MOX-Report No. 03/2024

A gentle introduction to interpolation on the Grassmann manifold

Ciaramella, G.; Gander, M.J.; Vanzan, T.

A GENTLE INTRODUCTION TO INTERPOLATION ON THE GRASSMANN MANIFOLD[*]

GABRIELE CIARAMELLA [†], MARTIN J. GANDER [‡], AND TOMMASO VANZAN [§]

Key words. Grassmann manifold, interpolation algorithms, exponential map, logarithmic map, Stiefel manifold, reduced order modelling, two-level stationary methods.

MSC codes. 15-01, 53-04, 65D05, 65F99, 97N50

1. Introduction. These notes originated from the authors' effort while studying interpolation techniques on the Grassmann manifold. This has been a hot topic recently since it is an important tool in parametric reduced order modelling. Fortunately, there is an extensive literature available with seminal contributions both from the engineering, e.g., [2, 3, 11, 15], and mathematical communities, e.g., [5, 29, 1, 4]. More generally, the development of numerical methods involving manifolds is a very active research area, see, e.g., [6] and references therein.
Given all these previous works, the reader may immediately ask the following question: is there any need for an additional introductory manuscript? It is the authors' belief that this is actually the case. The aim of these notes is to precisely fill a gap in the literature, by providing a reference which gently introduces numerical analysts to the very interesting research topic of interpolation on the Grassmann manifold. Indeed, on the one hand, the engineering literature often does not provide the necessary mathematical details needed by a numerical analyst to understand the subject and to solidly build new computational algorithms. On the other hand, manuscripts from the mathematical community, despite being seminal references, tend to be overwhelming in terms of details, and mathematically concepts that are often not familiar to numerical analysts approaching the topic for the first time. These notes are meant to be a first very gentle introduction to these numerical methods, before approaching the more organic references [1, 4, 29]. Further, the notes are self-contained concerning the derivation of geodesics, the algorithms to compute the exponential and logarithmic maps, and interpolation algorithms on the Grassmann manifold. These mathematical results are all well-known, but the original proofs are scattered across several manuscripts, often using different notations and level of detail, so that their study may not be immediate.
The manuscript is organized as follows. In Section 2 we introduce the very few concepts on general manifolds that will be needed in the rest of the notes. To guide the reader throughout our survey, we consider two examples: the unit circle and the Stiefel manifold. These are used systematically to smoothly introduce and visualise the mathematical concepts that are encountered in this manuscript. In Section 3 we discuss the Grassmann manifold and, retracing the same path of Section 2, present the necessary tools on the concrete case of the Grassmann manifold. Section 4

[*]

Funding: This work was supported through the program "Oberwolfach Research Fellows" by the Mathematisches Forschungsinstitut Oberwolfach in 2023.

[†]MOX, Dipartimento di Matematica, Politecnico di Milano, Italy , (gabriele.ciaramella@polimi.it).

[‡]Section de Mathématiques, Université de Genève, Switzerland, (martin.gander@unige.ch).

[§]CSQI Chair, Section de Mathématiques, École Polytechnique Fédérale de Lausanne, Switzerland, (tommaso.vanzan@epfl.ch.)

deals with interpolation algorithms by first describing in sequence a linear, a piecewise linear and a high-order interpolation scheme for univariate Grassmann-valued maps. Then, the well-known general high-order interpolation method for multi-variate Grassmann-valued maps [2], which is often just stated and taken for granted in several manuscripts, is introduced as a natural generalization of the previous simpler methods that are easier to understand intuitively. Finally, in Section 5 we discuss two numerical examples arising in the context of model order reduction and stationary methods for linear systems. For both examples, we include full-working Matlab codes which are also available for download as supplementary material.

2. What is a manifold?.

Differentiable manifolds are abstract mathematical objects that extend our intuition of smooth one-dimensional curves and two-dimensional surfaces to arbitrary dimensions and ambient spaces. In these notes, we focus on manifolds which are embedded into a larger ambient space, that is they can be seen as subsets of a larger linear space. As an example, consider a sphere which is a two-dimensional surface embedded into the three dimensional space $\mathbb{R}^3$. This special class of manifolds are often called submanifolds, but for the sake of simplicity we will simply call them manifolds.
Just as not all subsets of $\mathbb{R}^3$ represent a differentiable two-dimensional surface, differentiable manifolds must satisfy some conditions in order to deserve such a title.

DEFINITION 2.1 (Manifold). *A subset $\mathcal{M} \subset \mathbb{R}^{n+d}$ is called a d-dimensional differentiable manifold if there exists a collection of open sets $\{\mathcal{U}_\alpha\}_\alpha$ and functions (called coordinate charts) $\{\phi_\alpha\}_\alpha$ such that every $p \in \mathcal{M}$ belongs to at least one $\mathcal{U}_\alpha$ and the functions $\phi_\alpha : \mathcal{M} \supset \mathcal{U}_\alpha \to \phi_\alpha(\mathcal{U}_\alpha) \subset \mathbb{R}^d$ are diffeomorphisms. Further, whenever $\mathcal{U}_\alpha \cap \mathcal{U}_\beta \neq \emptyset$,*

$$\phi_\alpha \circ \phi_\beta^{-1} : \phi_\beta\left(\mathcal{U}_\alpha \cap \mathcal{U}_\beta\right) \to \phi_\alpha\left(\mathcal{U}_\alpha \cap \mathcal{U}_\beta\right) \tag{2.1}$$

is a diffeomorphism.

Recall that a *diffeomorphism* between two sets U and V is a bijective differentiable map $\phi : U \to V$ such that its inverse is also differentiable. Intuitively, a differential manifold looks locally (around every p) like a subset of $\mathbb{R}^d$, since we may find an open neighboorhod of p which is in one-to-one correspondance with an open subset of $\mathbb{R}^d$ through a smooth function, see Fig. 1 for a graphical description. Remark further that the inverse of a chart provides a useful smooth local parametrization of the manifold around a point p: let $\mathcal{U}_\alpha$ be an open set such that $p \in \mathcal{U}_\alpha$, then

$$\phi_\alpha^{-1} : \mathbb{R}^d \supset \phi_\alpha(\mathcal{U}_\alpha) \to \mathcal{U}_\alpha \subset \mathcal{M} \subset \mathbb{R}^{n+d}$$

is a local parametrization around the point p.
The adjective differentiable entails that differentiable manifolds enjoy some extra properties compared to "simple" manifolds. This is due to the requirement that the coordinate charts are diffeomorphisms, which results in the important property that, given an open set $\mathcal{U}_\alpha$ and a function $f : \mathcal{M} \supset \mathcal{U}_\alpha \to \mathbb{R}$ defined on the manifold, we may consider $\widehat{f} := f \circ \phi_\alpha$, which is now a function from $\mathbb{R}^d \to \mathbb{R}$ whose properties, such as, e.g., differentiability, can be studied with standard tools from calculus. The compatibility condition (2.1) guarantes that the definition of $\widehat{f}$ as well as its properties do not depend on the precise choice of the coordinate charts.
We next discuss a first classical example to clarify the concepts.

EXAMPLE 2.2 (The unit circle). *Consider the two dimensional unit circle*

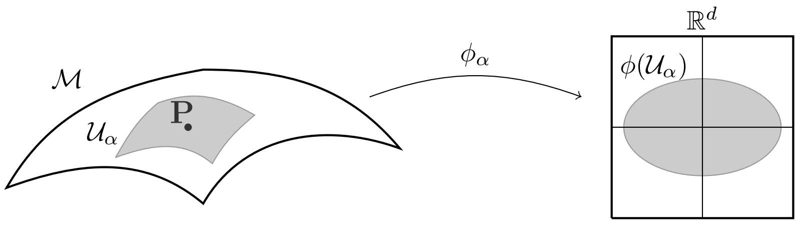

FIG. 1. *Graphical description of a coordinate chart. Given a point P on the manifold $\mathcal{M}$, we can find an open subset $\mathcal{U}_\alpha$, and a diffeomorphism ϕ_α which maps $\mathcal{U}_\alpha$ into an open subset $\phi_\alpha(\mathcal{U}_\alpha)$ of $\mathbb{R}^d$.*

$S^1 := \left\{\boldsymbol{x} = (x_1, x_2) \in \mathbb{R}^2 :\ x_1^2 + x_2^2 = 1\right\} \subset \mathbb{R}^2$, *the four sets*

$$\begin{aligned} U_1 &= S^1 \cap \left\{\boldsymbol{x} \in \mathbb{R}^2 : x_1 > 0\right\}, \quad & U_2 &= S^1 \cap \left\{\boldsymbol{x} \in \mathbb{R}^2 : x_1 < 0\right\}, \\ U_3 &= S^1 \cap \left\{\boldsymbol{x} \in \mathbb{R}^2 : x_2 > 0\right\}, \quad & U_4 &= S^1 \cap \left\{\boldsymbol{x} \in \mathbb{R}^2 : x_2 < 0\right\}, \end{aligned}$$

and the charts

$$\begin{aligned} \phi_1 &= U_1 \ni \boldsymbol{x} \mapsto x_2 \in (-1,1), \quad & \phi_2 &= U_2 \supset \boldsymbol{x} \to x_2 \in (-1,1), \\ \phi_3 &= U_3 \ni \boldsymbol{x} \mapsto x_1 \in (-1,1), \quad & \phi_4 &= U_4 \supset \boldsymbol{x} \to x_1 \in (-1,1). \end{aligned}$$

Clearly, each point $p \in S^1$ belongs to at least one set U_j, $j = 1, \ldots, 4$, and further the maps ϕ_j are diffeomorphism since they are bijective and, e.g., $\phi_1^{-1} : (-1,1) \to U_1$ is defined as $\phi_1^{-1}(x_2) = (\sqrt{1 - x_2^2}, x_2^2)$ and it is differentiable in the interval $(-1,1)$. Finally, one may check, e.g., that $U_1 \cap U_3 = \left\{\boldsymbol{x} \in \mathbb{R}^2 : 0 < x_1 < 1, x_2 = \sqrt{1 - x_1^2}\right\}$, and $\phi_3 \circ \phi_1^{-1} : (0,1) \ni x_1 \to \sqrt{1 - x_1^2}$ is a diffeomorphism of the unit invertal into itself. Hence we conclude that S^1 is a one-dimensional manifold.

Definition 2.1 relies on open sets and coordinate charts to define a manifold. Therefore, to check whether the unit circle is a manifold in Example 2.2, we had to come up with an explicit family of open sets and coordinate charts. In view of most common geometric surfaces and curves for which we have parametric and implicit representations, we may wonder if we could have an implicit definition of a manifold, thus avoiding the need to specify local parametrizations/coordinate charts. The next theorem provides a positive answer to this question.

THEOREM 2.3 (Implicit definition of a manifold, Prop. 18.7 [12]). *Let $h : \mathbb{R}^{n+d} \supset \Omega \to \mathbb{R}^d$ be a differentiable map and $c \in \mathbb{R}^d$ such that the differential $Dh(p) \in \mathbb{R}^{d\times(n+d)}$ has maximum rank for every $p \in \Omega$ with $h(p) = c$. Then, the preimage*

$$\mathcal{M} := h^{-1}(c) = \{p \in \Omega : h(p) = c\}$$

is a d-dimensional differentiable manifold of $\mathbb{R}^{n+d}$.

Theorem 2.3 allows us to check more easily that given sets have the structure of a differentiable manifold. Here are two classical examples.

EXAMPLE 2.4 (The unit circle revisited). *Let $h : \mathbb{R}^2 \to \mathbb{R}$ defined by $h(\boldsymbol{x}) = x_1^2 + x_2^2$. Obviously, $S^1 = h^{-1}(1)$. Further, h is differentiable and $Dh(\boldsymbol{x}) = (2x_1, 2x_2) \in \mathbb{R}^{1\times 2}$ is different from zero for every $\boldsymbol{x} \in S^1$. Hence, S^1 is a differential manifold.*

EXAMPLE 2.5 (The Stiefel manifold). *Let us consider the set of orthonormal matrices*

$$St(n,k) := \left\{X \in \mathbb{R}^{n\times k} \ \ \text{such that } X^\top X = I_k\right\}. \tag{2.2}$$

The set $St(n,k)$ coincides with the preimage of the null matrix through the map $F : \mathbb{R}^{n\times k} \to Sym(k)$ with $F(X) = X^\top X - I_k$, where $Sym(k) = \left\{A \in \mathbb{R}^{k\times k} : \ A = A^\top\right\}$ is the set of symmetric matrices. To show that $St(n,k)$ is a manifold, we show that the differential of F has full rank, which equivalently means that for every $\widetilde{V} \in Sym(k)$, there is a $V \in \mathbb{R}^{n\times k}$ such that $DF_X[V] = \widetilde{V}$. A direct calculation shows that $DF_X[V] = X^\top V + V^\top X$. It is then sufficient to choose $V = \frac{1}{2}X\widetilde{V}$ to verify

$$DF_X[V] = \frac{1}{2}X^\top X\widetilde{V} + \frac{1}{2}\widetilde{V}^\top X^\top X = \widetilde{V}.$$

Hence, we conclude that $St(n,k)$ is a manifold and it is called the Stiefel manifold.

We have previously mentioned that differentiable manifolds locally "look like" an Euclidean space. As a matter of fact, to each point p we may associate a *linear* vector space which locally approximates the manifold. This is the same concept as the tangent line to a curve in a specific point, which is nothing more than a linear vector space (spanned by the tangent vector) that locally approximates it. To do so, we consider all the smooth curves that lie entirely on $\mathcal{M}$ and pass through a given point p. The tangent space of $\mathcal{M}$ at p is defined as the set of all possible velocities (derivatives) of these curves.

Definition 2.6 (Tangent plane and tangent vectors). *Let $\mathcal{M}$ be a differentiable manifold and $I \subset \mathbb{R}$ an open set containing 0. For all $p \in \mathcal{M}$, the set*

$$T_p\mathcal{M} := \{\dot{c}(0) |\ c : I \to \mathcal{M} \text{ is smooth and } c(0) = p\} \tag{2.3}$$

is called tangent space of $\mathcal{M}$ at p. That is, $v \in T_p\mathcal{M}$ if and only if there exists a curve c on $\mathcal{M}$ that passes through p with velocity $v = \dot{c}(0)$. Elements of $T_p\mathcal{M}$ are called tangent vectors.

The next example shows how to use Definition 2.6 to calculate tangent spaces.

Example 2.7 (Tangent spaces to the unit circle and Stiefel manifold). *Let S^1 be the unit circle and $x(t) : (a,b) \to S^1$ a curve on the manifold such that $x(0) = \boldsymbol{x}$ for a fixed $\boldsymbol{x} \in S^1$. Since $x(t) \in S^1$ for every $t \in (a,b)$, it must hold $x_1(t)^2 + x_2(t)^2 = 1$. Taking the derivative with respect to t of this expression leads to*

$$0 = \frac{d}{dt}_{|t=0} 1 = \frac{d}{dt}_{|t=0} \left(x_1(t)^2 + x_2(t)^2\right) = 2\left(x_1(0)\dot{x}_1(0) + x_2(0)\dot{x}_2(0)\right) = 2\boldsymbol{x}^\top \dot{\boldsymbol{x}}.$$

Hence, the velocity $\boldsymbol{v}$ of each curve lying on S^1 and passing through x must satisfy $\boldsymbol{x}^\top\boldsymbol{v} = 0$, i.e. $T_xS^1 := \left\{\boldsymbol{v} \in \mathbb{R}^2 : \boldsymbol{x}^\top \boldsymbol{v} = 0\right\}$. The reader can easily check that T_xS^1 indeed coincides with the direction tangent to the unit circle S^1 at point $\boldsymbol{x}$.
Next, let $St(n,k)$ be the Stiefel manifold defined in (2.2), and $X(t)$ a curve lying on $St(n,k)$ with $X(0) = X$. The curve $X(t)$ must satisfy

$$0 = \frac{d}{dt}_{|t=0} I_k = \frac{d}{dt}_{|t=0} X(t)^\top X(t) = \dot{X}(0)^\top X(0) + X(0)^\top \dot{X}(0).$$

Therefore, $T_X St(n,k) = \left\{V \in \mathbb{R}^{n\times k} : V^\top X + X^\top V = 0\right\}$.

Direct calculations allow one to verify in these two examples that the tangent space is indeed a linear vector space. However, it is not clear at all from (2.3) that this property should hold more generally. Fortunately, it is possible to characterize the tangent space of an implicitely defined manifold as the kernel of the smooth defining function h. The kernel being a subspace, we then conclude that the tangent space is indeed a linear vector space.

THEOREM 2.8 (Characterization of the tangent space - Proposition 5.38 in [21]). *Let $h : \mathbb{R}^{n+d} \supset \Omega \to \mathbb{R}^d$ be a differentiable map, $c \in \mathbb{R}^d$ and $\mathcal{M} := h^{-1}(c)$ be a differentiable manifold. Then given a $p \in \mathcal{M}$, $T_p\mathcal{M} = KerDh_p$.*

The last concept we introduce is that of a distance between two points on a manifold. Let us first review the Euclidean case. Let $\boldsymbol{x}$ and $\boldsymbol{y}$ be two points on $\mathbb{R}^2$, and $c : [0,1] \to \mathcal{M}$ a smooth curve such that $c(0) = \boldsymbol{x}$ and $c(1) = \boldsymbol{y}$. Letting $\mathcal{L}(t) := \|\dot{c}(t)\|_2$, the length of the curve is defined as

$$L(c) := \int_0^1 \mathcal{L}(t)\, dt. \tag{2.4}$$

The distance between two points in $\mathbb{R}^2$ is then equal to length of the shortest curve c starting at $\boldsymbol{x}$ and ending at $\boldsymbol{y}$, namely

$$d(\boldsymbol{x}, \boldsymbol{y}) := \inf_{c:[0,1]\to\mathbb{R}^2 : c(0)=\boldsymbol{x},\ c(1)=\mathbf{y}} L(c). \tag{2.5}$$

The shortest curve between two points is called *geodesic*. In practice, the minimization problem (2.5) can be solved using the Euler-Lagrange equations,

$$\frac{d}{dt}\frac{d\mathcal{L}}{d\dot{c}_i} - \frac{d\mathcal{L}}{dc_i} = 0, \quad i = 1, 2. \tag{2.6}$$

A direct calculation shows that a minimizer is $c^\star(t) = (x_1 + t(y_1 - x_1), x_2 + t(y_2 - x_2))$, so that we recover the standard Euclidean distance $d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(y_1 - x_1)^2 + (y_2 - x_2)^2}$. When considering manifolds, it possible to define a distance between two points on a manifold by extending the construction just recalled. To do so, one needs first to introduce an appropriate norm for the vectors $\{\dot{c}(t)\}_{t\in(0,1)}$, which are now elements of the tangent space. The so-called Riemannian metric on $\mathcal{M}$ is a smooth[1] family of scalar products $(\langle\cdot,\cdot\rangle_p)_{p\in\mathcal{M}}$, $\langle\cdot,\cdot\rangle_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$, in order that for every $v \in T_p\mathcal{M}$, $\|v\| := \sqrt{\langle v, v\rangle_p}$. Consequently, the length of a curve c on the manifold is given by $L(c) := \int_0^1 \sqrt{\langle \dot{c}(t), \dot{c}(t)\rangle_{c(t)}}\, dt$, and the distance between two points $p, q \in \mathcal{M}$ is given by

$$d_{\mathcal{M}}(p, q) = \inf_{c:[0,1]\to\mathbb{R}^2 : c(0)=p,\ c(1)=q} \int_0^1 \sqrt{\langle \dot{c}(t), \dot{c}(t)\rangle_{c(t)}}\, dt.$$

A curve attaining the minimum, if it exists, is called *geodesic*.

3. The Grassmann manifold.

In this section, we focus on the Grassmann manifold and we introduce the main tools that will be needed to interpolate on such a manifold.

DEFINITION 3.1 (Grassmann manifold). *The Grassmann manifold is defined as the set of all k-dimensional subspaces of $\mathbb{R}^n$,*

$$Gr(n, k) := \{\mathcal{V} \subset \mathbb{R}^n :\ \mathcal{V} \text{ is a subspace}, dim(\mathcal{V}) = k\}.$$

There are several ways to represent an element $\mathcal{V}$ of the Grassmann manifold. A first approach uses a matrix $U \in \mathbb{R}^{n\times k}$ whose columns form a basis of $\mathcal{V}$, i.e. $\mathcal{V} = \text{span}(U)$.

[1] We will not dive into what "smooth" means here. The interested reader is refereed to [6, Definitions 3.44 and 3.52].

However, there are obviously several different matrices representing the same subspace $\mathcal{V}$. Therefore the latter is identified with the equivalence class

$$[U] := \left\{M \in \mathbb{R}^{n\times k} : \ M = UK, \ K \in \mathbb{R}^{k\times k} \text{ invertible}\right\},$$

containing all matrices of rank k whose columns span $\mathcal{V}$.
A second way associates $\mathcal{V}$ with one of its orthonormal bases, $\mathcal{V} = \text{span}(Q)$ with $Q \in \mathbb{R}^{n\times k}$ and $Q^\top Q = I_k$. Since again we have several different orthonormal bases, $\mathcal{V}$ is more correctly identified with the equivalence class

$$[Q] := \left\{U \in \mathbb{R}^{n\times k} : \ U = QK, \ K \in \mathbb{R}^{k\times k}, \ K^\top K = I_k\right\} \tag{3.1}$$

containing all (equivalent) orthonormal bases of $\mathcal{V}$. We therefore have the identification $\text{Gr}(n,k) = \{[Q] : Q \in \text{St}(n,k)\}$.
To eliminate any equivalence relation, a third approach identifies $\mathcal{V}$ with the unique orthogonal projector $P : \mathbb{R}^{n\times n} \to \mathcal{V}$, defined by $P = QQ^\top$, with Q being an orthonormal matrix. This representation is unique since if Q_1 is another orthonormal matrix in $[Q]$, we have $Q_1 = QK$, with $K \in \mathbb{R}^{k\times k}$ and $K^\top K = KK^\top = I_k$. Therefore, $Q_1 Q_1^\top = QKK^\top Q^\top = QQ^\top = P$. From the numerical point of view, it is quite convenient to represent $\mathcal{V}$ through an orthonormal representative matrix $V \in \text{St}(n,k)$ whose columns span $\mathcal{V}$, and this is what we will do from now on. To refer to a point on the manifold, we may use interchangeably $\mathcal{V}$ or $[V]$. We will use the second notation when we want to stress the matrix representative used.
The careful reader may have observed that it is definitely not clear at a first sight why the set $\text{Gr}(n,k)$ should have the structure of a differentiable manifold. As a matter of fact, this is quite technical, and we refer the interested reader to, e.g., [21, Examples 1.36 and 21.21] and [6, Chapter 9]. As a pure intuitive movitation, we notice from the second representation that $\text{Gr}(n,k)$ can be seen as a quotient set of $\text{St}(n,k)$ with respect to the equivalence relation defining the equivalence classes in (3.1), namely $Q, Q_1 \in \text{St}(n,k)$ satisfy

$$Q \sim Q_1 \text{ if and only if } \exists K \in \mathbb{R}^{k\times k}, \ K^\top K = I_k, \text{ s.t. } Q = Q_1 K.$$

It is then possible to prove that $\text{Gr}(n,k)$ inherits the structure of a smooth manifold from $\text{St}(n,k)$ ($\text{Gr}(n,k)$ is a quotient manifold) [21, Theorem 21.10]. Furthermore, for any matrix representative $V \in \text{St}(n,k)$ of $\mathcal{V} \in \text{Gr}(n,k)$, the tangent space of $\text{Gr}(n,k)$ at $\mathcal{V}$ is given by [6, Example 9.26],

$$T_{\mathcal{V}}\text{Gr}(n,k) := \left\{\Delta \in \mathbb{R}^{n\times k} \mid V^\top \Delta = 0\right\} \subset \mathbb{R}^{n\times k}. \tag{3.2}$$

Note that the tangent space does not depend on the representative V. Let V, V_1 be two members of the same equivalence class and $\Delta \in T_{[V]}\text{Gr}(n,k)$. Then $0 = V^\top \Delta = K^\top V_1^\top \Delta$ which implies, K being orthonormal, that $V_1^\top \Delta = 0$, and thus $\Delta \in T_{[V_1]}\text{Gr}(n,k)$. The other inclusion can be shown similarly.
It is also possible to define a Riemannian metric on $\text{Gr}(n,k)$. Let Δ and $\widetilde{\Delta}$ be two elements of the tangent space $T_{\mathcal{V}}\text{Gr}(n,k)$, and $V \in \text{St}(n,k)$ an orthonormal representative of $\mathcal{V}$. Then the Riemannian metric ([1, Section 3.3]) on the tangent space $T_{\mathcal{V}}\text{Gr}(n,k)$ is

$$\langle \Delta, \widetilde{\Delta}\rangle_{\mathcal{V}} := \text{Tr}\left((V^\top V)^{-1}\Delta^\top \widetilde{\Delta}\right), \tag{3.3}$$

where Tr denotes the trace operator. Note that, again, $\langle\cdot,\cdot\rangle_{\mathcal{V}}$ does not depend on the choice of the basis V for $\mathcal{V}$.

Next, let $\mathcal{Y}(t) \in \mathrm{Gr}(n,k)$ be a curve on the Grassmann manifold. At each time t, a point on the curve $\mathcal{Y}(t)$ is represented by an orthonormal matrix $Y(t) \in \mathrm{St}(n,k)$. Due to the independence of the Riemannian metric on the choice of the basis, to study the properties of $\mathcal{Y}(t)$ we can restrict ourselves to analyze $Y(t)$, whose length is given by the expression

$$L(\mathcal{Y}) = \int_0^1 \mathrm{Tr}\left((Y(t)^\top Y(t))^{-1} \dot{Y}(t)^\top \dot{Y}(t) \right) = \int_0^1 \mathcal{L}(Y(t), \dot{Y}(t))\, dt. \tag{3.4}$$

The interpolation algorithms discussed in Section 4 all depend on the concept of geodesic between two points on the manifold $\mathrm{Gr}(n,k)$. We therefore close this section by carefully characterizing a minimizer of (3.4) using the Euler-Lagrange equations.

Proposition 3.2 (Characterization of the geodesics). *If $Y : [0,1] \ni t \to Y(t) \in$ St(n,k) is a curve minimizing* (3.4)*, then $Y(t)$ satisfies the differential equation*

$$\ddot{Y}(t) - Y(t)\dot{Y}(t)^\top \dot{Y}(t) = 0, \quad \forall t \in [0,1]. \tag{3.5}$$

Proof. The minimizers of the functional (3.4) can be computed solving the Euler-Lagrangian equations

$$\frac{\partial \mathcal{L}}{\partial Y} - \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{Y}} = 0, \tag{3.6}$$

where

$$\begin{aligned}\mathcal{L}(Y,\dot{Y}) = \mathrm{Tr}\left((Y(t)^\top Y(t))^{-1} \dot{Y}(t)^\top \dot{Y}(t) \right) &= \mathrm{Tr}\left(\dot{Y}(t) (Y(t)^\top Y(t))^{-1} \dot{Y}(t)^\top \right) \\ &= \langle \dot{Y}(t)^\top, (Y(t)^\top Y(t))^{-1} \dot{Y}(t)^\top \rangle_F,\end{aligned}$$

and $\langle \cdot,\cdot \rangle_F$ denotes the Frobenius scalar product between matrices. Computing the directional derivatives with respect to $\dot{Y}$ we find

$$\begin{aligned}\frac{\partial L}{\partial \dot{Y}}[\delta \dot{Y}] = \langle \delta\dot{Y}^\top, (Y^\top Y)^{-1}\dot{Y}^\top\rangle_F + \langle \dot{Y}^\top, (Y^\top Y)^{-1}\delta\dot{Y}^\top\rangle_F &= 2\langle \delta\dot{Y}^\top, (Y^\top Y)^{-1}\dot{Y}^\top\rangle_F \\ &= 2\langle \dot{Y}(Y^\top Y)^{-1}, \delta\dot{Y}\rangle_F,\end{aligned}$$

and thus we conclude that $\frac{\partial L}{\partial \dot{Y}} = 2\dot{Y}(Y^\top Y)^{-1}$. Further,

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{Y}} = \frac{d}{dt} 2\dot{Y}(Y^\top Y)^{-1} = 2\ddot{Y}(Y^\top Y)^{-1} - 2\dot{Y}(Y^\top Y)^{-1}(\dot{Y}^\top Y + Y^\top \dot{Y})(Y^\top Y)^{-1}, \tag{3.7}$$

where in the last step we used [22, Formula 59]

$$\frac{d}{dt}(Y^\top Y)^{-1} = -(Y^\top Y)^{-1}(\dot{Y}^\top Y + Y^\top \dot{Y})(Y^\top Y)^{-1}.$$

For the directional derivative with respect to y, we recall the derivation formula [22, Formula 63]

$$\frac{\partial \mathrm{Tr}\left(AX^{-1}B\right)}{\partial X} = -(X^{-1}BAX^{-1})^\top.$$

Setting in our case $A = I_k$, I_k being the identity matrix in $\mathbb{R}^{k\times k}$, $X = Y^\top Y$, $B = \dot{Y}^\top \dot{Y}$, using the chain rule and the properties of the trace operator, we obtain for every matrix pertubation $\delta Y \in \mathbb{R}^{n\times k}$ that

$$\begin{aligned}\frac{\partial L}{\partial Y}[\delta Y] &= \frac{\partial L}{\partial X}\frac{\partial X}{\partial Y}[\delta Y] = -\langle (Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1}), Y^\top \delta Y + \delta Y^\top Y\rangle_F \\ &= -\mathrm{Tr}\left(\left[(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1})\right]^\top Y^\top \delta Y\right) \\ &\quad - \mathrm{Tr}\left(\left[(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1})\right]^\top \delta Y^\top Y\right) \\ &= -\langle Y(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1}), \delta Y\rangle \\ &\quad - \langle \delta Y, Y\left[(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1})\right]^\top\rangle,\end{aligned}$$

and hence

$$\frac{\partial L}{\partial Y} = -Y(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1}) - Y\left[(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1})\right]^\top. \tag{3.8}$$

Considering together (3.7) and (3.8), the Euler-Lagrange equations (3.6) read

$$\begin{aligned}&2\ddot{Y}(Y^\top Y)^{-1} - 2\dot{Y}(Y^\top Y)^{-1}(\dot{Y}^\top Y + Y^\top \dot{Y})(Y^\top Y)^{-1} \\ &\qquad + Y(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1} + Y\left[(Y^\top Y)^{-1}\dot{Y}^\top \dot{Y}(Y^\top Y)^{-1}\right]^\top = 0.\end{aligned}$$

To simplify the expression, we use that $Y(t) \in \mathrm{St}(n,k)$ at all times, thus $Y(t)^\top Y(t) = I_k$, and the property of elements of the tangent space (see (3.2)), that is $\dot{Y}(t)^\top Y(t) = 0$ $\forall t \in [0,1]$, to obtain

$$\ddot{Y}(t) + Y(t)\dot{Y}(t)^\top \dot{Y}(t) = 0, \quad t \in [0,1]. \qquad \square$$

3.1. Exponential and logarithmic maps. Equation (3.5) characterizes the curves which are minimizers of the length functional (3.4). To compute a geodesic between two points on $\mathrm{Gr}(n,k)$, we should solve (3.5) and specify the initial and final values $Y(0) = Y_0$ and $Y(1) = Y_1$, where $Y_0, Y_1 \in \mathrm{St}(n,k)$ span the initial and final subspaces $\mathcal{Y}_0$ and $\mathcal{Y}_1$. This procedure corresponds to solve a classical second-order *boundary value* problem.
On the other hand, we could solve (3.5) by specifying an initial value Y_0 and an *initial velocity* $\dot{Y} \in T_{[Y_0]}\mathrm{Gr}(n,k)$. This corresponds to transforming a boundary value problem into an *initial value* problem, much in the spirit of shooting methods [17]. Associated to these two different interpretations there are two maps which we discuss in the following: the logarithmic map and the exponential map.[2]
Given an initial point $\mathcal{Y}_0 \in \mathrm{Gr}(n,k)$, represented by $Y_0 \in \mathrm{St}(n,k)$, the exponential map is a function that to every tangent vector $\dot{Y} \in T_{[Y_0]}\mathrm{Gr}(n,k)$ associates the final point $\mathcal{Y}_1 := \mathrm{span}(Y_{Y_0,\dot{Y}}(1))$, where $Y_{Y_0,\dot{Y}}$ is a solution to the geodesic equation that starts from $[Y_0]$ and has an initial velocity $\dot{Y}$. In symbols,

$$\mathrm{Exp}_{[Y_0]} : T_{[Y_0]}\mathcal{M} \to \mathrm{Gr}(n,k), \quad \dot{Y} \mapsto \mathrm{span}(Y_{Y_0,\dot{Y}}(1)).$$

[2]These two maps can be generally defined on every differentiable manifold. We introduce them directly for the Grassmann manifold for the sake of simplicity.

Some authors require in the definition of the exponential map that the tangent vector is sufficiently small. This is due to the fact that if the tangent vector is sufficiently small, then $Y_{Y_0,\dot{Y}}(t)$ is actually the curve of minimum distance between $\mathcal{Y}_0$ and $\mathcal{Y}_1$. This may not always be true. Consider for instance the earth as a manifold and start from the north pole with an initial direction that makes you pass through Geneva. Clearly, by rescaling the initial velocity you may end up at time $t = 1$ exactly in Geneva, or, if the initial velocity is too large, in the middle of the Pacific ocean while passing through the south pole. The latter is clearly not the shortest path between the north pole and that point in the Pacific ocean.
We explicitely derive a closed formula for the exponential map in the following proposition.

Proposition 3.3 (Closed formula for the exponential map). *Let $\mathcal{Y}_0 \in Gr(n,k)$, represented by $Y_0 \in St(n,k)$, be an initial value on the Grassmann manifold and $\dot{Y} \in T_{[Y_0]}Gr(n,k)$ an initial velocity whose thin Singular Value Decomposition (SVD) is $[U,\Sigma,V] = svd(\dot{Y})$, with $U \in \mathbb{R}^{n\times k}$, $\Sigma \in \mathbb{R}^{k\times k}$ and $V \in \mathbb{R}^{k\times k}$. Then,*

$$Exp_{[Y_0]}(\dot{Y}) = span\left(Y_0 V \cos(\Sigma) + U \sin(\Sigma)\right),$$

where the functions $\cos(\cdot)$ *and* $\sin(\cdot)$ *act component-wise on the diagonal of* Σ.

Proof. Let us first observe that if $Y(t)$ is a solution of (3.5) then

$$\frac{d}{dt}\left(\dot{Y}(t)^\top \dot{Y}(t)\right) = \ddot{Y}(t)^\top \dot{Y}(t) + \dot{Y}(t)^\top \ddot{Y}(t) \tag{3.9}$$

$$= \left(\dot{Y}(t)^\top \dot{Y}(t) Y(t)^\top\right) \dot{Y}(t) + \dot{Y}(t)^\top \left(Y(t) \dot{Y}(t)^\top \dot{Y}(t)\right) = 0, \tag{3.10}$$

where we used the orthogonality property $\dot{Y}(t)^\top Y(t) = Y(t)^\top \dot{Y}(t) = 0$ of tangent vectors. Equation (3.9) implies that $\dot{Y}(t)^\top \dot{Y}(t) = \dot{Y}(0)^\top \dot{Y}(0)$ for every t. We then introduce a singular value decomposition of $\dot{Y}$ as $[U,\Sigma,V] = \text{svd}\left(\dot{Y}\right)$ which, using (3.9), implies $\dot{Y}(t)^\top \dot{Y}(t) = V\Sigma^2 V^\top$. Replacing this term into (3.5), multiplying from the right by V and defining $\Phi(t) = Y(t)V$ leads to

$$\ddot{\Phi}(t) + \Phi(t)\Sigma^2 = 0. \tag{3.11}$$

Σ^2 being diagonal, (3.11) represents a decoupled system of equations for the k columns of $\Phi(t) = [\phi_1(t)|\phi_2(t)|\ldots|\phi_k(t)] \in \mathbb{R}^{n\times k}$. Each equation is of the form $\ddot{\phi}_j(t) + \sigma_j^2 \phi_j(t) = 0$, σ_j being the singular values of $\dot{Y}$. The overall solution is then

$$\Phi(t) = A\cos(t\Sigma) + B\sin(t\Sigma),$$

or alternatively formulated in terms of Y,

$$Y(t)V = A\cos(t\Sigma) + B\sin(t\Sigma),$$

where $A, B \in \mathbb{R}^{n\times k}$. These two constant matrices are determined by the initial conditions. Evaluating at time $t = 0$, we find straightforwardly $A = Y_0 V$. Computing the derivative at time $t = 0$ gives

$$\dot{Y}(0)V = B\cos(t\Sigma)\Sigma_{|t=0} = B\Sigma \implies B = \dot{Y}(0)V\Sigma^{-1} = U\Sigma V^\top V \Sigma^{-1} = U.$$

Thus,

$$Y(t) = Y_0 V \cos(t\Sigma) V^\top + U \sin(t\Sigma) V^\top, \tag{3.12}$$

which, V being an orthonormal matrix, allows us to conclude that

$$\operatorname{span}(Y(t)) = \operatorname{span}(Y_0 V \cos(t\Sigma) + U \sin(t\Sigma)). \tag{3.13}$$

Evaluating (3.13) at time $t = 1$ shows the claim. □

Proposition 3.3 provides an easy way to compute the exponential map on the Grassmann manifold. The procedure can be summarized in the very short Matlab function:

```
function [Y]=exp_map(Y0,Ydot)
% EXP_MAP computes the exponential map on the Grassman manifold.
% Y=exp_map(Y0,Ydot) computes the matrix representative Y in St(n,k) of
% Exp_{[Y0]}(Ydot) in Gr(n,k), given a point Y0 in St(n,k) and a tangent
% vector Ydot in T_{[Y0]}Gr(n,k).

[U,Sigma,V]=svd(Ydot,'econ');
Y=Y0*V*diag(cos(diag(Sigma)))+U*diag(sin(diag(Sigma)));
```

Remark 3.4. Note that the closed formula (3.13) permits further to define the map $\gamma : (0,1] \ni t \to Y_{Y_0,\dot{Y}}(t)$, that is, instead of returning the final value at $t = 1$ of the geodesic starting from $\mathcal{Y}_0$ with velocity $\dot{Y}$, we can return any intermediate point that lies on such geodesic. Since $t\dot{Y}$ has an SVD equal to $[U, t\Sigma, V] = \operatorname{svd}\left(\dot{Y}\right)$, it follows that $\gamma(t) \equiv \operatorname{Exp}_{[Y_0]}\left(t\dot{Y}\right)$. This remark will be important in Section 4.

Remark 3.5 (On the name exponential map). To give an intuition on the origins of the name exponential map, let us consider the unit circle. Any geodesic on the circle is an arc since no other trajectories are possible. An arc on the circle starting from y_0 and with direction $\dot{y}_0$ can be written in parametric form as

$$y(t) = \cos(t)y_0 + \sin(t)\dot{y}_0. \tag{3.14}$$

Direct calculations show that $\dot{y}_0$ can be expressed as $\dot{y}_0 = Ay_0$, and that $\exp(tA) = \cos(t)I + \sin(t)A$, with the skew-symmetric matrix $A = [0, 1; -1, 0]$. Using this relation in (3.14), the matrix exponential arises,

$$y(t) = \cos(t)y_0 + \sin(t)Ay_0 = \exp(tA)y_0.$$

Let us now show that the matrix exponential satisfies the geodesic equation also in the general case of a Stiefel manifold with $k = n$. First, note that for a skew-symmetric matrix A, $Y(t) = \exp(tA)$ is orthogonal. Inserting the expression of $Y(t)$ into the geodesic formula, we obtain indeed

$$\begin{aligned}
\ddot{Y}(t) + Y(t)\dot{Y}(t)^T\dot{Y}(t) &= \exp(tA)A^2 + \exp(tA)A^\top \exp(tA)^\top \exp(tA)A \\
&= \exp(tA)A^2 + \exp(tA)A^\top A \\
&= \exp(tA)(A^2 + A^\top A) \\
&= \exp(tA)(A^2 - A^2) = 0,
\end{aligned}$$

where we used the properties of the matrix exponential and the skew-symmetry of A. More deeply, the exponential map name originates in the context of Lie groups and Lie algebras. For more details, we refer the interested reader to [21, Chapter 20].

We next consider the logarithm map on the Grassmann manifold. Given a base point $\mathcal{Y}_0 \in \operatorname{Gr}(n,k)$ the logarithmic maps receives a second point $\mathcal{Y}_1 \in \operatorname{Gr}(n,k)$ and returns the initial velocity $\dot{Y} \in T_{[Y_0]}\mathrm{Gr}(n,k)$ of the geodesic that starts from $\mathcal{Y}_0$ and arrives in $\mathcal{Y}_1$ at time $t=1$. In other words, the logarithmic map is the inverse of the exponential map, in the sense that if $\dot{Y} = \mathrm{Log}_{[Y_0]}([Y_1])$ then $\mathrm{Exp}_{[Y_0]}(\dot{Y}) = [Y_1]$. Note however that the logaritmic map is not always defined. As an example, consider the earth as a manifold (a sphere). Then, if $\mathcal{Y}_0$ and $\mathcal{Y}_1$ are the north and south pole, then there is not a unique geodesic connecting the two points (every terrestrial meridian is a geodesic), and thus the logaritmic map is not defined. It is however sufficient to restrict the domain of the logarithmic map to a sufficiently small neighbourhood of the initial point $\mathcal{Y}_0$.

Proposition 3.6 (Closed formula for the logarithmic map). *Let $\mathcal{Y}_0$ and $\mathcal{Y}_1$ be two points on $Gr(n,k)$ with Stiefel representatives Y_0, $Y_1 \in St(n,k)$. Assume that $M := Y_0^\top Y_1$ is invertible, and let $[R, S, \widetilde{Q}] = svd(M)$, and $L := (I - Y_0 Y_0^\top) Y_1 M^{-1}$, with $[\widehat{Q}, \widehat{S}, \widehat{R}] = svd(L)$. Further, let $\Sigma = atan\left(\widehat{S}\right)$. Then,*

$$Log_{[Y_0]}([Y_1]) = \widehat{Q}\Sigma R \in T_{[Y_0]}\mathcal{M}. \tag{3.15}$$

Proof. To compute the logarithmic map we do not solve directly the boundary value problem (3.5) with the appropriate boundary conditions, but we proceed with an algebraic derivation. We start with the calculation

$$Y_1 M^{-1} = (Y_0 Y_0^\top) Y_1 M^{-1} + \underbrace{(I - Y_0 Y_0^\top) Y_1 M^{-1}}_{L} = Y_0 + L,$$

which implies

$$Y_1 = Y_0 M + LM. \tag{3.16}$$

Note that (3.16) expresses the final point Y_1 in terms of Y_0, M and L. The rest of the proof consists in manipulating these three terms to get a formula similar to (3.13): A direct calculation shows that

$$\begin{aligned} L^\top L &= M^{-\top} Y_1^\top (I - Y_0 Y_0^\top)(I - Y_0 Y_0^\top) Y_1 M^{-1} \\ &= M^{-\top} Y_1^\top (I - Y_0 Y_0^\top) Y_1 M^{-1} \\ &= M^{-\top} M^{-1} - I, \end{aligned} \tag{3.17}$$

where we used that $Y_1^\top Y_1 = I_k$ since $Y_1 \in \mathrm{St}(n,k)$. Let $[R, S, \widetilde{Q}] = \mathrm{svd}(M)$ and $[\widehat{Q}, \widehat{S}, \widehat{R}] = \mathrm{svd}(L)$ be the SVDs of M and L. On the one hand, Eq (3.17) implies that $L^\top L = R(S^{-2} - I)R^\top$. On the other hand, $L^\top L = \widehat{R}\widehat{S}^2\widehat{R}^\top$ due to the thin SVD of L. Therefore, $\widehat{R} = R$ and $\widehat{S} = \sqrt{S^{-2} - I}$. We next define $\Sigma := \mathrm{atan}(\widehat{S})$ which implies $\widehat{S}^2 = \tanh \Sigma^2 = I - S^{-2}$, and thus $S = \cos(\Sigma)$ and $\widehat{S}S = \sin(\Sigma)$. Inserting the two SVDs of M and L and the two expressions for S and $\widehat{S}S$ into (3.16) we obtain

$$Y_1 = Y_0 R \cos(\Sigma) \widetilde{Q}^\top + \widehat{Q} \sin(\Sigma) \widetilde{Q}^\top. \tag{3.18}$$

Since $\widetilde{Q} \in \mathbb{R}^{k \times k}$ is an orthonormal matrix, we can neglect the two matrices $\widetilde{Q}^\top$ multiplying from the right in (3.18) when looking at the subspaces generated by the columns, namely,

$$\mathrm{span}\,(Y_1) = \mathrm{span}\left(Y_0 R \cos(\Sigma) + \widehat{Q}\sin(\Sigma)\right). \tag{3.19}$$

Note that (3.19) is exactly of the form (3.13) with $t = 1$. Thus, we finally deduce that the tangent vector is $\dot{Y} = \widehat{Q}\Sigma R$. ∎

A numerical procedure to compute the logarithmic map is described by the function `log_map_v0`. Note that, in constrast with the proof of Proposition 3.6, where the SVD of M is needed to derive (3.18), in practice we only need to perform one SVD, that of the matrix L, to get all quantities needed to compute the tangent vector.

```
function [Ydot]=log_map_v0(Y0,Y1)
% LOG_MAP_V0 computes the logarithmic map on the Grassman manifold.
% Ydot=log_map_v0(Y0,Y1) receives the matrix representatives Y0,Y1
% in St(n,k) of two points on G(n,k), and computes the matrix
% representative Ydot in St(n,k) such that [Exp_{Y0}(Ydot)]=[Y1]

[U,Sigma,V]=svd((eye(size(Y0,1))-Y0*Y0')*Y1/(Y0'*Y1),'econ');
Theta=diag(atan(diag(Sigma)));
Ydot=U*Theta*V';
```

Nevertheless, the function `log_map_v0` has two drawbacks. First, despite it correctly computes the logarithmic map on the Grassmann manifold, in the sense that $\text{Exp}_{\mathcal{Y}_0}(\text{Log}_{\mathcal{Y}_0}(\mathcal{Y}_1)) = \mathcal{Y}_1$, it does not necessarily preserve its Stiefel representative, that is,

$$\text{Exp}_{[Y_0]}\left(\text{Log}_{[Y_0]}([Y_1])\right) = [\widetilde{Y}_1],$$

where $\widetilde{Y}_1$ spans the same subspace of Y_1 but may be in a different basis. This can be seen as follows: Let $[\widehat{Q},\widehat{S},\widehat{R}]$ be the SVD of the tangent vector $\dot{Y}$ output of `log_map_v0`, then the exponential map returns the Stiefel representative

$$\widetilde{Y}_1 = Y_0\widehat{R}\cos(\Sigma)\widehat{R}^\top + \widehat{Q}\sin(\Sigma)\widehat{R}^\top,$$

which indeed spans the same subspace as Y_1 in (3.18), but it is a different basis since on the right-hand side it is multiplied by $\widehat{R}^\top$ and not by $\widetilde{Q}^\top$. The second drawback is that it requires M to be invertible which is an extra hypothesis that is actually not needed.

Recently, a different algorithm has been proposed in [29] that overcomes these two problems. The algorithm essentially computes a new representative $\widetilde{Y}_1$ of $\mathcal{Y}_1$, depending on the representative of the initial condition Y_0. This is done by solving a Procrustes problem [24] through the computation of the SVD of the matrix $Y_1^\top Y_0$, which we denoted with M in Proposition 3.6. This matrix does not need to be invertible, but only the calculation of its SVD is needed.

Proposition 3.7 (A second closed formula for the logarithmic map). *Let $\mathcal{Y}_0$ and $\mathcal{Y}_1$ be two points on $Gr(n,k)$ with Stiefel representatives Y_0, $Y_1 \in St(n,k)$. Let $[Q,S,R] = svd(Y_1^\top Y_0)$, $\widetilde{Y}_1 = Y_1QR^\top$, $L = (I - Y_0Y_0^\top)\widetilde{Y}_1$ with $[\widehat{Q},\widehat{S},\widehat{R}] = svd(L)$. Then,*

$$Log_{\mathcal{Y}_0}(\mathcal{Y}_1) = \widehat{Q}\,asin\,(\Sigma)\,\widehat{R}^\top \in T_{[Y_0]}\mathcal{M}, \tag{3.20}$$

and $Exp_{[Y_0]}\left(Log_{[Y_0]}([\widetilde{Y}_1])\right) = [\widetilde{Y}_1]$.

Proof. The first step is to solve the Procrustes problem which consists in finding a rotation of Y_1 such that it is as close as possible to the representative Y_0. In other words, we try to express the two (different) subspaces in a common basis,

$$\Phi = \underset{\Phi\in\mathbb{R}^{k\times k}:\Phi^\top\Phi=I_k}{\arg\min} \|Y_0 - Y_1\Phi\|_F.$$

To solve this problem, note that

$$\text{argmin}\|Y_0 - Y_1\Phi\|_F = \text{argmin}\langle Y_0 - Y_1\Phi, Y_0 - Y_1\Phi\rangle = \text{argmax}\langle \Phi, Y_1^\top Y_0\rangle.$$

Computing the SVD of $Y_1^\top Y_0$, $[Q,S,R] = \text{svd}\left(Y_1^\top Y_0\right)$, we have

$$\text{argmax}\langle \Phi, Y_1^\top Y_0\rangle = \text{argmax}\langle \Phi, QSR^\top\rangle = \text{argmax}\langle Q^\top \Phi R, S\rangle.$$

The matrix on the left is orthogonal, and the quantity is maximized when $Q^\top \Phi R$ is the identity,[3] thus $\Phi = QR^\top$. Let then $\widetilde{Y}_1 = Y_1 QR^\top$ be a new representative of $\mathcal{Y}_1$, and note that $Y_0^\top \widetilde{Y}_1 = RSR^\top$. The proof then proceeds similarly to that of Proposition 3.6. We split

$$\widetilde{Y}_1 = Y_0 Y_0^\top \widetilde{Y}_1 + (I - Y_0 Y_0^\top)\widetilde{Y}_1. \tag{3.21}$$

Defining $L := (I - Y_0^\top Y_0)\widetilde{Y}_1$, it holds that

$$L^\top L = \widetilde{Y}_1^\top (I - Y_0^\top Y_0)\widetilde{Y}_1 = R(I - S^2)R^\top,$$

which implies $[\widehat{Q}, \widehat{S}, R] = \text{svd}(L)$, where $\widehat{S} := \sqrt{I - S^2}$. Defining $\Sigma := \text{asin}(\widehat{S})$, we have $S = \cos(\Sigma)$ and inserting this into (3.21) leads to

$$\widetilde{Y}_1 = Y_0 R\cos(\Sigma)R^\top + \widehat{Q}\sin(\Sigma)R^\top. \tag{3.22}$$

Noticing that (3.22) is equal to $\text{Exp}_{[Y_0]}(\dot{Y})$, with $\dot{Y} = \widehat{Q}\Sigma R^\top$, we get the claim and further that $\text{Exp}_{[Y_0]}\left(\text{Log}_{[Y_0]}([\widetilde{Y}_1])\right) = \widetilde{Y}_1$. ◻

This improved alternative to compute the logarithmic map is implemented in the function `log_map`:

```
function [Ydot]=log_map(Y0,Y1)
% LOG_MAP computes the logarithmic map on the Grassman manifold.
% Ydot=log_map(Y0,Y1) receives the matrix representatives Y0,Y1
% in St(n,k) of two points on G(n,k), and computes a matrix
% representative Ydot in St(n,k) such that [Exp_{Y0}(Ydot)]=[Y1]

[Psi,S,R]=svd(Y1'*Y0,'econ');
[Q,Sigma,V]=svd((eye(size(Y0,1))-Y0*Y0')*Y1*Psi*R','econ');
Theta=diag(asin(diag(Sigma)));
Ydot=Q*Theta*V';
```

Incidentally, we remark that for some manifolds, as, e.g., the Stiefel manifold, there is no close expression available for the logarithmic map. The study of numerical methods to compute it is an active area of research, see, e.g., [7, 30, 25].

4. Algorithms for interpolation on the Grassmann manifold.

Given all the concepts introduced in the previous sections, we are now ready to discuss algorithms to interpolate on the Grassmann manifold. To do so, we first review a simple example of linear interpolation in Euclidean space. This allows us to provide a motivation for the algorithms discussed later.

4.1. Linear and piecewise linear univariate interpolation.

Let $c : [0,1] \ni t \to c(t) \in \mathbb{R}$ be an *unknown* one-dimensional smooth curve. Given only two values of $c(t)$, e.g., the initial and final time, $c(0) = c_0$ and $c(1) = c_1$, a standard problem in numerical analysis is to infer the value of $c(t)$ at any other $t \in (0,1)$. Interpolation is one of the numerical techniques to solve such problem [13]. It consists in deriving

[3]To see this, use the fact that S is diagonal and express the Frobenius scalar product as the sum of the elements of the product component-wise between the two matrices.

a second map $\widetilde{c} : [0,1] \to \mathbb{R}$ that can be easily evaluted for any $t \in [0,1]$ and satisfies $\widetilde{c}(0) = c_0$ and $\widetilde{c}(1) = c_1$. The value of c at any particular point $t \in (0,1)$ is then inferred to be $c(t) \approx \widetilde{c}(t)$. Since we only have two points, it is natural to use linear interpolation, namely to set $\widetilde{c}(t) = c_0 + (c_1 - c_0)t$, so that, e.g., $c(\frac{1}{2}) \approx \widetilde{c}(\frac{1}{2}) = \frac{c_0+c_1}{2}$. Let now Y be a curve on the Grassmann manifold, that is, $Y : [0,1] \ni t \to Y(t) \in \mathrm{Gr}(n,k)$. Given two points $Y(0) = [Y_0]$, $Y(1) = [Y_1]$, we would be tempted to use again linear interpolation and set, e.g., $[Y(\frac{1}{2})] \approx [\frac{1}{2}(Y_0 + Y_1)]$. This is equivalent to interpolate linearly component-wise the entries of the representative matrices. Unfortunately, this straight approach does not work. First, $\frac{1}{2}(Y_0 + Y_1)$ is not necessarily an orthonormal matrix. Second, non trivial manifolds are intrisically nonlinear objects, so that the sum of two elements does not lie on the manifold in general. For instance, if $\boldsymbol{x}, \boldsymbol{y} \in S^1$, $\frac{1}{2}(\boldsymbol{x} + \boldsymbol{y}) \notin S^1$. How can we then generalize the classical concept of linear interpolation on general smooth manifolds?
The solution is to interpret the action of perfoming linear interpolation in an Euclidean space from a broader perspective. As a matter of fact, linear interpolation consists actually in taking $\widetilde{c}$ as the geodesic (i.e. a straigth line, see Sec 2), that starts from c_0 at $t = 0$ and reaches c_1 at $t = 1$. In analogy with the Euclidean setting, when dealing with the Grassmann manifold it is natural to build the geodesic $\widetilde{Y}(t)$ that links $\mathcal{Y}_0$ and $\mathcal{Y}_1$ and approximate $Y(\frac{1}{2})$ with $\widetilde{Y}(\frac{1}{2})$. Using the logaritmic and exponential maps introduced in Sec. 3.1, $\widetilde{Y}(t)$ admits a simple close formula,

$$\widetilde{Y}(t) := \mathrm{Exp}_{[Y_0]}\left(t\mathrm{Log}_{\mathcal{Y}_0}([Y_1])\right), \tag{4.1}$$

that is, first we compute the tangent vector of the geodesic between $\mathcal{Y}_0$ and $\mathcal{Y}_1$ using the logarithmic map, then we move an amount $t \in [0,1]$ along the geodesic with the exponential map.

Remark 4.1 (Helpful intuitions for the generalization to higher-order/multivariate interpolation). First, note that the two data points $\mathcal{Y}_0$, $\mathcal{Y}_1$ do not play the same role. We could intuitively think that $\mathcal{Y}_0$ is selected as a *reference* point,[4] since we compute a tangent vector belonging to the tangent space of $\mathrm{Gr}(n,k)$ at $[Y_0]$, and we follow the geodesic starting from $[Y_0]$. We could have chosen as a reference point $\mathcal{Y}_1$, computed a tangent vector belonging to $T_{[Y_1]}\mathrm{Gr}(n,k)$, and then followed the geodesic starting from $\mathcal{Y}_1$. In this particular case, due to the uniqueness of the geodesic between two points, the choice of the reference point does not matter. But this will not be true for the high-order interpolation algorithms discussed next.
Second, we remark that we could write (4.1) as

$$\widetilde{Y}(t) := \mathrm{Exp}_{[Y_0]}\left(t\mathrm{Log}_{[Y_0]}([Y_1])\right) = \mathrm{Exp}_{[Y_0]}\left((1-t)\mathrm{Log}_{[Y_0]}([Y_0]) + t\mathrm{Log}_{[Y_0]}([Y_1])\right),$$

since $\mathrm{Log}_{[Y_0]}([Y_0]) = 0$. In other words, hidden in (4.1) there is a linear interpolation of the two tangent vectors belonging to $T_{[Y_0]}\mathrm{Gr}(n,k)$. This hidden interpolation step of the tangent vectors will actually become explicit in higher-order methods.

This one-dimensional linear interpolation procedure can be easily generalized to a piecewise linear variant. Let $\{\mathcal{Y}_1, \ldots, \mathcal{Y}_M\}$ be M points on the Grassmann manifold with representatives $\{Y_1, \ldots, Y_M\}$, and corresponding to the values of $Y(t)$ at times $\{t_1, \ldots, t_M\}$, $t_j < t_{j+1}$. Then, to infer the value of $\mathcal{Y}(t)$ at time t, we just find the

[4]The terminology "starting point" could be more appropriate for linear interpolation, but "reference point" is more suitable for the generalizations in the next subsections.

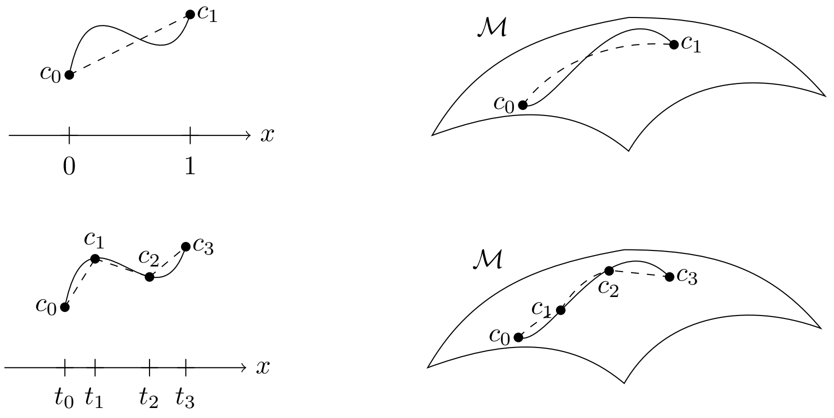

FIG. 2. *Top row: Linear interpolation of a curve $c(t)$ with values in $\mathbb{R}$ (left) and on a generic manifold $\mathcal{M}$ (right). The curve is represented by the continuous line, while the geodesic is the black dashed line. Bottom row: graphical representation of a piecewise linear interpolation on $\mathbb{R}$ (left) and on a manifold $\mathcal{M}$ (right).*

interval such that $t \in [t_j, t_{j+1}]$, build the geodesic $\widetilde{Y}(t)$ between $\mathcal{Y}_j$ and $\mathcal{Y}_{j+1}$, and finally return the value of $\widetilde{Y}(t)$ at time $t = \frac{t-t_j}{t_{j+1}-t_j}$, where the rescaling is due to the mapping of the interval $[t_j, t_{j+1}]$ to $[0, 1]$. This numerical procedure is summarized in the function `piecewise_linear_inter` and depicted in Fig. 2:

```
function Ynew=piecewise_linear_inter(Y,t,tnew)
% PIECEWISE_LINEAR_INTER evaluates the piecewise linear interpolant of t
    ->Y(t) in tnew.
% piecewise_linear_inter(Y,t,tnew) evaluates Y(t) at t=tnew, given
% a vector t containing the interpolation points, and a cell Y s.t.
% Y{i}=Y(t_i). The matrices Y{i} are orthogonal.

l=1;flag=0;
while( l<=length(t)-1 && flag==0) %find interval in which tnew lies
    if(t(l)<=tnew && tnew<=t(l+1))
        t0=t(l);                  %ends points of the interval
        t1=t(l+1);
        flag=1;
    end
    l=l+1;
end
l=l-1;
Ydot=log_map(Y{l},Y{l+1});        %Y{i} are orth.
r=(tnew-t0)/(t1-t0);
Ynew=exp_map(Y{l},r*Ydot);
```

4.2. High-order univariate interpolation. Suppose now that we would like to use not just a linear interpolation method, but a higher order one. For simplicity, we consider a quadratic interpolation. On $\mathrm{Gr}(n, k)$ we have data points $\mathcal{Y}_0, \mathcal{Y}_{\frac{1}{2}}$ and $\mathcal{Y}_1$, corresponding to the values $t \in \{0, \frac{1}{2}, 1\}$, and $t \in (0, 1)$. As a first step we choose a reference point among the data points available. Assume, e.g., that we choose $\mathcal{Y}_{ref} = \mathcal{Y}_{\frac{1}{2}}$. Next, we compute using the logarithmic map the three tangent vectors on $T_{[Y_{\frac{1}{2}}]}\mathrm{Gr}(n, k)$ associated to the geodesics starting from $\mathcal{Y}_{\frac{1}{2}}$ and reaching either $\mathcal{Y}_0$,

$\mathcal{Y}_{\frac{1}{2}}$ or $\mathcal{Y}_1$. We assume for the moment that the logarithmic map is well-defined in a sufficiently large neighboorhod containing the data points; See Sec. 4.3 to handle more general cases. We then perform a standard quadratic interpolation on $T_{[Y_{\frac{1}{2}}]}\mathrm{Gr}(n,k)$,

$$\widetilde{\dot{Y}}(t) = l_0(t)\mathrm{Log}_{[Y_{\frac{1}{2}}]}([Y_0]) + l_{\frac{1}{2}}(t)\mathrm{Log}_{[Y_{\frac{1}{2}}]}([Y_{\frac{1}{2}}]) + l_1(t)\mathrm{Log}_{[Y_{\frac{1}{2}}]}([Y_1]), \tag{4.2}$$

where $l_j(t)$ are Lagrange polynomials. Note that $T_{[Y_{\frac{1}{2}}]}\mathrm{Gr}(n,k)$ being a linear vector space, the result of the interpolation still lies in $T_{[Y_{\frac{1}{2}}]}\mathrm{Gr}(n,k)$. Finally, the output of the high-order interpolation is then

$$\widetilde{\mathcal{Y}} = \mathrm{Exp}_{[Y_{\frac{1}{2}}]}\left(\widetilde{\dot{Y}}(t)\right).$$

This algorithm can be readily generalized to arbitrary high-order methods and to several data points by replacing (4.2) with a suitable high-order interpolation formula. Note that the output of the algorithm does depend on the choice of the reference point, although its impact is generally limited [2]. To the best of our knowledge, there is not a rigourous mathematical argument to choose one particular point. Heuristically, it makes sense to "linearize" the manifold in a point closest to t.

4.3. Multivariate interpolation. So far we restricted ourselves to the interpolation of subspaces that depends on a scalar parameter t. We now review an algorithm presented in [2], that permits dependences on a set of parameters $\boldsymbol{t} = (t_1, \ldots, t_\ell) \in \mathbb{R}^\ell$. As the reader will notice, this algorithm is a direct generalization of the high-order scalar interpolation presented in Sec. 4.2.
Let $\{\mathcal{Y}_1, \ldots, \mathcal{Y}_M\}$ be M given points on $\mathrm{Gr}(n,k)$ corresponding to the input parameters $\{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_M\}$. We may assume the data is generated by an unknown map $\mathcal{Y} : \mathbb{R}^\ell \ni \boldsymbol{t} \to \mathcal{Y}(\boldsymbol{t})$, and given a sample parameter $\boldsymbol{t}$, we would like to estimate $\mathcal{Y}(\boldsymbol{t})$. The algorithm of [2] uses coordinate charts to map an open set of the manifold containing some input data to $\mathbb{R}^d$, perform interpolation in $\mathbb{R}^d$, and finally mapping the interpolation result back onto the manifold. The algorithm is reported in Alg 4.1 and it is described visually in Fig. 3. It consists in the following steps: First, we select a reference point $\mathcal{Y}_{ref}$. Then for $\widetilde{M} \leq M$ points that are sufficiently close to $\mathcal{Y}_{ref}$ (so that the logarithmic map is well-defined) we compute the tangent vector $\left\{\dot{Y}_j\right\}_{j=1}^{\widetilde{M}}$ of the geodesic starting from $\mathcal{Y}_{ref}$ and ending in $\mathcal{Y}_j$. Note that $\dot{Y}_j \in T_{\mathcal{Y}_{ref}}\mathrm{Gr}(n,k)$ for every j. Second, given the couples $(\boldsymbol{t}_j, \dot{Y}_j)$ we infer a tangent vector $\widetilde{\dot{Y}}$ using any standard Euclidean interpolant such that

$$\widetilde{\dot{Y}} = \sum_{j=1}^{\widetilde{M}} \omega_j \dot{Y}_j,$$

where $\omega_j \in \mathbb{R}$ are suitable interpolation weights. As a final step, we compute the output by moving along the geodesic using the exponential map, i.e. $\mathcal{Y}(\boldsymbol{t}) \approx \widetilde{Y} = \mathrm{Exp}_{\mathcal{Y}_{ref}}\left(\widetilde{\dot{Y}}\right)$.

5. Numerical examples. In this section we illustrate the potential of interpolation algorithms on the Grassman manifold in two simple, yet representative, problems arising in reduced order modeling. The first example considers a parametric time-dependent heat equation, where the interpolation algorithms are used to obtain more

Algorithm 4.1 Interpolation on the Grassmann manifold through tangent spaces

```
Require: A set $\{\mathcal{Y}_1, \dots, \mathcal{Y}_M\}$ of $M$ points on $\mathrm{Gr}(n,k)$, with representatives $\{Y_1, \dots, Y_M\}$ and corresponding to the parameters $\{\boldsymbol{t}_1, \dots, \boldsymbol{t}_M\}$, and an interpolation point $\boldsymbol{t}$.
1: Select a parameter $\boldsymbol{t}_i$ from the data set available (e.g., the closest to $\boldsymbol{t}$). Take $\mathcal{Y}_{ref} = \mathcal{Y}_i$ as the reference point.
2: For sufficiently close points, call log_map to compute $\dot{Y}_j := \mathrm{Log}_{[\mathcal{Y}_{ref}]}(\mathcal{Y}_j)$, $j = 1, \dots, \widetilde{M}$, $\widetilde{M} \leq M$.
3: Compute $\widetilde{\dot{Y}}$ using a classical interpolation algorithm with input the computed tangent vectors $\left\{\dot{Y}_1, \dots, \dot{Y}_{\widetilde{M}}\right\}$ at locations $\{\boldsymbol{t}_1, \dots, \boldsymbol{t}_{\widetilde{M}}\}$.
4: Call exp_map with arguments $\mathcal{Y}_{ref}$ and $\widetilde{\dot{Y}}$. Return its output.
```

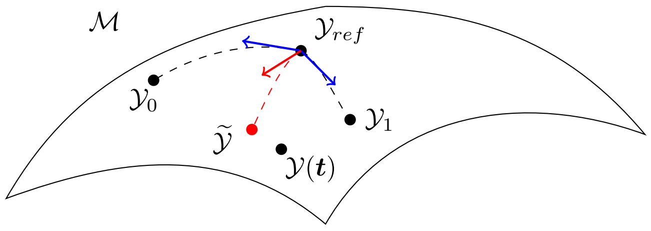

FIG. 3. *Graphical description of Alg. 4.1. The blue vectors represent the tangent vectors $\dot{Y}_0$ and $\dot{Y}_1$ of the geodesic starting from $\mathcal{Y}_{ref}$ and ending at $\mathcal{Y}_0$ and $\mathcal{Y}_1$. The red vector is the tangent vector $\widetilde{\dot{Y}}$ resulting from an interpolation of $\dot{Y}_0$ and $\dot{Y}_1$. Following the geodesic starting from $\mathcal{Y}_{ref}$ and determined by $\widetilde{\dot{Y}}$ we get the approximation $\widetilde{\mathcal{Y}}$ of $\mathcal{Y}(\boldsymbol{t})$.*

accurate reduced bases for new parameter queries, given a reduced basis dataset computed for some values of the parameter. The second problem instead deals with the solution of parametric linear systems using a simple two-level method with damped Jacobi smoothing, and shows how to use interpolation algorithms to derive efficient second level coarse spaces.

5.1. Time dependent parametric heat equation. Let us consider $\mathcal{D} := (0,1)$, $\mathcal{I}_\mu$ a bounded interval of $\mathbb{R}$, $T \in \mathbb{R}^+$, and the time dependent parametric partial differential equation

$$\partial_t u(x,t;\mu) - \nabla \cdot (\kappa(x;\mu)\nabla u(x,t;\mu)) = f(x,t), \quad x \in \mathcal{D},\ t \in [0,T],\ \mu \in \mathcal{I}_\mu, \tag{5.1}$$

equipped with the boundary conditions and the initial condition

$$u(0,t;\mu) = u(1,t;\mu) = 0, \qquad u(x,0;\mu) = u_0(x).$$

To ensure well-posedness for every admissible μ, we assume that there exist two positive numbers $\kappa_{\min}, \kappa_{\max} \in \mathbb{R}^+$ such that

$$0 < \kappa_{\min} \leq \kappa(x;\mu) \leq \kappa_{\max} \quad \forall x \in \mathcal{D},\ \forall \mu \in \mathcal{I}_\mu.$$

Our goal is the following: we would like to be able to evaluate in a fast and accurate way the solution map $\mu \to u(x,t;\mu)$.
Notice that for a fixed and given value of μ, (5.2) can be solved using a standard numerical methods: we introduce a piecewise linear finite element space $V_h =$

span$\{\phi_1,\dots,\phi_{N_h}\}$, associated to a quasi-uniform partition of the interval $[0,1]$ and consider a semi-discretization in space of (5.1). This results in the first order system of differential equations

$$M\partial_t \boldsymbol{u}(t;\mu) = A(\mu)\boldsymbol{u}(t;\mu) + \boldsymbol{f}(t), \tag{5.2}$$

where $(M)_{ij} = \int_{\mathcal{D}} \phi_i(x)\phi_j(x)\ dx$, $(A)_{ij} = \int_{\mathcal{D}} \kappa(x;\mu)\nabla\phi_i(x)\cdot\nabla\phi_j(x)\ dx$, $(\boldsymbol{f})_i = \int_{\mathcal{D}} f(x,t)\phi_i(x)\ dx$ and $\boldsymbol{u}(\mu) = (u_1(\mu),\cdots,u_{N_h}(\mu))^\top$ is the vector containing the coefficients of the expansion of the approximated solution $u_h(x,t;\mu)$ in the basis $\{\phi_j\}_{j=1}^{N_h}$ of V_h. Next, we use a classical integrator for ordinary differential equations to solve (5.2). Letting $(0,T] = \cup_{j=1}^{N_T}(t_j,t_{j+1}]$, $\Delta t := t_{j+1}-t_j$ be a uniform discretization of the time interval, and using, e.g., the Crank-Nicolson scheme, we end up solving the sequence of linear systems

$$\left(M + \frac{\Delta t}{2}A(\mu)\right)\boldsymbol{u}^n(\mu) = \left(M - \frac{\Delta t}{2}A(\mu)\right)\boldsymbol{u}^{n-1}(\mu) + \frac{\Delta t}{2}\left(\boldsymbol{f}^{n-1}+\boldsymbol{f}^n\right), \tag{5.3}$$

where $\boldsymbol{f}^n := \boldsymbol{f}(t_n)$ and our approximation is given by $u(x,t;\mu) \approx \sum_{j=1}^{N_h} u_j^n(\mu)\phi_j(x)$. Although simple, the approach described suffers from a high computational cost, especially if we wish to approximate $u(x,t\ \mu)$ for many values of μ.
A key observation here[5] is that for several problems, including the time-dependent parametric PDE (5.1), the solution $u(x,t;\mu)$ can be well described, both at different time instants and for different parameters μ, by a small dimensional subspace $\mathcal{V}$ whose dimension is $N_r \ll N_h$. Assuming for the moment that one has a basis $V = [\psi_1(x)|\dots|\psi_{N_r}(x)] \in \mathbb{R}^{N_h\times N_r}$ available for $\mathcal{V}$ (ψ_j are spatial functions like the $\phi_j(x)$), for every value of μ we could look for a reduced order solution of the form $u_r(x,t^n;\mu) = V\boldsymbol{u}_r^n(\mu) = \sum_{j=1}^{N_r}\psi_j(x)u_{j,r}^n(\mu)$, $\boldsymbol{u}_r^n(\mu) = (u_{1,r}^n(\mu),\dots,u_{N_r,r}^n(\mu))^\top \in \mathbb{R}^{N_r}$, by solving the reduced equations

$$V^\top\left(M+\frac{\Delta t}{2}A(\mu)\right)V\boldsymbol{u}_r^n(\mu)=V^\top\left(M-\frac{\Delta t}{2}A(\mu)\right)V\boldsymbol{u}_r^{n-1}(\mu)+\frac{\Delta t}{2}V^\top\left(\boldsymbol{f}^{n-1}+\boldsymbol{f}^n\right). \tag{5.4}$$

Notice that Eq (5.4) is derived from (5.3) by replacing $\boldsymbol{u}^n$ with $V\boldsymbol{u}_r^n$ and by projecting the equation onto the subspace $\mathcal{V}$. Further, (5.4) has the computational advantange of requiring to invert at each time step the smaller matrix $V^\top\left(M+\frac{\Delta t}{2}A\right)V \in \mathbb{R}^{N_r\times N_r}$, rather than $\left(M+\frac{\Delta t}{2}A\right) \in \mathbb{R}^{N_h\times N_h}$ appearing in (5.3).
To build an orthonormal basis V for $\mathcal{V}$, one of the most popular methods is the Proper Ortoghonal Decomposition (POD) (see [23, Section 6]) which consists in computing snaphots of the discrete solution $\left\{\boldsymbol{u}^0(\mu_i),\dots,\boldsymbol{u}^{N_T}(\mu_i)\right\}$ for given parameter values μ_i, $i=1,\dots,M$. All these discrete solutions are then inserted into a large matrix, called the snapshot matrix,

$$S = \left[\boldsymbol{u}^0(\mu_1),\dots,\boldsymbol{u}^{N_T}(\mu_1),\dots,\boldsymbol{u}^0(\mu_M),\dots,\boldsymbol{u}^{N_T}(\mu_M)\right] \in \mathbb{R}^{N_h\times(N_T+1)M}.$$

Finally, one computes a Singular Value Decomposition (SVD) of S and sets $V = [\psi_1,\dots,\psi_r]$, the ψ_j being the first N_r left singular vectors of S with $N_r \ll N_T M$. The intuition behind this construnction is that the columns of S will somehow be similar among them if $u(x,t;\mu)$ can be well approximated by a small dimensional subspace. Then, the left singular vectors of the SVD of S will span a subspace that

[5]Motivating more than two decades of intense research, see, e.g., the reference works [23, 18, 5].

describes most of the variability in the snapshots $\boldsymbol{u}^k(\mu_i)$, very similar to the principle behind Principal Component Analysis (PCA) in statistics, see, e.g., [23, Chapter 6]. Now, given this brief description of an approach to solve efficiently parametrized PDEs, how do interpolation algorithms on the Grassmann manifold come into play? The starting point is the observation that by performing the SVD of the snapshots matrix S, we are trying to capture *simultaneously* the dependence of the solution $u(x, t; \mu)$ on both the temporal and parametric variables t and μ. This may require to take a pretty large value of N_r to capture sufficiently well the variability of the columns of S, and thus leading to still quite a large linear system to be solved at each time step in (5.4). The same issue might arises if we consider (5.1) for a very large set of values μ, so that N_r might end up to be too large in a pure global approach, while we could hope to use a small N_r if we restricted ourselves to a subset of the parameter values.
A possible alternative is then to compute several subspaces $\mathcal{V}(\mu_j)$, for a given set of parameters μ_j decided a-priori, by perfoming the SVD of the snapshot matrices

$$S(\mu_j) = \left[\boldsymbol{u}^0(\mu_j), \dots, \boldsymbol{u}^{N_T}(\mu_j)\right].$$

Then, we can set $\mathcal{V}(\mu_j) = \text{span}\{\psi_1(\mu_j), \dots, \psi_{N_r}(\mu_j)\}$, $\psi_k(\mu_j)$ being the left singular vectors of $S(\mu_j)$, and perform Grassmann interpolation to derive a new subspace $\tilde{\mathcal{V}}(\mu)$ for a new, out of sample, parameter μ. We use the tilde-notation to stress that $\tilde{\mathcal{V}}(\mu)$ is not constructed as the span of the eigenvectors of some matrix, but as the result of an interpolation process between subspaces.
We now present a compact Matlab implementation that explores this idea and show its potential to compute an approximated solution of (5.1). We set $T = 1$, $\mathcal{I}_\mu = [\mu_{min}, \mu_{max}]$, $\kappa(x, \mu) = 1 + x^\mu$, $f(x, t) = 4\sin(\frac{\pi t}{2})\exp(2x^2)$ and $u_0(x) = \sin(3\pi x)x^2$. For any value of μ, the solution of (5.1) can be approximated using the Matlab function `Solve_given_mu` which implements a finite element discretization in space and Crank-Nicolson marching scheme in time. Notice that `Solve_given_mu` calls another function, `assemble_matrix`, which simply assembles the finite element mass and stiffness matrices. Due to space limitation, we do not report its implementation here, but it is available in the folder of codes in the supplementary material.

```
function [uvec]=Solve_given_mu(mu,P,El,coeff,f_fun,T,u0,V)
% SOLVE_GIVEN_MU computes the solution of the time dependent heat
    equation for a given parameter value mu.
% Solve_given_mu(mu,P,El,coeff,f_fun,T,u0,V) receives a parameter mu
% mesh matrices P and El, diffusion coefficent coeff, a force term f_fun
% a final time T, an initial condition u0, and an optional matrix V
% spanning a reduced subspace.
% It returns a matrix uvec, where uvec(:,j) approximates u(x,t^j;\mu)

n=size(El,2); dt=T(2)-T(1);                        % number of elements
M=assemble_matrix(@(x,el) 1,P,El,0,0);             % assemble mass matrix
if nargin <8 V=speye(n-1,n-1); end                 % no opt. input->Phi=
    identity
int=(2:n);                                         % interior nodes
A=assemble_matrix(@(x)coeff(x,mu),P,El,1,1);       % Assemble stiffness matrix
A=A(int,int);                                      % remove Dirichlet nodes
uvec=zeros(size(V,2),length(T));                   % allocate matrix
uvec(:,1)=V'*u0;                                   % initial condition
A=V'*A*V;                                          % reduced matrix if V in
    input
Mint=V'*M(int,int)*V;
bold=M*(f_fun(P,0)');                              % rhs at time t=0.
```

```
bold=bold(int);                                    % remove Dirichlet nodes
for k=1:length(T)-1                                % loop over time steps
    b=M*(f_fun(P,k*dt)');                          % compute time-dep. rhs
    b=b(int);                                      % remove Dirichlet nodes
    uvec(:,k+1)=(Mint+dt/2*A)\...                  % solve linear system
                ((Mint-dt/2*A)*uvec(:,k)+dt*V'*(b+bold)/2);
    bold=b;
end
uvec=V*uvec;                                       %final output.
```

The function `Solve_given_mu.m` allows us to compute easily the matrices $S(\mu_j)$ since they correspond to the function's output `uvec`. A first idea that we can try is to check the quality of the interpolation process, namely how close or far are the interpolated subspaces $\tilde{\mathcal{V}}(\mu_j)$ with respect to the exact $\mathcal{V}(\mu_j)$. To do so, we measure the angle[6] between $\tilde{\mathcal{V}}(\mu_j)$ and $\mathcal{V}(\mu_j)$ using the Matlab function `subspace` for several values of $\mu \in [1, 10]$. An angle close to $\frac{\pi}{2} \approx 1.5708$ means that the two subspaces are orthogonal, that is, they are very "distant" one from the other. On the contrary, a small angle means the two two subspaces are aligned, therefore they are very close to each other, and equal if the angle is zero. The following Matlab script computes the angle between $\mathcal{V}(\mu)$ and $\tilde{\mathcal{V}}(\mu)$ computed using the function `piecewise_linear_inter` described in 4.1. The result is shown on the left panel of Fig. 4.

```
dt=0.01; T=(0:dt:1);                        % time step and time interval
a=0; b=1;                                   % domain is interval (a,b)
h=0.01;                                     % mesh size
P=(a:h:b);                                  % nodes
n=(b-a)/h;                                  % number of elements
El=[(1:n);(2:n+1)];                         % connectivity matrix
int=(2:n);                                  % interior nodes
f_fun=@(x,t) 4*sin(pi*t/2)*exp(2*x.^2);     % force term
u0=(sin(3*pi*P(int)).*P(int).^2)';          % initial condition
kappa=@(x,mu,el) exp(mu*sin(pi*x));         % diff. coeff.
uvec=zeros(n-1,length(T));                  % matrix containing sols. at
    each t
uvec(:,1)=u0;                               % initial condition
Nr=10;                                      % dimension subspace
mumin=1; mumax=10;                          % range for values of mu in I_\
    mu.
nsamples=10;                                % number of interpolation points
musamples=linspace(mumin,mumax,nsamples);   %uniform samples in I_\mu
V=cell(nsamples,1);                         % cell containing subspaces
for j=1:length(musamples)                   % subspace at ref. values
    Smuj=Solve_given_mu(musamples(j),P,El,kappa,f_fun,T,u0);
    [V{j},~,~]=svds(Smuj,Nr);
end
muvec=(mumin:0.1:mumax);                    % range of testing values of \mu
anglevec=zeros(length(muvec),1);            % angles between subspaces
for j=1:length(muvec)                       % for every query parameter
    Smuj=Solve_given_mu(muvec(j),P,El,kappa,f_fun,T,u0);
    [Vj,~,~]=svds(Smuj,Nr);                 % compute exact Vj
    Vinter=piecewise_linear_inter(V,musamples,muvec(j)); % Interp.
        subspace
    anglevec(j)=subspace(Vinter,Vj);        % angle between subspaces.
end
plot(muvec,anglevec);xlim([mumin mumax]);ylim([0 pi/2]);grid on; %plot
ylabel('Angle between subspaces'); xlabel('Parameter \mu');set(gca,'
    fontsize',14);
```

[6]As we can measure the angle between two lines, it is possible to define an angle between subspaces [14].

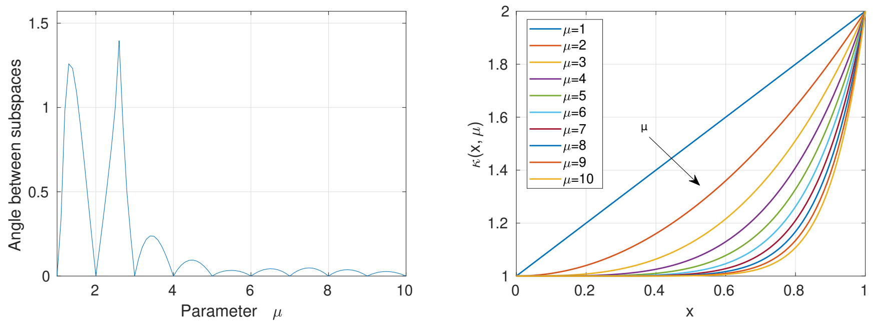

FIG. 4. *The left panel shows the angles between the subspace $\mathcal{V}(\mu)$ obtained through an eigen-decomposition of $S(\mu)$ and $\tilde{\mathcal{V}}(\mu)$ derived using piecewise linear interpolation on the Grassmann manifold. The right panel shows the diffusion coefficient $\kappa(x,\mu) = 1 + x^{\mu}$ for different values of μ.*

Notice that the angle is zero for $\mu = k$, $k = 1, \dots, 10$, since then the evaluation point coincides with an interpolation point (line 16), and thus the piecewise linear interpolation is exact. Further, we remark that the angle is quite large for μ close to 1, while it is small for larger values of μ. We can understand this behaviour by looking at the right panel of Fig. 4 which shows $\kappa(x,\mu) = 1 + x^{\mu}$ for different values of μ. Indeed as μ increases, $\kappa(x,\mu)$ varies a lot at the beginning, while it remains very similar for large values of μ. This small variability is reflected by a small variability of the matrices $S(\mu_j)$ (a slight change in $\kappa(x;\mu)$ will cause only a small change in $\{\boldsymbol{u}^n(\mu)\}_{n=1}^{N_T}$ due to the continuity of the solution map $\kappa \to \{\boldsymbol{u}^n\}_{n=1}^{N_T}$), and of the subspaces $\mathcal{V}(\mu_j)$, so that the interpolation algorithm delivers an accurate approximation. This observation should also make us consider the opportunity to use a non-uniform sampling by adding sampling points close to $\mu = 1$. We conclude this section by presenting a Matlab code that compares the solutions for $\mu = 3.5$ obtained by solving (5.3), and (5.4) with both $\mathcal{V}(\mu)$ computed from the POD of $S(\mu)$ and with $\tilde{\mathcal{V}}(\mu)$ computed using linear interpolation between the two subspaces $\mathcal{V}(3)$ and $\mathcal{V}(4)$.

```
dt=0.01; T=(0:dt:1);                           % time step, time interval
a=0; b=1;                                      % domain is interval (a,b)
h=0.01;                                        % mesh size
P=(a:h:b);                                     % nodes
n=(b-a)/h;                                     % number of elements
El=[(1:n);(2:n+1)];                            % connectivity matrix
int=(2:n);                                     % interior nodes
f_fun=@(x,t) 4*sin(pi*t/2)*exp(2*x.^2);        % force term
u0=(sin(3*pi*P(int)).*P(int).^2)';             % initial condition
coeff=@(x,mu,el) 1+x.^mu;                      % diffusion coefficient
Nr=5;                                          % number of functions
mueval=3.5;                                    % evaluation point
u=Solve_given_mu(mueval,P,El,coeff,f_fun,T,u0); % full solution
[U,~,~]=svds(u,Nr);                            % subspace for mueval
ur=Solve_given_mu(mueval,P,El,coeff,f_fun,T,u0,U); %reduced solution
muinit=3;                                      % extrema interval
uinit=Solve_given_mu(muinit,P,El,coeff,f_fun,T,u0);
[Uinit,~,~]=svds(uinit,Nr);                    % subspace for muinit
mufin=4;                                       %extrema interval
ufin=Solve_given_mu(mufin,P,El,coeff,f_fun,T,u0);
[Ufin,~,~]=svds(ufin,Nr);                      % subspace for mufin
Tangent_vector=log_map(Uinit,Ufin);            % compute tangent vector
Uinter=exp_map(Uinit,((mueval-muinit)/(mufin-muinit))*Tangent_vector);
uinter=Solve_given_mu(mueval,P,El,coeff,f_fun,T,u0,Uinter);
```

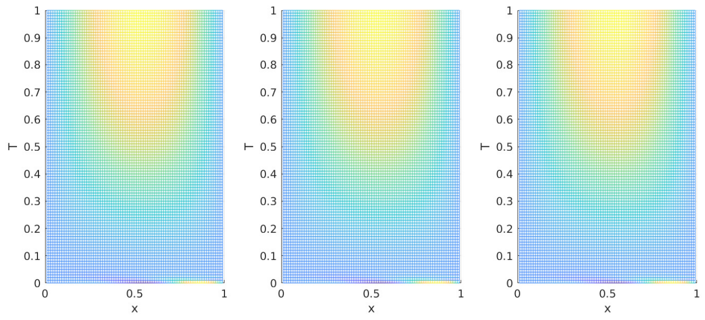

FIG. 5. *Comparison between the solutions computed by solving* (5.3) *(right plot), and* (5.4) *using both a subspace* $\mathcal{V}(\mu)$ *obtained computing the SVD of* $S(\mu)$ *(central panel) and a linearly interpolated subspace* $\tilde{\mathcal{V}}(\mu)$ *(right panel).*

```
25 figure(); [X,T]=meshgrid(P(int),T);                %plot
26 subplot(1,3,1); mesh(X,T,u');view([0 90]); xlabel('x');ylabel('T')
27 subplot(1,3,2); mesh(X,T,ur');view([0 90]); xlabel('x');ylabel('T')
28 subplot(1,3,3); mesh(X,T,uinter');view([0 90]); xlabel('x');ylabel('T')
```

As Fig 5 shows, $N_r = 5$ (line 11) basis functions are actually already sufficient to well-capture the solution behaviour. Most importantly from our point of view, Fig. 5 shows that we can avoid to recompute a subspace $\mathcal{V}$(3.5) through an expensive SVD, but instead safely use linear interpolation on the Grassmann manifold to derive a suitably accurate subspace.

5.2. Two-level iterative methods for parametric linear systems. In this subsection, we discuss the use of interpolation algorithms on the Grassmann manifold to design efficient solvers for the sequence of parametrized linear systems

$$A(\mu)\boldsymbol{u}(\mu) = \boldsymbol{f}, \quad \mu \in \mathcal{I}_\mu, \tag{5.5}$$

where $\mathcal{I}_\mu$ is a subset of $\mathbb{R}$ and $A \in \mathbb{R}^{N_h \times N_h}$. Such parametrized systems may arise for instance in the discretization of a stationary version of (5.1), or even in the time-dependent case, see, e.g., (5.3). More generally, they often appear in uncertainty quantification applications, where the repeated solution of linear systems for different realizations of the randomness is required to study the variability of a certain quantity of interest. Of course, the topic has been previously studied, see [8, 20, 19, 28]. In this paragraph, we propose an alternative and novel approach based on a two-level stationary iterative method that uses interpolation algorithms on the Grassmann manifold to adapt to new parameter queries.
Let us recall that a classical one-level stationary method to solve $A\boldsymbol{u} = \boldsymbol{f}$ is based on the splitting of the matrix A into $A = M - N$, and on the iteration

$$\boldsymbol{u}^n = \boldsymbol{u}^{n-1} + M^{-1}\left(\boldsymbol{f} - A\boldsymbol{u}^{n-1}\right).$$

Standard well-known methods can be embedded into this abstract form [9]. The Jacobi method is obtained setting $M = D$, D being the matrix containing only the diagonal of A, the damped Jacobi method is obtained setting $M = \frac{1}{\omega}D$, ω being a relaxation parameter, and the Gauss-Seidel method is derived setting $M = D + L$, L being the lower triangular part of A. Notice that the error $\boldsymbol{e}^n := \boldsymbol{u} - \boldsymbol{u}^n$ satisfies the recurrence relation $\boldsymbol{e}^n = G\boldsymbol{e}^{n-1}$, $G := M^{-1}N$ being the iteration matrix, and that, theoretically, by solving the error equation at every iteration n

$$A\boldsymbol{e} = \boldsymbol{f} - A\boldsymbol{u}^n, \tag{5.6}$$

we could immediately find the solution $\boldsymbol{u}$, since a direct calculation shows that $\boldsymbol{u} := \boldsymbol{u}^n + \boldsymbol{e}$ is the correct solution. Unfortunately, solving (5.6) is as expensive as solving the original system $A\boldsymbol{u} = \boldsymbol{f}$.

It is well-known that a stationary method converges if and only if the spectral radius of G is strictly smaller than one, that is, $\rho(G) < 1$ [9, Theorem 2.7]. Further, the spectrum of G provides important information about the convergence of the method. Assuming G is diagonalizable and letting $\{(\boldsymbol{v}_j, \lambda_j)\}_{j=1}^{N_h}$ be the eigenpairs of G, we can decompose $\boldsymbol{e}^n = \sum_{j=1}^{N} e_j^n \boldsymbol{v}_j$, so that $\boldsymbol{e}^{n+1} = G\boldsymbol{e}^n = \sum_{j=1}^{N} \lambda_j e_j^n \boldsymbol{v}_j$. In other words, at each iteration the error reduces by a factor λ_j along the direction of the $\boldsymbol{v}_j$ eigenvector. This is a very important remark since some classical iterative methods, as the Jacobi method, may convergence very fast along some eigenvector directions while they are extremely slow for others[7].

The idea of a two-level iterative method is to consider a subspace $\mathcal{V}$ which should be close to the subspace spanned by the eigenvectors of G that are responsible for the slow convergence of the one-level stationary iteration method. Then, we could use the one-level iterative method to reduce the error components that are efficiently handled by G, while we could solve a projected version of (5.6) onto the subspace $\mathcal{V}$, often called coarse problem, to reduce the remaining error components. Letting $V \in \mathbb{R}^{N_h \times N_c}$ be a matrix whose columns span $\mathcal{V}$, this procedure is described by Alg. 5.1. Notice that we have not discussed how to generate the subspace $\mathcal{V}$. Here and in the numerical codes, we assume for simplicity that the columns of V correspond to the eigenvectors associated to the N_c largest eigenvalues of G in modulus. However, this is not a practical choice due to the high computational cost. There are several alternatives to construct candidate subspaces in a more efficient way, but their discussion is beyond the scope of this manuscript (see, e.g., [10, 27]). Finally, we recall that a two-level method is still a stationary iterative method whose iteration matrix is

$$T_{\mathcal{V}} = G^{n_2}(I - V(V^\top A V)^{-1} V^\top A) G^{n_1}, \tag{5.7}$$

so that to study the convergence of a two-level method we can analyze the spectral radius of $T_{\mathcal{V}}$. Note that we use the subscript $_{\mathcal{V}}$ to stress the dependence on the coarse space $\mathcal{V}$.

After this brief overview of iterative methods, it might appear clear where we would like to use interpolation algorithms on the Grassmann manifold to solve (5.5) efficiently. As a matter of fact, for each parameter value μ, we have a different iteration matrix $G(\mu) := M^{-1}(\mu)N(\mu)$, and consequently a different coarse space $\mathcal{V}(\mu)$ which is expensive to compute. Therefore, we would like to construct the coarse spaces $\mathcal{V}(\mu_j)$ for a set of different parameters μ_j, $j = 1, \dots, M$, while for a new query parameter, we would use a coarse space $\widetilde{\mathcal{V}}(\mu_j)$ computed by interpolation. Let us see now how this idea works in practice in a Matlab example. We consider the stationary version of (5.1) with $\kappa(x, \mu) = 2 + \cos(\mu x)$ and for simplicity we consider a two-level method like in (5.7) with $n_1 = 1$ and $n_2 = 0$.

[7] Indeed, the damped Jacobi method eliminates very fast high frequency components of the error, while it struggles a lot to remove the low frequency ones [9, Sec. 4.10]. The key idea of the well-known multigrid methods is to add a second level to the Jacobi method based on a coarse geometric mesh, where the low frequency error components are still well represented and can be eliminated by a direct solve, see, e.g., [26, 16, 27].

Algorithm 5.1 Two-level stationary method

```
Require: $\boldsymbol{u}^0$                                 (initial guess)
1: $\boldsymbol{u}^n = \boldsymbol{u}^{n-1} + M^{-1}(\boldsymbol{f} - A\boldsymbol{u}^{n-1})$, $n = 1, \dots, n_1$   (pre-smoothing steps)
2: $\mathbf{r} = \mathbf{f} - A\mathbf{u}^{n_1}$                        (compute the residual)
3: Solve $(V^\top AV)\mathbf{e} = V^\top \mathbf{r}$               (solve the coarse problem)
4: $\mathbf{u}^0 = \mathbf{u}^{n_1} + V\mathbf{e}$                      (add coarse correction)
5: $\boldsymbol{u}^n = \boldsymbol{u}^{n-1} + M^{-1}(\boldsymbol{f} - A\boldsymbol{u}^{n-1})$, $n = 1, \dots, n_2$   (post-smoothing steps)
6: Set $\mathbf{u}^0 = \mathbf{u}^{n_2}$                           (update)
7: Repeat from 1 to 6 until convergence
```

```matlab
a=0; b=1;                           % domain is interval (a,b)
h=0.01;                             % mesh size
P=(a:h:b);                          % nodes
n=(b-a)/h;                          % number of elements
El=[(1:n);(2:n+1)];                 % connectivity matrix
int=(2:n);                          % interior nodes
coeff=@(x,mu,el) 2+cos(mu*x);       % diffusion coefficient
Nr=20;                              % Number of basis functions
mumin=0; mumax=10;                  % range of mu
Ninterpoints=7;
musamples=linspace(mumin,mumax,Ninterpoints);
V=cell(Ninterpoints,1);
omega=2/3;                          % parameter damped Jacobi
for k=1:length(musamples)           % coarse spaces for values of musamples
    A=assemble_matrix(@(x) coeff(x,musamples(k)),P,El,1,1);
    A=A(int,int);                   % remove boundary nodes
    D=spdiags(diag(A),0,size(A,1),size(A,2)); % extract diagonal
    G=@(v) v-omega*(D\(A*v));       % iteration matrix Jacobi
    [Vj,~]=eigs(G,size(A,1),Nr,'largestabs'); % eigen decomp. of G
    V{k}=Vj(:,1:Nr);                % keep first Nr eigenvectors
end
muvec=(mumin:0.1:mumax);            % range of testing values of mu
for k=1:length(muvec)
    A=assemble_matrix(@(x) coeff(x,muvec(k)),P,El,1,1); A=A(int,int);
    D=spdiags(diag(A),0,size(A,1),size(A,2));
    G=@(v) v-omega*(D\(A*v));       % iteration matrix Jacobi
    [Vex,E]=eigs(G,size(A,1),Nr,'largestabs'); % eigen decomp. of G
    eigen(k)=E(1,1);                % save spectral radius of G.
    Vinter=piecewise_linear_inter(V,musamples,muvec(k)); %linear interp.
    radius2lex(k)=abs(eigs(@(v) (G(v)-Vex*((Vex'*A*Vex)\(Vex'*A*(G(v))))
        ),size(A,1),1,'largestabs'));
    radius2linter(k)=abs(eigs(@(v) (G(v)-Vinter*((Vinter'*A*Vinter)\(
        Vinter'*A*(G(v))))),size(A,1),1,'largestabs'));
    anglevec(k)=subspace(Vex,Vinter);
end
figure(); plot(muvec,anglevec,'Linewidth',1.3); grid on; xlabel('\mu');
ylabel('Angles');
figure(); plot(muvec,eigen,'linewidth',1.3); hold on;
plot(muvec,radius2lex,'r','linewidth',1.3); plot(muvec,radius2linter,'k'
    ,'linewidth',1.3);
legend('Jacobi','Two-level Exact','Two-level Interp'); grid on;
xlabel('\mu');ylabel('Spectral radii');set(gca,'fontsize',14);
```

Fig. 6 shows on the left panel the angles between the exact and interpolated subspaces, while on the right one we show the behaviour of the spectral radii of the one-level damped Jacobi method (with parameter $\omega = 2/3$) and of two two-level methods, $T_{\mathcal{V}(\mu)}$ based on the exact coarse space $\mathcal{V}(\mu)$, and $T_{\tilde{\mathcal{V}}(\mu)}$ based on an interpolated coarse space.

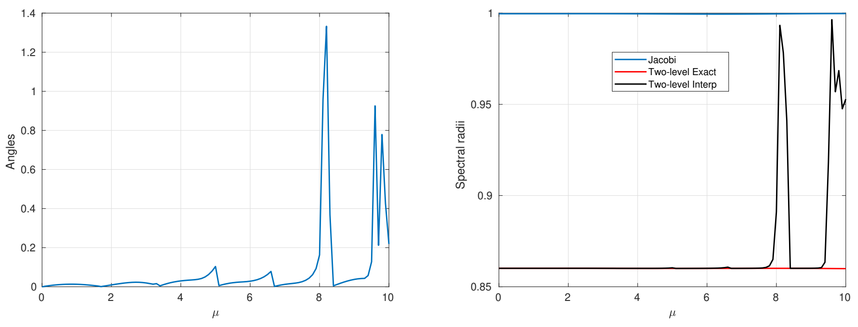

Fig. 6. *The left panel shows the angles between the exact subspaces $\mathcal{V}(\mu)$ computing using a SVD of $G(\mu)$, and the interpolated subspaces $\tilde{\mathcal{V}}(\mu)$. The right panel compares the spectral radius of a one-level Jacobi method, with two different two-level methods, one that uses $\mathcal{V}(\mu)$ and one that relies on $\tilde{\mathcal{V}}(\mu)$.*

We see that there is a strong relation between the subspace angles and the spectral radius of $T_{\tilde{\mathcal{V}}}(\mu)$: the smaller the subspace angle is, the closer is the spectral radius of $T_{\tilde{\mathcal{V}}}(\mu)$ to that of $T_{\mathcal{V}}(\mu)$. A related observation, important from our perspective, is that for quite a wide range of parameters μ, $\rho(T_{\tilde{\mathcal{V}}}(\mu))$ is very close to $\rho(T_{\mathcal{V}}(\mu))$, which corroborates the strategy proposed to adapt the coarse space of a two-level stationary method by relying on interpolation methods on the Grassmann manifold.

Acknowledgements. The authors gratefully acknowledge the financial support of the Mathematisches Forschungsinstitut Oberwolfach for a two-week stay at the Institute under the "Oberwolfach Research Fellows" program. G. Ciaramella and T. Vanzan are members of the INdAM-GNCS group.

REFERENCES

[1] P-A Absil, R. Mahony, and R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation. *Acta Applicandae Mathematica*, 80:199–220, 2004.
[2] D. Amsallem and C. Farhat. Interpolation method for adapting reduced-order models and application to aeroelasticity. *AIAA journal*, 46(7):1803–1813, 2008.
[3] D. Amsallem and C. Farhat. An online method for interpolating linear parametric reduced-order models. *SIAM Journal on Scientific Computing*, 33(5):2169–2198, 2011.
[4] T. Bendokat, R. Zimmermann, and P-A Absil. A Grassmann manifold handbook: Basic geometry and computational aspects. *arXiv preprint arXiv:2011.13699*, 2020.
[5] P. Benner, S. Gugercin, and K. Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.
[6] N. Boumal. *An introduction to optimization on smooth manifolds.* Cambridge University Press, 2023.
[7] D. Bryner. Endpoint geodesics on the Stiefel manifold embedded in Euclidean space. *SIAM Journal on Matrix Analysis and Applications*, 38(4):1139–1159, 2017.
[8] A. Carr, E. de Sturler, and S. Gugercin. Preconditioning parametrized linear systems. *SIAM Journal on Scientific Computing*, 43(3):A2242–A2267, 2021.
[9] G. Ciaramella and M. J. Gander. *Iterative Methods and Preconditioners for Systems of Linear Equations.* Society for Industrial and Applied Mathematics, Philadelphia, PA, 2022.
[10] G. Ciaramella and T. Vanzan. Spectral coarse spaces for the substructured parallel Schwarz method. *Journal of Scientific Computing*, 91(3):69, 2022.
[11] J. Degroote, J. Vierendeels, and K. Willcox. Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis. *International Journal for Numerical Methods in Fluids*, 63(2):207–230, 2010.
[12] J. Gallier. *Geometric Methods and Applications: For Computer Science and Engineering.* Texts in Applied Mathematics. Springer New York, 2011.

[13] W. Gander, M. J. Gander, and F. Kwok. *Scientific computing-An introduction using Maple and MATLAB*, volume 11. Springer Science & Business, 2014.

[14] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013.

[15] D. Goutaudier, F. Nobile, and J. Schiffmann. A new method to interpolate POD reduced bases–application to the parametric model order reduction of a gas bearings supported rotor. *International Journal for Numerical Methods in Engineering*, 124(18):4141–4170, 2023.

[16] W. Hackbusch. *Multi-Grid Methods and Applications*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2013.

[17] E. Hairer, S.P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2008.

[18] J. S. Hesthaven, G. Rozza, and B. Stamm. *Certified reduced basis methods for parametrized partial differential equations*, volume 590. Springer, 2016.

[19] E. Jarlebring and S. Correnty. Infinite GMRES for parameterized linear systems. *SIAM Journal on Matrix Analysis and Applications*, 43(3):1382–1405, 2022.

[20] D. Kressner and C. Tobler. Low-rank tensor Krylov subspace methods for parametrized linear systems. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1288–1316, 2011.

[21] J.M. Lee. *Introduction to Smooth Manifolds*. Graduate Texts in Mathematics. Springer, 2003.

[22] K. B. Petersen, M. S. Pedersen, et al. The matrix cookbook. *Technical University of Denmark*, 7(15):510, 2008.

[23] A. Quarteroni, A. Manzoni, and F. Negri. *Reduced basis methods for partial differential equations: an introduction*, volume 92. Springer, 2015.

[24] P. H. Schönemann. A generalized solution of the orthogonal Procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[25] M. Sutti. Shooting methods for computing geodesics on the Stiefel manifold. *arXiv:2309.03585*, 2023.

[26] U. Trottenberg, C.W. Oosterlee, and A. Schuller. *Multigrid Methods: Basics, Parallelism and Adaptivity*. Elsevier Science, 2001.

[27] J. Xu and L. Zikatanov. Algebraic multigrid methods. *Acta Numerica*, 26:591–721, 2017.

[28] O. Zahm and A. Nouy. Interpolation of inverse operators for preconditioning parameter-dependent equations. *SIAM Journal on Scientific Computing*, 38(2):A1044–A1074, 2016.

[29] R. Zimmermann. Manifold interpolation and model reduction. *arXiv preprint arXiv:1902.06502*, 2019.

[30] R. Zimmermann and K. Hüper. Computing the Riemannian logarithm on the Stiefel manifold: Metrics, methods, and performance. *SIAM Journal on Matrix Analysis and Applications*, 43(2):953–980, 2022.