enough". The methods are designed for ease of use: they only require to compute Riemannian exponentials and logarithms, they represent the curve by means of a number of tangent vectors that grows linearly with the number of data points, and, once the representation is computed, evaluating $\gamma(t)$ at any t requires a small number of exponentials and logarithms (independent of the number of data points). Among the proposed methods, the blended cubic spline technique combines the additional properties of interpolating the data when $\lambda \to \infty$ and reducing to the well-known cubic smoothing spline when the manifold is Euclidean. The methods are illustrated on synthetic and real data.

This work was supported by (i) the Fonds de la Recherche Scientifique – FNRS and the Fonds Wetenschappelijk Onderzoek – Vlaanderen under EOS Project no 30468160 and (ii) "Communauté française de Belgique - Actions de Recherche Concertées" (contract ARC 14/19-060).

P.-Y. Gousenbourger
Université catholique de Louvain
ICTEAM, Place du Levant, 2/L5.04.04 – 1348 Louvain-la-Neuve, Belgium
Tel.: +32-10-472307
E-mail: pierre-yves.gousenbourger@uclouvain.be

E. Massart · P.-A. Absil
Université catholique de Louvain
ICTEAM, Avenue Georges Lemaitre, 4/L4.05.01 – 1348 Louvain-la-Neuve, Belgium

Keywords Riemannian manifold · data fitting · curve fitting · interpolation · smoothing · blended cubic spline

1 Introduction

This paper concerns the problem of fitting a curve to data points on a manifold: we are given data points $d_0, \dots, d_n$ on a Riemannian manifold $\mathcal{M}$ as well as times $t_0 < \dots < t_n$, and we seek a curve $\gamma \colon [t_0, t_n] \to \mathcal{M}$ that strikes a balance between the conflicting goals of being "sufficiently straight" while passing "sufficiently close" to the data points at the given times. These intuitively formulated goals will be made precise momentarily.

1.1 Motivation

The data fitting problem is motivated by various applications that require to denoise or resample time-dependent—or more generally parameter-dependent—data on a Riemannian manifold.

For example, a crucial task in computational anatomy is to denoise and resample the evolution of the shape of an organ, yielding a curve fitting problem on a shape manifold. In Arnould *et al.* [3], for instance, different images of an organ are acquired in the space of closed 2D-shapes $\mathcal{S}$ at different body-depths, via MRI. Orientations of a probe are also registered at different times by 2D Transvaginal Ultrasound (TVUS) [4]. The final goal is to reconstruct the 3D volume $\gamma_{\text{MRI}}(z) \colon z \to \mathcal{S}$ of the imaged organ, where z is the depth inside the human body, but also the probe navigation path $\gamma_{\text{TVUS}}(t) \colon t \to \text{SE}(3)$.

Another application concerns interpolation or fitting of rotations of 3D objects lying on the special orthogonal group SO(3). This problem arises in robotics (for motion planning of rigid bodies) or in computer graphics, to animate 3D objects [27].

In projection-based model reduction of dynamical systems depending on one parameter, the computation of a suitable projector for a given value of the parameter requires usually a high computational effort. In Pyta *et al.* [30], the parameter is the Reynolds number and the dynamical system is the Navier-Stokes equations. Several projectors (i.e., elements of a Grassmann manifold) are computed with a Galerkin method for a small set of parameter values, and then curve fitting is used to approximate a new projector for other values of the Reynolds number.

We finally mention diffusion tensor imaging, where data points are elements of the manifold of 3×3 symmetric positive definite matrices [28], or the wind field modeling problem, where a wind field is characterized by a mean field and a covariance matrix C in the manifold $\mathcal{S}_+(p,r)$ of positive semidefinite matrices of size p and rank r [13].

1.2 State of the art

A well known strategy to handle the two conflicting goals (i.e., data fitting and smoothness) is to encapsulate them in an optimization problem

$$\min_{\gamma \in \Gamma} E_\lambda(\gamma) := \int_{t_0}^{t_n} \left\| \frac{\mathrm{D}^2\gamma(t)}{\mathrm{d}t^2} \right\|^2_{\gamma(t)} \mathrm{d}t + \lambda \sum_{i=0}^{n} \mathrm{d}^2(\gamma(t_i), d_i), \tag{1}$$

where the set Γ is an admissible set of curves on $\mathcal{M}$, $\frac{\mathrm{D}^2}{\mathrm{d}t^2}$ denotes the (Levi-Civita) second covariant derivative, $\|\cdot\|_{\gamma(t)}$ is the Riemannian metric at $\gamma(t)$, and $\mathrm{d}(\cdot,\cdot)$ is the Riemannian distance. The parameter λ (controlled by the user or by the algorithm itself, e.g., through a cross-validation procedure) sets the balance between the *regularizer* term $\int_{t_0}^{t_n} \|\frac{\mathrm{D}^2\gamma(t)}{\mathrm{d}t^2}\|^2_{\gamma(t)}\mathrm{d}t$ and the *goodness of fit* term $\sum_{i=0}^{n} \mathrm{d}^2(\gamma(t_i), d_i)$. When the Riemannian manifold $\mathcal{M}$ reduces to a Euclidean space and Γ is chosen to be the Sobolev space $H^2(t_0, t_n)$, a classical result (see, e.g., [14, Theorem 2.4]) states that the solution of (1) is a natural cubic spline. Specifically, it is the interpolating natural cubic spline when $\lambda \to \infty$ (see, e.g., [35] for its definition) and the least-squares linear regression as $\lambda \to 0$ (see [22, Proposition 4.5 and 4.6] for a result on manifolds).

Several methods exist to tackle problem (1) in general, and also more specifically in an imaging context. For instance, in Samir *et al.* [32], the problem is addressed in an infinite dimensional Sobolev space equipped with the Palais-metric, and (1) is minimized with a steepest-descent approach; that approach was applied to image processing by Su *et al.* [34]. In Boumal *et al.* [6], the curve is discretized with K points, reducing the search space to a (simpler) product manifold $\mathcal{M}^K$, and the covariant derivative is replaced by manifold-valued finite differences. Machado and Monteiro [23] handle the specific case of the sphere. More recently, Kim *et al.* [19] generalized to the shape space a technique from Jupp and Kent [17] called "unwrapping and unrolling"; this technique has the advantage to "transform" the manifold-valued problem to an equivalent Euclidean problem and solve it with classical Euclidean techniques.

The limit case where $\lambda = 0$, i.e., geodesic regression, was studied in the work of Rentmeesters [31] and Fletcher [12]. In the former, the problem is solved with a gradient descent technique for Riemannian symmetric spaces, while in the latter, the least-square problem is studied intrinsically as a minimization of the sum of squared geodesic distances on $\mathcal{M}$. An extension of geodesic regression is polynomial regression, for which Riemannian techniques were proposed by Hinkle *et al.* [15] (intrinsic method) and Lin *et al.* [21] (extrinsic methods).

We also mention the other limit case where $\lambda \to \infty$, studied by Arnould *et al.* [3] and Absil *et al.* [1]. In that work, the search space Γ is chosen to be a space of C^1 composite Bézier curves (resp. surfaces), and the optimality of the returned curve is guaranteed only when $\mathcal{M}$ is Euclidean. The advantage compared to [32,6] is that the search space is less complex, as the curves are only represented by a few control points on the manifold. Furthermore, an advantage compared to [32,19,12,31] consists in the simplicity of the proposed method. Indeed, only two objects on the manifold are required: the exponential map and the logarithm, while most of the techniques mentioned above require a gradient or heavy computations for parallel transportation. Worth mentioning is also the work of Modin *et al.* [25] on interpolation with C^2 composite cubic Bézier curves on Riemannian symmetric spaces, and the extension to bivariate interpolation in [1].

Finally, note that several curve fitting methods on manifolds do not fit the optimization framework (1), such as subdivision schemes [10,37] or Lie-algebraic methods [33].

1.3 Our contribution

In this paper, our purpose is to extend the C^1 composite Bézier curve *interpolation* method of [1,3] to a curve *fitting* method, where the data points do not need to be interpolated exactly (i.e., $\lambda \in (0,\infty)$). Among the several techniques developed in this paper, the "blended cubic spline" (see Section 5) stands out as it combines the following desirable properties:

(i) As $\lambda \to \infty$, the data points are interpolated at the given times;
(ii) The curve is of class C^1;
(iii) If the manifold $\mathcal{M}$ reduces to a Euclidean space, then the produced curve is the natural cubic spline that minimizes the energy function (1) over the (infinite-dimensional) Sobolev space $H^2(t_0, t_n)$;
(iv) The only knowledge that the method requires from the manifold is the Riemannian exponential (a function that solves the initial-value geodesic problem) and the Riemannian logarithm (a function that solves the endpoint geodesic problem);
(v) The produced curve is represented by $\mathcal{O}(n)$ tangent vectors to the manifold (or simply points on the manifold);
(vi) Computing $\gamma(t)$ for any given t requires $\mathcal{O}(1)$ exp and log operations once the representation by $\mathcal{O}(n)$ tangent vectors is available.

In a nutshell, the proposed blended cubic spline method consists in building polynomial pieces by solving the optimization problem (1) in various tangent spaces, and then blending together corresponding pieces by means of carefully chosen weights in order to satisfy all the above-mentioned properties.

Throughout the paper we assume that the fitting problem instance (i.e., the data points and associated times) is such that the considered algorithm evaluates the Riemannian exponentials and logarithms only where they are well defined (so that the resulting curve is well defined as well) and C^1 (so that property (ii) holds). This standing assumption always holds when $\mathcal{M}$ is a Hadamard manifold, and is in general not a concern when $\mathcal{M}$ is complete and the cut loci on $\mathcal{M}$ have codimension greater than one.

Though it stems from Bézier-related considerations, the method to construct the blended cubic spline fairly strongly departs from the composite Bézier approach used for interpolation in [1,3]. Indeed, it can be described without appealing to Bézier concepts such as the Bernstein basis polynomials and the De Casteljau algorithm, and the curve obtained at the limit $\lambda \to \infty$ is in general not the curve obtained by the interpolation method of [1,3]. We also present several Bézier-like curve fitting methods (see Section 6), but each of them fails to satisfy at least one of the above-mentioned properties. These Bézier-like methods are nevertheless worth considering because they are not necessarily inferior in practical applications (see Section 7). Moreover, in Section 4, we present a genuine composite Bézier fitting method (Algorithm 2) that satisfies properties (ii)–(vi), and also property (i) when the data points are not too far spread out. The various methods are compared in Section 7, where we also show that addressing the fitting problem in a single tangent space—another strategy that satisfies all the above-mentioned properties—is less satisfactory in practice than the proposed blending method.

1.4 Outline

The paper is organized as follows. Preliminaries on Bézier curves and Riemannian manifolds are given in Section 2. The background on C^1 piecewise-Bézier interpolation is recalled in Section 3. Then, a C^1 piecewise Bézier fitting technique is analysed in Section 4 as a generalization of the interpolation technique, and its limitations as an interpolation method when $\lambda \to \infty$ are highlighted. In Section 5, the proposed blending method (which fixes these limitations) is described and analyzed. Several Bézier-like alternatives are presented in Section 6. Numerical experiments are conducted in Section 7. A summary and some perspectives are drawn in Section 8.

2 Preliminaries

2.1 Euclidean composite Bézier curves

In this section, we briefly summarize the concept of Bézier curves in a Euclidean space $\mathbb{R}^m$. We also define the composite Bézier curve and the conditions needed to obtain C^1-continuity along this curve. Finally, we present the De Casteljau algorithm that evaluates Bézier curves in a recursive way and admits a well-known generalization to manifolds [29]. More details about Bézier curves can be found in [11].

Bézier curves in $\mathbb{R}^m$, defined next, are nothing else than polynomials expressed in a particular basis.

Definition 2.1 (Bézier curve)**.** Consider a sequence of control points $b_0, \ldots, b_K \in \mathbb{R}^m$, $K \in \mathbb{N}$. The *Bézier curve* $\beta_K\colon [0,1] \to \mathbb{R}^m$ of degree K is defined as

$$\beta_K(\cdot; b_0, \ldots, b_K)\colon [0,1] \to \mathbb{R}^m, t \mapsto \sum_{j=0}^{K} b_j B_{jK}(t), \qquad (2)$$

where $B_{jK}(t) = \binom{K}{j} t^j (1-t)^{K-j}$ are the Bernstein basis polynomials. In particular, the cubic Bézier curve $\beta_3 = \beta_3(t; b_0, b_1, b_2, b_3)$ driven by the control points b_0, b_1, b_2, b_3 is given by

$$\beta_3 := b_0(1-t)^3 + 3b_1(1-t)^2 t + 3b_2(1-t)t^2 + b_3 t^3. \quad (3)$$

The Bézier curve interpolates its first and last control points, and its velocity (i.e., time-derivative) $\dot\beta_K$ at $t=0$ (resp. $t=1$) is tangent to the segment joining its two first (resp. two last) control points. In short:

$$\beta_K(0;b_0,\dots,b_K)=b_0, \tag{4}$$
$$\beta_K(1;b_0,\dots,b_K)=b_K, \tag{5}$$
$$\dot\beta_K(0;b_0,\dots,b_K)=K(b_1-b_0), \tag{6}$$
$$\dot\beta_K(1;b_0,\dots,b_K)=K(b_K-b_{K-1}). \tag{7}$$

A simple algorithm to evaluate $\beta_K(t;b_0,\dots,b_K)=:x_0^K$ at time $t\in[0,1]$ is the so called De Casteljau algorithm. As it is based only on convex combinations of two points, it has a simple geometric interpretation, as represented in Figure 1.

Definition 2.2 (De Casteljau)**.** The De Casteljau algorithm on $\mathbb{R}^m$ reads

$$\begin{aligned} x_i^0 &:= b_i && \text{for } i=0,\dots,K\\ x_j^k &:= (1-t)x_j^{k-1}+tx_{j+1}^{k-1} && \text{for } k=1,\dots,K\\ &&& \text{and } j=0,\dots,K-k. \end{aligned}$$

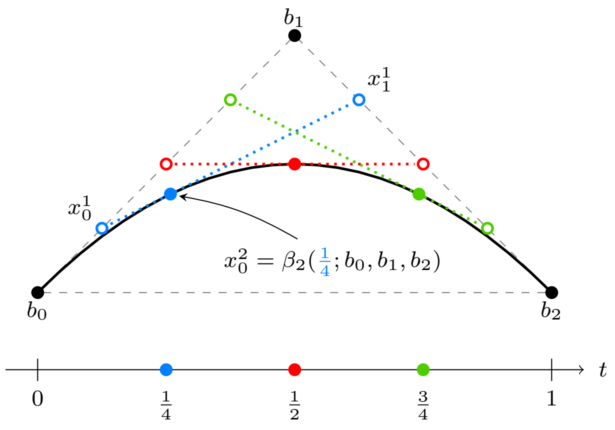


Fig. 1: Geometric interpretation of the De Casteljau algorithm [11] applied to a quadratic Bézier curve $\beta_2(t;b_0,b_1,b_2)\in\mathbb{R}^2$. At each step, one computes a convex combination of two consecutive points.

Definition 2.3 (Composite Bézier curve)**.** Consider now a sequence $(\beta_{K_i}^i)_{i=0}^{N-1}$ of N Bézier curves of degree K_i, $i=0,\dots,N-1$, determined by control points $b_0^i,\dots,b_{K_i}^i\in\mathbb{R}^m$. The *composite Bézier curve* $\mathbf{B}$ is defined as

$$\mathbf{B}\colon[0,N]\to\mathbb{R}^m,\ t\mapsto\beta_{K_i}^i(t-i;b_0^i,\dots,b_{K_i}^i),\ i=\lfloor t\rfloor, \tag{8}$$

where $\lfloor t\rfloor$ is the largest integer $i\le t$, with an exception for $\lfloor N\rfloor:=N-1$.

Remark 2.4. For simplicity, we restrict to composite curves whose pieces are defined on intervals $[i,i+1]$, $i\in\mathbb{Z}$, but the step to define them on any interval $[t_i,t_{i+1}]$ is direct.

The next proposition follows directly from (4) and (5).

Proposition 2.5 (Continuity of the composite Bézier curve)**.** *The composite Bézier curve made of N segments is continuous if the last and first control points of every two consecutive Bézier curves are the same. We introduce them as $p_i:=b_{K_i}^i=b_0^{i+1}$, $i=0,\dots,N-2$.*

In addition to continuity, we state now conditions for $\mathbf{B}$ to be continuously differentiable at $t=i$ (differentiability is ensured on the rest of the domain since Bézier curves are polynomials). We introduce the notations b_i^- for the second to last control point of the $(i-1)^{\text{th}}$ Bézier curve of $\mathbf{B}$ (namely, $b_{K_{i-1}-1}^{i-1}$), and b_i^+ for the second control point of the i^{th} Bézier curve (i.e., b_1^i). The conditions follow from (6) and (7).

Proposition 2.6 (Differentiability of the composite Bézier curve)**.** *The composite Bézier curve is differentiable at $t=i$ if the following C^1-conditions hold:*

$$p_i=\frac{K_{i-1}b_i^-+K_ib_i^+}{K_{i-1}+K_i},\quad i=1,\dots,N-1. \tag{9}$$

Geometrically, this condition means that p_i, b_i^- and b_i^+ must be aligned.

Example 2.7 (C^1 composite cubic Bézier curve). A composite cubic Bézier curve $\mathbf{B}$ is represented in Figure 2. $\mathbf{B}\colon[0,5]\to\mathbb{R}$ is composed of five cubic Bézier curves and is defined as $\mathbf{B}(t)=\beta_3(t-i;p_i,b_i^+,b_{i+1}^-,p_{i+1})$ for $i=\lfloor t\rfloor$. Continuity is trivial. Continuous differentiability is given by $p_i=0.5(b_i^-+b_i^+)$, $i=1,\dots,4$.

2.2 Riemannian Manifolds

Before reviewing the generalization of Bézier curves to manifolds in Section 2.3, we now introduce the necessary concepts of Riemannian geometry. Detailed information on Riemannian manifolds can be found for example in [2,8,26].

Consider a Riemannian manifold $\mathcal{M}$. $T_a\mathcal{M}$ denotes the (Euclidean) tangent space to $\mathcal{M}$ at $a\in\mathcal{M}$; the tangent bundle to $\mathcal{M}$ is denoted by $T\mathcal{M}$. Tangent spaces on Riemannian manifolds are endowed with a smoothly varying inner product. Let $\langle\cdot,\cdot\rangle_a$ denote the inner product between two tangent vectors $\xi,\eta\in T_a\mathcal{M}$, with $a\in\mathcal{M}$ and let $\|\xi\|_a=\sqrt{\langle\xi,\xi\rangle_a}$ be the corresponding norm.

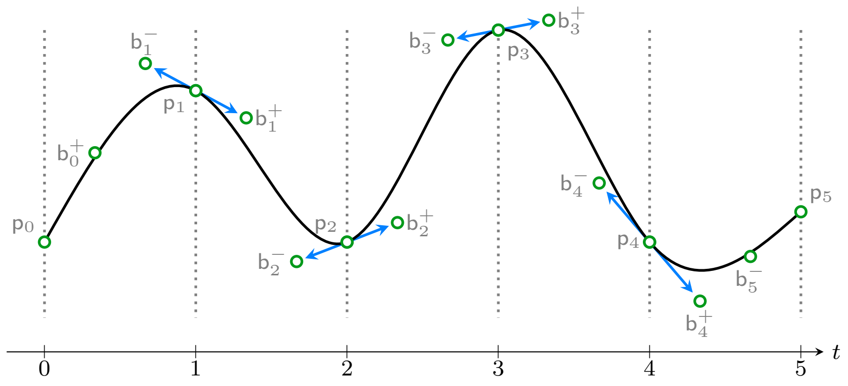


Fig. 2: Schematic representation of the composite cubic Bézier curve $\mathbf{B}\colon \mathbb{R} \to \mathcal{M}$, for $\mathcal{M} = \mathbb{R}$. The control points (green circles) fully determine the curve; continuous differentiability is reached at the junction p_i of the segments if the consecutive control points (b_i^-, p_i, b_i^+), $i = 1, \ldots, n-1$, are aligned on a geodesic (the blue arrows draw the first derivative of $\mathbf{B}$).

It is now possible to define the length and the energy of a path $\gamma\colon [0,1] \to \mathcal{M}$ as

$$L_\gamma = \int_0^1 \sqrt{\langle \dot\gamma(t), \dot\gamma(t)\rangle_{\gamma(t)}}\mathrm{d}t \tag{10}$$

and

$$E_\gamma = \int_0^1 \langle \dot\gamma(t), \dot\gamma(t)\rangle_{\gamma(t)}\mathrm{d}t, \tag{11}$$

where $\dot\gamma$ stands for the time-derivative of the curve γ. The paths γ minimizing (11) subject to the endpoint conditions $\gamma(0) = a$ and $\gamma(1) = b$ are termed *minimizing geodesics*, and their length is the *Riemannian distance* $\mathrm{d}_\mathcal{M}(a,b)$. When it is unique (a generic property when the manifold is complete [16]), the minimizing geodesic is denoted by $g(t; a, b)$ (note that, for any pair of points outside a set of measure zero, this unique minimizing geodesic exists [16]). A *geodesic* is a locally length-minimizing curve. Note that, as $g(t; a, b)$ minimizes (11), it also minimizes (10) as mentioned in [8, Chapter 3].

We denote by $\exp_a\colon T_a\mathcal{M} \to \mathcal{M}, \xi \mapsto b = \exp_a(\xi)$, the Riemannian exponential map, where b is the point attained at time $t = 1$ by the geodesic γ satisfying $\gamma(0) = a$ and $\dot\gamma(0) = \xi$. This map is bijective from $\mathrm{D}_a := \{\xi \in T_a\mathcal{M} : \|\xi\|_a < r_a\}$ to $\mathcal{D}_a := \{b \in \mathcal{M} : d_\mathcal{M}(a,b) < r_a\}$, where $r_a \in \mathbb{R}$ is called the injectivity radius at a. The inverse exponential map is called the Riemannian logarithm map. It is noted $\log_a\colon \mathcal{D}_a \to T_a\mathcal{M}, b \mapsto \xi = \log_a(b)$ and provides the initial velocity $\xi = \dot g(t; a, b)|_{t=0} \in T_a\mathcal{M}$ of the geodesic $g(t; a, b)$. Therefore, for $a \in \mathcal{M}$, $b \in \mathcal{D}_a$, and under the standing assumption from the introduction, one can compute the minimizing geodesic between a and b as $g(t; a, b) = \exp_a(t\log_a(b))$. Note that $\exp_a \circ \log_a$ is the identity on $\mathcal{D}_a$ (and actually everywhere on $\mathcal{M}$ except on the cut locus of a), and $\log_a \circ \exp_a$ is the identity on D_a.

Finally, we introduce notions related to the weighted geodesic average between two points $x_1, x_2 \in \mathcal{M}$ and for $w \in [0,1]$, noted $y = \mathrm{av}[(x_1, x_2), (1-w, w)]$. This average corresponds to the value $y \in \mathcal{M}$ that solves $\min_{y\in\mathcal{M}}(1-w)\mathrm{d}^2_\mathcal{M}(x_1, y) + w\mathrm{d}^2_\mathcal{M}(x_2, y)$ (if this minimizer is unique and exists). In has been shown [1, Remark 3], that this average reduces to the geodesic evaluated at $t = w$, i.e.,

$$\mathrm{av}[(x_1, x_2), (1-w, w)] = \exp_{x_1}\big(w\log_{x_1}(x_2)\big)\,.$$

The multigeodesically convex hull $\mathrm{co}(U)$ of a set $U \subset \mathcal{M}$ is the smallest set $S \subset \mathcal{M}$ containing U, such that it contains any weighted geodesic average between any of its points. This set $\mathrm{co}(U)$ is called *proper* if the weighted averages between any finitely many points in U are unique and smoothly depend on their weights and points [1].

2.3 Bézier curves on manifolds

We can now recall the definition of composite Bézier curves on manifolds. Crouch [9] and Lin and Walker [20] proposed a generalization of the De Casteljau algorithm on Lie Groups and on Riemannian manifolds. Furthermore, the C^1 conditions on the composite Bézier curve can be obtained thanks to the works of Popiel and Noakes [29]. We briefly review these results.

The following definition generalizes the Bézier curves to manifolds through the De Casteljau algorithm (Definition 2.2). It only requires Riemannian exponentials and logarithms.

Definition 2.8 (Bézier curve on manifold)**.** Consider a set of control points $b_0, \ldots, b_K \in \mathcal{M}$, $K \in \mathbb{N}$, such that the convex hull $\mathrm{co}(b_0, \ldots, b_K)$ is proper. The *Bézier* *curve* $\beta_K(\cdot; b_0, \dots, b_K)\colon [0,1] \to \mathcal{M}$ is defined recursively at t by

$$\begin{aligned} x_i^0 &:= b_i && \text{for } i = 0, \dots, K \\ x_j^k &:= \exp_{x_j^{k-1}}\left(t \log_{x_j^{k-1}} x_{j+1}^{k-1}\right) && \text{for } k = 1, \dots, K \\ & && \text{and } j = 0, \dots, K-k. \end{aligned}$$

Finally, $\beta_K(t; b_0, \dots, b_K) := x_K^0$. Note that, the exp-log formulation of x_j^k gives a geodesic between x_j^{k-1} and x_{j+1}^{k-1}, and replaces the straight line from Definition 2.2.

Popiel and Noakes [29] have shown that properties (4)–(7) carry over to manifolds in the following form:

$$\beta_K(0; b_0, \dots, b_K) = b_0, \tag{12}$$
$$\beta_K(1; b_0, \dots, b_K) = b_K, \tag{13}$$
$$\dot\beta_K(0; b_0, \dots, b_K) = K \log_{b_0}(b_1), \tag{14}$$
$$\dot\beta_K(1; b_0, \dots, b_K) = -K \log_{b_K}(b_{K-1}). \tag{15}$$

These properties confirm that the first and last control points of the manifold-valued Bézier curve are interpolated. They also mean that the first (resp. last) temporal derivative of the curve is tangent to the geodesic joining the two first (resp. two last) control points of the curve. We can now generalize the composite Bézier curve to manifolds.

Definition 2.9 (Composite Bézier curve on manifold)**.** Consider the sequence of control points $(p_i = b_0^i, b_i^+ = b_1^i, \dots, b_{i+1}^- = b_{K_i-1}^i, p_{i+1} = b_{K_i}^i)_{i=0}^{N-1} \in \mathcal{M}$, defining a sequence of N manifold-valued Bézier curves $(\beta_{K_i}^i)_{i=0}^{N-1}$ with $\beta_{K_i}^i\colon [0,1] \to \mathcal{M}$ for all i. The composite Bézier curve $\mathbf{B}\colon [0,N] \to \mathcal{M}$ is defined analogously to the Euclidean case, i.e., according to equation (8).

The next proposition states the conditions for a composite Bézier curve to be of class C^1. Along with the fact that Definition 2.8 only requires exponential and logarithm maps, the simplicity of these C^1 conditions is the main motivation for resorting to composite Bézier curves in order to generalize C^1 piecewise-polynomial curves to manifolds.

Proposition 2.10 (Continuity and differentiability)**.** *The composite Bézier curve composed of N segments on a manifold $\mathcal{M}$ is continuous if $b_{K_i}^i = b_0^{i+1}$, for $i = 0, \dots, N-2$ (the notation p_i introduced in Definition 2.9 thus anticipated the continuity condition). As mentioned in [3, 1], it will be C^1-continuous if*

$$\frac{1}{K_{i-1}} \log_{p_i}\left(b_i^-\right) = -\frac{1}{K_i} \log_{p_i}\left(b_i^+\right), \quad i = 1, \dots, N-1. \tag{16}$$

Similarly to (9), this condition holds if p_i is on a geodesic between b_i^+ and b_i^- and satisfies

$$p_i = g\left(\tfrac{K_{i-1}}{K_{i-1}+K_i}; b_i^-, b_i^+\right).$$

Example 2.11 (C^1 composite cubic Bézier curve on manifold). We consider the same composite cubic Bézier curve as in Example 2.7, but now defined on a Riemannian manifold $\mathcal{M}$. The curve is defined as $\mathbf{B}\colon [0,5] \to \mathcal{M}\colon \mathbf{B}(t) = \beta_3(t-i; p_i, b_i^+, b_{i+1}^-, p_{i+1})$, for $i = \lfloor t \rfloor$. Continuity is again trivial. Continuous differentiability is verified if

$$p_i = g(0.5; b_i^-, b_i^+), \quad i = 1, \dots, 4. \tag{17}$$

3 Interpolation with composite Bézier curves

Interpolation on manifolds with composite Bézier functions was studied in previous papers. In the approach of Arnould *et al.* [3], the composite Bézier curve interpolates n data points by means of $n-1$ Bézier curves. The first and last ones are *quadratic* Bézier curves while all the others are *cubic*. In Absil *et al.* [1], the question of surfacic (i.e., depending on two parameters) Bézier interpolation is treated.

In this section, we consider interpolation with a composite cubic Bézier curve. The method to determine the control points of the $n-1$ segments of Bézier is similar to [3] but the main difference is that all pieces are now cubic curves, such that the final composite Bézier curve is the natural cubic spline when the manifold is a Euclidean space. In short, the approach works as follows. Given the manifold-valued data points $d_0, \dots, d_n$ at parameter-values $t_0, \dots, t_n$, one seeks the interpolating C^1 composite cubic Bézier curve $\mathbf{B}\colon [t_0, t_n] \to \mathcal{M}$ minimizing (1) for $\lambda \to \infty$, i.e.,

$$\min_{\mathbf{B}} \int_{t_0}^{t_n} \left\| \frac{\mathrm{D}^2 \mathbf{B}(t)}{\mathrm{d}t^2} \right\|_{\mathbf{B}(t)}^2 \mathrm{d}t \quad \text{s.t. } \mathbf{B}(t_i) = d_i. \tag{18}$$

Without loss of generality, we will assume throughout this section that the time parameters $t_i = i$, $i = 0, \dots, n$, and that

$$\mathbf{B}(t) = \beta_3(t-i; d_i, b_i^+, b_{i+1}^-, d_{i+1}), \; i = \lfloor t \rfloor. \tag{19}$$

As the first and last control points of each segment are the data points, interpolation is ensured by equations (12) and (13). As a result, the optimization problem reduces to finding the remaining control points $b_i^+, b_{i+1}^- \in \mathcal{M}$, $i = 0, \dots, n-1$, that minimize the mean square acceleration of $\mathbf{B}$ under the differentiability conditions (16). This problem, however, leads to a time-consuming non-linear constrained optimization problem on manifolds. Therefore, we proceed as follows: we consider the case where $\mathcal{M} = \mathbb{R}^m$ and compute the optimal (Euclidean) control points (b_i^+, b_{i+1}^-), $i = 0, \ldots, n-1$, of $\mathbf{B}$ such that (18) is minimized. Since the composite cubic Bézier form encompasses all cubic splines, the optimal control points correspond to the interpolating natural cubic (which is known to minimize the mean squared acceleration under interpolation conditions [14, Theorem 2.3]), and thus satisfy the differentiability conditions of Proposition 2.6. The optimality conditions on the control points are then generalized to a manifold $\mathcal{M}$. The resulting C^1 interpolating composite cubic Bézier curve is not guaranteed to have a minimal mean squared acceleration, even among all C^1 interpolating composite cubic Bézier curve. However, by construction, it is optimal when the manifold is Euclidean. Hence, it can be expected to be close to optimal when the manifold is "sufficiently flat".

3.1 Computation of the control points

Let us consider the case $\mathcal{M} = \mathbb{R}^m$ and the data points $d_0, \ldots, d_n \in \mathbb{R}^m$ at parameter-values $t_i = i$, $i = 0, \ldots, n$. The optimization problem (18) becomes

$$\min_{\Gamma} \int_0^n \|\ddot{\mathbf{B}}(t)\|_2^2 \mathrm{d}t, \tag{20}$$

as the covariant second derivative $\frac{D^2\cdot}{\mathrm{d}t^2}$ is the classical time-derivative on $\mathbb{R}^m$, and the manifold-valued norm is the classical 2-norm. The search space Γ is reduced to the space of composite cubic Bézier curves $\Gamma_{\mathbf{B}}$ on $\mathbb{R}^m$ that interpolate the data points, namely,

$$\Gamma_{\mathbf{B}} = \{\mathbf{B}(t) = \beta_3(t - i; d_i, b_i^+, b_{i+1}^-, d_{i+1}),\ i = \lfloor t \rfloor\}.$$

The remaining optimization variables are

$$\Gamma'_{\mathbf{B}} = \{b_0^+, b_1^-, \ldots, b_{n-1}^+, b_n^-\}.$$

The differentiability constraints (16), on $\mathbb{R}^m$, read $b_i^+ = 2d_i - b_i^-$ (Proposition 2.6). Hence, the only remaining degrees of freedom of the Euclidean problem are $\{b_0^+, b_i^-\}$, $i = 1 \ldots, n$, which will be our optimization variables.

In summary, the optimization problem reduces to

$$\min_{\Gamma'_{\mathbf{B}}} \int_0^n \|\ddot{\mathbf{B}}(t)\|_2^2 \mathrm{d}t,\ \text{s.t. } b_i^+ = 2d_i - b_i^-, \tag{21}$$

for $i = 1, \ldots, n-1$.

This problem is actually quadratic in its $n+1$ variables, as $\mathbf{B}(t)$ depends linearly on the control points. Furthermore, it can be split into m smaller scalar problems in each of the components of $\mathbb{R}^m$. The optimality conditions of (21) take then the form of m independent linear systems, that we formulate as $A \cdot X = C \cdot D$, where $X = [b_0^+, b_1^-, b_2^-, \ldots, b_n^-]^T \in \mathbb{R}^{(n+1)\times m}$ contains the remaining optimization variables, $D = [d_0, \ldots, d_n]^T \in \mathbb{R}^{(n+1)\times m}$ contains the data points of the problem, and $A, C \in \mathbb{R}^{(n+1)\times(n+1)}$ are matrices of coefficients. The details are given in Appendix A.1.

The solution of this linear system is given by

$$X = A^{-1} \cdot C \cdot D =: W \cdot D.$$

In other words, each control point x_i of X is obtained as a simple weighted sum of the data points $(d_i)_{i=0}^n$:

$$x_i = \sum_{j=0}^n w_{ij} d_j, \qquad i = 0, \ldots, n. \tag{22}$$

To generalize this result to Riemannian manifolds, we do not solve the manifold-valued version of (21) directly (note that very recent work about this approach can be found in [5]), but we rather generalize the optimality conditions (22) to get a formula to compute the control points on $\mathcal{M}$. Note that the resulting piecewise-Bézier curve is not guaranteed to be a solution of (18), except if $\mathcal{M}$ is flat (see Proposition (3.2)).

To do so, we observe that $\sum_{j=0}^n w_{ij} = 1$. This property can be deduced from the fact that the problem is invariant to translations. Indeed, one can write identically the shifted optimality conditions (22) as

$$x_i - d_{\text{ref}} = \sum_{j=0}^n w_{ij}(d_j - d_{\text{ref}}), \qquad i = 0, \ldots, n,$$

for any d_{ref}. These differences can be interpreted as Riemannian logarithms $\log_a(b) = b - a$ on the Euclidean space. In other words, one can interpret the shifting of the optimality conditions as a mapping of the point x_i to the tangent space $T_{d_{\text{ref}}}\mathcal{M}$ at d_{ref}.

Consider now a general manifold $\mathcal{M}$, the list of data points $d_0, \ldots, d_n \in \mathcal{M}$, the search space $\Gamma'_{\mathbf{B}} \subseteq \mathcal{M}^{2n}$ and the remaining optimization variables $x_i \in \mathcal{M}$, $i = 0, \ldots, n$. The optimality conditions (22) are naturally generalized as

$$\tilde{x}_i = \log_{d_{\text{ref}}}(x_i) = \sum_{j=0}^n w_{ij} \log_{d_{\text{ref}}}(d_j) \in T_{d_{\text{ref}}}\mathcal{M}, \tag{23}$$

for $i = 0, \ldots, n$. Then, $x_i = \exp_{d_{\text{ref}}}(\tilde{x}_i)$.

The remaining task consists now in choosing a suitable reference point d_{ref} for each variable x_i. A first possible choice is to use the same reference point for all variables. However, for points far from d_{ref}, the tangent space $T_{d_{\text{ref}}}\mathcal{M}$ will generally be a bad approximation of $\mathcal{M}$, and the resulting curve may be of poor quality. Instead, we propose to choose a different reference point for each x_i, such that the most important data points averaged in equation (23) (i.e., the data points associated with the largest weights w_{ij}) are well approximated on $T_{d_{\mathrm{ref}}}\mathcal{M}$. In the case of the control point $x_i = b_i^-$, $i = 1, \dots, n$, the largest weight is w_{ii}, so $d_{\mathrm{ref}} = d_i$. For $x_0 = b_0^+$, the same rule is applied and $d_{\mathrm{ref}} = d_0$.

In view of the differentiability constraints (16), we get the other control points as

$$b_i^+ = \exp_{d_i}(-\tilde{x}_i), \quad i = 1, \dots, n-1, \tag{24}$$

and the final composite Bézier curve $\mathbf{B}$ is reconstructed with the De Casteljau algorithm (see Definition 2.8).

The previous development results in the following definition for an interpolating composite cubic Bézier curve on manifold.

Definition 3.1 (Interpolating C^1 composite cubic Bézier curve on manifold)**.** The proposed composite cubic Bézier curve $\mathbf{B}\colon [0,n] \to \mathcal{M}$, interpolating the manifold-valued data points $d_0, \dots, d_n$ at parameter values $t_0 = 0, \dots, t_n = n$, is defined as

$$\mathbf{B}(t) = \beta_3(t-i; d_i, b_i^+, b_{i+1}^-, d_{i+1}), \qquad i = \lfloor t \rfloor,$$

where β_3 is as in Definition 2.8. The control points b_0^+, b_i^- for $i = 1, \dots, n$ are defined by (23), with $d_{\mathrm{ref}} = d_i$, while the remaining control points b_i^+, $i = 1, \dots, n-1$, are defined by the C^1 conditions (24).

The full algorithm to generate the curve of Definition 3.1 is presented in Algorithm 1.

Algorithm 1 Interpolating C^1 composite cubic Bézier curve approaching the solution of (18).

```
Require: $d_0, \dots, d_n \in \mathcal{M}$, $A$ and $C$ from Appendix A.1
  Init: $s_0 = \dots s_n = 0$.
  $W \leftarrow A^{-1}C$                                % matrix of weights
  for $i = 0, \dots, n$ do
    $d_{\mathrm{ref}} \leftarrow d_i$                     % reference point
    for $j = 0, \dots, n$ do
      $s_j \leftarrow \log_{d_{\mathrm{ref}}}(d_j)$       % mapping to $T_{d_{ref}}\mathcal{M}$
    end for
    $\tilde{x} \leftarrow \sum_{k=0}^{n} w_{ik} s_k$      % cp generation
    $x \leftarrow \exp_{d_{\mathrm{ref}}}(\tilde{x})$     % mapping to $\mathcal{M}$
    if $i \neq 0$, $i \neq n$ then
      $b_i^- \leftarrow x$
      $b_i^+ \leftarrow \exp_{d_{\mathrm{ref}}}(-\tilde{x})$   % $C^1$ condition (16)
    else if $i = 0$ then
      $b_0^+ \leftarrow x$
    else $\{i = n\}$
      $b_n^- \leftarrow x$
    end if
  end for
  % De Casteljau algorithm (Definition 2.8)
  $\mathbf{B}(t) = \beta_3(t-i; d_i, b_i^+, b_{i+1}^-, d_{i+1})$,   $i = \lfloor t \rfloor$.
```

3.2 Properties of the interpolating curve

As mentioned in the introduction, we seek a C^1 interpolating curve that results in a low value of the cost function of (18). We see here how the curve of Definition 3.1 fulfills these specifications.

Proposition 3.2 (Natural cubic spline)**.** *For $\mathcal{M} = \mathbb{R}^m$, the interpolating composite cubic Bézier curve $\mathbf{B}\colon [0,n] \to \mathbb{R}^m$ of Definition 3.1 is the natural cubic spline that minimizes the mean square acceleration under the interpolation conditions.*

Proof. By construction. □

Theorem 3.3 (C^1 interpolation)**.** *Consider the data points $d_0, \dots, d_n \in \mathcal{M}$ associated with parameter values $t_i = i$, $i = 0, \dots, n$. The composite cubic Bézier curve $\mathbf{B}(t)\colon [0,n] \to \mathcal{M}$ of Definition 3.1 fulfills the following properties:*

(i) $\mathbf{B}(i) = d_i$, *$i = 0, \dots, n$;*
(ii) $\mathbf{B}(t)$ *is differentiable at $t \in [0,n]$ if $\|\tilde{x}_i\| < r_{d_i}$, $i = 1, \dots, n-1$.*

Proof. *(i)* follows from (12) and (13) (Property 2.10). Let us prove *(ii)*. By definition, the Bézier curve is differentiable on its domain [29]. Therefore, there only remains to prove that the *composite* Bézier curve is differentiable at $t = i$, $i = 1, \dots, n-1$, i.e., at the points where two consecutive segments join. Consider $i \in \{1, \dots, n-1\}$. As $\|\tilde{x}_i\| < r_{d_i}$, we have that $\log_{d_i}\left(b_i^-\right) = \log_{d_i}\left(\exp_{d_i}(\tilde{x}_i)\right) = \tilde{x}_i = -\log_{d_i}\left(\exp_{d_i}(-\tilde{x}_i)\right) = -\log_{d_i}\left(b_i^+\right)$. By equations (14) and (15), one has

$$\left.\frac{\mathrm{d}\mathbf{B}(t)}{\mathrm{d}t}\right|_{t=i^-} = \left.\frac{\mathrm{d}\mathbf{B}(t)}{\mathrm{d}t}\right|_{t=i^+}.$$

□

Proposition 3.4 (Minimal representation of the interpolating curve)**.** *The interpolating curve (Definition 3.1) is uniquely represented by $n+1$ tangent vectors.*

Proof. All the control points mentioned in Definition 3.1 are obtained from the $n+1$ tangent vectors $\tilde{x}_i \in T_{d_i}\mathcal{M}$, $i = 0, \dots, n$, as stated in the relevant parts of Algorithm 1. □

Proposition 3.5 (Exponential and logarithm maps required)**.** *The number of exponential and logarithm maps required to compute the interpolating curve (Definition 3.1) is*

- *$n(n+1)$ logarithms for the construction of the minimal representation of the curve of Proposition 3.4;*

- 8 *exponential and* 6 *logarithm maps to reconstruct the curve* $\mathbf{B}(t)$*, for a given parameter value t, given that minimal representation.*

Proof. The $n(n+1)$ logarithms maps are required for the evaluation of equation (23). Then, the reconstruction of the curve at a given time t requires 1 exponential map for $b^-_{\lfloor t\rfloor}$, 1 exponential map for $b^+_{\lfloor t\rfloor}$, and 6 additional exponential and logarithm maps for the De Casteljau algorithm. $\square$

4 Fitting with composite Bézier curves

In [13], composite Bézier curves were used to fit manifold-valued data points. The method developed is based on the same framework as [3], i.e., quadratic Bézier curves for the first and last segment, and cubic Bézier curves for inner segments.

In this section, we generalize the result of Section 3 to the fitting problem. We are again given $n+1$ data points $d_0,\dots,d_n$ on a Riemannian manifold $\mathcal{M}$, at times $t_i = i$, $i = 0,\dots,n$. We seek now the C^1 composite cubic Bézier curve $\mathbf{B}\colon [0,n] \to \mathcal{M}$ defined as

$$\mathbf{B}(t) = \beta_3(t-i; p_i, b^+_i, b^-_{i+1}, p_{i+1}),\ i = \lfloor t\rfloor, \tag{25}$$

minimizing (1) for $\lambda > 0$. Note that, compared to Section 3, the data points d_i are no longer the end-points of the Bézier curves.

Instead of solving the problem (1) directly, we use, similarly as in the previous section, a suboptimal route: we first determine optimality conditions on the control points of $\mathbf{B}$, such that (1) is minimized when $\mathcal{M}$ is Euclidean. Then, we generalize these conditions to any manifold $\mathcal{M}$. However, we will see that this method sometimes fails to satisfy the first property we required in the introduction, namely, the fact that the curve obtained should interpolate the data points when $\lambda \to \infty$.

4.1 Control points generation for cubic Bézier curves

Similarly as in Section 3, one needs to determine the position of the $3n+1$ control points of $\mathbf{B}$.

We consider the Euclidean case $\mathcal{M} = \mathbb{R}^m$ and the data points $d_i \in \mathbb{R}^m$ corresponding to parameter values $t_i = i$, $i = 0,\dots,n$. Now, we also consider $\lambda > 0$, the regularization parameter. The problem (1) becomes then

$$\min_{\mathbf{B}\in\Gamma'_{\mathbf{B}}} \int_0^n \|\ddot{\mathbf{B}}(t)\|_2^2 \mathrm{d}t + \lambda \sum_{j=0}^n \|d_j - p_j\|_2^2.$$

The search space is the set of control points

$$\Gamma'_{\mathbf{B}} = \{p_0, b^+_0, b^-_i, p_i, b^+_i, b^-_n, p_n\}_{i=1}^{n-1},$$

subject to the $n-1$ differentiability conditions (17), namely, $p_i = \frac{b^-_i + b^+_i}{2}$, $i = 1,\dots,n-1$. The final problem is now

$$\min_{\mathbf{B}\in\Gamma'_{\mathbf{B}}} \int_0^n \|\ddot{\mathbf{B}}(t)\|_2^2 \mathrm{d}t + \lambda \sum_{j=0}^n \|d_j - p_j\|_2^2, \tag{26}$$

$$\text{s.t. } p_i = \frac{b^-_i + b^+_i}{2},\ i = 1,\dots,n-1. \tag{27}$$

Since in $\mathbb{R}^m$ the set of C^1 composite cubic Bézier curves encompasses the cubic splines, the optimal control points correspond to the well-known cubic smoothing spline, see, e.g., [14, Theorem 2.4].

The cost function (26) is a quadratic function in its control points. Following the same path as in Section 3, we decouple the problem into m independent problems, and we formulate these as $(A_0+\lambda A_1)\cdot X = \lambda C\cdot D$ where

$$X = \left[p_0, b^+_0, b^-_1, b^+_1 \dots, b^+_{n-1}, b^-_n, p_n\right]^T \in \mathbb{R}^{(2n+2)\times m}$$

contains the $2n+2$ remaining control points to optimize, stored as row vectors, and where $D = \left[d_0,\dots,d_n\right]^T \in \mathbb{R}^{(n+1)\times m}$ contains the data points. The matrices A_0, $A_1 \in \mathbb{R}^{(2n+2)\times(2n+2)}$ and $C \in \mathbb{R}^{(2n+2)\times(n+1)}$ are matrices of coefficients. They are given in Appendix A.2.

The Euclidean optimality conditions $X = \lambda(A_0 + \lambda A_1)^{-1} C\cdot D =: W\cdot D$ are weighted combinations of the data points, so that each optimization variable x_i of X satisfies

$$x_i = \sum_{j=0}^n w_{ij} d_j, \quad i = 0,\dots,2n+1, \tag{28}$$

where $\sum_{j=0}^n w_{ij} = 1$.

Consider now the data points $d_0,\dots,d_n$ on a Riemannian manifold $\mathcal{M}$. We consider also the search space $\Gamma'_{\mathbf{B}} \subseteq \mathcal{M}^{3n+1}$ and the optimization variables $x_i \in \mathcal{M}$, $i = 0,\dots,2n+1$. Similarly to Section 3, we rewrite the optimality conditions (28) as:

$$\tilde{x}_i = \log_{d_{\mathrm{ref}}}(x_i) = \sum_{j=0}^n w_{ij} \log_{d_{\mathrm{ref}}}(d_j) \in T_{d_{\mathrm{ref}}}\mathcal{M}, \tag{29}$$

for $i = 0,\dots,2n+1$. The control point x_i is finally obtained by $x_i = \exp_{d_{\mathrm{ref}}}(\tilde{x}_i)$. To minimize distortions due to the projection of the data on the tangent space $T_{d_{\mathrm{ref}}}\mathcal{M}$ and back on the manifold, we choose d_{ref} as the closest data point from the control point x_i. Namely, $d_{\mathrm{ref}} = d_j$, if $x_i = b^-_j$ or b^+_j, $j = 1,\dots,n-1$, and we choose $d_{\mathrm{ref}} = d_0$ (resp. $d_{\mathrm{ref}} = d_n$) for the case $x_i = p_0$ or $x_i = b^+_0$ (resp. b^-_n or p_n). Afterwards, one can retrieve p_i, $i = 1,\dots,n-1$ with the differentiability constraints (17):

$$p_i = g(0.5; b^-_i, b^+_i), \qquad i = 1,\dots,n-1. \tag{30}$$

The associated composite Bézier curve $\mathbf{B}(t)\colon [0,n] \to \mathcal{M}$ is finally reconstructed with the De Casteljau algorithm presented in Definition 2.8.

The previous reasoning leads to the following definition for a fitting composite cubic Bézier curve on a manifold.

Definition 4.1 (Fitting C^1 composite cubic Bézier curve on manifold)**.** For a given value of the parameter $\lambda > 0$, the proposed composite cubic Bézier curve $\mathbf{B}\colon [0,n] \to \mathcal{M}$, fitting the manifold-valued data points $d_0,\dots,d_n$ at parameter values $t_0 = 0,\dots,t_n = n$, is defined as

$$\mathbf{B}(t) = \beta_3(t-i; p_i, b_i^+, b_{i+1}^-, p_{i+1}), \qquad i = \lfloor t \rfloor,$$

where β_3 is as in Definition 2.8. The control points $p_0, b_i^+, b_{i+1}^-, p_n$, $i = 0\dots,n-1$, are defined by (29) with $d_{\text{ref}} = d_i$, while the remaining control points p_i, $i = 1,\dots,n-1$, are defined by (30).

The full algorithm to generate a fitting composite cubic Bézier curve on manifold is presented in Algorithm 2.

Algorithm 2 Fitting C^1 composite cubic Bézier curve approaching the solution of (1).

```
Require: $d_0,\dots,d_n$, $\lambda > 0$, $A_0$, $A_1$ and $C$ (Appendix A.2).

Init: $s_0 = \dots = s_n = 0$.
$W \leftarrow (A_0 + \lambda A_1)^{-1} C$                      % matrix of weights

for $i = 0,\dots,n$ do
    $d_{\text{ref}} \leftarrow d_i$                             % reference point
    for $j = 0,\dots,n$ do
        $s_j \leftarrow \log_{d_{\text{ref}}}(d_j)$             % mapping to $T_{d_{ref}}\mathcal{M}$
    end for
    $\tilde{x} \leftarrow \sum_{k=0}^n w_{(2i)k} s_k$           % cp generation
    $\tilde{y} \leftarrow \sum_{k=0}^n w_{(2i+1)k} s_k$
    $x \leftarrow \exp_{d_{\text{ref}}}(\tilde{x})$             % mapping to $\mathcal{M}$
    $y \leftarrow \exp_{d_{\text{ref}}}(\tilde{y})$
    if $i \neq 0$, $i \neq n$ then
        $b_i^- \leftarrow x$
        $b_i^+ \leftarrow y$
        $p_i \leftarrow g(0.5; x, y)$                           % $C^1$ condition (17)
    else if $i = 0$ then
        $p_0 \leftarrow x$
        $b_0^+ \leftarrow y$
    else $\{i = n\}$
        $b_n^- \leftarrow x$
        $p_n \leftarrow y$
    end if
end for
% De Casteljau algorithm (Definition 2.8)
$\mathbf{B}(t) = \beta_3(t-i; p_i, b_i^+, b_{i+1}^-, p_{i+1})$,   $i = \lfloor t \rfloor$.
```

An example of a composite cubic Bézier curve fitting a set of data on $\mathbb{R}^2$ is given in Figure 3a, and on the sphere $\mathbb{S}^2$ in Figure 3b. Note that both results were obtained with the exact same code, where the only difference was the definition of the exponential and logarithm maps. Exponential and logarithm maps on $\mathbb{S}^2$ and SO(3) are given in Appendix A.3.

4.2 Properties of the fitting composite cubic Bézier curve

We present here some properties of the fitting composite cubic Bézier curve of Definition 4.1.

Proposition 4.2 (Natural cubic spline)**.** *When $\mathcal{M} = \mathbb{R}^m$, the composite cubic Bézier curve $\mathbf{B}\colon [0,n] \to \mathbb{R}^m$ of Definition 4.1 is the cubic smoothing spline that minimizes (1) over the Sobolev space $H^2(0,n)$.*

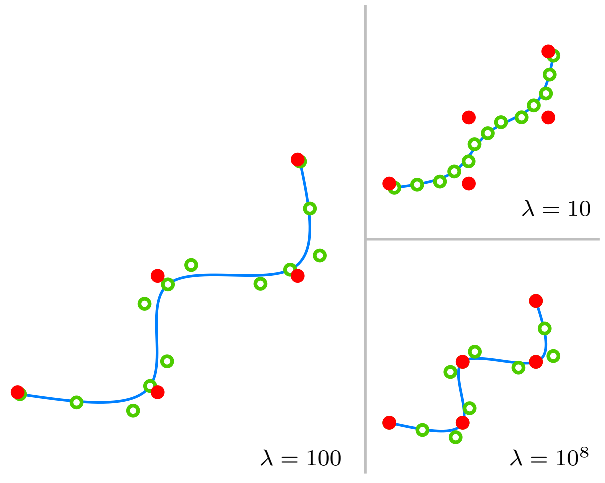
Proof. By construction. □

Theorem 4.3 (C^1 interpolation when $\lambda \to \infty$)**.** *Consider the data points $d_0,\dots,d_n \in \mathcal{M}$ associated with parameter values $t_i = i$, $i = 0,\dots,n$, and the control points p_0, b_0^+, b_i^-, p_i, b_i^+, b_n^-, and p_n, $i = 1,\dots,n-1$. Let $r = \inf_{a\in\mathcal{M}} r_a$ be the injectivity radius of $\mathcal{M}$. The composite cubic Bézier curve $\mathbf{B}(t)$ defined in Definition 4.1 satisfies the following properties:*

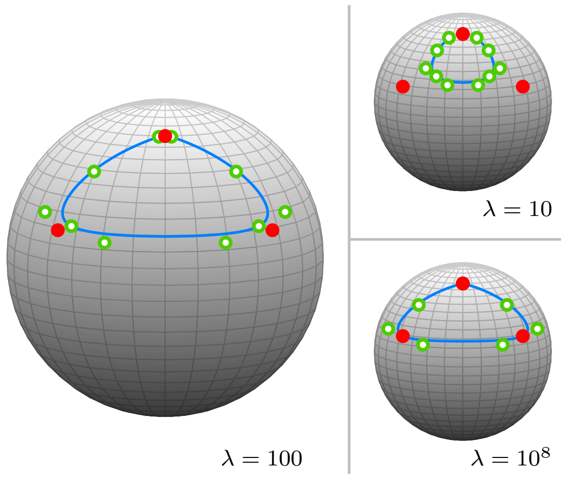
(i) If $\lambda \to \infty$ and $\|\tilde{x}_i\|_{\mathcal{M}} < \frac{r}{2}$, for $\tilde{x}_i$ defined by (29) then $\mathrm{d}_{\mathcal{M}}(\mathbf{B}(i), d_i) = \mathrm{d}_{\mathcal{M}}(p_i, d_i) \to 0$;
(ii) $\mathbf{B}(t)$ is differentiable at $t \in [0,n]$.

Proof. We show directly property *(ii)*, as by construction $\log_{p_i}\big(b_i^+\big) = -\log_{p_i}\big(b_i^-\big)$, so we only have to verify Property *(i)*. By hypothesis, b_i^- and b_i^+ both lie in the set $\mathcal{D}_{d_i}(\frac{r}{2}) \coloneqq \{y \in \mathcal{M} : d_{\mathcal{M}}(d_i, y) < \frac{r}{2}\}$. Due to the triangle inequality, $d_{\mathcal{M}}(b_i^-, b_i^+) \leq d_{\mathcal{M}}(d_i, b_i^-) + d_{\mathcal{M}}(d_i, b_i^+) < r$. Hence $(b_i^-, b_i^+) \mapsto \exp_{b_i^-}\Big(\frac{1}{2}\log_{b_i^-}\big(b_i^+\big)\Big)$ is continuous. Since interpolation holds on the Euclidean case when $\lambda \to \infty$, we have that $\lim_{\lambda\to\infty} \tilde{x}_i = -\lim_{\lambda\to\infty} \tilde{y}_i$ (see Algorithm 2). Hence, d_i is the midpoint of the minimizing geodesic between $x \coloneqq \lim_{\lambda\to\infty} b_i^-$ and $y \coloneqq \lim_{\lambda\to\infty} b_i^+$, i.e., $d_i = \exp_x\big(\frac{1}{2}\log_x(y)\big)$. It follows that $\lim_{\lambda\to\infty} p_i = d_i$. □

This theorem shows that, when $\lambda \to \infty$, the composite cubic Bézier curve $\mathbf{B}$ of Definition 4.1 interpolates the data points *if they are not too spread out*, since $\|\tilde{x}_i\|_{\mathcal{M}}$ is then small for all i. Otherwise, interpolation as $\lambda \to \infty$ may not hold, and we show some examples in the next section.



(a) On the Euclidean space $\mathbb{R}^2$



(b) On the sphere $\mathbb{S}^2$.

Fig. 3: Composite cubic Bézier curve fitting a set of data points on a manifold $\mathcal{M}$ for different values of the regularization parameter λ. **Left**: on the Euclidean space $\mathbb{R}^2$. **Right**: on the sphere $\mathbb{S}^2$. The (solid red) data points are fitted by the (blue) curve defined by the (circled green) control points. Fitting becomes interpolation when $\lambda \to \infty$.

4.3 Lack of interpolation when $\lambda \to \infty$

A first illustration of the limitation suggested by Theorem 4.3*(i)* can be observed on $\mathcal{S}_+(p, q)$, the manifold of positive semidefinite matrices of size p and rank q, equipped with the metric from [36, §7.2]. A second illustration will be also given on the unit circle $\mathbb{S}^1$.

Example 4.4 (Lack of interpolation on $\mathcal{S}_+(p, q)$). The $\mathcal{S}_+(p, q)$ manifold arises in the development of efficient and safe navigating tools for UAV's (unmanned aerial vehicles). These navigating tools rely on faithful models for the wind field. In [13], the wind field is modeled as a Gaussian process, characterized by a mean field and a covariance matrix $C \in \mathcal{S}_+(3024, 20)$. Those two parameters depend on some external meteorological conditions (in [13], they are parametrized by the orientation θ of the prevailing wind in the area of interest). For a given prevailing wind orientation θ, the mean field and the corresponding covariance matrix can be estimated from time-consuming numerical simulations. The approach proposed in [13] is to run those simulations for some key orientations of the prevailing wind, and to obtain values for intermediate orientations by fitting a curve $\mathbf{B}(\theta)$ to those points. We use the dataset of [13], made of 33 covariance matrices $C_i \in \mathcal{S}_+(3024, 20)$, $i = 1, \dots, 33$, corresponding to 33 different orientations $\theta_i = (i-1)\pi/64$. The manifold $\mathcal{S}_+(p, q)$ is here seen as a quotient manifold $\mathbb{R}_*^{p\times q}/\mathrm{O}(q)$, where $\mathbb{R}_*^{p\times q}$ is the set of full rank matrices of size $p \times q$, and $\mathrm{O}(q)$ is the manifold of orthogonal matrices of size $q \times q$. This quotient manifold is endowed with the metric proposed in [36, §7.2] and its geometry is studied in [24].

Applying Algorithm 2 with $\lambda = 10^8$ to those data, we observe in Figure 4 that the composite cubic Bézier curve $\mathbf{B}(\theta)$ does not interpolate one of the data points, the interpolation error at that point being several orders of magnitude higher than at the other points. The

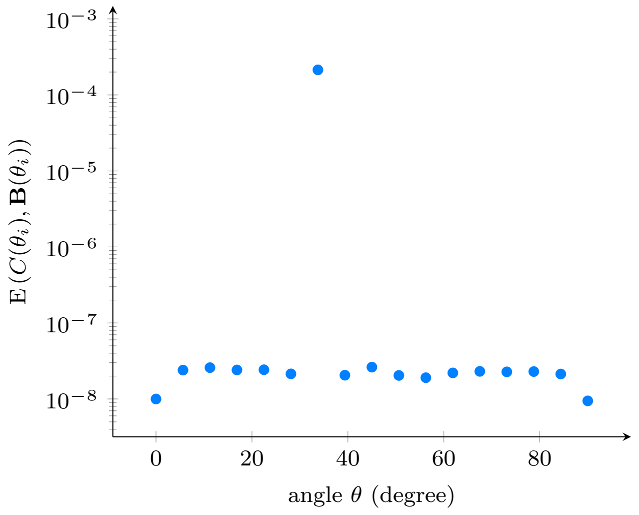


Fig. 4: Interpolation error between the curve $\mathbf{B}(\theta)$ and the data points $C_1, C_3, C_5, \dots, C_{33} \in \mathcal{S}_+(3024, 20)$ extracted from [13]. The parameter λ is set to 10^8. We observe that the error made on the data point C_{13} is several orders of magnitude higher than the error made on the others.

reason of this interpolation error is directly related to Theorem 4.3 whose hypotheses are not met. Indeed, there is no guarantee that $\|\tilde{x}_i\|_{\mathcal{M}} < \frac{r}{2}$. Actually, we can even show that, for the metric considered here, the local injectivity radius around a point $a \in \mathcal{S}_+(p, q)$ is equal to the square root of the smallest eigenvalue of a, which means that it tends towards zero as a tends towards the border of the manifold.

The next example deals with the circle $\mathbb{S}^1$, a manifold whose injectivity radius is equal to π. We provide a configuration of data points that violates the assumptions of Theorem 4.3, and for which interpolation is indeed not achieved.

Example 4.5 (Lack of interpolation in $\mathbb{S}^1$). Consider the set of data points $d_0, \dots, d_4$ represented on Figure 5a. The position of the data points d_0 and d_4 has been chosen respectively on the lower half circle and the upper half circle (here, at $\xi_0 = -120°$ and $\xi_4 = 120°$). The data point d_2 corresponds to an angle $\xi_2 = 0°$, while data points d_1 and d_3 correspond respectively to angles $\xi_1 = 179°$ and $\xi_3 = 181°$.

On Figure 5b, we represent the curve obtained when we try to interpolate the set of data points from Figure 5a with Algorithm 2. At time $t = 2$, the angle ξ is equal to $-\pi$, instead of the desired value 0. Indeed, following (17), $p_2 = -\pi$ is the midpoint of the shortest endpoint geodesic between the control points b_2^- and b_2^+. Unfortunately, as indicated on Figure 5a, $d_2 = 0$ is the midpoint of a non-minimizing geodesic $\bar{\gamma}$ between b_2^- and b_2^+ (because $\|\log_{d_2}(b_2^-)\| > \frac{r}{2}$, as well as $\|\log_{d_2}(b_2^+)\|$), while the midpoint of the shortest geodesic γ between the two data points is given by $\xi = -\pi$. Therefore, d_2 is not interpolated by $\mathbf{B}(t)$.

4.4 Alternative definition of the control points to ensure interpolation while losing differentiability

To overcome this limitation for any general manifold, we propose now a solution that enforces the fitting curve to interpolate the data points when $\lambda \to \infty$, regardless of the curvature of the manifold. This new condition will fix the interpolation problem, but we will see that this conditions yields to a loss of the differentiability condition that we had before. As a consequence of that, it will be necessary to modify the classical De Casteljau algorithm (usually used at the reconstruction step) to propose a new definition of Bézier curves, compatible with the new C^1-condition (Sections 5 and 6).

The solution to have interpolation modifies the definition of the intermediary control points $p_i, i = 1, \dots, n-1$, as follows.

Proposition 4.6 (Interpolation conditions). *Let us consider the tangent vectors $\tilde{b}_i^+$ and $\tilde{b}_i^- \in T_{d_i}\mathcal{M}$, $i = 1, \dots, n-1$, computed by (29). To interpolate the data points d_i when $\lambda \to \infty$, the control points p_i, can be computed as (see Figure 6)*

$$p_i = \exp_{d_i}\left(\frac{\tilde{b}_i^- + \tilde{b}_i^+}{2}\right), i = 1, \dots, n-1. \tag{31}$$

Proposition 4.6 is nothing more than another way to generalize the Euclidean conditions of differentiability (9). The two resulting definitions for the control points p_i, $i = 1, \dots, n-1$, are equivalent on the Euclidean space, while they are not on a general Riemannian manifold. The difference compared to (30) is that here the C^1 condition is directly computed in the tangent space at d_i, and not at b_i^-.

Applying these new interpolation conditions results in the following definition for the fitting composite cubic Bézier curve.

Definition 4.7 (Fitting composite cubic Bézier curve on manifold–II). For a given value of the parameter $\lambda > 0$, the proposed composite cubic Bézier curve $\mathbf{B}\colon [0, n] \to \mathcal{M}$, fitting the manifold-valued data points $d_0, \dots, d_n$ at parameter values $t_0 = 0, \dots, t_n = n$, is defined as

$$\mathbf{B}(t) = \beta_3(t - i; p_i, b_i^+, b_{i+1}^-, p_{i+1}), \qquad i = \lfloor t \rfloor.$$

The control points $p_0, b_i^+, b_{i+1}^-, p_n$, $i = 0, \dots, n-1$, are defined by (29) with $d_{\text{ref}} = d_i$. The remaining control points p_i, $i = 1, \dots, n-1$, are computed as in Proposition 4.6. The Bézier curve β_3 is reconstructed according to Definition 2.8.

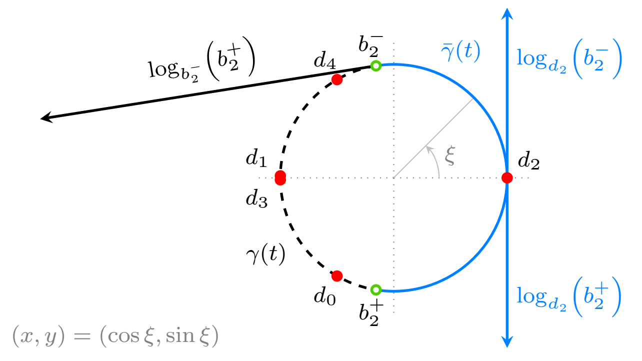
We now list a set of properties that are verified by the fitting curve of Definition 4.7.

Proposition 4.8. *The composite cubic Bézier curve $\mathbf{B}(t)$ of Definition 4.7 interpolates the data points when $\lambda \to \infty$.*

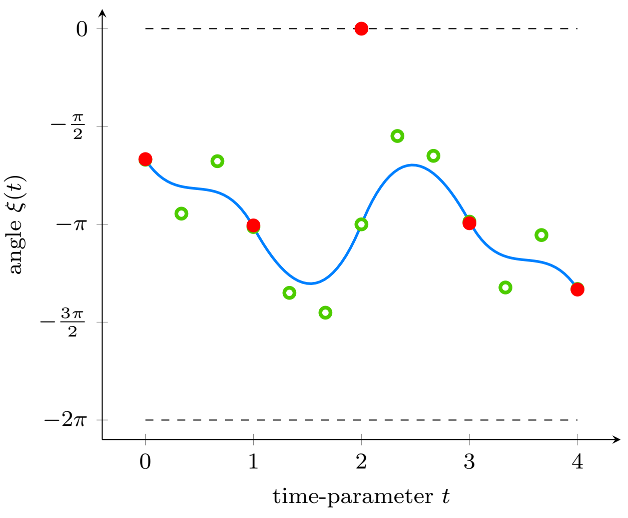
Proof. We show here that $\mathbf{B}(i) \to d_i$, $i = 0, \dots, n$, when $\lambda \to \infty$. Let $i \in \{1, \dots, n-1\}$. When $\lambda \to \infty$, the weights w_{ij} in equation (29) are such that $\lim_{\lambda\to\infty} \tilde{b}_i^+ = -\lim_{\lambda\to\infty} \tilde{b}_i^-$. Therefore, $\lim_{\lambda\to\infty} \tilde{p}_i = 0 = \tilde{d}_i$, and thus $\lim_{\lambda\to\infty} p_i = d_i$. For $i \in \{0, n\}$, interpolation is ensured by construction. □

Proposition 4.9 (Minimal representation of the fitting curve). *The fitting curve of Definition 4.7 is uniquely represented by $2(n+1)$ tangent vectors.*

Proof. The proof is similar to the one of Proposition 3.4. The control points are obtained from the $2(n+1)$ tangent vectors $\tilde{x}$, $\tilde{y} \in T_{d_i}\mathcal{M}$, $i = 0, \dots, n$, as stated in the relevant parts of Algorithm 2. □


$\log_{b_2^-}\left(b_2^+\right)$
b_2^-
d_4
$\bar{\gamma}(t)$
$\log_{d_2}\left(b_2^-\right)$
d_1
ξ
d_2
d_3
$\gamma(t)$
d_0
b_2^+
$\log_{d_2}\left(b_2^+\right)$
$(x, y) = (\cos\xi, \sin\xi)$

(a) Representation of the data set used for the counter-example on the circle, as well as the two curves γ and $\bar{\gamma}$, for $i = 2$. The injectivity radius of the circle $\mathbb{S}^1$ is π. As the geodesic length $L_{\bar{\gamma}} = |\log_{d_2}(b_i^-)| + |\log_{d_2}(b_i^+)| > \pi$, the geodesic $\bar{\gamma}$ is not minimizing, while $\gamma = g(t; b_i^-, b_i^+)$ is. The consequence of this is that $p_i \neq d_i$.



(b) The curve $\xi(t) = \tan^{-1}(\mathbf{B}(t))$ (solid blue) obtained by applying Algorithm 2 to the (solid red) data points, with $\lambda = 10^8$, does not interpolate all data points. Specifically, $\xi(2) \neq \tan^{-1}(d_2)$.

Fig. 5: Second counter-example for interpolation by the fitting curve of Definition 4.1, when $\lambda \to \infty$.

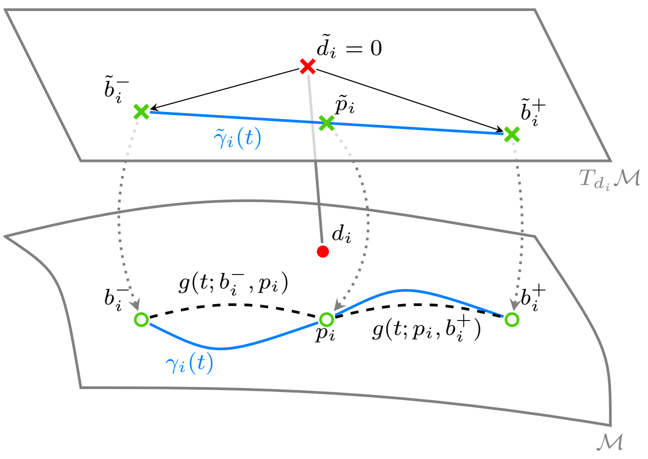


Fig. 6: Choice of the point p_i with the interpolation conditions (31). As $\tilde{p}_i \in T_{d_i}\mathcal{M}$ is in the middle of the straight line $\tilde{\gamma}_i(t)$ joining $\tilde{b}_i^-$ and $\tilde{b}_i^+$, $\lim_{\lambda\to\infty} p_i = d_i$. Furthermore, $\gamma_i(t) = \exp_{d_i}(\tilde{\gamma}_i(t))$ is differentiable at p_i even if b_i^-, p_i and b_i^+ are not aligned on the same geodesic.

Proposition 4.10 (Exponential and logarithm maps required)**.** *The number of exponential and logarithm maps required by the fitting curve of Definition 4.7 is*

- $n(n+1)$ *logarithms for the construction of the minimal representation of the curve of Proposition 4.9*
- 10 *exponential maps and* 6 *logarithm maps to reconstruct the curve* $\mathbf{B}(t)$*, for a given parameter value* t*, given that minimal representation.*

Proof. The $n(n+1)$ logarithms maps are required for the evaluation of (23). The reconstruction of the curve at a given value of t require 4 exponential maps (to obtain the representation on the manifold of the 4 control points associated to the segment $i = \lfloor t \rfloor$), and the De Casteljau algorithm requires 6 additional exponential and logarithm maps. $\square$

Proposition 4.11. *The composite cubic Bézier curve of Definition 4.7 is not differentiable at* p_i.

Proof. Consider a set of data points $d_i \in \mathbb{R}^2$, $i = 0, \ldots, n$, and the corresponding parameter values $t_0 = 0, \ldots, t_n = n$. Let $\mathbf{B}$ be the fitting curve (Definition 4.7), for a given value of λ. Without loss of generality, we fix i, and consider that the associated data point d_i is equal to $(0, 0)$. Assume now that the data points belong actually to a manifold $\mathcal{M}$ similar to $\mathbb{R}^2$. More precisely, let $\mathcal{M}$ be the Euclidean space $\mathbb{R}^2$ except in a small region between b_i^- and b_i^+ where the metric is smoothly reduced such that $\mathrm{d}_{\mathcal{M}}(b_i^-, b_i^+) < \mathrm{d}_{\mathbb{R}^2}(b_i^-, b_i^+) = \|b_i^+ - b_i^-\|$, as represented in Figure 7. Finally, let us denote by $\tilde{x}$ the representation in $T_{d_i}\mathcal{M} = \mathbb{R}^2$ of a point $x \in \mathcal{M}$.

Proposition 4.6 states that $\tilde{p}_i := \log_{d_i}(p_i)$ is obtained as $\tilde{p}_i = 0.5(\tilde{b}_i^- + \tilde{b}_i^+)$. As $d_i = (0, 0)$, we observe

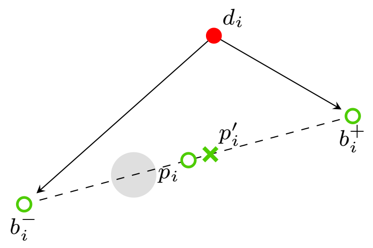


Fig. 7: Proposition 4.6 is not compatible with the differentiability constraints, as the middle of the geodesic between b_i^- and b_i^+ is p_i' and not p_i.

immediately that $b_i^+ = \tilde{b}_i^+$, as well as $b_i^- = \tilde{b}_i^-$. However, $p_i = 0.5(b_i^- + b_i^+)$ does not correspond to the midpoint $p_i' = g(0.5; b_i^-, b_i^+)$ of a (possibly non-minimizing) geodesic g between b_i^- and b_i^+. This is due to the shrinking of the metric in the above-mentioned area. Therefore, the two velocities $\lim_{\epsilon\to 0} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{B}(i-\epsilon) = -\log_{p_i}\left(b_i^-\right)$ and $\lim_{\epsilon\to 0} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{B}(i+\epsilon) = \log_{p_i}\left(b_i^+\right)$ are in general different in $T_{p_i}\mathcal{M}$, which results in the non-differentiability of the curve. □

On the one hand, the new fitting curve of Definition 4.7 recovers the interpolation property as $\lambda \to \infty$, but on the other hand, losing differentiability is a major drawback.

In the next section, we propose a new curve definition, obtained using a so-called blending process, in order to overcome this drawback.

5 Fitting with blended Bézier curves

We now present a way to construct a *differentiable* curve $\mathbf{B}(t)$ that satisfies the six properties mentioned in Section 1. The method is also less costly than those of Section 4, as it requires fewer exp and log evaluations to evaluate the curve at a given parameter value t. We still resort to Bézier curves, but here it is merely for convenience in order to reuse (29); actually all the Bézier curve are now computed in (Euclidean) tangent spaces and are thus nothing else than polynomials expressed in a specific form.

5.1 Blending technique

The basic idea of this method is that the control points of the i^{th} curve of $\mathbf{B}(t)$ are computed in the tangent spaces of the data points d_i and d_{i+1}. A (Euclidean) Bézier curve is computed on each of these tangent spaces and then mapped to the manifold. These two solutions are finally blended together on $\mathcal{M}$, via a carefully chosen weighted mean that will be defined in a moment. This last step of this sketched algorithm gives its name to the reconstructed *blended* spline $\mathbf{B}$. We provide now a more formal description of this method.

Definition 5.1 (Blended curve)**.** Let $d_{\text{ref},1}$, $d_{\text{ref},2} \in \mathcal{M}$ be two different reference points. Consider two sequences of tangent vectors $\tilde{b}_0, \dots, \tilde{b}_K \in T_{d_{\text{ref},1}}\mathcal{M}$ and $\hat{b}_0, \dots, \hat{b}_K \in T_{d_{\text{ref},2}}\mathcal{M}$ named *control vectors*. The *blended curve* $\beta_K\colon [0,1] \to \mathcal{M}$ of degree K is defined as

$$\beta_K(t) = \operatorname{av}[(L(t), R(t)), (1-w(t), w(t))], \tag{32}$$

where

$$L(t) = \exp_{d_{\text{ref},1}}\left(\beta_K(t; \tilde{b}_0, \dots, \tilde{b}_K)\right) \tag{33}$$

$$R(t) = \exp_{d_{\text{ref},2}}\left(\beta_K(t; \hat{b}_0, \dots, \hat{b}_K)\right) \tag{34}$$

and where $w(t) = 3t^2 - 2t^3$.

Remark 5.2. In order to obtain the differentiability property of Theorem 5.7, the weight function $w(t)$ has to be chosen such that $\beta_K(0) = L(0)$, $\beta_K(1) = R(1)$, $\dot{\beta}_K(0) = \dot{L}(0)$, and $\dot{\beta}_K(1) = \dot{R}(1)$. This is achieved when $w(0) = 0$, $w(1) = 1$, $w'(0) = 0$, and $w'(1) = 0$. Among all functions w that satisfy these conditions, we opted for the one that minimizes the mean square second derivative. Remark also that $w(t) \in [0,1]$, for any $t \in [0,1]$.

Proposition 5.3 (Continuity of the blended curve)**.** *The blended curve (Definition 5.1) is well defined and smooth under the assumptions from the introduction, i.e., that the exponential and the logarithm map involved are well defined and smooth. This holds in particular if $\mathcal{M}$ is complete and if the distance $\mathrm{d}_{\mathcal{M}}(L(t), R(t)) < r^*(t)$, where $r^*(t) = \max(r_{R(t)}, r_{L(t)})$, for all $t \in [0,1]$. Furthermore, it interpolates the point $b_0 = \exp_{d_{ref,1}}(\tilde{b}_0)$ and the point $b_K = \exp_{d_{ref,2}}(\hat{b}_K)$.*

Proof. The first claim is immediate by composition. For the second claim, as $\mathrm{d}_{\mathcal{M}}(L(t), R(t)) < r^*(t)$, the weighted average between $L(t)$ and $R(t)$ never crosses the cut loci of the points, so that the blended curve is never discontinuous. Interpolation is obtained as follows. As $w(0) = 0$, $\beta_K(0) = \operatorname{av}[(L(0), R(0)), (1,0)] = L(0) = b_0$, and similarly for $\beta_K(1)$, as $w(1) = 1$. □

The natural next step is now to define the notion of *composite blended curve.*

Definition 5.4 (Composite blended curve)**.** Consider now the sequence $(\beta^i_{K_i})_{i=0}^{n-1}$ of n blended curves. The *composite blended curve* $\mathbf{B}(t)$ is the curve

$$\mathbf{B}\colon [0,n] \to \mathcal{M}, t \mapsto \beta^i_{K_i}(t-i), \qquad i = \lfloor t \rfloor. \tag{35}$$

Consider now the data points $d_0, \dots, d_n \in \mathcal{M}$ corresponding to parameter values $t_i = i$, $i = 0, \dots, n$, and the regularization parameter $\lambda > 0$. Our objective is now to define the *blended cubic spline* $\mathbf{B}(t) = \beta_3^i(t-i)$, $i = \lfloor t \rfloor$ minimizing (1) when $\mathcal{M}$ is flat. To do so, one has to choose the two reference points and find the control vectors of each blended curve.

Without loss of generality, consider the ith blended curve $\beta_3^i(t)$ of $\mathbf{B}(t)$. We define its intermediate functions $L(t)$ and $R(t)$ as

$$L(t) = \exp_{d_i}\Big(\beta_K(t; \tilde{p}_i, \tilde{b}_i^+, \tilde{b}_{i+1}^-, \tilde{p}_{i+1})\Big) \tag{36}$$

$$R(t) = \exp_{d_{i+1}}\Big(\beta_K(t; \hat{p}_i, \hat{b}_i^+, \hat{b}_{i+1}^-, \hat{p}_{i+1})\Big), \tag{37}$$

where we have chosen $d_{\mathrm{ref},1} = d_i$ and $d_{\mathrm{ref},2} = d_{i+1}$.

The control vectors of the Euclidean Bézier curves on $T_{d_i}\mathcal{M}$ and $T_{d_{i+1}}\mathcal{M}$ are obtained such that they are cubic smoothing splines on their respective tangent spaces. In other words, the control vectors $\tilde{x} \in \{\tilde{p}_i, \tilde{b}_i^+, \tilde{b}_{i+1}^-, \tilde{p}_{i+1}\}$ (resp. $\hat{x}$) correspond to the tangent vectors computed using equation (29) with $d_{\mathrm{ref}} = d_i$ (resp. $d_{\mathrm{ref}} = d_{i+1}$) on $T_{d_i}\mathcal{M}$ (resp. $T_{d_{i+1}}\mathcal{M}$). Finally, in view of Proposition 2.6 and the C^1 (actually C^2) property of a natural cubic spline, the remaining control vectors are

$$\tilde{p}_i = \frac{\tilde{b}_i^- + \tilde{b}_i^+}{2}, \quad i = 1, \dots, n-1, \tag{38}$$

and accordingly for $\hat{p}_i$.

We can now define the *blended cubic spline.*

Definition 5.5 (Blended cubic spline). For a given value of the parameter $\lambda > 0$, the proposed *blended cubic spline* $\mathbf{B}\colon [0,n] \to \mathcal{M}$ fitting the manifold-valued data points $d_0, \dots, d_n \in \mathcal{M}$ at parameter values $t_0 = 0, \dots, t_n = n$, is defined as

$$\mathbf{B}(t) \coloneqq \beta_3^i(t-i), \quad i = \lfloor t \rfloor,$$

where $\beta_3^i(t-i)$ is as in Definition 5.1. The intermediate functions $L(t)$ and $R(t)$ of β_3^i are defined by (36) and (37), and their control vectors are computed on $T_{d_i}\mathcal{M}$ and $T_{d_{i+1}}\mathcal{M}$ using (29) and (38).

The whole method is summarized in Algorithms 3 and 4, and this last algorithm is represented on Figure 8.

Algorithm 3 Control points generation for the blended cubic spline of Definition 5.5.

```
Require: $d_0, \dots, d_n$, $\lambda > 0$, $A_0$, $A_1$ and $C$ (Appendix A.2).

  Init: $s_0 = \dots = s_n = u_0 = \dots = u_n = 0$.
  $W \leftarrow (A_0 + \lambda A_1)^{-1} C$                % matrix of weights
  $d_{\mathrm{ref},2} = d_0$.
  for $j = 0, \dots, n$ do
    $u_j = \log_{d_{\mathrm{ref},2}}(d_j)$
  end for

  for $i = 0, \dots, n-1$ do
    $d_{\mathrm{ref},1} \leftarrow d_i$                  % reference points
    $d_{\mathrm{ref},2} \leftarrow d_{i+1}$
    for $j = 0, \dots, n$ do
      $s_j \leftarrow u_j$                       % mapping to $T_{d_{ref,\star}}\mathcal{M}$
      $u_j \leftarrow \log_{d_{\mathrm{ref},2}}(d_j)$
    end for
    for $j = 0, \dots, 3$ do
      $\tilde{x}_j \leftarrow \sum_{k=0}^{n} w_{(2i+j)k} s_k$          % cp generation
      $\hat{x}_j \leftarrow \sum_{k=0}^{n} w_{(2i+j)k} u_k$
    end for

    % first point of the segment
    if $i = 0$ then
      $\tilde{p}_0 \leftarrow \tilde{x}_0$ and $\hat{p}_0 \leftarrow \hat{x}_0$
    else
      $\tilde{p}_i \leftarrow 0.5(\tilde{x}_0 + \tilde{x}_1)$              % $C^1$-condition
      $\hat{p}_i \leftarrow 0.5(\hat{x}_0 + \hat{x}_1)$
    end if

    % last point of the segment
    if $i = n-1$ then
      $\tilde{p}_n \leftarrow \tilde{x}_3$ and $\hat{p}_n \leftarrow \hat{x}_3$
    else
      $\tilde{p}_{i+1} \leftarrow 0.5(\tilde{x}_2 + \tilde{x}_3)$            % $C^1$-condition
      $\hat{p}_{i+1} \leftarrow 0.5(\hat{x}_2 + \hat{x}_3)$
    end if

    % inner points of the segment
    $\tilde{b}_i^+ \leftarrow \tilde{x}_1$ and $\hat{b}_i^+ \leftarrow \hat{x}_1$
    $\tilde{b}_{i+1}^- \leftarrow \tilde{x}_2$ and $\hat{b}_{i+1}^- \leftarrow \hat{x}_2$
  end for
```

Algorithm 4 Reconstruction of the C^1 blended cubic spline of Definition 5.5 at time t

```
Require:
  $i \in \{0, \dots, n-1\}$, $t \in [i, i+1]$,
  $(\tilde{p}_i, \tilde{b}_i^+, \tilde{b}_{i+1}^-, \tilde{p}_{i+1}) \in T_{d_i}\mathcal{M}$,
  $(\hat{p}_i, \hat{b}_i^+, \hat{b}_{i+1}^-, \hat{p}_{i+1}) \in T_{d_{i+1}}\mathcal{M}$

  $w \leftarrow 3t^2 - 2t^3$
  $\tilde{x} \leftarrow \beta_3(t; \tilde{p}_i, \tilde{b}_i^+, \tilde{b}_{i+1}^-, \tilde{p}_{i+1})$,    $x \leftarrow \exp_{d_i}(\tilde{x})$
  $\hat{y} \leftarrow \beta_3(t; \hat{p}_i, \hat{b}_i^+, \hat{b}_{i+1}^-, \hat{p}_{i+1})$,    $y \leftarrow \exp_{d_{i+1}}(\hat{y})$
  $z \leftarrow \mathrm{av}[(x, y), (1-w, w)]$
  return $z$
```

5.2 Properties of the composite cubic blended curve

In this section, we analyse the properties of the composite cubic blended curve.

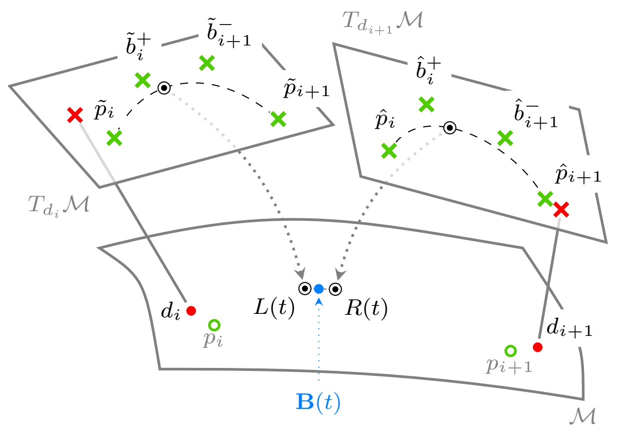


Fig. 8: Illustration of the reconstruction of the blended cubic spline (Algorithm 4). The points $(\tilde{p}_i, \tilde{b}_i^+, \tilde{b}_{i+1}^-, \tilde{p}_{i+1})$ and $(\hat{p}_i, \hat{b}_i^+, \hat{b}_{i+1}^-, \hat{p}_{i+1})$ are computed respectively in the tangent space of d_i and d_{i+1} according to (29) and (38). The points $L(t)$ and $R(t)$ (black circles) are obtained as the mapping to $\mathcal{M}$ of the Euclidean Bézier curve obtained on $T_{d_i}\mathcal{M}$ and $T_{d_{i+1}}\mathcal{M}$ respectively. Finally, these points are averaged on $\mathcal{M}$ via equation (32) to obtain the value of the blended cubic spline $\mathbf{B}(t)$ (blue dot).

Lemma 5.6. *Consider the tangent spaces $T_{d_i}\mathcal{M}$, $i = 1, \dots, n-1$ and the control vectors $\hat{p}_{i-1}$, $\hat{b}_{i-1}^+$, $\hat{b}_i^-$, $\hat{p}_i = \tilde{p}_i$, $\tilde{b}_i^+$, $\tilde{b}_{i+1}^-$, and $\tilde{p}_{i+1}$ obtained by* (29) *and* (38)*. The curve $\tilde{\gamma}\colon [0,2] \to T_{d_i}\mathcal{M}\colon t \mapsto \tilde{\gamma}(t)$ given by*

$$\tilde{\gamma}(t) = \begin{cases} \beta_3(t; \hat{p}_{i-1}, \hat{b}_{i-1}^+, \hat{b}_i^-, \hat{p}_i) & \text{for } t \in [0,1] \\ \beta_3(t-1; \tilde{p}_i, \tilde{b}_i^+, \tilde{b}_{i+1}^-, \tilde{p}_{i+1}) & \text{for } t \in [1,2], \end{cases}$$

is a natural cubic spline on $T_{d_i}\mathcal{M}$.

Proof. By construction of the control points, the curve $\tilde{\gamma}$ corresponds to two successive pieces of the optimal fitting curve in the Euclidean space $T_{d_i}\mathcal{M}$ for the data points $\log_{d_i}(d_j)$, $j = 0, \dots, n$. This optimal fitting curve is known to be a natural cubic spline; see, e.g., [14]. □

Theorem 5.7. *Consider the data points $d_0, \dots, d_n \in \mathcal{M}$ associated with parameter-values $t_i = i$, $i = 0, \dots, n$. The blended cubic spline $\mathbf{B}(t)\colon [0,n] \to \mathcal{M}$ of Definition 5.5 satisfies the following properties:*

(i) $\mathbf{B}(i) \to d_i$ *when* $\lambda \to \infty$, *for* $i = 0, \dots, n$;
(ii) $\mathbf{B}(t)$ *is well defined and smooth under the assumptions from the introduction, i.e., that the exponential and the logarithm map well defined and smooth. This holds in particular if $\mathcal{M}$ is complete and, on each piece, $\mathrm{d}_{\mathcal{M}}(L(\tau), R(\tau)) < r^*(\tau)$, $r^*(\tau) = \max(r_{R(\tau)}, r_{L(\tau)})$, for all $\tau \in [0,1]$.*
(iii) when $\mathcal{M}$ is a Euclidean space, $\mathbf{B}(t)$ is the cubic smoothing spline that minimizes (1) *over the Sobolev space $H^2(0,n)$.*

Proof. By Proposition 5.3, $\beta_3^i(0) = p_i$ and $\beta_3^i(1) = p_{i+1}$, for $i = 0, \dots, n-1$. Therefore, $\mathbf{B}(i) = p_i$. When $\lambda \to \infty$, $\tilde{p}_i = 0 \in T_{d_i}\mathcal{M}$ by (29) and (38), so $p_i = d_i \in \mathcal{M}$, and property (i) is verified $\forall i$. The proof of property (iii) is direct from Lemma 5.6, as $T_{d_0}\mathbb{R}^m = T_{d_1}\mathbb{R}^m = \cdots = T_{d_n}\mathbb{R}^m$. Let us now prove property (ii). Again by Proposition 5.3, $\mathbf{B}$ is C^1 on $t \neq i$. There remains to show that $\mathbf{B}(t)$ is differentiable at $t = i$. For $i \in \{0, n\}$ differentiability is trivial. Let $i \in \{1, \dots, n-1\}$. Consider the curve $\tilde{\gamma}(t)$ of Lemma 5.6 and $\gamma(t) = \exp_{d_i}(\tilde{\gamma}(t))$. By definition, $w'(t) = 0$ at $t \in \{0,1\}$. Therefore,

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{B}(t)|_{t=i^+} &= \frac{\mathrm{d}}{\mathrm{d}t}\mathrm{av}[(L_i(t), R_i(t)), (1-w(t), w(t))]|_{t=0^+} \\ &= \frac{\mathrm{d}}{\mathrm{d}t}L_i(t)|_{t=0^+} = \frac{\mathrm{d}}{\mathrm{d}t}\gamma(t)|_{t=1^+}, \end{aligned}$$

where $R_i(t)$ and $L_i(t)$ are the intermediate functions (36) and (37) of the ith blended curve of $\mathbf{B}$. Similarly,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{B}(t)|_{t=i^-} = \frac{\mathrm{d}}{\mathrm{d}t}R_{i-1}(t)|_{t=1^-} = \frac{\mathrm{d}}{\mathrm{d}t}\gamma(t)|_{t=1^-}.$$

As $\exp_{d_i}(\cdot)$ is a smooth mapping, $\gamma(t)$ is differentiable at $t = 1$, so $\mathbf{B}(t)$ is differentiable at $t = i$, $\forall i$. □

Remark 5.8. Note that the condition of Theorem 5.7, *(ii)*, is not easy to check in practice. Let us pose $\tilde{L}(t)$ and $\hat{R}(t)$, the smoothing splines computed in $T_{d_i}\mathcal{M}$ and $T_{d_{i+1}}\mathcal{M}$, respectively, evaluated at $t = s - i$, $s \in [i, i+1]$. Let t be given. By the triangular inequality, one can say that

$$\mathrm{d}_{\mathcal{M}}\big(L(t), R(t)\big) \le \|\tilde{L}(t)\| + \mathrm{d}_{\mathcal{M}}(d_i, d_{i+1}) + \|\hat{R}(t)\|.$$

By definition, the smoothing splines read

$$\tilde{L}(t) = \sum_{i=0}^{3}\sum_{j=0}^{n} B_{i3}(t) D_{ij} \tilde{d}_j,$$

and accordingly for $\hat{R}(t)$. We pose $\Delta = \max_{ij} \mathrm{d}_{\mathcal{M}}(d_i, d_j)$. As $B_{i3}(t) \ge 0$ and $\sum_{i=0}^{3} B_{i3}(t) = 1$, one has

$$\mathrm{d}_{\mathcal{M}}\big(L(t), R(t)\big) \le (n+1)\max_{ij}(D_{ij})\big(2\Delta\big) + \Delta.$$

The condition of Theorem 5.7, *(ii)*, is verified if

$$\Delta \le \frac{r^*(t)}{1 + 2(n+1)\max_{ij}(D_{ij})}.$$

Note that this condition can be checked a priori, is sufficient for arbitrary manifolds with nonzero injectivity radius, but is by no means necessary. In practice, we observed that the property (ii) holds true in all experiments we conducted, and we were not even able to choose the data points malicously enough to make it wrong.

Proposition 5.9 (Minimal representation of the curve)**.** *The blended cubic spline of Definition 5.5 is uniquely represented by* $6(n-1)+8$ *tangent vectors.*

Proof. The curve is represented by the following tangent vectors:

- 4 vectors $\tilde{p}_0$, $\tilde{b}_0^+$, $\tilde{b}_1^-$, $\tilde{b}_1^+ \in T_{d_0}\mathcal{M}$
- 6 vectors $\tilde{p}_0$, $\tilde{b}_0^+$, $\tilde{b}_1^-$, $\tilde{b}_1^+$, $\tilde{b}_2^-$, $\tilde{b}_2^+ \in T_{d_1}\mathcal{M}$
- $6(n-3)$ vectors $\tilde{b}_{i-1}^-$, $\tilde{b}_{i-1}^+$, $\tilde{b}_i^-$, $\tilde{b}_i^+$, $\tilde{b}_{i+1}^-$, $\tilde{b}_{i+1}^+ \in T_{d_i}\mathcal{M}$, for $i=2,\dots,n-2$;
- 6 vectors $\tilde{b}_{n-2}^-$, $\tilde{b}_{n-2}^+$, $\tilde{b}_{n-1}^-$, $\tilde{b}_{n-1}^+$, $\tilde{b}_n^-$, $\tilde{p}_n \in T_{d_{n-1}}\mathcal{M}$
- 4 vectors $\tilde{b}_{n-1}^-$, $\tilde{b}_{n-1}^+$, $\tilde{b}_n^-$, $\tilde{p}_n \in T_{d_n}\mathcal{M}$.

□

Proposition 5.10 (Exponential and logarithm maps required)**.** *The number of exponential and logarithm maps required by the blended cubic spline (Definition 5.5) is*

- $n(n+1)$ *logarithms for the construction of the minimal representation of the curve of Proposition 5.9*
- 3 *exponential maps and* 1 *logarithm map to reconstruct the curve* $\mathbf{B}(t)$*, for a given parameter value* t*, given that minimal representation.*

Proof. The $n(n+1)$ logarithms maps are here again required for the representation of the data points d_j, $j \neq i$, in the tangent space $T_{d_i}\mathcal{M}$. The reconstruction of the curve at a given value of t requires 2 exponential maps to map on the manifold the values of the Euclidean Bézier curves computed in the tangent spaces at d_i and d_{i+1}, with $i=\lfloor t \rfloor$, and one additional logarithm and exponential map, to compute the weighted average of those two values. □

6 Fitting with composite Bézier-like curves

In this section, we present another approach to fix the differentiability problem identified in Proposition 4.11. This approach is based on the observation that the curve

$$\bar{\gamma}_i\colon [0,2] \to \mathcal{M}\colon \bar{\gamma}_i(t) = \begin{cases} g(t; b_i^-, p_i) & \text{for } t \in [0,1], \\ g(t; p_i, b_i^+) & \text{for } t \in [1,2] \end{cases}$$

(see Figure 6) is not differentiable at $t=1$. This curve, however, is used at the first step of the De Casteljau algorithm to reconstruct Bézier curves of $\mathbf{B}$ afterwards (Definition 2.8). A natural (but naive) idea to fix this problem would be to replace these two geodesics (in the De Casteljau algorithm) by any differentiable curve $\gamma_i(t)\colon [0,2] \to \mathcal{M}$ satisfying $\gamma_i(0)=b_i^-$, $\gamma_i(1)=p_i$ and $\gamma_i(2)=b_i^+$ (for instance, $\gamma(t)=\exp_{d_i}(\tilde{\gamma}(t))$ from Lemma 5.6). We explain here why this approach does not fix the differentiability problem identified in Proposition 4.11. We discuss also how this approach can be modified (by replacing the De Casteljau method by a more general algorithm) in order to obtain C^1 fitting curves satisfying the interpolation property as $\lambda \to \infty$. However, the curve obtained will no longer reduce to a natural cubic spline in the case $\mathcal{M}=\mathbb{R}^m$.

This section relies on the notion of *blossom* of a recursive function. Blossoms can be viewed as a manner to generalize the De Casteljau algorithm. Indeed, at each step of the recursion, one can choose a different function and a different evaluation time, while the De Casteljau algorithm is limited to performing only geodesics at a given time t. The section is composed of two parts. We first recall the general theory and remarkable theorems about blossoms. Then, we present and analyse two new fitting methods based on that theory.

6.1 Blossom functions

Blossoms are conceptual mathematical objects that are described, e.g., in [11,20]. We refer the reader to these documents for more details and extend here the results to a more general setting that will be useful for our developments.

Definition 6.1 (Manifold-valued blossom)**.** Consider a set of points $x_0,\dots,x_K \in \mathcal{M}$, $K \in \mathbb{N}$, associated with parameter values $t_0,\dots,t_K \in [0,1]$, and a set of smooth functions $f_i(\cdot;x,y)\colon [0,1] \to \mathcal{M}$, $i=1,\dots,K$ such that $f_i(0;x,y)=x$ and $f_i(1;x,y)=y$ for all i. We will later refer to such functions as *endpoint functions*. The associated recursive function $h_K(\cdot;x_0,\dots,x_K)\colon [0,1] \to \mathcal{M}$ is defined as

$$h_i(t;x_0,\dots,x_i) = \\ f_i\big(t; h_{i-1}(t;x_0,\dots,x_{i-1}), h_{i-1}(t;x_1,\dots,x_i)\big),$$

with $h_0(t;x_j)=x_j$, $j=0,\dots,K$. The notion of *blossom* is a generalization of h_K, where different values of t can be used at each step of the recursion. The *blossom* of h_K is thus the map $\psi_K\colon \mathbb{R}^{K+1} \to \mathcal{M}$ given by

$$\psi_i(t_0,\dots,t_i;x_0,\dots,x_i) = \\ f_i\big(t_i; \psi_{i-1}(t_0,\dots,t_{i-1};x_0,\dots,x_{i-1}), \\ \psi_{i-1}(t_0,\dots,t_{i-1};x_1,\dots,x_i)\big), \quad (39)$$

with $\psi_0(t_0;x_j)=x_j$ for all j. Observe therefore that

$$\psi_K(t,\dots,t;x_0,\dots,x_K) = h_K(t;x_0,\dots,x_K). \quad (40)$$

Example 6.2 (De Casteljau algorithm). One can of course define a blossom for the Bézier curves of degree K on manifolds, based on the De Casteljau algorithm (Definition 2.2). In that case, the curve $f_i(\cdot; x, y) = g(\cdot; x, y)$ is the geodesic between x and y, for all i.

Proposition 6.3 (Endpoint interpolation and velocity). *For any set of data points $x_0, \dots, x_K \in \mathcal{M}$, the following properties hold:*

(i) $h_K(0; x_0, \dots, x_K) = x_0$,
(ii) $h_K(1; x_0, \dots, x_K) = x_K$,
(iii) $\dot{h}_K(0; x_0, \dots, x_K) = \sum_{i=1}^{K} \left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=0} f_i(t; x_0, x_1)$,
(iv) $\dot{h}_K(1; x_0, \dots, x_K) = \sum_{i=1}^{K} \left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=1} f_i(t; x_{K-1}, x_K)$.
(v) if $f_i(\cdot; x, y) \in C^1$, $h_K(\cdot; x_0, \cdot, x_K) \in C^1$ *as well.*

Proof. Properties (i) and (ii) follow directly from the definition of the functions f_i: indeed, one has $f_i(0; x, y) = x$ and $f_i(1; x, y) = y$, for all i. Properties (iii) and (iv) are proven in a similar way as [29, Theorem 1]. By (40) one has

$$\dot{h}_K(0; x_0, \dots, x_K) = \sum_{i=1}^{K} \left.\frac{\partial}{\partial t_i}\right|_{t_i=0} \psi_K(0, \dots, 0, t_i, 0, \dots, 0; x_0, \dots, x_K).$$

As $f_i(0; x, y) = x$ for $x, y \in \mathcal{M}$, one has, by [29, Lemma 3 (ii)],

$$\psi_K(0, \dots, 0, t_i, 0, \dots, 0; x_0, \dots, x_K) = f_i(t_i; x_0, x_1).$$

As a result, we obtain

$$\dot{h}_K(0; x_0, \dots, x_K) = \sum_{i=1}^{K} \left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=0} f_i(t; x_0, x_1).$$

(iv) is obtained symmetrically. Finally (v): the smoothness of h_K is preserved by composition. □

6.2 Bézier-like fitting curves

In this section, we propose two modifications of the classical De Casteljau algorithm (Definition 2.8) in order to produce a fitting curve for a set of data points $d_0, \dots, d_n$ associated with parameters $t_0 = 0, \dots, t_n = n$. The underlying idea is to modify the functions f_i used to construct the function h_K from Definition 6.1.

Definition 6.4 (Bezier-like fitting curve type-I). Let $\lambda > 0$. For $i \in \{0, \dots, n-1\}$, we define $\tilde{p}_i$, $\tilde{b}_i^+ \in T_{d_i}\mathcal{M}$ and $\hat{b}_{i+1}^-$, $\hat{p}_{i+1} \in T_{d_{i+1}}\mathcal{M}$, the vectors computed with (29) and (38) in their respective tangent spaces. We consider the corresponding points p_i, b_i^+, b_{i+1}^- and $p_{i+1} \in \mathcal{M}$. The *cubic Bézier-like curve* $h_3(\cdot; p_i, b_i^+, b_{i+1}^-, p_{i+1})\colon [0,1] \to$ $\mathcal{M}$ is recursively computed, according to Definition 6.1, with the following endpoint functions:

$$\begin{aligned}
f_1(t; p_i, b_i^+) &:= \gamma_i(1+t), \\
f_1(t; b_i^+, b_{i+1}^-) &:= g(t; b_i^+, b_{i+1}^-), \\
f_1(t; b_{i+1}^-, p_{i+1}) &:= \gamma_{i+1}(t), \\
f_k(t; x, y) &:= \mathrm{av}[(x, y), (1 - w(t), w(t))], \quad k = 2, 3,
\end{aligned}$$

where $w(t) = 3t^2 - 2t^3$ and $\gamma_i\colon [0,2] \to \mathcal{M}, \gamma_i(t) := \exp_{d_i}\left(\frac{(2-t)}{2}\tilde{b}_i^- + \frac{t}{2}\tilde{b}_i^+\right)$ is a straight line on $T_{d_i}\mathcal{M}$, mapped to $\mathcal{M}$. As illustrated in Figure 6, $\gamma(1) = p_i$. Observe that, in a mild abuse of notation, the definition of f_1 depends on the name of its arguments.

The *composite cubic Bézier-like curve of type I* $\mathbf{B}(t)$ is then defined according to Definition 2.9. This definition is illustrated in Figure 9.

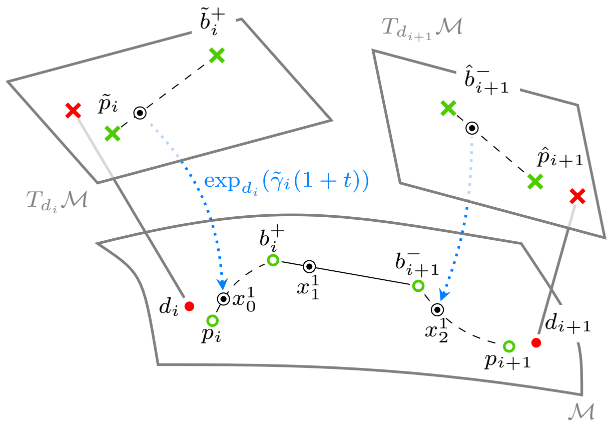


Fig. 9: Composite Bézier-like curve of type I. The first step of the algorithm is a hybrid method where the first (resp. last) geodesic is replaced by the curve $\gamma_i(1+t)$ (resp. $\gamma_{i+1}(t)$). The middle one is still a classical geodesic between b_i^+ and b_{i+1}^-. The next steps of the algorithm are classical weighted averagings.

Remark 6.5. This definition of Bézier-like curve is directly related to the interpolation conditions (38) but not restricted to cubic curves. Indeed, it can be easily generalized to curves of degree K, where $f_1(t; p_i, b_i^+)$ and $f_1(t; b_{i+1}^-, p_{i+1})$ would correspond to the Euclidean operation mapped to $\mathcal{M}$, while the "other" (endpoint) functions $f_1(t; x, y)$ would be classical geodesics between x and y.

Lemma 6.6. *Let $x, y \in \mathcal{M}$ and $w(t) = 3t^2 - 2t^3$. Let $z := f(t; x, y) = \mathrm{av}[(x, y), (1 - w(t), w(t))]$. Then*

$$\left.\frac{\mathrm{d}}{\mathrm{d}t}\right|_{t=s} f(t; x, y) = 0, \ \textit{for } s \in \{0, 1\}.$$

Proof. The optimality condition of $z = f(t;x,y)$ are given by [18, Thm 1.2]:

$$0 = \log_z(x)\,(1-w(t)) + \log_z(y)\,w(t) \coloneqq F(t,z).$$

By the implicit function theorem, one has

$$\frac{\mathrm{d}}{\mathrm{d}t} f(t;x,y) = -\left(D_z F(t,z)\right)^{-1} D_t F(t,z),$$

where $D_t F(t,z) = (\log_z(y) - \log_z(x))\,w'(t)$. As $w'(t) = 6t - 6t^2$, we see that $D_t F(t,z)|_{t=0} = D_t F(t,z)|_{t=1} = 0$, and $\frac{\mathrm{d}}{\mathrm{d}t}\big|_{t=s} f(t;x,y) = 0$ for $s \in \{0,1\}$. $\square$

Theorem 6.7. *Consider the data points $d_0,\dots,d_n \in \mathcal{M}$ associated with parameter-values $t_0 = 0,\dots,t_n = n$. The composite cubic Bézier-like curve $\mathbf{B}(t)$ of type I (Definition 6.4) satisfies the following properties:*

(i) $\mathbf{B}(i) \to d_i$, when $\lambda \to \infty$;
(ii) $\mathbf{B}(t)$ is differentiable for $t \in [0,n]$.

Proof. As $\mathrm{av}[(x,y),(1,0)] = x$ and $\mathrm{av}[(x,y),(0,1)] = y$, the cubic Bézier-like curves h_3 from Definition 6.4 are recursive functions as defined in Definition 6.1. Therefore, we can apply Proposition 6.3 (i–ii). By condition (38), we prove (i) because $\tilde{p}_i \to 0$ when $\lambda \to \infty$, so $p_i \to d_i$. For condition (ii), let $i = 1,\dots,n-1$. $\mathbf{B}(t)$ is smooth for $t \neq i$, as h_3 is smooth (Proposition 6.3 (v)). For $t = i$, we have by Lemma 6.6 that $\frac{\mathrm{d}}{\mathrm{d}t}\big|_{t=s} f_k(t;x,y) = 0$, for $s \in \{0,1\}$. Therefore, by Proposition 6.3 (iii–iv), one has

$$\left.\frac{\mathrm{d}\mathbf{B}(t)}{\mathrm{d}t}\right|_{i^-} = \left.\frac{\mathrm{d}\gamma_i(t)}{\mathrm{d}t}\right|_{1^-}$$

and

$$\left.\frac{\mathrm{d}\mathbf{B}(t)}{\mathrm{d}t}\right|_{i^+} = \left.\frac{\mathrm{d}\gamma_i(t)}{\mathrm{d}t}\right|_{1^+}.$$

As $\gamma_i(t)$ is differentiable, so is $\mathbf{B}(t)$. $\square$

Remark 6.8. In Definition 6.4, one could be tempted to simply use classical geodesics in place of $f_k(t;x,y)$, $k = 2,3$. However, by Proposition 6.3 the left and right velocities of $\mathbf{B}$ at $t = i$ will in general not be the same: indeed, as shown in Proposition 4.11, b_i^-, p_i and b_i^- are not always aligned.

Proposition 6.9 (Minimal representation of the curve)**.** *The fitting Bézier-like curve of Definition 6.4 is uniquely represented by $2(n+1)$ tangent vectors.*

Proof. The proof is similar to the one of Proposition 4.9, the same tangent vectors can be used to represent the curve. $\square$

Proposition 6.10 (Exponential and logarithm maps required)**.** *The number of exponential and logarithm maps required by the fitting Bézier-like curve of Definition 6.4 is*

- *$n(n+1)$ logarithms for the construction of the minimal representation of the curve;*
- *8 exponential maps and 4 logarithm maps to reconstruct the curve $\mathbf{B}(t)$, for a given parameter value t, given that minimal representation.*

Proof. The proof is here also similar to the one of Proposition 4.10. The differences between the two methods is that two geodesics are spared at the first step of the algorithm. $\square$

One strength of the method is that differentiability of the composite curve $\mathbf{B}$ (Definition 6.4) at $t = i$ depends only on the differentiability of γ_i. The curve $\gamma_i(t)$ can be replaced by any differentiable curve between b_i^- and b_i^+ and such that $\gamma(0.5) = p_i$. This property is possible because $\frac{\mathrm{d}}{\mathrm{d}t}\,\mathrm{av}[(x,y),(1-w,w)]|_{t=s} = 0$, for $s \in \{0,1\}$ and $w(t) = 3t^2 - 2t^3$. Then, most of the computation of the pieces of $\mathbf{B}(t)$ can be transferred to the tangent spaces $T_{d_i}\mathcal{M}$ and $T_{d_{i+1}}\mathcal{M}$, $i = 0,\dots,n-1$. For cubic curves, for instance, the curve $\gamma_i\colon [0,2] \to \mathcal{M}$ can be the mapping to $\mathcal{M}$ of $\tilde{\gamma}_i(t)$, composed of two C^1-patched quadratic Bézier curves computed on the tangent space of d_i, as

$$\tilde{\gamma}_i(t) = \begin{cases} \beta_2(t;\tilde{b}_{i-1}^+,\tilde{b}_i^-,\tilde{p}_i) & \text{for } t \in [0,1] \\ \beta_2(t-1;\tilde{p}_i,\tilde{b}_i^+,\tilde{b}_{i+1}^-) & \text{for } t \in [1,2], \end{cases} \tag{41}$$

where $\tilde{x}$ is the point $x \in \mathcal{M}$ represented in $T_{d_i}\mathcal{M}$. The resulting curve h_3 would thus be an averaging of the two curves $\gamma_i(t)$ and $\gamma_{i+1}(t)$, as represented on Figure 10.

This leads us to the following definition.

Definition 6.11 (Bézier-like fitting curve type-II)**.** Let $\lambda > 0$. For $i \in \{0,\dots,n-1\}$, let $p_i, b_i^+, b_{i+1}^-, p_{i+1}$ be the control points computed with (29) and (31), and let $\tilde{x} = \log_{d_i}(x) \in T_{d_i}\mathcal{M}$ and $\hat{x} = \log_{d_{i+1}}(x) \in T_{d_{i+1}}\mathcal{M}$, the representation of these control points in the corresponding tangent spaces. Let also $\tilde{b}_{i+1}^- = \log_{d_i}(\exp_{d_{i+1}}(\hat{b}_{i+1}^-))$, $i = 0,\dots,n-1$, and $\hat{b}_i^+ = \log_{d_{i+1}}(\exp_{d_i}(\tilde{b}_i^+))$, $i = 0,\dots,n-1$. The *cubic Bézier-like curve (type II)* $y(t)\colon [0,1] \to \mathcal{M}$ is computed with the following iterative procedure:

$$\begin{aligned} \tilde{x}_i^L &\coloneqq \beta_2(t;\tilde{p}_i,\tilde{b}_i^+,\tilde{b}_{i+1}^-) \\ \hat{x}_i^R &\coloneqq \beta_2(t;\hat{b}_i^+,\hat{b}_{i+1}^-,\hat{p}_{i+1}) \\ x_i^L &\coloneqq \exp_{d_i}\left(\tilde{x}_i^L\right) \\ x_i^R &\coloneqq \exp_{d_{i+1}}\left(\hat{x}_i^R\right) \\ y(t) &\coloneqq \mathrm{av}[(x_i^L,x_i^R),(1-w(t),w(t))] \end{aligned}$$

where $w(t) = 3t^2 - 2t^3$. The *composite cubic Bézier-like curve (type II)* $\mathbf{B}(t)$ is then defined as (8). This definition is illustrated at Figure 10.

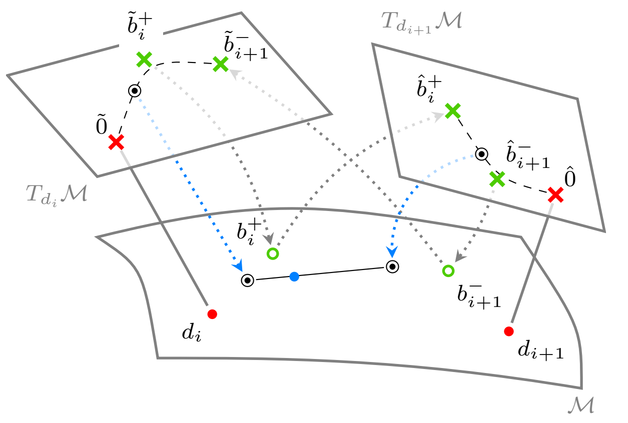


Fig. 10: Composite Bézier-like curve of type II. The first step of the algorithm consists in mapping on the manifold the quadratic Bézier curves computed on the tangent spaces $T_{d_i}\mathcal{M}$ and $T_{d_{i+1}}\mathcal{M}$. The value of the composite Bézier-like curve is then obtained by a classical weighted averaging of these two points (black circles).

Remark 6.12. The exponential and logarithm map evaluations represent usually the major part of the computation effort. Indeed, these maps are not always closed form and might require lengthy iterative procedures to be evaluated (e.g., on the space of shapes [19]). A direct way to spare $2n-2$ Exp-Log evaluations is to compute the control points directly on the dedicated tangent space, to avoid transfers from one tangent space to another, as in Algorithm 3. The curve will remain C^1 by Properties 6.3, (iii–iv).

Theorem 6.13. *Consider the data points $d_0, \dots, d_n \in \mathcal{M}$ associated with parameter-values $t_0 = 0, \dots, t_n = n$. The composite cubic Bézier-like curve $\mathbf{B}(t)$ of type II (Definition 6.11) satisfies the following properties:*

(i) $\mathbf{B}(i) \to d_i$, *when* $\lambda \to \infty$*;*
(ii) $\mathbf{B}(t)$ *is differentiable for* $t \in [0, n]$*.*

Proof. The proof is similar to the proof of Theorem 6.7. The condition (i) is reached via Proposition 6.3 (i–ii) and condition (31). Let $i = 1, \dots, n-1$. Differentiability for $t \neq i$ is trivial. For $t = i$, condition (ii) follows from Lemma 6.6: as $\gamma_i(t) = \exp_{d_i}(\tilde{\gamma}_i(t))$ (Equation (41)) is differentiable, so is $\mathbf{B}(t)$. □

Proposition 6.14 (Minimal representation of the curve)**.** *The fitting Bézier-like curve of Definition 6.11 is uniquely represented by $2(n+1)$ tangent vectors.*

Proof. The proof is similar to the one of Proposition 4.9, the same tangent vectors can be used to represent the curve. □

Remark 6.15. There is no need to store the tangent vectors $\tilde{b}_{i+1}^- = \log_{d_i} b_{i+1}^-$ and $\hat{b}_i^+ = \log_{d_{i+1}} b_i^+$, as they can be recovered afterwards. Indeed, $\tilde{b}_{i+1}^- = \log_{d_i}\left(\exp_{d_{i+1}} \hat{b}_{i+1}^-\right)$, and $\hat{b}_i^+ = \log_{d_{i+1}}\left(\exp_{d_i} \tilde{b}_i^+\right)$.

Proposition 6.16 (Exponential and logarithm maps required)**.** *The number of exponential and logarithm maps required by the fitting Bézier-like curve of Definition 6.11 is*

- *$n(n+1)$ logarithms for the construction of the minimal representation of the curve of Proposition 6.14*
- *5 exponential maps and 3 logarithm maps to reconstruct the curve $\mathbf{B}(t)$, for a given parameter value t, given that minimal representation.*

Proof. The $n(n+1)$ logarithm maps are required to represent the data points d_j in the tangent space $T_{d_i}\mathcal{M}$, $j \neq i$. At the reconstruction step, 2 exponentials and 2 logarithms are required to compute $\hat{b}_i^+$ and $\tilde{b}_{i+1}^-$, as mentioned in Remark 6.15. Finally, 2 exponential maps are required to map $\tilde{x}_i^L$ and $\hat{x}_i^R$ to $\mathcal{M}$, and the averaging of these two points costs 1 exponential map and 1 logarithm map. □

Remark 6.17. A last thing to remark about Definitions 6.4 and 6.11 is that the reconstructed composite cubic Bézier-like curves no longer reduce to natural cubic splines when $\mathcal{M} = \mathbb{R}^m$. This will be shown numerically in the next section.

7 Numerical examples

In this section, we compare the four fitting methods described in the paper. That is, the Bézier fitting curve from Definition 4.7 (Bézier), the blended fitting curve from Definition 5.5 (Blend), and the Bézier-like fitting curves defined in Section 6, i.e., Definitions 6.4 (BL-I) and 6.11 (BL-II). The goals of the tests we have performed are threefold. The first goal is to validate numerically some of the target properties from the introduction. We consider here two such properties: property (i) (interpolation of the data as $\lambda \to \infty$), see Section 7.1, and property (iii) (recovering of the natural cubic spline in the Euclidean case), see Section 7.2. The second goal of this section is to compare the curves regarding the value of the optimization problem (1). This is done in Section 7.3, for two different manifolds: the sphere $\mathbb{S}^2$ and the special orthogonal group SO(3). Finally, the third and last goal of this section is to compare the methods on a curve fitting application. For this, we use the wind field data from [13] (the one we already used in Section 4.3) to illustrate the ability of the different methods to recover some "hidden" data points. This is done in Section 7.4.

As a comparison, we also consider the simpler approach in which the fitting curve is entirely computed in a unique tangent space $T_{d_{\mathrm{ref}}}\mathcal{M}$. One of the drawbacks of this approach is that the result usually depends on the tangent space chosen. In short, we compute the optimal (Euclidean) smoothing spline in $T_{d_{\mathrm{ref}}}\mathcal{M}$ using the control points (29) and the condition (38), with a unique $d_{\mathrm{ref}} = d_{\mathrm{mid}}$, where $d_{\mathrm{mid}} = d_{n/2}$ if n is even, and

$$d_{\mathrm{mid}} = g(0.5, d_{(n-1)/2}, d_{(n+1)/2})$$

if n is odd. This (Euclidean) fitting curve is then mapped back to $\mathcal{M}$. The latter method will be referred to as TS (for Tangent Space) in our results.

7.1 Validation of property (i): interpolation as $\lambda \to \infty$

We consider the two data sets from Section 4.3 (namely, the unit circle $\mathbb{S}^1$ represented on Figure 5a and the wind field data, lying on $\mathcal{S}_+(p, q)$, the manifold of positive semidefinite matrices of rank q). We compute the different fitting curves for $\lambda \to \infty$ on these datasets. Similarly to Section 4.3, the parameter λ is set to 10^8.

The resulting curves are represented in Figure 11a and Figure 12. These two figures indicate that all the methods satisfy the interpolation property, for both datasets. Moreover, Figure 11a indicates that all the methods tend to behave in a similar way, except the reference method TS, which leads to a considerably different curve. This can be explained by the fact that the data points are spread out on $\mathbb{S}^1$. Therefore, the mappings $\log_{d_{\mathrm{mid}}}(d_i)$, $i = 0, \ldots, 4$, are in general very distorded representations of the data points $d_i \in \mathbb{S}^1$. It can also be observed on Figure 12 that the errors for the method TS are larger on the boundaries of the interval, i.e., in the region in which the distortions expected from the mapping into the tangent space $T_{d_{\mathrm{mid}}}\mathcal{M}$ are the largest.

As a comparison, we computed the optimal solution on the circle (Opt). Indeed, on that particular manifold, is it possible to compute the optimal solution to (1). The only particularity of the circle, with respect to the Euclidean space, is that two points distant of an angle 2π are equivalent. So, we can solve the problem (1) in two steps: first, compute the optimal representative for the angles characterizing the data, which is a combinatorial problem, and then, compute the composite Bézier curve in $\mathbb{R}$ that fits those values.

By comparing the mean squared accelerations of the different curves (Bézier: 52.2, Blend: 66.7, BL-I: 56.3, BL-II: 34.2, TS: 71.9, Opt: 17.8), we observe that BL-II performs surprisingly well. We see that the blended cubic spline performs a bit less good, but as we show in Section 7.3, this is generally not observable when working with other (randomly generated) datasets. Actually, we expect this acceleration to be due to the choice of the data points, that are in our case far away from each other. Indeed, this causes here a large difference between the two blended curves $L(t)$ and $R(t)$, as represented in Figure 11b (see for instance between the points at $t = 0$ and $t = 1$). This difference highlights here the importance of the reference points chosen in all proposed methods. How to tackle this problem while keeping the properties (i–vi) is still an open question.

7.2 Validation of property (iii): natural cubic splines in the Euclidean case

Figure 13 illustrates the curves obtained for a fitting task ($\lambda = 10^2$), on the bidimensional Euclidean space $\mathbb{R}^2$. We verify here that the curves reconstructed by the methods Blend, TS and Bézier are the natural cubic spline, and that the curves obtained via BL-I and BL-II differ from the optimal solution. Each component of the data points $d_i = (d_{i,x}, d_{i,y}) \in \mathbb{R}^2$, $i = 0, \ldots, 5$, was chosen randomly as $d_{i,x}, d_{i,y} \sim \mathcal{N}(0, 1)$.

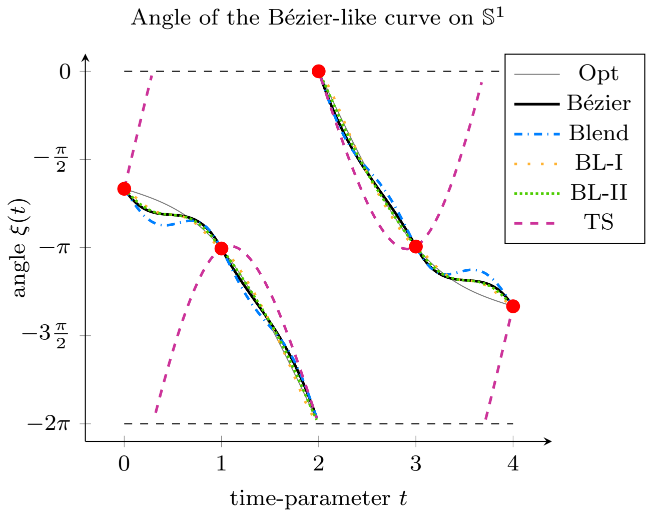
We know by Theorem 5.7 and Proposition 4.2 that $\mathbf{B}(t)$ is the natural spline $\mathbf{B}^\star(t)$ when it is reconstructed as a blended cubic spline (Blend), or as a classical Bézier curve (Bézier and TS), as shown in Figure 13 (left). Figure 13 (right) shows us that the curves reconstructed by the methods from Section 6 (BL-I and BL-II) are not the natural spline $\mathbf{B}^\star(t)$ and that their speed and their path differ strongly.

Figure 13 (left) also suggests that all methods would return the same curve position at $t = \frac{k}{2}$, $k \in \mathbb{Z}$. Indeed, as $w(0) = 0$, $w(0.5) = 0.5$ and $w(1) = 1$, the curves reconstructed by all methods are identical there.

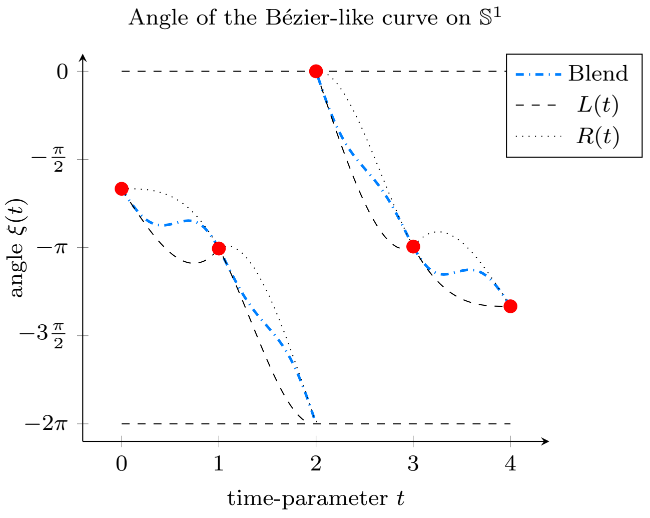
7.3 Mean acceleration of the curves

We compare here the acceleration of the different curves, on two manifolds: the sphere $\mathbb{S}^2$ and the special orthogonal group SO(3).

For each manifold, we generated randomly $N = 1000$ geodesics $\gamma_k \colon [0, 1] \to \mathcal{M}$. From these geodesics, we extracted 6 points $x_{p,k} = \exp_{\gamma_k(p/5)}(v)$, $p = 0, \ldots, 5$, where $v \in T_{\gamma_k(p/5)}\mathcal{M}$ is a random vector whose components are distributed according to a classical normal law $\mathcal{N}(0, 0.1^2)$ of mean 0 and standard deviation 0.1. On each data set $(x_{p,k})_{p=0}^5$, with $k = 1, \ldots, N$, we built the five fitting curves at 1000 equispaced times $t \in [0, 5]$, and evaluated the acceleration $\ddot{\mathbf{B}}_k(t)$ of each of them using second order manifold valued finite differences [6,



(a) The five methods interpolate the data points.



(b) Explanation of the behavior of the blended cubic spline.

Fig. 11: Comparison of the five methods on the counter-example of Section 4.3, on $\mathbb{S}^1$. **Left**: the fitting curves are computed for a parameter value $\lambda = 10^8$, with all presented methods. In accordance with Proposition 4.6, the data points are almost interpolated by all methods. We see that the method TS can lead to a drastically different curve (pink dashed). **Right**: The blended cubic spline is computed, at each time, as a weighted average of $L(t)$ and $R(t)$ (see Definition 5.1). The fact that the two curves are computed on two different tangent spaces results here in strong differences between them after projection on the manifold.

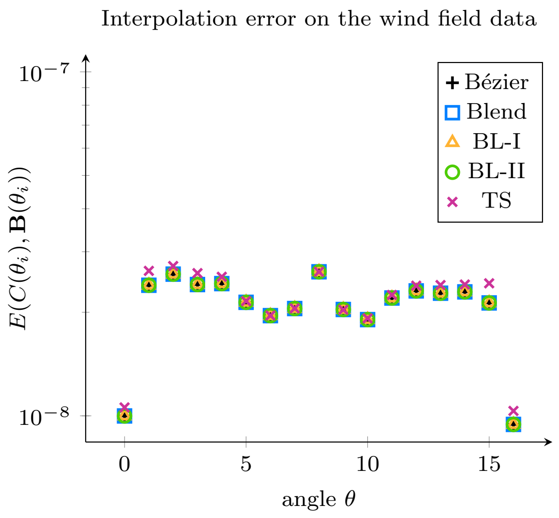


Fig. 12: The five reconstruction methods interpolate the data points of the counter-example on $\mathcal{S}_+(p,q)$ of Section 4.3, when condition (31) is respected. As a comparison, Figure 4 shows the situation when condition (31) is not respected.

§3.1]:

$$\ddot{\mathbf{B}}_k(t_i) = \frac{\log_{\mathbf{B}_k(t_i)}(\mathbf{B}_k(t_{i+1})) + \log_{\mathbf{B}_k(t_i)}(\mathbf{B}_k(t_{i-1}))}{\Delta\tau^2},$$

where $\Delta\tau = t_i - t_{i-1}$.

Figure 14a displays, for the sphere $\mathbb{S}^2$, the mean acceleration of the curves $\mathbb{E}[\|\ddot{\mathbf{B}}(t)\|]$, estimated by averaging the results on the N datasets given by

$$\mathbb{E}[\|\ddot{\mathbf{B}}(t)\|] \simeq \frac{1}{N}\sum_{k=1}^{N}\|\ddot{\mathbf{B}}_k(t)\|.$$

This figure indicates that the method TS results on average in a curve with a considerably larger acceleration than the four other methods.

Figure 14b presents the results of the same tests, but on the special orthogonal group SO(3). We observe here similar results as on the sphere.

7.4 Ability to recover left out data points

Finally, we compare here the different fitting methods in a real-life fitting application, namely the wind field application considered in [13]. We built a training set using one data point out of three, the two other data points being used for the validation set. The quotient geometry of the manifold of fixed-rank positive-semidefinite matrices is studied in [24].

For each fitting method considered, we computed the mean square error in dB between the validation data and the values taken by the curve, at the corresponding angles:

$$\mathrm{MSE}(\mathbf{B}) = 10\log\left(\frac{\sum_{i\in I_V}\|C(\theta_i) - \mathbf{B}(\theta_i)\|_F^2}{\sum_{i\in I_V}\|C(\theta_i)\|_F^2}\right),$$

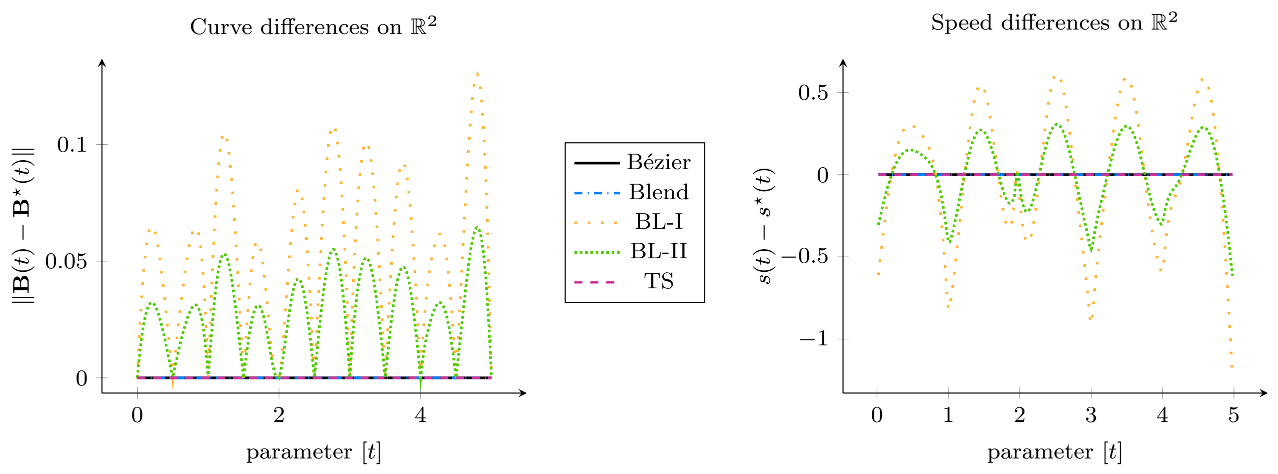
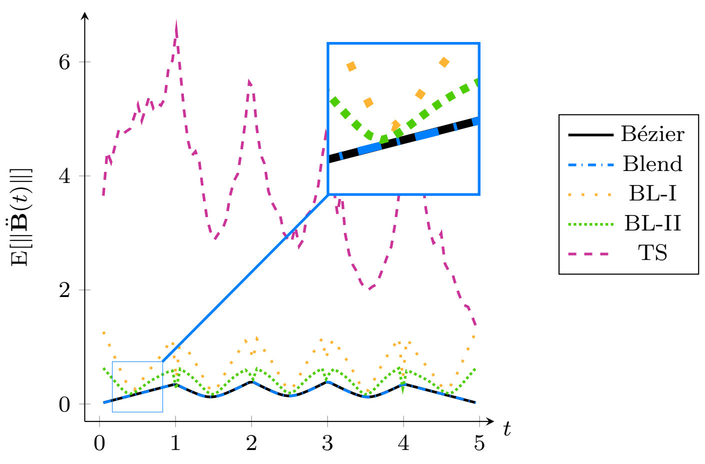
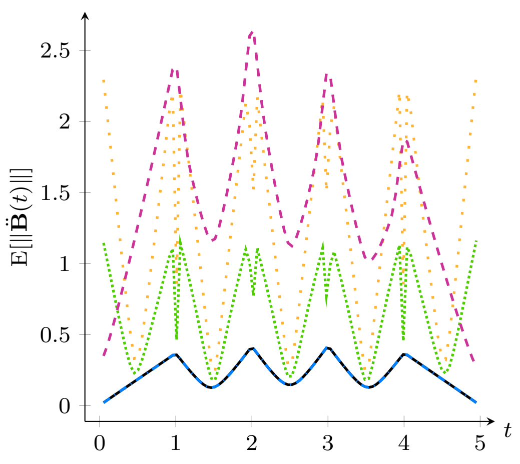


Fig. 13: The curves $\mathbf{B}(t)$ computed by BL-I and BL-II are different from the optimal curve $\mathbf{B}^\star(t)$ (left) and their speed $s(t)$ then differ from the speed $s^\star(t)$ of the optimal curve (right).



(a) Mean acceleration on the sphere $\mathbb{S}^2$.



(b) Mean acceleration on SO(3).

Fig. 14: Average acceleration of the curves, computed on 1000 random datasets, with $\lambda = 100$. The blending cubic spline outperforms the methods BL-I, BL-II and TS.

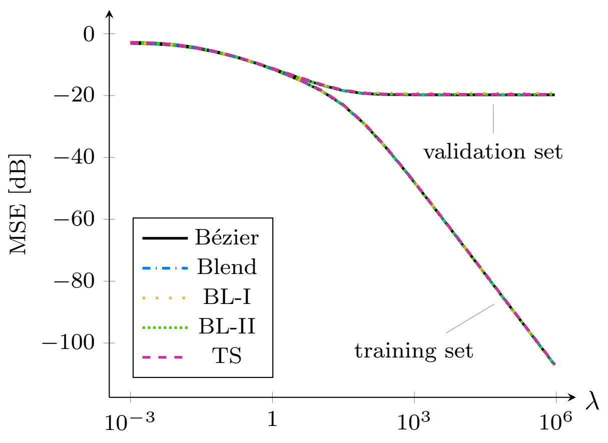
where I_{V} stands for the indices of the points belonging to the validation set.

This process was repeated for several values of the fitting parameter λ, and the results are displayed on Figure 15a. This figure indicates that the different fitting methods considered yield comparable results, the methods BL-I and TS being slightly less accurate than the others.

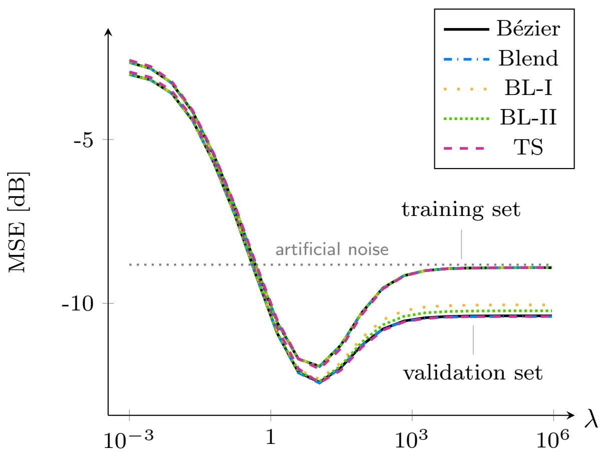
8 Summary and further work

To conclude, we summarize here the differences between the methods proposed in the paper. We also include in our discussion the TS method introduced in Section 7.

Among all presented methods, only two of them satisfy the six target properties from the introduction (see Table 1): the blended cubic spline (Definition 5.1) and the TS method. All other methods fail to satisfy at least one of the properties: Bézier curves either may fail to interpolate the data points when $\lambda \to \infty$ (Definition 4.1), or loose differentiability (Definition 4.7); BL-I and BL-II validate properties (i–ii) but no longer reduce to the natural smoothing spline when $\mathcal{M}$ is a Euclidean space. All methods, however, are based on simple computation. They all meet the properties (iv–vi) to only require the exponential and logarithm maps of the manifold (iv), to only need $\mathcal{O}(n)$ tangent vectors to represent the fitting curve (v), and to only need $\mathcal{O}(1)$ operations to reconstruct it (vi). Finally, we have seen in Section 7 that TS leads usually to curves with a larger acceleration than the blended cubic spline, when the manifold is not flat.



(a) Without additional noise.



(b) With additional noise.

Fig. 15: Results on the wind field fitting problem (Section 4.3). The training set was made of one point out of three from the dataset [13], while the validation set is made of all remaining data points. The mean square error (MSE) is obtained on the training and validation sets, depending on the value of the regularization parameter λ. All the fitting methods considered behave similarly on this fitting problem. **Left**: The error on the training set decreases as λ increases (the problem becomes an interpolation problem), while the error on the validation set reaches a constant level (remaining error, due to the inability of the model to recover the left out data). **Right**: The data have been corrupted as follows: the covariance matrices of the training set were perturbed by a Gaussian perturbation: $C(\theta_i) \leftarrow C(\theta_i) + 0.05N(\theta_i)$, with $N(\theta_i)_{lm} \sim \mathcal{N}(0,1)$. This artificial noise amounts to adding a MSE of about -9 dB on the training set. The error displayed is computed with respect to the original (non-corrupted) data. As $\lambda \to \infty$, the error on the training set reaches a constant level (we are interpolating the noisy data). However, for $\lambda > 1$, the error on the validation set reaches a smaller value, with an optimal denoising parameter at $\lambda \simeq 100$.

	Bézier – Def. 4.1	Bézier – Def. 4.7	Blend	BL-I	BL-II	TS
Interpolation when $\lambda \to \infty$	✗	✓	✓	✓	✓	✓
Differentiability	✓	✗	✓	✓	✓	✓
Natural spline when $\mathcal{M} = \mathbb{R}^r$	✓	✓	✓	✗	✗	✓
Only exps and logs	✓	✓	✓	✓	✓	✓
Curve encoded by $\mathcal{O}(n)$ tangent vectors	✓	✓	✓	✓	✓	✓
Reconstruction by $\mathcal{O}(1)$ exps and logs	✓	✓	✓	✓	✓	✓

Table 1: Properties of the methods. The blended cubic spline gathers all properties. The TS method introduced in Section 7 also satisfies all those properties, but was shown numerically to return a curve with a larger acceleration when the manifold is curved.

The complexity of the various methods is nearly equivalent with a small advantage to the blended cubic spline. Indeed, the number of exponential and logarithm maps needed to generate the tangent vectors representing the curve (a step that can be done once and for all, independently of the discretization of the curve to reconstruct) is the same for all methods. The difference stands in the complexity of the evaluation of $\mathbf{B}$ for a given t and in the number of tangent vectors to store to represent the curve. The blended cubic spline, even if it requires to store more tangent vectors to represent the whole curve, has the advantage to be less consuming in operations to construct $\mathbf{B}$ at a given t. The TS method requires fewer evaluations of exponential and logarithm maps than any other, but performs poorly qualitatively, as we can see in Figure 11a. The details of the complexity are presented in Table 2 and Table 3.

Therefore, in view of the results presented in Section 7, we can conclude that the blended cubic spline method is a good tradeoff between accuracy, efficiency and low storage requirements.

We now draw some perspectives. A consequence of property (iii) is that, if the manifold $\mathcal{M}$ reduces to a Euclidean space and $\lambda \to 0$, then the solution curve γ converges to the least-squares linear regression solution. None of the various methods presented in this paper satisfy the following stronger version of this property: for any manifold $\mathcal{M}$, when $\lambda \to 0$, the solution curves converges to the least-squares geodesic regression of the data points. How to design an algorithm that satisfies

	Computation of the tangent vectors representing the curve		Reconstruction of the curve at a given time t	
	# Exps	# Logs	# Exps	# Logs
Bézier	0	$n(n+1)$	10	6
Blend	**0**	$\mathbf{n(n+1)}$	**3**	**1**
BL-I	0	$n(n+1)$	8	4
BL-II	0	$n(n+1)$	5	3
TS	0	$n(n+1)$	1	0

Table 2: Number of exponential and logarithm maps required by the algorithms, for $n+1$ data points. The blended cubic spline has the second fastest reconstruction algorithm.

	Tangent vectors required
Bézier	$2(n+1)$
Blend	$6(n-1)+8$
BL-I	$2(n+1)$
BL-II	$2(n+1)$
TS	$2(n+1)$

Table 3: Minimal number of tangent vectors to store to represent the curve, for $n+1$ data points. The blended cubic spline requests more memory but compensates with a reconstruction faster than the others.

this stronger property along with properties (i)–(vi) remains an open problem.

We have observed in Figure 11b that the blending method could sometimes behave in an unexpected way, due to the choice of the reference points $d_{\text{ref},1}$ and $d_{\text{ref},2}$. How to optimally choose these two points also remains an open question. A possible solution to avoid such behaviour would be to choose the weight $w(t)$ differently in the blending step.

Appendix

A.1 Interpolating Bézier curves: coefficient matrices for control points generation

The problem (21) on $\mathcal{M} = \mathbb{R}^m$ is a quadratic function to be optimized with respect to the $n+1$ optimization variables $X = (b_0^+, b_1^-, b_2^-, \dots, b_n^-)$. The solution of that problem reduces to m independent linear systems $A \cdot X_k = C \cdot D_k$, where X_k is the vector of the k^{th} component of the points of X, and D_k is the vector of the k^{th} component of the data points $(d_i)_{i=0}^n$. We obtain

$$\int_i^{i+1} \|\ddot{\beta}_3(t-i; d_i, b_i^+, b_{i+1}^-, d_{i+1})\|_2^2 \mathrm{d}t =$$
$$12(-3d_i b_i^+ + 3b_i^+ b_i^+ - 3b_i^+ b_{i+1}^- + 3b_{i+1}^- b_{i+1}^- - 3b_{i+1}^- d_{i+1} + K),$$

with $i = \lfloor t \rfloor$. K gathers the terms that are independent from the optimization variables. Introducing the differentiability constraints (16) $b_i^+ = 2p_i - b_i^-$, and β_3^i, the i^{th} segment of the composite cubic Bézier curve, one has

$$\frac{\partial \beta_3^0}{\partial b_0^+} = 6b_0^+ - 3b_1^- - 3d_0 \tag{42}$$

$$\frac{\partial \beta_3^0}{\partial b_1^-} = -3b_0^+ + 6b_1^- - 3d_1 \tag{43}$$

and for $i = 1, \dots, n-1$, $n \geq 2$

$$\frac{\partial \beta_3^i}{\partial b_i^-} = 6b_i^- + 3b_{i+1}^- - 9d_i \tag{44}$$

$$\frac{\partial \beta_3^i}{\partial b_{i+1}^-} = 3b_i^- + 6b_{i+1}^- - 6d_i - 3d_{i+1}. \tag{45}$$

By definition 2.9, (21) is minimized when these quantities vanish, which yields the linear system $A \cdot X = C \cdot D$, for $n \geq 2$ where (in matlab indexing)

$$\begin{aligned} A(1,1:2) &= \begin{bmatrix} 6 & -3 \end{bmatrix} \\ A(2,1:3) &= \begin{bmatrix} -3 & 12 & 3 \end{bmatrix} \\ A(\mathtt{k},\mathtt{k}-1:\mathtt{k}+1) &= \begin{bmatrix} 3 & 12 & 3 \end{bmatrix} \quad k = 3, \dots, n \\ A(\mathtt{n}+1,\mathtt{n}:\mathtt{n}+1) &= \begin{bmatrix} 3 & 6 \end{bmatrix}. \end{aligned}$$

and

$$\begin{aligned} C(1,1) &= 3 \\ C(2,2) &= 12 \\ C(\mathtt{k},\mathtt{k}-1:\mathtt{k}) &= \begin{bmatrix} 6 & 12 \end{bmatrix} \quad k = 3, \dots, n \\ C(\mathtt{n}+1,\mathtt{n}:\mathtt{n}+1) &= \begin{bmatrix} 6 & 3 \end{bmatrix}. \end{aligned}$$

The third lines in the definition of A and C only hold for $n > 2$. All the other entries are equal to zero.

A.2 Fitting curves : coefficient matrices for control points generation

The problem (26) on $\mathcal{M} = \mathbb{R}^m$ is a quadratic function to be optimized with respect to the $2n+2$ optimization variables $X = (p_0, b_0^+, b_1^-, b_1^+, \dots, b_n^-, p_n)$. As in appendix A.1, the solution of that problem reduces to m independent linear systems $(A_0 + \lambda A_1) \cdot X_k = \lambda C \cdot D_k$. This system depends on the regularization parameter $\lambda > 0$, on X, and on the points $(d_i)_{i=0}^n$ in D.

For $n \geq 4$, the matrices of coefficients $A_0, A_1 \in \mathbb{R}^{(2n+2)\times(2n+2)}$ and $C \in \mathbb{R}^{(2n+2)\times(n+1)}$ are given by the following sparse matrices.

A_0 is given, for $i = 2, \dots, n-2$, by

$$\begin{aligned}
A_0(\mathtt{1,1:4}) &= \begin{bmatrix}24 & -36 & 6 & 6\end{bmatrix}\\
A_0(\mathtt{2,1:4}) &= \begin{bmatrix}-36 & 72 & -36 & 0\end{bmatrix}\\
A_0(\mathtt{3,1:6}) &= \begin{bmatrix}6 & -36 & 48 & -24 & 3 & 3\end{bmatrix}\\
A_0(\mathtt{4,1:6}) &= \begin{bmatrix}6 & 0 & -24 & 48 & -33 & 3\end{bmatrix}\\
A_0(\mathtt{2i+1,2i-1:2i+6}) &= \begin{bmatrix}3 & -33 & 48 & -24 & 3 & 3\end{bmatrix}\\
A_0(\mathtt{2i+2,2i-1:2i+6}) &= \begin{bmatrix}3 & 3 & -24 & 48 & -33 & 3\end{bmatrix}\\
A_0(\mathtt{2n-1,2n-3:2n+2}) &= \begin{bmatrix}3 & -33 & 48 & -24 & 0 & 6\end{bmatrix}\\
A_0(\mathtt{2n,2n-3:2n+2}) &= \begin{bmatrix}3 & 3 & -24 & 48 & -36 & 6\end{bmatrix}\\
A_0(\mathtt{2n+1,2n-1:2n+2}) &= \begin{bmatrix}0 & -36 & 72 & -36\end{bmatrix}\\
A_0(\mathtt{2n+2,2n-1:2n+2}) &= \begin{bmatrix}6 & 6 & -36 & 24\end{bmatrix}.
\end{aligned}$$

The coefficients of A_1, are

$$\begin{aligned}
A_1(\mathtt{1,1}) &= \lambda\\
A_1(\mathtt{2,2}) &= 0\\
A_1(\mathtt{2i-1,2i-1:2i}) &= \tfrac{1}{2}\begin{bmatrix}\lambda, \lambda\end{bmatrix}\\
A_1(\mathtt{2n+1,2n+1}) &= 0\\
A_1(\mathtt{2n+2,2n+2}) &= \lambda,
\end{aligned}$$

for $i = 2, \dots, n$. Finally, the coefficients of C are given, for $i = 2, \dots, n$, by

$$\begin{aligned}
C(\mathtt{1,1}) &= 2\lambda\\
C(\mathtt{2i-1,i}) &= \lambda\\
C(\mathtt{2i,i}) &= \lambda\\
C(\mathtt{2n+2,n+1}) &= 2\lambda.
\end{aligned}$$

The other entries are equal to zero.

A.3 Elements of differential geometry

The tables 4 and 5 give the explicit formulae used to evaluate the exponential map and the logarithm in this paper. They are implemented in Manopt [7] as a proper factory.

Sphere $\mathbb{S}^{m-1}$: the set of normed vectors of size m.	
$\mathbb{S}^{m-1} = \{x \in \mathbb{R}^m \ : \ x^\top x = 1\}$	
$T_x\mathbb{S}^{m-1} = \{v \in \mathbb{R}^m \ : \ x^\top v = 0\}$	
Inner product	$\langle v_1, v_2\rangle_x = v_1^\top v_2$
Distance	$\mathrm{d}(x,y) = \arccos(x^\top y)$
Exponential	$\exp_x(v) = x\cos(\lVert v\rVert) + \frac{v}{\lVert v\rVert}\sin(\lVert v\rVert)$
Logarithm	$\log_x(y) = \frac{(I_d - xx^\top)y}{\sqrt{1-(x^\top y)^2}}\arccos(x^\top y)$

Table 4: Riemannian operators for $\mathbb{S}^{m-1}$ extracted from [31].

The special orthogonal group SO(m).	
$\mathrm{SO}(m) = \{X \in \mathbb{R}^{m\times m} \ : \ X^\top X = I,\ \det(X) = 1\}$	
$T_X\mathrm{SO}(m) = \{H \in \mathbb{R}^{m\times m} \ : \ X^\top H + H^\top X = 0\}$	
Inner product	$\langle H_1, H_2\rangle = \mathrm{trace}\left(H_1^\top H_2\right)$
Distance	$\mathrm{d}(X,Y) = \lVert \log\left(X^\top Y\right)\rVert_\mathrm{F}$
Exponential	$\exp_X(H) = X\exp\left(X^\top H\right)$
Logarithm	$\log_X(Y) = X\log\left(X^\top Y\right)$

Table 5: Riemannian operators for SO(m) extracted from [6]. Note the difference between $\exp_X(H)$, the Riemannian exponential, and $\exp(X)$, the matrix exponential.

Acknowledgements The codes developed for this paper use the Manopt toolbox [7]. Special thanks to Benedikt Wirth for the very productive discussions: several methods presented here were conceived during an ongoing joint project on Bézier fitting surfaces.

References

1. Absil, P.A., Gousenbourger, P.Y., Striewski, P., Wirth, B.: Differentiable Piecewise-Bézier Surfaces on Riemannian Manifolds. SIAM Journal on Imaging Sciences **9**(4), 1788–1828 (2016). DOI 10.1137/16M1057978
2. Absil, P.A., Mahony, R., Sepulchre, R.: Optimization Algorithms on Matrix Manifolds. Princeton University Press, Princeton, NJ (2008). URL http://press.princeton.edu/chapters/absil/
3. Arnould, A., Gousenbourger, P.Y., Samir, C., Absil, P.A., Canis, M.: Fitting Smooth Paths on Riemannian Manifolds : Endometrial Surface Reconstruction and Preoperative MRI-Based Navigation. In: F. Nielsen, F. Barbaresco (eds.) Geometric Science of Information, *Lecture Notes in Computer Sciences*, vol. 9389, pp. 491–498. Springer, Berlin, Heidelberg (2015). DOI 10.1007/978-3-319-25040-3_53
4. Bagaria, S.J., Rasalkar, D.D., Paunipagar, B.K.: Imaging Tools for Endometriosis: Role of Ultrasound, MRI and Other Imaging Modalities in Diagnosis and Planning Intervention, chap. 24, pp. 437–447 (2012)
5. Bergmann, R., Gousenbourger, P.Y.: A variational model for data fitting on manifolds by minimizing the acceleration of a Bézier curve. arXiv preprint arXiv:1807.10090 (2018)
6. Boumal, N., Absil, P.A.: A discrete regression method on manifolds and its application to data on SO(n). In: IFAC Proceedings Volumes (IFAC-PapersOnline), vol. 18, pp. 2284–2289 (2011). DOI 10.3182/20110828-6-IT-1002.00542
7. Boumal, N., Mishra, B., Absil, P.A., Sepulchre, R.: Manopt, a Matlab Toolbox for Optimization on Manifolds. Journal of Machine Learning Research **15**, 1455–1459 (2014). URL http://www.manopt.org
8. do Carmo, M.P.: Riemannian Geometry. Mathematics (Birkhäuser) theory. Birkhäuser Boston (1992). DOI 10.1007/978-0-387-29403-2
9. Crouch, P., Kun, G., Silva Leite, F.: De Casteljau algorithm on Lie groups and spheres. Journal of Dynamical and Control Systems **5**(3), 397–429 (1999). DOI 10.1023/A:1021770717822
10. Dyn, N.: Linear and Nonlinear Subdivision Schemes in Geometric Modeling. In: F. Cucker, A. Pinkus, M.J. Todd (eds.) FoCum, *London Math. Soc. Lecture Note Ser.*, vol. 363, pp. 68–92. Cambridge University Press (2009). DOI 10.1017/CBO9781139107068.004
11. Farin, G.E.: Curves and Surfaces for CAGD. Academic Press, fifth edition (2002)
12. Fletcher, P.T.: Geodesic regression and the theory of least squares on Riemannian manifolds. Int. J. Comput. Vis. **105**(2), 171–185 (2013). DOI 10.1007/s11263-012-0591-y

13. Gousenbourger, P.Y., Massart, E., Musolas, A., Absil, P.A., Jacques, L., Hendrickx, J.M., Marzouk, Y.: Piecewise-Bézier C1 smoothing on manifolds with application to wind field estimation. In: ESANN2017, pp. 305–310. Springer (2017)
14. Green, P.J., Silverman, B.W.: Nonparametric regression and generalized linear models: a roughness penalty approach. CRC Press (1993)
15. Hinkle, J., Fletcher, P.T., Joshi, S.: Intrinsic polynomials for regression on Riemannian manifolds. Journal of Mathematical Imaging and Vision **50**(1), 32–52 (2014). DOI 10.1007/s10851-013-0489-5
16. Itoh, J.i., Tanaka, M.: The dimension of a cut locus on a smooth Riemannian manifold. Tohoku Mathematical Journal, Second Series **50**(4), 571–575 (1998)
17. Jupp, P.E., Kent, J.T.: Fitting Smooth Paths to Spherical Data. Journal of Applied Statistics **36**(1), 34–46 (1987). DOI 12/61765
18. Karcher, H.: Riemannian center of mass and mollifier smoothing. Communications on pure and applied mathematics **30**(5), 509–541 (1977)
19. Kim, K.R., Dryden, I.L., Le, H.: Smoothing splines on Riemannian manifolds, with applications to 3D shape space. arXiv preprint arXiv:1801.04978 pp. 1–23 (2018)
20. Lin, A., Walker, M.: CAGD techniques for differentiable manifolds. In: Proceedings of the 2001 International Symposium Algorithms for Approximation IV, pp. 36–43 (2001)
21. Lin, L., St. Thomas, B., Zhu, H., Dunson, D.B.: Extrinsic Local Regression on Manifold-Valued Data. Journal of the American Statistical Association **112**(519), 1261–1273 (2017). DOI 10.1080/01621459.2016.1208615
22. Machado, L., Leite, F.S.: Fitting smooth paths on Riemannian manifolds. Int. J. Appl. Math. Stat. **4**(J06), 25–53 (2006)
23. Machado, L., Monteiro, M.T.T.: A numerical optimization approach to generate smoothing spherical splines. Journal of Geometry and Physics **111**, 71–81 (2017). DOI 10.1016/j.geomphys.2016.10.007
24. Massart, E., Absil, P.A.: Quotient geometry of the manifold of fixed-rank positive-semidefinite matrices. Tech. Rep. UCL-INMA-2018.06, UCLouvain (2018). URL http://sites.uclouvain.be/absil/2018.06
25. Modin, K., Bogfjellmo, G., Verdier, O.: Numerical Algorithm for C2-splines on Symmetric Spaces. arXiv preprint arXiv:1703.09589 (2018)
26. O'Neill, B.: Elementary differential geometry. Academic Press INC, London (1966)
27. Park, J.: Interpolation and tracking of rigid body orientations. In: ICCAS, pp. 668–673 (2010)
28. Pennec, X., Fillard, P., Ayache, N.: A Riemannian framework for tensor computing. International Journal of Computer Vision **66**(1), 41–66 (2006). DOI 10.1007/s11263-005-3222-z
29. Popiel, T., Noakes, L.: Bézier curves and C2 interpolation in Riemannian manifolds. Journal of Approximation Theory **148**(2), 111–127 (2007). DOI 10.1016/j.jat.2007.03.002
30. Pyta, L., Abel, D.: Interpolatory Galerkin Models for Navier-Stokes-Equations. IFAC-PapersOnLine **49**(8), 204–209 (2016). DOI 10.1016/j.ifacol.2016.07.442
31. Rentmeesters, Q.: A gradient method for geodesic data fitting on some symmetric Riemannian manifolds. In: Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on, pp. 7141–7146 (2011). DOI 10.1109/CDC.2011.6161280
32. Samir, C., Absil, P.A., Srivastava, A., Klassen, E.: A Gradient-Descent Method for Curve Fitting on Riemannian Manifolds. Foundations of Computational Mathematics **12**(1), 49–73 (2012). DOI 10.1007/s10208-011-9091-7
33. Shingel, T.: Interpolation in special orthogonal groups. IMA journal of numerical analysis **29**(3), 731–745 (2008)
34. Su, J., Dryden, I.L., Klassen, E., Le, H., Srivastava, A.: Fitting smoothing splines to time-indexed, noisy points on nonlinear manifolds. Image and Vision Computing **30**(6–7), 428–442 (2012). DOI 10.1016/j.imavis.2011.09.006
35. Süli, E., Mayers, D.: An Introduction to Numerical Analysis. Cambridge University Press (2003)
36. Vandereycken, B., Absil, P.A., Vandewalle, S.: Embedded geometry of the set of symmetric positive semidefinite matrices of fixed rank. In: Statistical Signal Processing, 2009. SSP'09. IEEE/SP 15th Workshop on, pp. 389–392. IEEE (2009)
37. Wallner, J., Nava Yazdani, E., Grohs, P.: Smoothness properties of Lie group subdivision schemes. Multiscale Model. Simul. **6**(2), 493–505 (electronic) (2007). DOI 10.1137/060668353