SIAM J. SCI. COMPUT.
Vol. 42, No. 5, pp. A2593–A2619

HERMITE INTERPOLATION AND DATA PROCESSING ERRORS ON RIEMANNIAN MATRIX MANIFOLDS*

RALF ZIMMERMANN†

Abstract. The main contribution of this paper is twofold: On the one hand, a general framework for performing Hermite interpolation on Riemannian manifolds is presented. The method is applicable if algorithms for the associated Riemannian exponential and logarithm mappings are available. This includes many of the matrix manifolds that arise in practical Riemannian computing applications such as data analysis and signal processing, computer vision and image processing, structured matrix optimization problems, and model reduction. On the other hand, we expose a natural relation between data processing errors and the sectional curvature of the manifold in question. This provides general error bounds for manifold data processing methods that rely on Riemannian normal coordinates. Numerical experiments are conducted for the compact Stiefel manifold of rectangular column-orthogonal matrices. As use cases, we compute Hermite interpolation curves for orthogonal matrix factorizations such as the singular value decomposition and the QR-decomposition.

Key words. Hermite interpolation, matrix manifold, Riemannian logarithm, Riemannian exponential, SVD, QR-decomposition

AMS subject classifications. 15A16, 15B10, 33B30, 33F05, 53-04, 65F60

DOI. 10.1137/19M1282878

1. Introduction.

Given a data set that consists of locations $t_0, \ldots, t_k \in \mathbb{R}$, function values $p_0 = f(t_0), \ldots, p_k = f(t_k)$, and derivatives $v_0 = \dot{f}(t_0), \ldots, v_k = \dot{f}(t_k)$, the (first-order) Hermite interpolation problem reads as follows:

Find a polynomial P of suitable degree such that

$$P(t_i) = p_i, \quad \dot{P}(t_i) = v_i, \quad i = 0, \ldots, k. \tag{1}$$

Local cubic Hermite interpolation is the special case of Hermite-interpolating a two-points data set $\{p_i, v_i, p_{i+1}, v_{i+1}\}$ on $t_i, t_{i+1} \in \mathbb{R}$. *Cubic Hermite interpolation* is achieved by joining the local pieces on each subinterval $[t_i, t_{i+1}]$. By construction, the derivative at the end point of $[t_i, t_{i+1}]$ coincides with the derivative of the start point of $[t_{i+1}, t_{i+2}]$ so that the resulting curve is globally C^1 [12, Remark 7.7].

In this paper, we address the Hermite interpolation problem for a function that takes values on a Riemannian manifold $\mathcal{M}$ with tangent bundle $T\mathcal{M}$. More precisely, consider a differentiable function

$$f : [a, b] \to \mathcal{M}, \quad t \mapsto f(t)$$

and a sample plan $a = t_0, \ldots, t_k = b$. Sampling of the function values and the derivatives of f at the parameter instants t_i produces a data set consisting of manifold locations $p_i = f(t_i) \in \mathcal{M}$ and velocity vectors $v_{p_i} = \dot{f}(t_i) \in T_{p_i}\mathcal{M}$ in the respective tangent spaces of $\mathcal{M}$ at p_i. The Hermite manifold interpolation problem is as follows:

Find a curve $c : [a, b] \to \mathcal{M}$ of class C^1 such that

$$c(t_i) = p_i \in \mathcal{M}, \quad \dot{c}(t_i) = v_{p_i} \in T_{p_i}\mathcal{M}, \quad i = 0, \ldots, k. \tag{2}$$

*Submitted to the journal's Methods and Algorithms for Scientific Computing section August 22, 2019; accepted for publication (in revised form) June 4, 2020; published electronically September 8, 2020.

https://doi.org/10.1137/19M1282878

†Department of Mathematics and Computer Science, University of Southern Denmark (SDU) Odense, Odense, DK-5230 Denmark (zimmermann@imada.sdu.dk).

1.1. Original contributions. (1) We introduce a method to tackle problem (2) that is a direct analogue to Hermite interpolation in Euclidean spaces. The method has the following features:

(i) The approach works on arbitrary Riemannian manifolds, i.e., no special structure (Lie group, homogeneous space, symmetric space, etc.) is required.
In order to conduct practical computations, only algorithms for evaluating the Riemannian exponential map and the Riemannian logarithm map must be available.

(ii) Once the curve data is computed, evaluating the Hermite interpolant is of the same computational complexity as quasi-linear, geodesic interpolation, which is arguably the simplest and cheapest of all manifold interpolation methods.

(2) In addition, we expose a natural relation between data processing errors and the sectional curvature of the manifold in question. This provides general error bounds for data processing methods (including but not limited to interpolation) that work via a back-and-forth mapping of data between the manifold and its tangent space, or, more precisely, data processing methods that rely on Riemannian normal coordinates.

For convenience, the exposition will focus on cubic polynomial Hermite interpolation. However, the techniques may be readily combined with any interpolation method that is linear in the sampled locations and derivative values. Apart from polynomial interpolation, this includes radial basis function approaches [3] and gradient-enhanced kriging [34].

As a use case, we provide an explicit and efficient method for the cubic Hermite interpolation of column-orthogonal matrices, which form the so-called Stiefel manifold $St(n,r) = \{U \in \mathbb{R}^{n\times r} | U^T U = I\}$. This allows for the Hermite interpolation of the matrix factors in a parameter-dependent singular value decomposition (SVD) or QR-decomposition. In particular, Hermite interpolation of a truncated low-rank SVD is enabled.

1.2. Related work. Interpolation problems with manifold-valued sample data and spline-related approaches have triggered an extensive amount of research work.

It is well known that cubic splines in Euclidean spaces are acceleration-minimizing. This property allows for a generalization to Riemannian manifolds in the form of a variational problem for the intrinsic, covariant acceleration of curves, whose solutions can be interpreted as generalized cubic polynomials on Riemannian manifolds. The variational approach to interpolation on manifolds has been investigated, e.g., in [26, 11, 10, 32, 8, 29, 20]; see also [27] and references therein. While the property of minimal mean-acceleration is certainly desirable in many contexts, including automobile, aircraft, and ship designs and digital animations, there is no conceptual reason to impose this condition when interpolating general smooth nonlinear manifold-valued functions.

A related line of research is the generalization of Bézier curves and the De Casteljau algorithm [5] to Riemannian manifolds [27, 21, 25, 1, 15, 30]. Bézier curves in Euclidean spaces are polynomial splines that rely on a number of so-called control points. A Bézier curve starts at the first control point and ends at the last control point; the starting velocity is tangent to the line between the first two pairs of control points; the velocity at the end point is tangent to the line between the penultimate and the last control point. This is illustrated in Figure 1. The number of control points determines the degree of the polynomial spline. To obtain the value $B(t)$ of a Bézier curve at time t, a recursive sequence of straight-line convex combinations of two locations must be computed. The transition of this technique to Riemannian

manifolds is via replacing the inherent straight lines with geodesics [27]. The start and end velocities of the resulting spline are proportional to the velocity vectors of the geodesics that connect the first two and the last two control points, respectively [27, Theorem 1].

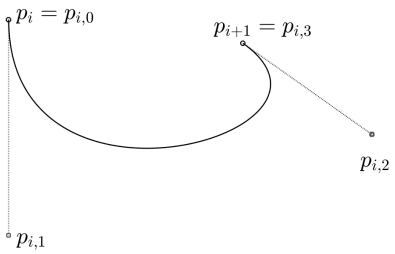

FIG. 1. *A cubic Bézier curve based on four control points $p_{i,0}, p_{i,1}, p_{i,2}, p_{i,3}$. The "inner" control points $p_{i,1}, p_{i,2}$ may be used to prescribe tangent directions at $p_i = p_{i,0}$ and $p_{i+1} = p_{i,3}$, which are interpolated.*

Note that the actual applications and use cases featured in the work referenced above are almost exclusively on low-dimensional matrix manifolds like $S^2, S^3, SO(3)$, and $SE(3)$.

A Hermite-type method that is specifically tailored for interpolation problems on the Grassmann manifold is sketched in [3, section 3.7.4]. Hermitian interpolation has been considered explicitly in [18] for data on compact, connected Lie groups with a bi-invariant metric. The idea is as follows: Given two points $p, q \in \mathcal{M}$ on a manifold and two tangent directions $v_p \in T_p\mathcal{M}, v_q \in T_q\mathcal{M}$, the authors of [18] approach the task to construct a connecting curve $c : [t_i, t_{i+1}] \to \mathcal{M}$ such that $c(t_i) = p, \dot{c}(t_i) = v_p, c(t_{i+1}) = q, \dot{c}(t_{i+1}) = v_q$ by constructing a "left" arc l_i that starts at $t = t_i$ from p with the prescribed velocity v_p and a "right" arc r_i that ends at $t = t_{i+1}$ at q with the prescribed velocity v_q. The two arcs are then blended to a single spline arc via a certain geometric convex combination. In Euclidean spaces, this would read $s(t) = (1 - \Phi(t))l_i(t) + \Phi(t)r_i(t)$, where Φ is a suitable weight function. Because a general Riemannian manifold lacks a vector space structure, [18] develops a manifold analogue of convex combinations, which works for derivative data up to arbitrary order but requires the algebraic structure of compact, connected Lie groups.

This same idea of blending a left arc and a right arc has been followed up in [15]. Here, the Euclidean convex combination is replaced with a geodesics average $s(t) = \operatorname{Exp}_{l_i(t)}(\Phi(t) \operatorname{Log}_{l_i(t)}(r_i(t)))$. In combination, this would constitute a valid approach for solving (2) in arbitrary Riemannian manifolds. In practice, the building arcs $l_i(t)$ and $r_i(t)$ may be taken to be the geodesics with the prescribed velocities in their respective start and end points.

It should be mentioned that none of the papers on Bézier curves referenced above tackles the Hermite interpolation problem explicitly. However, the Bézier approach can be turned into a Hermite method by choosing the control points such that the sampled start and terminal velocities are met. It is clear that this requires at least 4 control points in each subinterval $[t_i, t_{i+1}]$; see Figure 1.

Interpolation problems on Stiefel manifolds have been considered in [21]—however using quasi-geodesics rather than geodesics. The work [37] includes preliminary numerical experiments for interpolating orthogonal frames on the Stiefel manifold that relies on the canonical Riemannian Stiefel logarithm [28, 35].

To the best of the author's knowledge, numerical Hermite interpolation on arbitrary Riemannian manifolds has not yet been considered in the research literature, and neither has the specific Hermite interpolation of QR or SVD matrix factors.

Remark. (Hermite) interpolation of curves on Riemannian manifolds, i.e., of manifold-valued functions $f : [a, b] \to \mathcal{M}$, must not be confused with (Hermite) interpolation of real-valued functions with domain of definition on a manifold, $f : \mathcal{M} \to \mathbb{R}$. The latter line of research is pursued, e.g., in [24], but is not considered here.

1.3. Organization. The paper is organized as follows: Starting from the classical Euclidean case, section 2 introduces an elementary approach to Hermite interpolation on general Riemannian manifolds. Section 3 relates the data processing errors of calculations in Riemannian normal coordinates to the curvature of the manifold in question. In section 4, the specifics of performing Hermite interpolation of column-orthogonal matrices are discussed, and section 5 illustrates the theory by means of numerical examples. Conclusions are given in section 6.

1.4. Notational specifics. The $(r \times r)$-*identity matrix* is denoted by $I_r \in \mathbb{R}^{r\times r}$, or simply I if the dimensions are clear. The $(r \times r)$-*orthogonal group* is denoted by

$$O(r) = \{\Phi \in \mathbb{R}^{r\times r} | \Phi^T\Phi = \Phi\Phi^T = I_r\}.$$

Throughout, the QR-decomposition $A = QR$ of $A \in \mathbb{R}^{n\times r}$, $n \geq r$, is understood as the "economy size" QR-decomposition with $Q \in \mathbb{R}^{n\times r}$, $R \in \mathbb{R}^{r\times r}$.

The standard matrix exponential and the principal matrix logarithm are given by

$$\exp_m(X) := \sum_{j=0}^{\infty} \frac{X^j}{j!}, \quad \log_m(I + X) := \sum_{j=1}^{\infty} (-1)^{j+1} \frac{X^j}{j}.$$

The latter is well defined for matrices that have no eigenvalues on $\mathbb{R}^-$.

For a Riemannian manifold $\mathcal{M}$, the geodesic that starts from $p \in \mathcal{M}$ with velocity $v \in T_p\mathcal{M}$ is denoted by $t \mapsto c_{p,v}(t)$. The Riemannian exponential function at p is

$$\text{Exp}_p : T_p\mathcal{M} \supset D_0 \to \mathcal{D}_p \subset \mathcal{M}, \quad v \mapsto \text{Exp}_p(v) := c_{p,v}(1) \tag{3}$$

and is visualized in Figure 2. It maps a star-shaped domain D_0 around the origin in $T_p\mathcal{M}$ diffeomorphically to a domain $\mathcal{D}_p \subset \mathcal{M}$. The size of D_0 depends on the *injectivity radius* $r_{\mathcal{M}}(p)$ of $\mathcal{M}$ at p. The injectivity radius at p is the Riemannian distance of p to its cut locus C_p. The cut locus, in turn, is the set of points beyond which the geodesics starting at p cease to be length-minimizing. Finally, the global injectivity radius is $r_{\mathcal{M}} := \inf\{r_{\mathcal{M}}(p) | \, p \in \mathcal{M}\} = \inf_{p\in\mathcal{M}} \text{dist}(p, C_p)$; see [13, p. 271].

The Riemannian logarithm at p is

$$\text{Log}_p : \mathcal{M} \supset \mathcal{D}_p \to D_0 \subset T_p\mathcal{M}, \quad q \mapsto v = (\text{Exp}_p)^{-1}(q). \tag{4}$$

The Riemannian exp and log maps for some of the most prominent matrix manifolds are collected, e.g., in [36].

For a differentiable function $f : \mathcal{M} \to \mathcal{N}$, the differential at p is a linear map between the tangent spaces

$$df_p : T_p\mathcal{M} \to T_{f(p)}\mathcal{N}. \tag{5}$$

The chain rule extends to the manifold setting. For $f : \mathcal{M} \to \mathcal{N}$ and $g : \mathcal{N} \to \mathcal{P}$,

$$d(g \circ f)_p : T_p\mathcal{M} \to T_{g(f(p))}\mathcal{P}, \quad d(g \circ f)_p(v) = dg_{f(p)}(df_p(v)).$$

The differentials of the exponential and logarithm mappings read

$$d(\mathrm{Exp}_p)_v : \ T_v(T_p\mathcal{M}) \cong T_p\mathcal{M} \ \to \ T_{\mathrm{Exp}_p(v)}\mathcal{M}, \tag{6}$$

$$d(\mathrm{Log}_q)_p : \qquad T_p\mathcal{M} \qquad \to \ T_{\mathrm{Log}_q(p)}(T_q\mathcal{M}) \cong T_q\mathcal{M}, \tag{7}$$

where $\cong$ indicates the usual identification of a linear space with its tangent space. For $\mathbb{R}^n$ this is the familiar relation that $T_x\mathbb{R}^n \cong \mathbb{R}^n$.

2. Hermite interpolation on Riemannian manifolds. In this section, we construct a quasi-cubic spline between two data points $p, q \in \mathcal{M}$ on a manifold with prescribed velocities $v_p \in T_p\mathcal{M}$ and $v_q \in T_q\mathcal{M}$. To this end, we develop a manifold analogue to the classical local cubic Hermite interpolation in Euclidean spaces [12, section 7].

2.1. The Euclidean case. We start with a short recap of Hermite cubic space curve interpolation, where the following setting is of special interest to our considerations. Let $\mathbb{V}$ be a real vector space and let $f : [t_0, t_1] \to \mathbb{V}$ be differentiable with $f(t_0) = p \in \mathbb{V}$, $f(t_1) = q \in \mathbb{V}$, and derivative data $v_p = \dot{f}(t_0), v_q = \dot{f}(t_1)$. When applied to vector-valued functions, the classical local cubic Hermite interpolating spline is the space curve $c(t)$ that is obtained via a linear combination of the sampled data,

$$c(t) = a_0(t)p + a_1(t)q + b_0(t)v_p + b_1(t)v_q. \tag{8}$$

For the reader's convenience, the basic cubic Hermite coefficient polynomials $a_0(t)$, $a_1(t)$, $b_0(t), b_1(t)$ are listed in section S1 of the supplement. It is an elementary yet often overlooked fact that for functions $t \mapsto f(t) \in \mathbb{R}^n$, componentwise polynomial interpolation of the coordinate functions $t \mapsto f_j(t) \in \mathbb{R}$, $j = 1, \ldots, n$, is equivalent to interpolating the coefficients in a linear combination of the sampled data vectors.

2.2. Transfer to the manifold setting. Let $\mathcal{M}$ be a Riemannian manifold and consider a differentiable function

$$f : [t_0, t_1] \to \mathcal{M}, \quad t \mapsto f(t).$$

Suppose that $f(t_0) = p, f(t_1) = q \in \mathcal{M}$ and $\dot{f}(t_0) = v_p \in T_p\mathcal{M}, \dot{f}(t_1) = v_q \in T_q\mathcal{M}$. By default, but for no mathematical reasons, we use q as the center for the Riemannian normal coordinates. Assume further that $\mathrm{dist}(p, q) < r_{\mathcal{M}}(q)$, where $r_{\mathcal{M}}(q)$ is the injectivity radius of $\mathcal{M}$ at q. The latter condition ensures that the sample data lies within a domain, where the Riemannian exponential is a diffeomorphism; see subsection 1.4. Our approach is to express the interpolating curve in terms of normal coordinates centered at $q = f(t_1) \in \mathcal{M}$,

$$c : [t_0, t_1] \to \mathcal{M}, \quad c(t) = \mathrm{Exp}_q(\gamma(t)).$$

Hence, the task is transferred to constructing a curve $t \mapsto \gamma(t) \subset T_q\mathcal{M}$ such that the image curve c under the exponential function solves the Hermite interpolation problem (2). Because $T_q\mathcal{M}$ is a vector space, we can utilize the ansatz of (8) but for $\mathbb{V} = T_q\mathcal{M}$,

$$\gamma(t) = a_0(t)\Delta_p + a_1(t)\Delta_q + b_0(t)\hat{v}_p + b_1(t)\hat{v}_q.$$

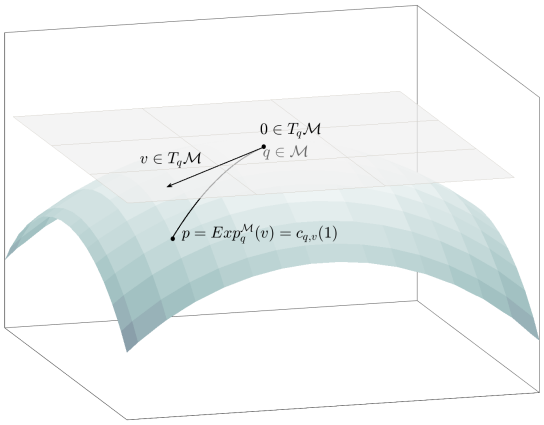

FIG. 2. *Visualization of the Riemannian exponential map: The tangent velocity* $\Delta \in T_q\mathcal{M}$ *is mapped to the end point of the geodesic* $p = c_{q,\Delta}(1) \in \mathcal{M}$. *The Riemannian distance* $\mathrm{dist}(q,p)$ *equals the norm* $\|\Delta\|$ *as measured by the Riemannian metric on* $T_q\mathcal{M}$.

Here, $\Delta_p = \mathrm{Log}_q(p), \Delta_q = \mathrm{Log}_q(q) = 0 \in T_q\mathcal{M}$ are the normal coordinate images of the locations p and q. The tangent vectors $\hat{v}_p, \hat{v}_q \in T_q\mathcal{M}$ play the role of the velocity vectors and must be chosen such that

$$\dot{c}(t_0) = \frac{d}{dt}\big|_{t=t_0} \mathrm{Exp}_q(\gamma(t)) \stackrel{!}{=} v_p = \dot{f}(t_0), \tag{9}$$

$$\dot{c}(t_1) = \frac{d}{dt}\big|_{t=t_1} \mathrm{Exp}_q(\gamma(t)) \stackrel{!}{=} v_q = \dot{f}(t_1). \tag{10}$$

Since the interpolating curve $c(t) = \mathrm{Exp}_q(\gamma(t))$ is expressed in normal coordinates centered at $q = f(t_1)$, condition (10) is readily fulfilled by selecting $\hat{v}_q = v_q$: According to the properties of the cubic Hermite coefficient functions $a_0(t), b_0(t), b_1(t)$, the Taylor expansion of $\gamma(t)$ around t_1 is $\gamma(t_1 + h) = h\hat{v}_q + \mathcal{O}(h^2)$. Therefore, up to first order, $\gamma(t)$ is a ray *emerging from the origin* $0 \in T_q\mathcal{M}$ with direction $\hat{v}_q \in T_q\mathcal{M}$. Hence, the directional derivative of the exponential function is

$$\frac{d}{dt}\big|_{t=t_1} \mathrm{Exp}_q(\gamma(t)) = \frac{d}{dt}\big|_{h=0} \mathrm{Exp}_q(h\hat{v}_q + \mathcal{O}(h^2)) = d(\mathrm{Exp}_q)_0(\hat{v}_q) = \hat{v}_q = v_q.$$

The latter equation holds because $d(\mathrm{Exp}_q)_0 = \mathrm{id}_{T_q\mathcal{M}}$ [13, section 3, Prop. 2.9].

The condition (9) is more challenging because the Taylor expansion of $\gamma(t)$ around t_0 is $\gamma(t_0 + h) = \Delta_p + h\hat{v}_p + \mathcal{O}(h^2)$ and is *not* a ray emerging from the origin $0 \in T_q\mathcal{M}$. As the differential $d(\mathrm{Exp}_q)_v$ is *not* the identity for $v \neq 0$, the computation of $\frac{d}{dt}\big|_{t=t_1} \mathrm{Exp}_q(\Delta_p + h\hat{v}_p + \mathcal{O}(h^2)) = d(\mathrm{Exp}_q)_{\Delta_p}(\hat{v}_p)$ is more involved. In fact, it is related to the Jacobi fields on a Riemannian manifold; see [13, section 5], [22, section 10], and the upcoming section 3. Yet, for our purposes, it is sufficient to determine the tangent vector $\hat{v}_p$ such that

$$v_p = d(\mathrm{Exp}_q)_{\Delta_p}(\hat{v}_p). \tag{11}$$

As long as the sample points $p = \mathrm{Exp}_q(\Delta_p)$ and q are not conjugate, we can make use of the fact that Exp_q is a local diffeomorphism around Δ_p [22, Prop. 10.11].

Hence, under this assumption, (11) is equivalent to

$$\begin{aligned} d(\mathrm{Log}_q)_p(v_p) &= \left(d(\mathrm{Log}_q)_p \circ d(\mathrm{Exp}_q)_{\Delta_p}\right)(\hat{v}_p) \\ &= \frac{d}{dt}\big|_{t=t_0} \left(\mathrm{Log}_q \circ \mathrm{Exp}_q\right)(\Delta_p + h\hat{v}_p) \\ &= \frac{d}{dt}\big|_{t=t_0} \mathrm{id}_{T_q\mathcal{M}}(\Delta_p + h\hat{v}_p) = \hat{v}_p. \end{aligned}$$

Recall that $v_p = \dot{f}(t_0)$ is the given sample data. In summary, we have proved the following theorem.

THEOREM 1. *Let $t_0 < t_1 \in \mathbb{R}$ and let $f : [t_0, t_1] \to \mathcal{M}$ be a differentiable function on a Riemannian manifold $\mathcal{M}$. Suppose that*

$$f(t_0) = p,\ f(t_1) = q \in \mathcal{M}, \quad \dot{f}(t_0) = v_p \in T_p\mathcal{M},\ \dot{f}(t_1) = v_q \in T_q\mathcal{M}$$

and assume that p and q are not conjugate along the geodesic $t \mapsto \mathrm{Exp}_q(t\,\mathrm{Log}_q(p))$ that connects p and q. Set $\Delta_p = \mathrm{Log}_q(p)$ and

$$\hat{v}_p = d(\mathrm{Log}_q)_p(v_p) \in T_q\mathcal{M}, \quad \hat{v}_q = v_q \in T_q\mathcal{M}.$$

Then

$$c : [t_0, t_1] \to \mathcal{M}, \quad t \mapsto \mathrm{Exp}_q\big(a_0(t)\Delta_p + b_0(t)\hat{v}_p + b_1(t)\hat{v}_q\big) \tag{12}$$

with the classical cubic coefficient functions $a_0(t), b_0(t), b_1(t)$ as stated in the supplement in (S1)–(S4) *Hermite-interpolates the given sample data.*

Remark 2. Consider a Hermite sample set $p_i = f(t_i) \in \mathcal{M}$, $v_{p_i} = \dot{f}(t_i) \in T_{p_i}\mathcal{M}$, $i = 0, \ldots, k$. Then, by construction and in complete analogy to the Euclidean case, the composite curve

$$C : [t_0, t_k] \to \mathcal{M}, \quad t \mapsto c_{[t_i, t_{i+1}]}(t) \text{ for } t \in [t_i, t_{i+1}], \tag{13}$$

that combines the local quasi-cubic spline arcs $c = c_{[t_i, t_{i+1}]}$ of Theorem 1 is of class C^1 and solves the Hermite manifold interpolation problem (2).

Practical computation of $\hat{v}_p$. In cases where an explicit formula for the Riemannian logarithm is at hand, the directional derivative $\hat{v}_p = d(\mathrm{Log}_q)_p(v_p)$ can be directly computed. For general nonlinear manifolds $\mathcal{M}$, computing the differentials of the Riemannian exponential and logarithm is rather involved. According to (5), (6), (7), it holds that

$$d(\mathrm{Log}_q)_p : T_p\mathcal{M} \ \to\ T_q\mathcal{M}.$$

In order to evaluate $d(\mathrm{Log}_q)_p(v_p)$, we can take any differentiable curve $\tilde{c}(s) \subset \mathcal{M}$ that satisfies $\tilde{c}(0) = p$ and $\dot{\tilde{c}}(0) = v_p$. Then,

$$d(\mathrm{Log}_q)_p(v_p) = d(\mathrm{Log}_q)_{\tilde{c}(0)}(\dot{\tilde{c}}(0)) = \frac{d}{ds}\big|_{s=0} \mathrm{Log}_q\left(\tilde{c}(s)\right). \tag{14}$$

An obvious choice is $\tilde{c}(s) = \mathrm{Exp}_p(sv_p) \subset \mathcal{M}$. The final equation for computing $\hat{v}_p$ as required by (9) is

$$\hat{v}_p = \frac{d}{ds}\big|_{s=0} \left(\mathrm{Log}_q \circ \mathrm{Exp}_p\right)(sv_p). \tag{15}$$

Due to the different base points, this composition of Log_q and Exp_p is not the identity. Rather, the composite map $\mathrm{Log}_q \circ \mathrm{Exp}_p : T_p\mathcal{M} \supset D_0 \to T_q\mathcal{M}$ is a transition function for the normal coordinate charts. It is defined on an open subset of a Hilbert space and maps to a Hilbert space; see [23, Fig. 1.6, p. 12] for an illustration. Hence, we can approximate the directional derivative $\hat{v}_p = \frac{d}{ds}\big|_{s=0} \left(\mathrm{Log}_q \circ \mathrm{Exp}_p\right)(sv_p)$ via finite difference approaches:

$$\hat{v}_p = \frac{\left(\mathrm{Log}_q \circ \mathrm{Exp}_p\right)(hv_p) - \left(\mathrm{Log}_q \circ \mathrm{Exp}_p\right)(-hv_p)}{2h} + \mathcal{O}(h^2). \tag{16}$$

2.3. Computational effort and preliminary comparison to other methods. Computationally, the most involved numerical operations are the evaluations of the Riemannian Log- and Exp-mappings. Therefore, as in [15], we measure the computational effort associated with the Hermite interpolation method as the number of such function evaluations.

Constructing a quasi-cubic Hermite interpolant as in Remark 2 requires on each subinterval $[t_i, t_{i+1}]$

- one Riemannian logarithm to compute $\Delta_p = \mathrm{Log}_q(p)$,
- two Riemannian Log- and Exp-evaluations for the central difference approximation of (16),

which results in a total of $3k$ Riemannian Log-evaluations and $2k$ Riemannian Exp-evaluations for the whole composite curve. The data to represent the curve (13) can be precomputed and stored.

Evaluating a quasi-cubic Hermite interpolant at time t requires *a single* Riemannian Exp-evaluation.

As mentioned in the introduction, Bézier-like approaches may be used to tackle the Hermite interpolation problem (2). This requires a cubic degree and at least four control points on each subinterval $[t_i, t_{i+1}]$ to impose the derivative constraints; see Figure 1. The most efficient of such methods in [15] requires $\mathcal{O}(k^2)$ Riemannian Log-evaluations for constructing the curve data. Evaluating the curve at time t requires 3 Riemannian Exp-evaluations plus 1 Riemannian Log-evaluation [15, Prop. 5.10].

3. Error propagation. The approach introduced in subsection 2.2 follows the standard principle of (1) mapping the sampled data onto the tangent space, (2) performing data processing (in this case, interpolation) in the tangent space, and (3) mapping the result back to the curved manifold. In this section, we perform a general qualitative analysis of the behavior of the actual errors on the manifold in question in relation to the data processing errors that accumulate in the tangent space. In particular, this allows us to obtain error estimates for any manifold interpolation procedure based on the above standard principle and also applies to other data processing operations that subordinate to this pattern. In essence, the error propagation is related to the manifold's curvature via a standard result from differential geometry on the spreading of geodesics [13, Chapter 5, section 2].

THEOREM 3. *Let $\mathcal{M}$ be a Riemannian manifold, let $q \in \mathcal{M}$, and consider tangent vectors $\Delta, \tilde{\Delta} \in T_q\mathcal{M}$, which are to be interpreted as exact data and associated approximation with error $E := \Delta - \tilde{\Delta}$. Write $\delta = \|\Delta\|$, $\tilde{\delta} = \|\tilde{\Delta}\|$, where it is understood that the norm is that of $T_q\mathcal{M}$. Assume that $\delta, \tilde{\delta} < 1$. Let $\sigma = span(\Delta, \tilde{\Delta}) \subset T_q\mathcal{M}$ and let $K_q(\sigma)$ be the sectional curvature at q with respect to the 2-plane σ.*

If $s_0 = \angle(\tilde{\Delta}, \Delta)$ is the angle between $\tilde{\Delta}$ and Δ, then the Riemannian distance

between the manifold locations $p = \mathrm{Exp}_q(\Delta)$ *and* $\tilde{p} = \mathrm{Exp}_q(\tilde{\Delta})$ *is*

$$\mathrm{dist}_{\mathcal{M}}(p, \tilde{p}) \leq |\delta - \tilde{\delta}| + s_0 \delta \left(1 - \frac{K_q(\sigma)}{6}\delta^2 + o(\delta^2)\right) + \mathcal{O}(s_0^2), \tag{17}$$

with the underlying assumption that all data are within the injectivity radius at q.

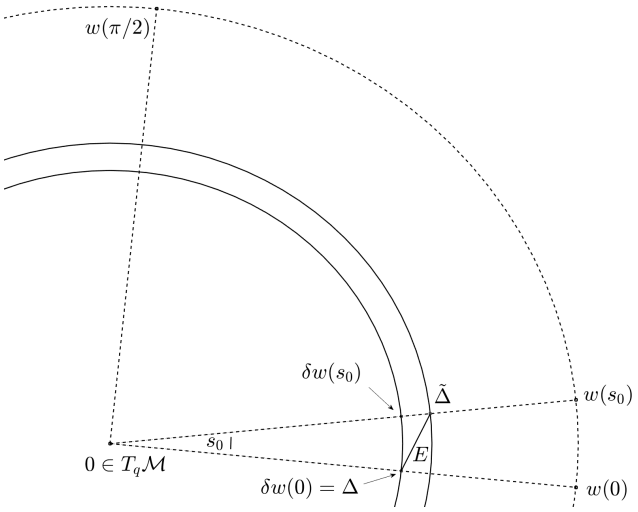

FIG. 3. *The geometric setting in the proof of Theorem* 3. *The length of the gap vector* E *is estimated against the length of the arc that connects* $\delta w(0)$ *and* $\delta w(s_0)$ *plus the difference of the radii* $\delta = \|\Delta\|$ *and* $\tilde{\delta} = \|\tilde{\Delta}\|$.

Proof. Let $q \in \mathcal{M}$. Consider tangent vectors $\Delta, \tilde{\Delta} \in T_q\mathcal{M}$, where $\tilde{\Delta}$ is thought of as an approximation of Δ that is, for example, obtained from an interpolation procedure in $T_q\mathcal{M}$. The approximation error in the tangent space is $\|E\|_{T_q\mathcal{M}} = \|\Delta - \tilde{\Delta}\|_{T_q\mathcal{M}}$. The manifold locations associated with Δ and $\tilde{\Delta}$ are p and $\tilde{p}$, respectively. Formally, it holds that $\mathrm{dist}_{\mathcal{M}}(p, \tilde{p}) = \|\log_p(\tilde{p})\|_{T_p\mathcal{M}}$. However, the data are given in normal coordinates around $q \in \mathcal{M}$ and not around p (or $\tilde{p}$). The Riemannian exponential is a radial isometry (lengths of rays starting from the origin of the tangent space equal the lengths of the corresponding geodesics). Yet, it is not an isometry so that $\mathrm{dist}_{\mathcal{M}}(p, \tilde{p}) \neq \|E\|_{T_q\mathcal{M}}$, unless $\mathcal{M}$ is flat. Therefore, we will estimate the distance $\mathrm{dist}_{\mathcal{M}}(p, \tilde{p})$ against a component that corresponds to the length of a ray in $T_q\mathcal{M}$ and a circular segment in $T_q\mathcal{M}$. To this end, we introduce an orthonormal basis $w, w^{\perp}$ for the plane $\sigma = \mathrm{span}(\Delta, \tilde{\Delta}) \subset T_q\mathcal{M}$ via

$$w := \frac{\Delta}{\|\Delta\|_{T_q\mathcal{M}}}, \quad w^{\perp} = \frac{\tilde{\Delta} - \langle w, \tilde{\Delta}\rangle w}{\|\tilde{\Delta} - \langle w, \tilde{\Delta}\rangle w\|_{T_q\mathcal{M}}}.$$

The circular segment $w(s)$ of the unit circle in the σ-plane that starts from $w = w(0)$ and ends in $w^{\perp} = w(\pi/2)$ can be parameterized via the curve

$$w : [0, \pi/2] \to T_q\mathcal{M}, \quad s \mapsto w(s) = \cos(s)w + \sin(s)w^{\perp}.$$

Let $s_0 \in [0, \pi/2]$ be the angle such that $\tilde{\Delta} = \tilde{\delta} w(s_0)$. This setup is illustrated in Figure 3, where the outer dashed circular arc indicates the unit circle and the solid

circular arcs are the circles of radii δ and $\tilde{\delta}$, respectively. By the triangle inequality,

$$\text{dist}_{\mathcal{M}}(p,\tilde{p}) = \text{dist}_{\mathcal{M}}\left(\text{Exp}_q(\delta w(0)), \text{Exp}_q(\tilde{\delta} w(s_0))\right) \tag{18}$$

$$\leq \text{dist}_{\mathcal{M}}\left(\text{Exp}_q(\delta w(0)), \text{Exp}_q(\delta w(s_0))\right) \tag{19}$$

$$+ \text{dist}_{\mathcal{M}}\left(\text{Exp}_q(\delta w(s_0)), \text{Exp}_q(\tilde{\delta} w(s_0))\right). \tag{20}$$

Since the points $\delta w(s_0)$ and $\tilde{\delta} w(s_0) = \tilde{\Delta}$ are on a ray that emerges from the origin in $T_q\mathcal{M}$, the distance term in line (20) is exactly $|\delta - \tilde{\delta}|$; see Figure 3. Note that $\text{Exp}_q(\delta w(0)) = \text{Exp}_q(\Delta) = p$. Hence, the distance term in line (19) is

$$\text{dist}_{\mathcal{M}}\left(p, \text{Exp}_q(\delta w(s_0))\right) = \| \text{Log}_p(\text{Exp}_q(\delta w(s_0)))\|_{T_p\mathcal{M}}.$$

A Taylor expansion centered at $s = 0$ of the transition function along the circular segment $s \mapsto \left(\text{Log}_p \circ \text{Exp}_q\right)(\delta w(s))$ gives

$$\begin{aligned}
\text{Log}_p(\text{Exp}_q(\delta w(s_0))) &= \text{Log}_p(\text{Exp}_q(\delta w(0))) \\
&\quad + s_0 \frac{d}{ds}\Big|_{s=0} \left(\text{Log}_p \circ \text{Exp}_q\right)(\delta w(s)) + \mathcal{O}(s_0^2) \\
&= \text{Log}_p(p) + s_0 d(\text{Log}_p)_p \circ d(\text{Exp}_q)_\Delta(\delta \dot{w}(0)) + \mathcal{O}(s_0^2) \\
&= 0 + s_0 d(\text{Exp}_q)_{\delta w}(\delta w^\perp) + \mathcal{O}(s_0^2).
\end{aligned}$$

To arrive at the last line, $d(\text{Log}_p)_p = \text{id}_{T_p\mathcal{M}}$ was used, which follows from the standard result $d(\text{Exp}_p)_0 = \text{id}_{T_p\mathcal{M}}$ [13, section 3, Prop. 2.9, p. 65] with the usual identification of $T_p\mathcal{M} \cong T_p(T_p\mathcal{M})$; cf. (7), (6).

By construction, $\delta \mapsto d(\text{Exp}_q)_{\delta w}(\delta w^\perp)$ is a Jacobi field along the geodesic ray that starts from $q = \text{Exp}_q(0)$ with unit velocity $w \in T_q\mathcal{M}$; see [13, section 5, Prop. 2.7, p. 114]. Moreover, $w = w(0), w^\perp = w(\pi/2)$ constitute an orthonormal basis of the σ-plane in $T_q\mathcal{M}$. Therefore, the results [13, section 5, Cor. 2.9, Cor. 2.10, p. 115] apply and give

$$\|d(\text{Exp}_q)_{\delta w}(\delta w^\perp)\|_{T_p\mathcal{M}} = \delta - \frac{K_q(\sigma)}{6}\delta^3 + o(\delta^3).$$

In summary,

$$\text{dist}_{\mathcal{M}}(p,\tilde{p}) \leq |\delta - \tilde{\delta}| + \delta s_0 \left(1 - \frac{K_q(\sigma)}{6}\delta^2 + o(\delta^2)\right) + \mathcal{O}(s_0^2),$$

which establishes the theorem. ∎

Remark 4.

(i) The approximation error in the tangent space $\epsilon := \|E\| = \|\tilde{\Delta} - \Delta\|$ can be related by elementary trigonometry to the angle $s_0 = \angle(\tilde{\Delta}, \Delta)$: The point $\tilde{\Delta}$ lies on the circle $D_\epsilon(\Delta)$ of radius ϵ around Δ. Assume that $\epsilon < \|\Delta\| = \delta$. (This corresponds to assuming that the relative error between $\tilde{\Delta}$ and Δ is below 100%.) The angle between Δ and any point on $D_\epsilon(\Delta)$ is bounded by the angle between Δ and a line that starts at the origin and is tangent to the boundary of the circle, the value of which is $\arcsin(\frac{\epsilon}{\delta})$. Thus, for fixed δ and $\epsilon \to 0$,

$$\delta s_0 \leq \delta \arcsin\left(\frac{\epsilon}{\delta}\right) = \epsilon + \mathcal{O}\left(\epsilon^3\right).$$

In terms of the error ϵ, the distance estimate (17) reads

$$\text{dist}_{\mathcal{M}}(p,\tilde{p}) \le |\delta-\tilde{\delta}| + \epsilon\left(1-\frac{K_q(\sigma)}{6}\delta^2+o(\delta^2)\right)+\mathcal{O}(\epsilon^2). \tag{21}$$

(ii) If we travel from Δ to $\tilde{\Delta}$ in the tangent space on the corresponding curves as in the proof of Theorem 3, i.e., first along the circular arc from $\Delta=\delta w(0)$ to $\delta w(s_0)$ and then along the ray from $\delta w(s_0)$ to $\tilde{\delta}w(s_0)=\tilde{\Delta}$, then we cover precisely a distance of $|\delta-\tilde{\delta}|+\delta s_0$. Comparing this with the right-hand side of (17), we see that the corresponding distances of the manifold images are asymptotically shorter (longer) if $\mathcal{M}$ features positive (negative) sectional curvatures. The underlying principle is the well-known effect that geodesics on positively curved spaces spread apart less than straight rays in a flat space, while they spread apart more on negatively curved spaces; see [13, section 5, Rem. 2.11, pp. 115–116]. From the numerical point of view, this means that data processing operations that work in the tangent space followed by a transition to the manifold are rather well behaved on manifolds of positive curvature, while the opposite holds on negatively curved manifolds. In subsection 5.3, we will show an illustration of Theorem 3 on an interpolation problem on the compact Stiefel manifold; see in particular Figure 6.

With the help of Theorem 3, explicit error bounds for manifold interpolation methods can be obtained. For example, cubic Hermite interpolation comes with a standard error bound [12, Thm. 7.16] that applies to the interpolant in the tangent space. This can be forwarded to a manifold error via Theorem 3.

4. Cubic Hermite interpolation of column-orthogonal matrices. In the numerical experiments of the next section, we will consider Hermite interpolation problems, where the sampled data are column-orthogonal matrices that stem from SVDs or QR-decompositions of given input matrices. The set of column-orthogonal matrices

$$St(n,r) := \{U\in\mathbb{R}^{n\times r} |\ U^TU = I_r\}$$

is called the (compact) *Stiefel manifold*. The next subsections review the essential aspects of the numerical treatment of Hermite interpolation problems on Stiefel manifolds. For more information on Stiefel manifolds, see [2, 14, 36].

4.1. Geometric essentials of Stiefel manifolds. The Stiefel manifold is a compact homogeneous matrix manifold. The *tangent space* $T_USt(n,r)$ at a point $U\in St(n,r)$ can be thought of as the space of velocity vectors of differentiable curves on $St(n,r)$ passing through U:

$$T_USt(n,r) = \{\dot{c}(t_0) |\ c:(t_0-\epsilon,t_0+\epsilon)\to St(n,r), c(t_0)=U\}.$$

For any matrix representative $U\in St(n,r)$, the tangent space of $St(n,r)$ at U is

$$T_USt(n,r) = \left\{\Delta\in\mathbb{R}^{n\times r} |\ U^T\Delta = -\Delta^TU\right\}\subset\mathbb{R}^{n\times r}.$$

Every tangent vector $\Delta\in T_USt(n,r)$ may be written as

$$\Delta = UA+(I-UU^T)T,\quad A\in\mathbb{R}^{r\times r}\text{ skew},\quad T\in\mathbb{R}^{n\times r}\text{ arbitrary}. \tag{22}$$

The dimension of both $T_USt(n,r)$ and $St(n,r)$ is $nr-\frac{1}{2}r(r+1)$.

Each tangent space carries an inner product $\langle \Delta, \tilde{\Delta} \rangle_U = tr(\Delta^T (I - \frac{1}{2} UU^T) \tilde{\Delta})$ with corresponding norm $\|\Delta\|_U = \sqrt{\langle \Delta, \Delta \rangle_U}$. This is called the *canonical metric* on $T_U St(n,r)$. It is derived from the quotient space representation $St(n,r) = O(n)/O(n-r)$ that identifies two square orthogonal matrices in $O(n)$ as the same point on $St(n,r)$ if their first r columns coincide [14, section 2.4]. For a condensed introduction to quotient spaces, see [36, section 2.5]. Endowing each tangent space with this metric (that varies differentiably in U) turns $St(n,r)$ into a *Riemannian manifold*. The associated sectional curvature is nonnegative and is bounded by $0 \leq K_U(\sigma) \leq \frac{5}{4}$ for all $U \in St(n,r)$ and all two-plans $\sigma = \text{span}(\Delta, \tilde{\Delta}) \subset T_U St(n,r)$ [28, section 5].

Given a start point $U \in St(n,r)$ and an initial velocity $\Delta \in T_U St(n,r)$ the Stiefel geodesic $c_{U,\Delta}$ (and thus the Riemannian exponential) is

$$c_{U,\Delta}(t) = Exp_U(t\Delta) = (U, Q) \exp_m \left(t \begin{pmatrix} A & -R^T \\ R & 0 \end{pmatrix} \right) \begin{pmatrix} I_r \\ 0 \end{pmatrix}, \tag{23}$$

where

$$\Delta = UU^T \Delta + (I - UU^T)\Delta \overset{\text{(QR-decomp. of } (I-UU^T)\Delta)}{=} UA + QR$$

is the decomposition of the tangent velocity into its horizontal and vertical components with respect to the base point U [14]. Because Δ is tangent, $A = U^T \Delta \in \mathbb{R}^{r \times r}$ is skew. The Riemannian Stiefel logarithm has no known closed-form expression but can be computed with the algorithm of [35], which is stated in section S4 of the supplement.

4.2. Differentiating the Stiefel exponential. In this section, we compute the directional derivative of the Stiefel exponential

$$\frac{d}{dt}\Big|_{t=0} \text{Exp}_U(\Delta_0 + tV), \quad \Delta_0, V \in T_U St(n,r). \tag{24}$$

This is important for two reasons.

1. *Differentiable gluing of interpolation curves.* Consider a manifold data set $t_i, p_i = f(t_i)$, $i = 0, \ldots, j, j+1, \ldots, k$, where p_j is selected as the base point for the Riemannian normal coordinates, but where the Riemannian distance of the sample points p_j and p_0 exceeds the injectivity radius of $\mathcal{M}$ at p_j. Then, tangent space interpolation with mapping the full data set to $T_{p_j}\mathcal{M}$ is not possible. A remedy is to split the data set at p_j and to compute two interpolation curves, one for the sample set $t_i, p_i = f(t_i)$, $i = 0, \ldots, j$, and one for the sample set $t_i, p_i = f(t_i)$, $i = j, j+1, \ldots, k$. Assuming that the data $p_0, \ldots, p_j$ lie within the injectivity radius of $\mathcal{M}$ at, say, $p_{\lfloor j/2 \rfloor}$, this data set can be mapped to the tangent space $T_{p_{\lfloor j/2 \rfloor}}\mathcal{M}$. Suppose that the remaining sample points can be mapped to $T_{p_{j+l_0}}\mathcal{M}$ for an index $1 < l_0 < k-j$. With the standard method of tangent space interpolation, the curves have the expressions $c_1(t) = \text{Exp}_{p_{\lfloor j/2 \rfloor}}(\sum_{i=0}^{j} a_i(t)\Delta_i)$ and $c_2(t) = \text{Exp}_{p_{j+l_0}}(\sum_{i=j}^{k} \tilde{a}_i(t)\tilde{\Delta}_i)$, where $\Delta_i = \text{Log}_{p_{\lfloor j/2 \rfloor}}(p_i)$, $i = 0, \ldots, j$, for c_1 and $\tilde{\Delta}_i = \text{Log}_{p_{j+l_0}}(p_i)$, $i = j, \ldots, k$, for c_2. Concatenating the curves c_1, c_2 will result in a nondifferentiable kink at the intersection location p_j, where c_1 ends and c_2 starts. In order to avoid this, one can compute the derivative $\dot{c}_1(t_j)$ and use $\dot{c}_1(t_j) = \dot{c}_2(t_j)$ as a Hermitian derivative sample when constructing c_2, or proceed the other way around. For obtaining $\dot{c}_1(t_j)$, a derivative of the form of (24) must be computed.
2. *Method validation.* The cubic Hermite manifold interpolation method of Theorem 1 requires the computation of $\hat{v}_p = d(\text{Log}_q)_p(v_p)$. As mentioned in

subsection 2.2, the differential of the Log-mapping cannot be computed explicitly for general manifolds $\mathcal{M}$. In order to assess the numerical quality of a finite-differences approximation, we can first compute $\hat{v}_p$ by (16) and then recompute (11):

$$v_{p,rec} = d(\mathrm{Exp}_q)_{\Delta_p}(\hat{v}_p) = \frac{d}{dt}\big|_{t=0} \mathrm{Exp}_q(\Delta_p + t\hat{v}_p).$$

Thus, the numerical accuracy can be assessed via the error

$$\mathcal{E} := \frac{\|v_{p,rec} - v_p\|_p}{\|v_p\|_p}. \tag{25}$$

Again, a derivative of the form of (24) must be computed. We will exploit formula (25) in the numerical example of subsection 5.1.

Now, let us address the derivative (24) for $\Delta_0, V \in T_U St(n,r)$. The underlying computational obstacle is that the exponential law does not hold for the matrix exponential and two noncommuting matrices $\exp_m(X+tY) \neq \exp_m(X)\exp_m(tY)$. Write $\Delta(t) = \Delta_0 + tV$ and let $Q(t)R(t) = (I - UU^T)\Delta(t)$ be the t-dependent QR-decomposition of the tangent space curve. Moreover, $A(t) := U^T\Delta(t)$ and $\dot{A}(0) = U^TV$. Then, by the product rule,

$$\begin{aligned}
\frac{d}{dt}\big|_{t=0} \mathrm{Exp}_U(\Delta(t)) &= \frac{d}{dt}\big|_{t=0} (U, Q(t)) \exp_m\left(\begin{pmatrix} A(t) & -R^T(t) \\ R(t) & 0 \end{pmatrix}\right)\begin{pmatrix} I_r \\ 0 \end{pmatrix} \\
&= (0, \dot{Q}(0)) \exp_m\left(\begin{pmatrix} A(0) & -R^T(0) \\ R(0) & 0 \end{pmatrix}\right)\begin{pmatrix} I_r \\ 0 \end{pmatrix} \\
&\quad + (U, Q(0)) \frac{d}{dt}\big|_{t=0} \exp_m\left(\begin{pmatrix} A(t) & -R^T(t) \\ R(t) & 0 \end{pmatrix}\right)\begin{pmatrix} I_r \\ 0 \end{pmatrix}.
\end{aligned}$$

Introduce the matrix function $M(t) = \left(\begin{smallmatrix} A(t) & -R^T(t) \\ R(t) & 0 \end{smallmatrix}\right)$. It is sufficient to compute $d(\exp_m)_{M(0)}(\dot{M}(0)) = \frac{d}{dt}\big|_{t=0} \exp_m(M(0) + t\dot{M}(0))$. This is a common problem in Lie group theory; see [16, section 5.4]. The solution is formally an infinite sequence of nested commutator products in $[M, \dot{M}] = M\dot{M} - \dot{M}M$,

$$\frac{d}{dt}\big|_{t=0} \exp_m(M + t\dot{M}) = \exp_m(M)\left(\dot{M} - \frac{1}{2!}[M,\dot{M}] + \frac{1}{3!}[M,[M,\dot{M}]] - \cdots\right),$$

which is ill-suited for numerical calculations. Here, the parameter t is omitted with the implicit understanding that all quantities are evaluated at $t = 0$. Yet, by Mathias's theorem [17, Thm. 3.6, p. 58], it holds that

$$\exp_m\left(\begin{array}{c|c} M & \dot{M} \\ \hline 0 & M \end{array}\right) = \left(\begin{array}{c|c} \exp_m(M) & \frac{d}{dt}\big|_{t=0} \exp_m(M + t\dot{M}) \\ \hline 0 & \exp_m(M) \end{array}\right). \tag{26}$$

Hence, for data stemming from $St(n,r)$, a $(4r \times 4r)$-matrix exponential must be computed. The advantage is that $\exp_m(M)$ and $\frac{d}{dt}\big|_{t=0} \exp_m(M + t\dot{M})$ are obtained in one go and both are needed for evaluating (24). Moreover, usually $r \ll n$ in practical applications. For details and alternative algorithms for computing $\frac{d}{dt}\big|_{t=0} \exp_m(M + t\dot{M})$, see [17, section 10.6]. In summary, we have the following.

LEMMA 5. *With all quantities as introduced above, let*

$$\exp_m\left(\begin{array}{c|c} M & \dot{M} \\ \hline 0 & M \end{array}\right) = \left(\begin{array}{c|c} \begin{pmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{pmatrix} & \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix} \\ \hline \mathbf{0} & \begin{pmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{pmatrix} \end{array}\right)$$

be written in terms of subblocks of size $r \times r$. Then

$$\frac{d}{dt}\Big|_{t=0} \mathrm{Exp}_U(\Delta(t)) = \dot{Q}E_{21} + UD_{11} + QD_{21}. \tag{27}$$

The derivatives of the QR-factors of the decomposition $Q(t)R(t) = (I - UU^T)\Delta(t)$ that are required to compute $\dot{Q}$ and $\dot{M} = \left(\begin{smallmatrix} \dot{A} & -\dot{R}^T \\ \dot{R} & 0 \end{smallmatrix}\right)$ can be obtained from Algorithm S1 in the supplement.

4.3. Alternative options for Hermite data preprocessing on $St(n, r)$. As outlined in section 2, the Hermite interpolation problem with local Stiefel sample data $f(t_0) = U, \dot{f}(t_0) = \Delta, f(t_1) = \tilde{U}, \dot{f}(t_1) = \tilde{\Delta}$ requires us to translate the derivative samples to a common tangent space. On $St(n, r)$, this amounts to computing

$$\hat{\Delta} = \frac{d}{ds}\Big|_{s=0} \mathrm{Log}_{\tilde{U}}\left(\tilde{c}(s)\right)$$

for some differentiable curve $\tilde{c}(s) \subset St(n, r)$ that satisfies $\tilde{c}(0) = U$ and $\dot{\tilde{c}}(0) = \Delta$. There are options other than $\tilde{c}(s) = \mathrm{Exp}_U(s\Delta)$ with this property which might be cheaper to evaluate, depending on the context: For a skew-symmetric M_0, the *Cayley transformation* [33, eq. (7)], [6, p. 284]

$$M : s \mapsto M(s) = \left(I + \frac{s}{2}M_0\right)\left(I - \frac{s}{2}M_0\right)^{-1} = I + sM_0 + \frac{s^2}{2}M_0^2 + \ldots \tag{28}$$

produces a curve of orthogonal matrices that matches the matrix exponential $s \mapsto \exp_m(sM_0)$ up to terms of order $\mathcal{O}(s^2)$. As a consequence,

$$M_0 = \frac{d}{ds}\Big|_{s=0} \exp_m(sM_0) = \frac{d}{ds}\Big|_{s=0} M(s)$$

and the matrix curve $s \mapsto (U, Q)M(s)\left(\begin{smallmatrix} I_r \\ 0 \end{smallmatrix}\right)$ with $M(s)$ from (28) based on $M_0 = \left(\begin{smallmatrix} A & -R^T \\ R & 0 \end{smallmatrix}\right)$ as in (23) may be used as the curve $\tilde{c}(s)$ in (14), (15) instead of the Stiefel exponential (23).

Another option is to use *retractions* as a replacement for the Riemannian exponential [2, section 4.1]. By definition, the differential of a retraction map at the origin of the tangent space is the identity map and thus coincides with the differential of the Riemannian exponential at the origin. Suitable matrix curves $s \mapsto \tilde{c}(s)$ that match $\mathrm{Exp}_U(s\Delta)$ up to terms of first order based on Stiefel retractions are

$$\begin{aligned} &\tilde{c} : s \mapsto (U + s\Delta)\Phi(I + s^2\Lambda)^{-\frac{1}{2}}\Phi^T, && \Phi\Lambda\Phi^T \overset{\text{EVD}}{=} \Delta^T\Delta, \\ &\tilde{c} : s \mapsto qr(U + s\Delta), && \text{(compact QR-decomposition);} \end{aligned}$$

see [2, Example 4.1.3].

Remark. With the QR-based retraction, there is the challenge of computing a *differentiable* QR-path. Numerical QR-algorithms in high-level programming environments like MATLAB and SciPy might provide discontinuous matrix paths, e.g., because of different internal pivoting strategies.

5. Examples and experimental results. In this section, we conduct various numerical experiments that put the theoretical findings in perspective. All examples are coded and performed in the SciPy programming environment [19] and address interpolation problems on the Stiefel manifold. As a rule, the following interpolation schemes are conducted for comparison in the experiments.

- *Piecewise geodesic interpolation:* In this case, the Stiefel samples are connected by geodesics as described in [36, section 3.1]. No derivative information is used. This is the manifold version of linear interpolation.
- *RBF tangent space interpolation:* In this case, all data are mapped to a single tangent space attached at $Q(t_j)$, $j = \lfloor k/2 \rfloor$, where $k+1$ is the number of sample points. Then, RBF interpolation is performed on tangent vectors as described in [4], [36, section 3.1]. Here, as an RBF kernel, the inverse multiquadric is selected. No derivative information is used.
- *Hermite quasi-cubic interpolation*, as introduced in subsection 2.2.

This selection comprises two piecewise methods (with and without derivatives) and a global method. For the latter, RBF was chosen because it features most frequently in engineering applications. It produces a smooth interpolant but does not match the sampled derivatives. RBF methods rely on interpoint distance information and are not designed to perform well on just a few sample points. Therefore, the RBF tangent space interpolation is excluded from the comparison for sample data sets of size less than or equal to 3.

5.1. The numerical accuracy of the derivative translates. Before we start with the actual interpolation problems, we assess the numerical accuracy of the process of mapping a velocity sample $v_U \in T_U St(n,r)$ to a tangent velocity $\hat{v}_U \in T_{\tilde{U}} St(n,r)$ for two different Stiefel locations $U, \tilde{U} \in St(n,r)$. This requires the numerical computation of $\hat{v}_U = d(\mathrm{Log}_{\tilde{U}})_U(v_U)$ with the help of central finite differences as in (16).

Then, we reverse this process to recover the original input via

$$v_{U,rec} = \frac{d}{dt}\Big|_{t=0} \mathrm{Exp}_{\tilde{U}}(\mathrm{Log}_{\tilde{U}}(U) + t\hat{v}_U).$$

To this end, we utilize formula (27) of Lemma 5. We compute the error $\mathcal{E} = \frac{\|v_{U,rec} - v_U\|_F}{\|v_U\|_F}$. As data points, we use samples of the Stiefel function $\mu \mapsto U(\mu) \in St(n,r)$, $n = 1001, r = 6$, that is featured in the upcoming subsection 5.4: More precisely, $U = U(0.9)$, $\tilde{U} = U(1.4)$. The Riemannian distance is $\mathrm{dist}(U, \tilde{U}) \approx 0.356$. The tangent direction to be translated is chosen as $v_U = Log_U(U(1.9)) \in T_U St(n,r)$. All Riemannian log computations are performed with the algorithm of [35] and a numerical convergence threshold of $\tau = 10^{-14}$. Table 1 shows the reconstruction error $\mathcal{E}$ versus the finite difference step size h used in (16). Even though there are various numerical processes involved (matrix exp, matrix log, numerical QR-differentiation, iterative Stiefel logarithm, etc.), the accuracy of the finite difference approach is surprisingly high. In the following experiments, a step size of $h = 10^{-4}$ is used to calculate (16). Section S5 of the supplement features an additional experiment with pseudorandom data, where the point $\tilde{U}$ is pushed ever farther away from U. Yet, the accuracy of (16) is hardly affected and stays high as long as the iterative Stiefel log algorithm keeps converging.

5.2. Hermite interpolation of the Q-factor of a QR-decomposition. As a first example, consider a cubic matrix polynomial

$$Y(t) = Y_0 + tY_1 + t^2 Y_2 + t^3 Y_3, \quad Y_i \in \mathbb{R}^{n \times r}, n = 500, r = 10.$$

TABLE 1
Error term (25) *versus finite difference step size* h *of* (16).

Step size h	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}
Error $\mathcal{E}$	1.2e-8	1.2e-10	4.3e-12	4.2e-11	4.1e-10	5.0e-9

The matrices Y_i were produced as random matrices with entries uniformly sampled from $[0,1]$ for Y_0, entries uniformly sampled from $[0,0.5]$ for Y_1, Y_2, and entries uniformly sampled from $[0,0.2]$ for Y_3. The t-dependent QR-decomposition is

$$Y(t) = Q(t)R(t), \quad \dot{Y}(t) = \dot{Q}(t)R(t) + Q(t)\dot{R}(t).$$

The matrix curve $Y(t)$ is sampled at 6 Chebyshev roots in $[-1.1, 1.1]$. The Chebyshev locations read $t_0 \approx -1.0625$, $t_1 \approx -0.7778$, $t_2 \approx -0.2847$, $t_3 \approx 0.2847$, $t_4 \approx 0.7778$, $t_5 \approx 1.0625$.

At each sample point t_i the Q-factor $Q(t_i)$ of the QR-decomposition is computed. The corresponding derivative $\dot{Q}(t_i)$ is obtained from Algorithm S1 in section S2 in the supplement. This constitutes the Hermite sample data set

$$Q(t_i) \in St(n,r), \quad \dot{Q}(t_i) \in T_{Q(t_i)}St(n,r), \quad i = 0, \ldots, 5.$$

As announced above, the *piecewise geodesic interpolation*, the *RBF tangent space interpolation*, and the *Hermite quasi-cubic interpolation* are applied to the sample data set. Since the piecewise geodesic and the quasi-cubic approach rely on piecewise splines, it is only the "global" RBF tangent space interpolation that benefits from the Chebyshev sampling.

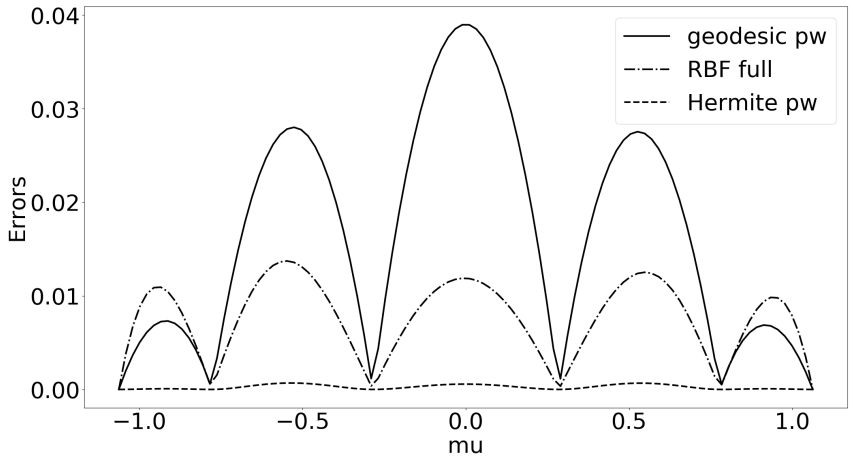

FIG. 4. *Relative errors of the various interpolation approaches in the matrix Frobenius norm for the experiment of subsection* 5.2.

For $t \in [-1.1, 1.1]$, the relative interpolation errors are computed in the matrix Frobenius norm as $\frac{\|Q^*(t)-Q(t)\|_F}{\|Q(t)\|_F}$, where $Q^*(t)$ denotes the manifold interpolant and $Q(t)$ is the reference solution. The error curves are displayed in Figure 4.

The relative Frobenius errors are stated in Table 2. The L_2-norm gives the integrated squared errors on a discrete representation of the interval $[t_0, t_k]$ with a resolution of 100 points.

TABLE 2
Relative Frobenius norm errors and computational costs associated with the experiment of subsection 5.2.

	Geodesic interp.	RBF tan. interp.	Hermite interp.
Errors			
Max. relative errors	0.039	0.014	0.0007
L_2-norm of error data	0.030	0.016	0.0005
Computation times			
Data preprocessing	0.49s	0.80s	1.51s
Probing at 100 locations	0.12s	0.17s	0.14s

The computation times are also listed in Table 2. The computation times split in the preprocessing phase, which comprises the computation of the Riemannian normal coordinate images of the sample data. For the quasi-cubic Hermite approach, this means computing the curve data $\Delta_{Q(t_i)}, \hat{v}_{Q(t_i)}$ of (12) in every subinterval $[t_i, t_{i+1}], i = 0, \dots k-1$. For the piecewise geodesic approach, only the tangent locations $\Delta_{Q(t_i)}$ are required. For the RBF tangent space interpolation, the sample data $\{Q(t_i)\}_i$ are mapped to $\{\Delta_{Q(t_i)}\}_i$ on the same tangent space $T_{Q(t_2)}St(n,r)$. As outlined in subsection 2.3, the Hermite approach requires thrice the number of Riemannian log evaluations as the piecewise geodesic approach, which is also reflected in the computation times. The RBF approach needs the same number of Riemannian log evaluations as the piecewise geodesic approach, but the outer sample locations have a larger distance to the chosen center $Q(t_2)$, which means that the Stiefel logarithm of Algorithm S4 needs more iterations to converge. Once the various interpolants are constructed, they are probed at 100 locations in the sampled interval. Apart from basic matrix summation in the tangent space, this requires a single Riemannian exponential for all three approaches. The associated timings are in the last row of Table 2 and include the featured error calculations.

In section S6 of the supplement, we repeat this experiment but select $p = Q(t_i)$ as the center for the Riemannian normal coordinates. Hence, the derivative data are mapped to $T_{Q(t_i)}St(n,r)$ instead of $T_{Q(t_{i+1})}St(n,r)$ and the tangent space interpolation curve is of the form

$$\gamma(t) = a_0(t)\mathbf{0}_p + a_1(t)\Delta_q + b_0(t)\hat{v}_p + b_1(t)\hat{v}_q \subset T_pSt(n,r) \text{ instead of}$$
$$\gamma(t) = a_0(t)\Delta_p + a_1(t)\mathbf{0}_q + b_0(t)\hat{v}_p + b_1(t)\hat{v}_q \subset T_qSt(n,r),$$

where $p = Q(t_i), q = Q(t_{i+1})$.

5.3. Hermite interpolation of a low-rank SVD. Next, we consider an academic example of a nonlinear matrix function with fixed low rank. The goal is to perform a quasi-cubic interpolation of the associated SVD. As above, we construct a cubic matrix polynomial

$$Y(t) = Y_0 + tY_1 + t^2Y_2 + t^3Y_3, \quad Y_i \in \mathbb{R}^{n\times r}, n = 10\,000, r = 10,$$

with random matrices Y_i, with entries uniformly sampled from $[0, 1]$ for Y_0 and from $[0, 0.5]$ for Y_1, Y_2, Y_3. Then, a second matrix polynomial is considered:

$$Z(t) = Z_0 + tZ_1 + t^2Z_2, \quad Z_i \in \mathbb{R}^{r\times m}, r = 10, m = 300.$$

Here, the entries of Z_0 are sampled uniformly from $[0,1]$ while the entries of Z_1, Z_2 are sampled uniformly from $[0, 0.5]$. The nonlinear low-rank matrix function is set as

$$W(t) = Y(t)Z(t) \in \mathbb{R}^{n\times m}.$$

By construction, $W(t)$ is (with high probability) of fixed $\operatorname{rank}(W(t)) \equiv r = 10\ \forall t$. The low-rank SVD

$$W(t) = U_r(t)\Sigma_r(t)V_r(t)^T, \quad U_r(t) \in St(n,r), V_r(t) \in St(m,r), \Sigma \in \mathbb{R}^{r\times r},$$

is sampled at the two Chebyshev nodes $t_0 = 0.0732, t_1 = 0.4268$ in the interval $[0.0, 0.5]$. (The Chebyshev sampling is implemented as a standard in the program code that was written for the numerical experiments and is not of importance in this case.) The Hermite sample data set

$$U_r(t_i), \dot{U}_r(t_i), \quad V_r(t_i), \dot{V}_r(t_i), \quad \Sigma_r(t_i), \dot{\Sigma}_r(t_i), \quad i = 0, 1,$$

is computed with Algorithm S3 in section S3 of the supplement. (To this end $V(t_i) = (V_r(t_i), V_{m-r}(t_i)) \in O(m)$ is required.)

Remark 6. Computing an analytic path of an SVD and thus a proper sample data set is a challenge in its own right; see [9]. This is in part because of the inherent ambiguity of the SVD even in the case of mutually distinct singular values, where $W = U\Sigma V^T = (US)\Sigma(SV^T)$ for any orthogonal and diagonal matrix $S = \operatorname{diag}(\pm 1, \ldots, \pm 1)$ [17, B.11, p. 334]. SVD algorithms from numerical linear algebra packages may return a different "sign-matrix" S for the SVD of $W(t)$ and $W(s)$, even when t and s are close to each other. This introduces discontinuities in the sampled U and V matrices. In the experiments performed in this work, we normalize the SVD as follows. A reference SVD $U_0\Sigma_0V_0^T = W(t_0)$ is computed. At each t, we compute an SVD $U_t\Sigma_tV_t^T$ and determine $S = \operatorname{sign}(\operatorname{diag}(U_t^TU_0))$, where the sign-function is understood to be applied entrywise on the diagonal elements. Then, we replace $U_t \leftarrow U_tS$, $V_t \leftarrow V_tS$. In the test cases considered here, this hands-on approach is sufficient to ensure a differentiable SVD computation. In general, one has to allow for negative singular values to ensure differentiability [9].

For comparison, the *piecewise geodesic interpolation* and the *Hermite quasi-cubic interpolation* of subsection 2.2 are conducted. For the reasons explained in the introduction to section 5, the RBF tangent space interpolation is excluded from the comparison. Yet, we remark that the linear RBF tangent space interpolation coincides with the piecewise geodesic interpolation and all other RBF kernels tested by the author ("thin plate spline," "cubic," "inverse multiquadric," "Gauß") performed worse on this example; the exception being the "multiquadric" kernel, which performed slightly better than the linear RBF. For t in the sampled range, the relative interpolation errors are computed in the Frobenius norm as $\frac{\|U^*(t)\Sigma^*(t)(V^*(t))^T - W(t)\|_F}{\|W(t)\|_F}$, where $U^*(t)$, $\Sigma^*(t)$, $V^*(t)$ are the interpolants of the matrix factors of the low-rank SVD of $W(t)$ and $W(t)$ is the reference solution. Figure 5 displays the error curves for the piecewise geodesic and the quasi-cubic Hermite interpolation approaches. The values of relative errors are listed in Table 3, which also includes the computational costs. With respect to the computational effort, the same remarks and observations as in Table 2 apply. Keep in mind that the costs of probing the interpolants also include the error calculations.

For the sake of completeness, we repeat this experiment but with selecting $p = U(t_i)$ instead of $q = U(t_{i+1})$ as the center for the Riemannian normal coordinates.

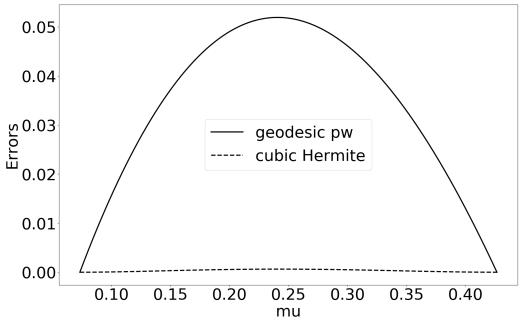

FIG. 5. *Corresponding to subsection* 5.3*: Relative errors associated with the interpolation process of the low-rank SVD in terms of the Frobenius matrix norm.*

TABLE 3
Relative Frobenius norm errors and computational costs associated with the experiment of subsection 5.3.

	Geodesic interp.	Hermite interp.
Errors		
Max. relative errors	0.0519	0.00063
L_2-norm of error data	0.0225	0.00024
Computation times		
Data preprocessing	0.12s	0.33s
Probing at 100 locations	7.11s	7.19s

This leads to virtually indistinguishable plots, which are therefore omitted. The maximum relative errors are $6.3023 \cdot 10^{-4}$ (q-based) versus $6.3374 \cdot 10^{-4}$ (p-based). The L_2-norms of the relative errors are $2.3819 \cdot 10^{-4}$ (q-based) versus $2.3954 \cdot 10^{-4}$ (p-based).

Next, we will show an illustration of Theorem 3. Recall that the local cubic Hermite interpolation scheme works by performing Hermite interpolation in a selected tangent space and subsequently mapping the result to the manifold. Hence, the interpolation error first arises in the tangent space as the gap between the interpolated tangent vector and the exact reference. The final interpolation error is the gap between the manifold image of the interpolated tangent vector and the exact reference on the manifold. For the U-factor interpolation featured in the above example, Figure 6 shows the absolute interpolation errors of the tangent space data in the *canonical Riemannian metric* together with interpolation errors of final manifold data in terms of the *Riemannian distance*. The manifold errors are very close to the tangent space errors but are actually slightly smaller, despite the additional downstream translation of the tangent space interpolants to the manifold via the Riemannian exponential, which is an additional source of numerical errors. This is in line with Theorem 3, since the Stiefel manifold features positive sectional curvature.

5.4. Hermite interpolation of the left singular vectors of nonlinear function snapshots. In the next experiment, we consider the SVD of discrete snapshots

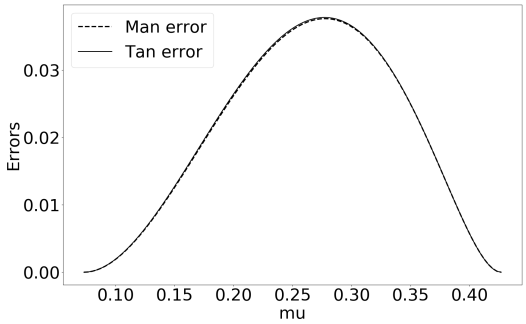

FIG. 6. *Interpolation of the U-factor of the SVD data in subsection* 5.3. *Absolute Hermite interpolation errors in terms of the Riemannian metric on the tangent space (Tan error) and as measured by the Riemannian distance function on the manifold (Man error).*

of a nonlinear multiparameter function. To this end, define

$$f : [0,1] \times [0,2] \times [1,4] \to \mathbb{R}, (x,t,\mu) \mapsto x^t \sin\left(\frac{\pi}{2}\mu x\right) \text{ and}$$
$$F : [0,1] \times [0,2] \times [1,4] \to \mathbb{R}, (x,t,\mu) \mapsto \frac{f(x,t,\mu)}{\|f(\cdot,t,\mu)\|_{L_2}},$$

where $\langle f_1, f_2\rangle_{L_2} = \int_0^1 f_1(x)f_2(x)dx$ and $\|\cdot\|_{L_2} = \sqrt{\langle\cdot,\cdot\rangle_{L_2}}$ on $L_2([0,1])$. We will discretize F in x, take function "snapshots" at selected values of t, and eventually Hermite-interpolate the left singular vectors of the discrete snapshot matrices with respect to μ. The partial derivative of F by μ is

$$\partial_\mu F(x,t,\mu) = \frac{1}{\|f(\cdot,t,\mu)\|_{L_2}}\partial_\mu f(x,t,\mu) - \frac{\langle f(\cdot,t,\mu), \partial_\mu f(\cdot,t,\mu)\rangle_{L_2}}{\|f(\cdot,t,\mu)\|_{L_2}^3} f(x,t,\mu).$$

For the spatial discretization, we use an equidistant decomposition of the unit interval, $0 = x_1, x_2, \ldots, x_n = 1$, $n = 1001$. Then, we take $r = 6$ function snapshots in t at time instants $t = 1.0,\ 1.6,\ 2.2,\ 2.8,\ 3.4,\ 4.0$. In this way, a μ-dependent snapshot matrix function $Y(\mu) := (F(x,t_1,\mu), \ldots, F(x,t_6,\mu)) \in \mathbb{R}^{n\times r} = \mathbb{R}^{1001\times 6}$ with SVD

$$Y(\mu) = U(\mu)\Sigma(\mu)V(\mu)^T, \quad U(\mu) \in St(n,r), \Sigma(\mu) \in \mathbb{R}^{r\times r}, V(\mu) \in O(r),$$

is obtained. Figure 7 displays the snapshot matrices at some selected parameter values together with the associated left singular value matrices. For $\mu \in [0,2]$, the values of f are nonnegative and so are all entries in the corresponding snapshot matrices. Beyond $\mu = 2.0$, negative entries arise in the snapshot vectors. Figure 8 tracks the smallest singular value $\sigma_r(\mu)$ of the snapshot matrices $Y(\mu)$ for $\mu \in [1.7, 2.3]$. A substantial nonlinear change in $\sigma_r(\mu)$ is apparent around the value of $\mu = 2.0$. This makes SVD interpolation beyond the parameter location $\mu = 2.0$ a challenging problem.

We sample the left singular vector matrix $U(\mu)$ together with the derivative $\dot{U}(\mu)$ at 6 Chebyshev samples $\mu_i \in \{1.7102, 1.7879, 1.9224, 2.0776, 2.2121, 2.2898\}$ in the interval $[1.7, 2.3]$. As in subsection 5.2, we juxtapose the results of piecewise geodesic interpolation, RBF tangent space interpolation, and quasi-cubic Hermite interpolation. For μ in the sampled range, the relative interpolation errors

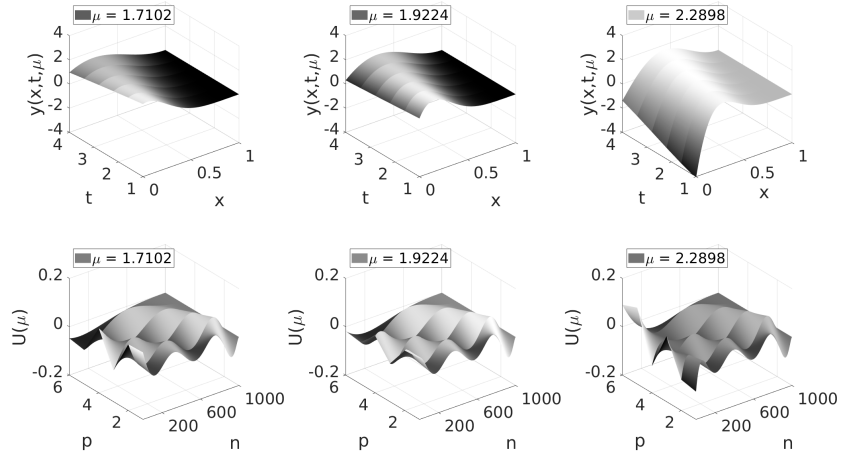

FIG. 7. *Sample data featured in subsection* 5.4. *Upper row: Snapshot matrices* $Y(\mu)$. *Lower row: Corresponding left singular vector matrices* $U(\mu)$.

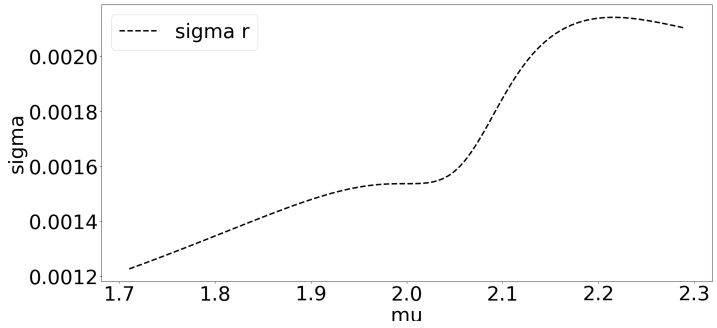

FIG. 8. *Corresponding to subsection* 5.4. *The smallest singular value* $\sigma_r(\mu)$ *of the* μ*-dependent SVD* $Y(\mu) = U(\mu)\Sigma(\mu)V(\mu)^T$ *in the range* $\mu \in [1.7, 2.3]$.

are computed in the Frobenius norm as $\frac{\|U^*(\mu)-U(\mu)\|_F}{\|U(\mu)\|_F}$, where $U^*(\mu)$ denotes the manifold interpolant and $U(\mu)$ is the reference solution. The error curves are displayed in Figure 9. According to the figure, the RBF tangent space interpolation method fails to interpolate the samples at the first two parameter locations $\mu_0 \approx 1.7102, \mu_1 \approx 1.7879$. This is explained as follows. In the RBF tangent space interpolation method, all the Stiefel samples $U(\mu_i)$ are mapped to the tangent space attached at $U(\mu_3)$, $\mu_3 \approx 2.0776$, via $\Delta(\mu_i) = \mathrm{Log}_{U(\mu_3)}(U(\mu_i))$. Note that the base point μ_3 happens to lie beyond the "2.0-threshold value," after which negative function values appear. On the other hand, $\mu_0, \mu_1 < 2.0$. It turns out that the Riemannian Stiefel log-algorithm is not well defined for $i = 0, 1$. Put in different words, $U(\mu_0)$ and $U(\mu_1)$ are too far from $U(\mu_3)$ to be mapped to $T_{U(\mu_3)}St(n, r)$ by the Stiefel log-algorithm. For a more quantitative analysis, the distance $\mathrm{dist}(U(\mu_i), U(\mu_3))$, $i = 0, 1$, is decisive. Yet, without an improved Log-algorithm, it is also not possible to compute

$\operatorname{dist}(U(\mu_0), U(\mu_3)) = \|Log_{U(\mu_3)}(U(\mu_0))\|$ explicitly. An estimate can be obtained via

$$\operatorname{dist}(U(\mu_0), U(\mu_3)) \leq \operatorname{dist}(U(\mu_0), U(\mu_2)) + \operatorname{dist}(U(\mu_2), U(\mu_3)) \approx 0.64 + 1.53 = 2.17.$$

Likewise, an upper bound for $\operatorname{dist}(U(\mu_1), U(\mu_3))$ is 2.001. From [28, eq. (5.13)], the injectivity radius on $St(n,p)$ is bounded from below by $\operatorname{inj}(St(n,p)) \geq \sqrt{\frac{4}{5}}\pi \approx 2.81$. This shows that the data points are actually within the injectivity radius. Hence, the interpolation failure of the RBF method is due to a shortcoming of the numerical Stiefel log-algorithm rather than due to a mathematical obstacle.

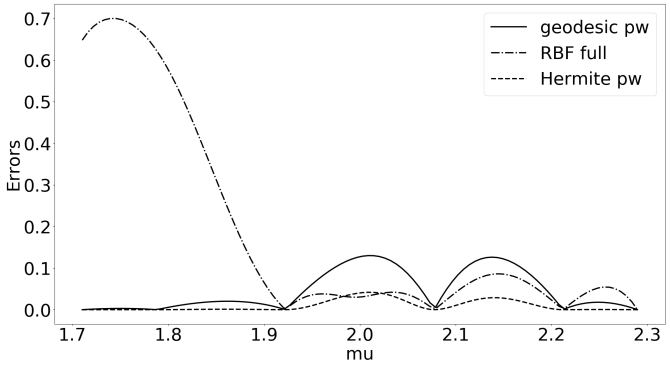

FIG. 9. *Relative errors of the various interpolation approaches in the matrix Frobenius norm for the experiment of subsection* 5.4.

The relative Frobenius errors and computation times associated with the test case at hand are listed in Table 4. With respect to the computation times, the same discussion as in subsection 5.2 and subsection 5.3 applies.

TABLE 4
Relative Frobenius norm errors and computation times associated with the experiment of subsection 5.4.

	Geodesic interp.	RBF tan. interp.	Hermite interp.
Errors			
Max. relative errors	0.1301	0.7003	0.0418
L_2-norm of error data	0.0501	0.2336	0.0123
Computation times			
Data preprocessing	0.48s	0.98s	1.30s
Probing at 100 locations	0.11s	0.18s	0.11s

The results of the Hermite approach with selecting $p = U(t_i)$ instead of $q = U(t_{i+1})$ as the center for the Riemannian normal coordinates are stated in section S6 of the supplement.

5.5. Parametric dimension reduction. SVD interpolation may be used for parametric dimension reduction. Consider samples $Y_0, \ldots, Y_k$ of a matrix curve $\mu \mapsto Y(\mu) \subset \mathbb{R}^{n\times m}$. Suppose that instead of the original data, only a low-rank SVD approximation is stored $Y_i \approx U_i \Sigma_i V_i^T$, where $U_i \in \mathbb{R}^{n\times r}, \Sigma_i \in \mathbb{R}^{r\times r}, V_i \in \mathbb{R}^{m\times r}$ for a fixed $r \ll \min\{n,m\}$. Then a low-rank approximant to any $Y(\mu)$ in the sampled range can be obtained via interpolating the SVD data.

In this section, we apply this approach to an application from computational option pricing. The value function $y(T, s; K, r, \sigma)$ that gives a fair price for a European call option is determined via the *Black–Scholes equation* [7],

$$0 = \partial_t y(t,s) + \frac{1}{2}\sigma^2 s^2 \partial_s^2 y(t,S) + rs\partial_s y(t,s) - ry(t,s), \quad s \geq 0, \quad 0 < t \leq T,$$

with terminal condition $y(T,s) = \max\{s-K, 0\}, s > 0$, and boundary conditions $V(0,t) = 0$, $V(s,t) = s - Ke^{-r(T-t)} \sim s$, $s \to \infty$, as discussed in [31, section 4]. This is a parabolic PDE that depends on time t, the stock value s, and a number of additional system parameters, namely the strike price K, the interest rate r, the volatility σ, and the exercise time T. In this experiment, we consider a fixed interest rate of $r = 0.01$ and an exercise time of $T = 2$ units. The dependency on the underlying $s \in [50, 150]$ is resolved via a discretization of the interval by equidistant steps of $\Delta s = 0.01$, while the strike price $K \in [30, 170]$ is discretized in steps of $\Delta K = 1$. Eventually, the volatility σ will act as the interpolation parameter. Hermite interpolation requires the option price y as well as its derivative $\partial_\sigma y$, which in economics is referred to as the "vega" of set of the set of "greeks." A similar test case was considered in [37].

The Black–Scholes equation for a single underlying has a closed-form solution. Yet, here, we will approach it via a numerical scheme in order to mimic the corresponding procedure for real-life problems. Application of a finite volume scheme to the Black–Scholes PDE yields snapshot matrices

$$Y(\sigma) = (Y_{s,k}(\sigma))_{\substack{s=1,\ldots,10\,001 \\ k=1,\ldots,141}}, \quad \partial_\sigma Y(\sigma)_{\substack{s=1,\ldots,10\,001 \\ k=1,\ldots,141}}$$

for $\sigma \in [0.1, 0.2, \ldots, 1.0]$. The computation time for each data pair $Y(\sigma), \partial_\sigma Y(\sigma)$ is ca. 17min on a standard laptop computer. For each sampled snapshot matrix $Y(\sigma) \in \mathbb{R}^{n\times m}$, $n = 10\,001$, $m = 141$, an SVD is performed and is truncated to the $r = 5$ dominant singular values/singular vector triples. This yields a compressed representation $Y(\sigma) = U(\sigma)\Sigma(\sigma)V^T(\sigma)$, with $U(\sigma) \in St(n,r)$, $\Sigma(\sigma) \in diag(r,r)$, $V(\sigma) \in St(m,r)$ and consumes ca. $0.1s$ on a laptop computer. The *relative information content* is $\mathrm{ric}(r) = \frac{\sum_{j=1}^r \sigma_j^2}{\sum_{k=1}^m \sigma_k^2} \geq 0.99999$. The storage requirements for a $(10\,001 \times 141)$-matrix are 11.3MB; all the low-rank SVD factors truncated to $r = 5$ require a total of 0.4MB of disk space, which is ca. 3.5% of the uncompressed representation. We sample full solution data sets $Y(\sigma), \partial_\sigma Y(\sigma)$ at $\sigma \in \{0.1, 0.4, 0.9\}$. The sample data sets are displayed in Figure 10. In order to assess the approximation accuracy, we interpolate at $\sigma^* \in \{0.2, 0.3,\ 0.5, 0.6, 0.7, 0.8\}$ and compute the relative Frobenius norm errors of the interpolated low-rank SVD representation $U(\sigma^*)\Sigma(\sigma^*)V(\sigma^*)^T$ with respect to the exact full rank data matrix $Y(\sigma^*)$.

We compare the *piecewise geodesic interpolation* (without derivative data) to the *cubic Hermite approach*. As explained in subsection 5.3, employing a global method like the RBF tangent space interpolation is inadequate with as few as three sample points at hand. The manifold interpolation approaches exploit (and preserve) the low-rank representation of the snapshot data. An alternative is to keep the full, uncompressed data matrices $Y_i \in \mathbb{R}^{10\,001\times 141}$ and to interpolate in the flat Euclidean matrix space. In this case, the classical linear and Hermite interpolation procedures can be performed as outlined in subsection 2.1. We refer to this as the *linear full interpolation* and the *Hermitian full interpolation* approach, respectively. The errors are displayed in the bar plot Figure 11. The values underlying the bar plot are listed in

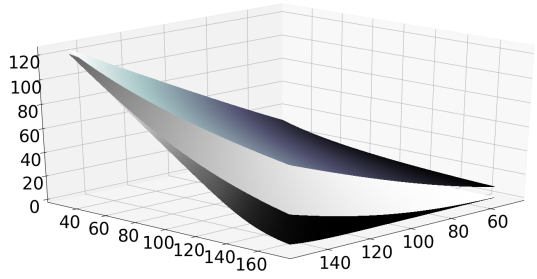

FIG. 10. *Corresponding to subsection* 5.5. *Sampled data Black–Scholes solution data sets* $Y(0.1), Y(0.4), Y(0.9)$.

Table 5. Note that the errors for the geodesic and the Hermite low-rank interpolation include both the effects of interpolation *and* dimension reduction. Even though the data sets underwent a substantial reduction in dimension, the relative errors are of a comparable order of magnitude. To better judge the results, we note that the relative errors between the two consecutive samples $Y(0.1)$, $Y(0.4)$ and $Y(0.4)$, $Y(0.9)$ are 27% and 39%, respectively. The computation time to produce the low-rank SVD interpolants at the six trial locations with the manifold Hermite approach is $0.028s$, which compares to $0.083s$ for the Euclidean Hermite method on the uncompressed matrices. Even though the dimensions of this test case are still rather academic, the low-rank manifold Hermite approach not only saves storage but is already roughly three times faster than the Euclidean method on the uncompressed data.

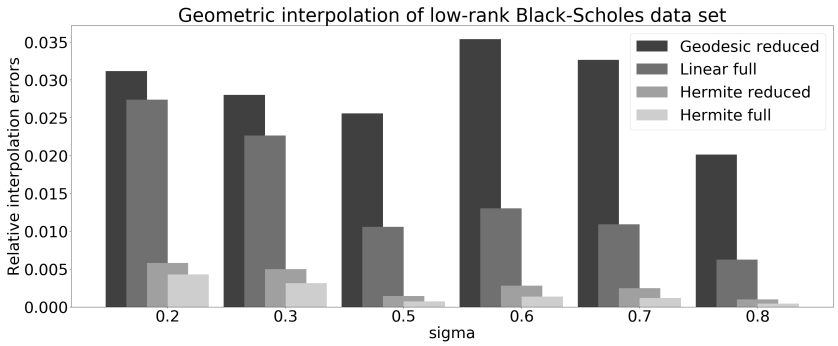

FIG. 11. *Corresponding to subsection* 5.5. *Relative interpolation errors associated with the Black–Scholes test case. Compressed, low-rank SVD obtained via geodesic interpolation and Hermite interpolation on* $St(n,r), St(m,r)$, *respectively, versus uncompressed standard linear and Hermite interpolation on the flat matrix space* $\mathbb{R}^{n\times m}$.

The technique could be used as a reduced online storage scheme: store the truncated (Hermite) SVD data at some selected sample locations; interpolate when a prediction at any in-between location is required online.

6. Conclusions and final remarks. We have presented an elementary, general approach to Hermite interpolation on Riemannian manifolds that is applicable to

TABLE 5
Numerical values associated with Figure 11.

Volatility σ	0.2	0.3	0.5	0.6	0.7	0.8
Geodesic reduced	0.031178	0.028025	0.025577	0.035379	0.032635	0.0201280
Linear full	0.027382	0.022640	0.010562	0.013015	0.010908	0.0062528
Hermite reduced	0.005791	0.004969	0.001416	0.002799	0.002482	0.0009872
Hermite full	0.004298	0.003142	0.000734	0.001374	0.001156	0.0004374

practical problems whenever algorithms to compute the Riemannian exp and log mappings are available. While our focus was on the manifold counterpart of local cubic Hermite interpolation, the method is flexible and may be combined with any Hermite method that is linear in the sample data. In fact, only the coefficient functions a_0, b_0, b_1 in (12) need to be replaced; no additional changes are necessary. Moreover, combinations of Hermite and Lagrange methods are straightforward generalizations.

In addition, we have exposed a relation between the sectional curvature of the manifold in question and the data processing errors that arise for computations in Riemannian normal coordinates.

As an example, Hermite interpolation of Stiefel data was discussed in more detail. From the observations in the numerical experiments, the main practical constraint on the sampled data is that two consecutive samples be close enough so that the Riemannian Stiefel logarithm is well defined. As a rule of thumb, if the data points are close enough so that the Riemannian log algorithm converges, then the Hermite interpolation method provides already quite accurate results. This observation is supported by the experiments of subsection 5.4, where in particular the Riemannian distance between the data points $U(\mu_2)$ and $U(\mu_3)$ is almost $\pi/2$ and is thus comparably large, while the maximum relative approximation error is below 5%.

As presented here, the method constructs piecewise cubic manifold splines between data points $p = f(t_i)$ and $q = f(t_{i+1})$ in terms of normal coordinates centered at q. Thus, it is not symmetric in the sense that computations in normal coordinates centered at p will lead to different results. Yet, as shown in the numerical experiment of subsection 5.3 as well as in section S6 of the supplement, these effects prove to be minor. Symmetry in the choice of the base point could be established by using the technique of blending arcs as in [15]: construct one Hermite arc centered at p, construct one Hermite arc centered at q, and blend the arcs by "geodesic averaging." However, this requires evaluating more than twice as many Riemannian Exp and Log-mappings. While desirable from a theoretical viewpoint, there is no reason to expect that a "symmetric method" provides better approximation accuracy.

Acknowledgment. The data set featured in subsection 5.5 was kindly provided by my colleague Kristian Debrabant from the Department for Mathematics and Computer Science (IMADA), SDU Odense.

REFERENCES

[1] P.-A. ABSIL, P.-Y. GOUSENBOURGER, P. STRIEWSKI, AND B. WIRTH, *Differentiable piecewise-Bézier surfaces on Riemannian manifolds*, SIAM J. Imaging Sci., 9 (2016), pp. 1788–1828, https://doi.org/10.1137/16M1057978.

[2] P.-A. ABSIL, R. MAHONY, AND R. SEPULCHRE, *Optimization Algorithms on Matrix Manifolds*, Princeton University Press, Princeton, NJ, 2008, http://press.princeton.edu/titles/8586.html.

[3] D. Amsallem, *Interpolation on Manifolds of CFD-Based Fluid and Finite Element-Based Structural Reduced-Order Models for On-Line Aeroelastic Prediction*, Ph.D. thesis, Stanford University, Stanford, CA, 2010.

[4] D. Amsallem and C. Farhat, *Interpolation method for adapting reduced-order models and application to aeroelasticity*, AIAA J., 46 (2008), pp. 1803–1813.

[5] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann Ser. Comput. Graphics, Elsevier, New York, 1995.

[6] R. Bhatia, *Matrix Analysis*, Grad. Texts in Math. 169, Springer-Verlag, New York, Berlin, Heidelberg, 1997.

[7] F. Black and M. Scholes, *The pricing of options and corporate liabilities*, J. Political Economy, 81 (1973), pp. 637–654.

[8] N. Boumal and P.-A. Absil, *A discrete regression method on manifolds and its application to data on SO(n)*, IFAC Proc. Vol., 44 (2011), pp. 2284–2289, https://doi.org/10.3182/20110828-6-IT-1002.00542.

[9] A. Bunse-Gerstner, R. Byers, V. Mehrmann, and N. K. Nichols, *Numerical computation of an analytic singular value decomposition of a matrix valued function*, Numer. Math., 60 (1991), pp. 1–39.

[10] M. Camarinha, F. Silva Leite, and P. Crouch, *On the geometry of Riemannian cubic polynomials*, Differential Geom. Appl., 15 (2001), pp. 107–135, https://doi.org/10.1016/S0926-2245(01)00054-7.

[11] P. Crouch and F. Silva Leite, *The dynamic interpolation problem: On Riemannian manifolds, Lie groups, and symmetric spaces*, J. Dynam. Control Syst., 1 (1995), pp. 177–202, https://doi.org/10.1007/BF02254638.

[12] P. Deuflhard and A. Hohmann, *Numerical Analysis in Modern Scientific Computing: An Introduction*, Texts Appl. Math. 43, Springer, New York, 2003, https://doi.org/10.1007/978-0-387-21584-6.

[13] M. P. do Carmo, *Riemannian Geometry*, Mathematics: Theory & Applications, Birkhäuser Boston, Boston, 1992.

[14] A. Edelman, T. A. Arias, and S. T. Smith, *The geometry of algorithms with orthogonality constraints*, SIAM J. Matrix Anal. Appl., 20 (1998), pp. 303–353, https://doi.org/10.1137/S0895479895290954.

[15] P.-Y. Gousenbourger, E. Massart, and P.-A. Absil, *Data fitting on manifolds with composite Bézier-like curves and blended cubic splines*, J. Math. Imaging Vis., 61 (2019), pp. 645–671, https://doi.org/10.1007/s10851-018-0865-2.

[16] B. C. Hall, *Lie Groups, Lie Algebras, and Representations: An Elementary Introduction*, 2nd ed., Grad. Texts in Math. 222, Springer-Verlag, Cham, 2015.

[17] N. J. Higham, *Functions of Matrices: Theory and Computation*, SIAM, Philadelphia, PA, 2008, https://doi.org/10.1137/1.9780898717778.

[18] J. Jakubiak, F. S. Leite, and R. Rodrigues, *A two-step algorithm of smooth spline generation on Riemannian manifolds*, J. Comput. Appl. Math., 194 (2006), pp. 177–191, https://doi.org/10.1016/j.cam.2005.07.003.

[19] E. Jones, T. Oliphant, P. Peterson, et al., *SciPy: Open Source Scientific Tools for Python*, 2001–, http://www.scipy.org/ (accessed July 2019).

[20] H. Le, K. R. Kim, and I. L. Dryden, *Smoothing Splines on Riemannian Manifolds, with Applications to 3D Shape Space*, preprint, https://arxiv.org/abs/1801.04978, 2018.

[21] K. A. Krakowski, L. Machado, F. Silva Leite, and J. Batista, *Solving interpolation problems on Stiefel manifolds using quasi-geodesics*, in Pré-Publicaçiões do Departamento de Matemática, Universidade de Coimbra, 2015, 15–36.

[22] J. M. Lee, *Riemannian Manifolds: An Introduction to Curvature*, Springer-Verlag, New York, Berlin, Heidelberg, 1997.

[23] J. M. Lee, *Introduction to Smooth Manifolds*, 2nd ed., Grad. Texts in Math. 218, Springer, New York, 2013.

[24] F. Narcowich, *Generalized Hermite interpolation and positive definite kernels on a Riemannian manifold*, J. Math. Anal. Appl., 190 (1995), pp. 165–193.

[25] E. Nava-Yazdani and K. Polthier, *De Casteljau's algorithm on manifolds*, Comput. Aided Geom. Des., 30 (2013), pp. 722–732, https://doi.org/10.1016/j.cagd.2013.06.002.

[26] L. Noakes, G. Heinzinger, and B. Paden, *Cubic splines on curved spaces*, IMA J. Math. Control Inform., 6 (1989), pp. 465–473, https://doi.org/10.1093/imamci/6.4.465.

[27] T. Popiel and L. Noakes, *Bézier curves and C2 interpolation in Riemannian manifolds*, J. Approx. Theory, 148 (2007), pp. 111–127.

[28] Q. Rentmeesters, *Algorithms for Data Fitting on Some Common Homogeneous Spaces*, Ph.D. thesis, Université Catholique de Louvain, Louvain, Belgium, 2013, http://hdl.handle.net/2078.1/132587.

[29] C. Samir, P.-A. Absil, A. Srivastava, and E. Klassen, *A gradient-descent method for curve fitting on Riemannian manifolds*, Found. Comput. Math., 12 (2012), pp. 49–73, https://doi.org/10.1007/s10208-011-9091-7.

[30] C. Samir and I. Adouani, *$C1$ interpolating Bézier path on Riemannian manifolds, with applications to $3D$ shape space*, Appl. Math. Comput., 348 (2019), pp. 371–384, https://doi.org/10.1016/j.amc.2018.11.060.

[31] R. Seydel, *Tools for Computational Finance*, 6th ed., Universitext, Springer-Verlag, London, 2017.

[32] F. Steinke, M. Hein, J. Peters, and B. Schoelkopf, *Manifold-valued thin-plate splines with applications in computer graphics*, Comput. Graphics Forum, 27 (2008), pp. 437–448, https://doi.org/10.1111/j.1467-8659.2008.01141.x.

[33] Z. Wen and W. Yin, *A feasible method for optimization with orthogonality constraints*, Math. Program., 142 (2013), pp. 397–434, https://doi.org/10.1007/s10107-012-0584-1.

[34] R. Zimmermann, *On the maximum likelihood training of gradient-enhanced spatial Gaussian processes*, SIAM J. Sci. Comput., 35 (2013), pp. A2554–A2574, https://doi.org/10.1137/13092229X.

[35] R. Zimmermann, *A matrix-algebraic algorithm for the Riemannian logarithm on the Stiefel manifold under the canonical metric*, SIAM J. Matrix Anal. Appl., 38 (2017), pp. 322–342, https://doi.org/10.1137/16M1074485.

[36] R. Zimmermann, *Manifold Interpolation and Model Reduction*, preprint, https://arxiv.org/abs/1902.06502v1, 2019.

[37] R. Zimmermann and K. Debrabant, *Parametric model reduction via interpolating orthonormal bases*, in Numerical Mathematics and Advanced Applications ENUMATH 2017, F. A. Radu, K. Kumar, I. Berre, D. N. Nordbotten, and I. S. Pop, eds., Springer, Cham, 2018, https://doi.org/10.1007/978-3-319-96415-7.