Interpolation in Special Orthogonal Groups

Tatiana Shingel
DAMTP, Centre for Mathematical Sciences
University of Cambridge

August 7, 2007

Abstract

The problem of constructing smooth interpolating curves in non-Euclidean spaces finds applications in different areas of science. In this paper we propose a scheme to generate interpolating curves in Lie groups, focusing on a special orthogonal group $\mathrm{SO}(n)$. Our technique is based on the exponential representation of elements of the group, which allows to transfer the problem to the corresponding Lie algebra $\mathfrak{so}(n)$ and benefit from the linearity of this space. Due to the exponential representation we can obtain a high degree of smoothness of an interpolating curve at relatively low costs. The underlying problem is challenging because the standard $\mathrm{SO}(n) \longrightarrow \mathfrak{so}(n)$ map is multi-valued.

1 Introduction

In the present paper we develop an algorithm in order to provide a constructive solution to the following problem:

> Let $Q_1, Q_2, \ldots, Q_m \in \mathrm{SO}(n)$ be an ordered set of elements and $s_1 < s_2 < \cdots < s_m$ be given points in $[0, 1]$. Construct a sufficiently smooth interpolation function $Q : [0, 1] \longrightarrow \mathrm{SO}(n)$ such that
>
> $$Q(s_k) = Q_k, \quad \text{for } k = 1, \ldots, m. \tag{1}$$

A Lie group $\mathrm{SO}(n)$ can be naturally embedded in the Euclidean space $\mathrm{M}_n(\mathbb{R})$ of all $n \times n$ matrices. Therefore, we will identify the elements of $\mathrm{SO}(n)$ with real $n \times n$ orthogonal matrices of unit determinant.

Construction of smooth interpolating curves in non-Euclidean spaces is an interesting theoretical problem which finds many applications in engineering and physics. Interpolation in the 3-dimensional rotation group SO(3) has immediate application in robotics for path planning of a rigid body, in computer graphics for the animation of 3D objects, and elsewhere. Interpolation in Lie groups can be applied in geometric integration, when one needs to approximate a solution at intermediate points in the integration interval. There exist several methods and algorithms which solve the interpolation problem. Depending on which properties of the trajectory the algorithm is focused, it has certain implementation costs, which usually grow with the degree of smoothness. There are other restrictions such as considering a problem of interpolation only locally, i.e. when the interpolation

data can be covered by a single coordinate chart on the manifold. We believe that under this restriction the interpolation problem falls short of having an adequate approach, since interpolation is a global concept by its very nature and should not be considered in a local setting.

Among the methods that we have encountered in literature very common are appropriate generalizations of classical interpolation techniques [4], [16]. The De Casteljau algorithm, which is used to generate polynomial spline curves in Euclidian spaces, is popular due to its simple geometric construction, based on the application of successive linear interpolations. A sequence of $d-1$ points known as control points is used to construct a polynomial of degree d joining two given points in $\mathbb{R}^n$. A concatenation of polynomial segments generates a spline curve. The natural generalization of a straight line in a manifold is the minimizing geodesic. The existence of *unique* minimizing geodesics between the points to be interpolated is the key idea in the extension of the De Casteljau algorithm to non-Euclidian spaces, in particular to complete Riemannian manifolds. However, in practice the algorithm can be implemented only when the computation of the minimizing geodesics is feasible and affordable. Numerous examples in literature reveal that this is a computationally expensive task even in simple cases. In addition, the modified De Casteljau algorithm is used as a computational device to construct at most C^1-splines on manifolds, since the computational cost increases substantially when higher degree of smoothness of the interpolating curve is required. Regarding this issue, reductions in complexity were recently presented in the work [18], in which the authors introduced a smoothing function which allows to construct C^k-polynomial splines on complete Riemannian manifolds in three steps. The choice of the smoothing function depends on the degree of smoothness k prescribed in advance. In case of compact Lie groups there is an easy construction of a curve on the group, connecting two points, which makes the algorithm easy to implement. In particular, for $Q_k, Q_{k+1} \in \mathrm{SO}(n)$ the corresponding curve has the form $Q_k e^{t \log(Q_k^T Q_{k+1})}$ with $t \in [0,1]$. We would like to point out that the issue of uniqueness of such curves, which does not seem to be covered in [18], is closely related to the nature of the logarithmic function. Due to its importance, we will discuss in the sequel the role of the logarithm in interpolation in considerably greater detail.

There have been many attempts to solve similar interpolation problems on Lie groups in terms of the coordinates of the embedding space [10], [13]. The idea is to find a suitable mapping in order to express the information in a linear space, solve the interpolation problem there and pull the trajectory back to a manifold. Popular SO(3) interpolation algorithms adopt various re-parametrizations of the rotation matrices (e.g. rotation axes and angles, unit quaternions) and perform cubic spline interpolation based on such representations [10]. These algorithms, however, often do not generalize to higher-dimensional manifolds. Recently the authors of [7] proposed to combine the pull back/push forward technique with rolling a manifold (e.g. sphere, $\mathrm{SO}(n)$ or a Graßmann manifold) on its affine tangent space like a rigid body. In this manner the rolling motion is described by the action of the Euclidian group $\mathrm{SE}(n)$ on the embedding space. The interpolation problem is then solved on the affine space and rolled back to the manifold, avoiding in this manner trajectory distortions. The solution is obtained in closed form which is convenient for implementations. However, the outlined method is only applied to a localized set of data on the manifold, i.e. situated within a single chart with respect to a chosen local diffeomorphism.

In this work we adopt a projection technique that exploits the mathematical elegance of the Lie group $\mathrm{SO}(n)$ and the corresponding Lie algebra $\mathfrak{so}(n)$. Note that the exponential

map $\exp : \mathfrak{so}(n) \longrightarrow \mathrm{SO}(n)$ is surjective, which implies that for any $Q \in \mathrm{SO}(n)$ there exists at least one $A \in \mathfrak{so}(n)$ such that $Q = \exp(A)$. Consequently, we would like to transfer the data onto the Lie algebra $\mathfrak{so}(n)$, interpolate the points using tools of the classical interpolation theory for linear spaces, and then map the path obtained in the algebra back to the group by the exponential map. Exponential representation of the interpolation curve insures relatively easy means to obtain high degrees of smoothness, which is a main source of complexity in the algorithms mentioned earlier. The key point in our approach is a novel projection method from the group to the algebra, which is presented in the sequel. The necessity of the method is justified by a simple observation that the exponential map is not injective, so a "natural" way to choose one of the corresponding preimages in the Lie algebra space is often not apparent. In particular, we will provide a simple example showing that using standard logarithmic map for the projection may create large distortions of an approximating curve. For the implementation of the algorithm we require that any two consecutive interpolation points are situated within a single chart (with exponential map as a local diffeomorphism), but the whole set of data we consider globally on $\mathrm{SO}(n)$. Each step in the algorithm is based only on the knowledge of the projection in the previous step, which in turn is obtained via a specially constructed, rapidly convergent iterative procedure.

Lie groups can provide a nonlinear motion on manifolds by means of Lie group actions. A manifold which is transitively acted upon by a Lie group is called a *homogeneous space*. A popular example which has clear geometric interpretation is the unit sphere S^2 which is a homogeneous space with respect to an SO(3)-action. Other well-known examples of homogeneous spaces under an action of $\mathrm{SO}(n)$ are Stiefel manifold, Graßman manifold and a torus. Hence, the question of interpolation in homogeneous spaces is closely related to the question of interpolation in Lie groups. However, the procedure is not as straightforward as it seems due to the presence of *isotropy* for Lie group actions [15]. In general, the isotropy subgroup is defined pointwise on a manifold, and is a set of those elements in the group which fix a given element on the manifold under the group action. For example, for any point x on a sphere S^2 there is a nontrivial isotropy group in SO(3), which is a subgroup of rotations in the plane perpendicular to the vector $\vec{x}$. In case when an isotropy group is nontrivial, it is unclear what elements to use in the interpolation process. However, we believe that the extra freedom which is introduced by isotropy can be used to improve the quality of interpolating function, similarly to an approach that has been pursued in [11], where the authors used isotropy to improve significantly the numerical accuracy of Lie group methods. This important application of interpolation in Lie groups is the major target for our future research. In this work we will use a sphere S^2 acted upon by the rotation group SO(3) only for illustration purposes, meaning that particular rotations for points on the sphere are prescribed in advance.

The rest of the paper is structured as follows: Section 2 presents few relevant notions from the theory of Lie groups. Here we introduce a splitting formula for matrix exponential, which is obtained by solving a linear differential equation on a Lie group using a *Magnus expansion* [12]. Being an interesting theoretical result the formula is also a major tool for the iterative scheme which we propose to use for interpolation; Section 3 reports a failure of simple approaches suggested by techniques from matrix linear algebra which underlies the difficulties of the interpolation problem. The iterative algorithm which is used for rapid computation of interpolating curves is introduced in Section 4; Section 5 reports some numerical examples, which are then followed by a Conclusion.

2 Basic definitions and background theory

We wish to keep Lie-group and Lie-algebra notation and knowledge to minimum and expect the reader to be familiar with basic concepts. These fundamental notions are covered in detail in [1],[19].

When measuring distances between the elements of the group we will consider $\mathrm{SO}(n)$ as naturally embedded into a linear space of matrices $\mathrm{M}_n(\mathbb{R})$. In what follows we will use the spectral norm of a matrix (denoted simply by $\|\cdot\|$), unless stated otherwise. By definition,

$$\|A\| = \max_{x\neq 0} \frac{\|Ax\|_2}{\|x\|_2} = (\text{maximal eigenvalue of } AA^T)^{1/2},$$

where $\|\cdot\|_2$ is the Euclidean norm of a vector. Note that for any $Q \in \mathrm{SO}(n)$ its spectral norm is equal to 1, since $QQ^T = \mathrm{I}$. There are other obvious ways to measure distances between the elements on manifolds, for example along geodesics, which we, however, will not pursue in this paper.

The exponential map for the Lie group $\mathrm{SO}(n)$ is the usual exponential map of matrices, given by the Taylor series

$$\exp:\ A \in \mathfrak{so}(n) \longrightarrow \exp(A) \in \mathrm{SO}(n), \qquad \exp(A) = \sum_{k=0}^{\infty} \frac{A^k}{k!}.$$

Similarly, the logarithm is defined by the following power series:

$$\log:\ Q \in \mathrm{SO}(n) \longrightarrow \log(Q) \in \mathfrak{so}(n), \qquad \log Q = \sum_{k=1}^{\infty} (-1)^{k+1} \frac{(Q-I)^k}{k},$$

which converges for matrices Q with $\|Q - I\| < 1$. The fact that this definition is invalid outside the specified domain causes major problems for interpolation. The logarithm is not unique, which follows from the non-uniqueness of the logarithm of a complex number. The easiest way to see this in this setting is to apply the above power series expansion to the diagonalized form of Q, namely to PDP^{-1}, where P is orthogonal and D is diagonal with eigenvalues of Q as diagonal entries. There are countably many choices of a logarithm branch for each of the eigenvalue and complexity increases exponentially with the dimension of the matrix Q.

We will make use of the following Lie-group and Lie-algebra operators [1], [19]:

$$\begin{aligned} &\mathrm{Ad}_Q E = QEQ^{-1}, && Q \in \mathrm{SO}(n),\ E \in \mathfrak{so}(n),\\ &\mathrm{ad}_A E = [A, E], && A, E \in \mathfrak{so}(n), \end{aligned}$$

where $[\cdot,\cdot]$ is the Lie bracket, which is in this case just the commutator of matrices. There is an important formula, relating Ad, ad and the exponential map:

$$\mathrm{Ad}_{\exp(A)} = \exp(\mathrm{ad}_A).$$

It is also useful to recall the definition of the dexp operator [9],

$$\mathrm{dexp}_A = \left.\frac{e^u - 1}{u}\right|_{u=\mathrm{ad}_A} = \sum_{k=0}^{\infty} \frac{1}{(k+1)!} \mathrm{ad}_A^k, \qquad A \in \mathfrak{so}(n),$$

where ad_A^k is defined as an iterated commutator,

$$\begin{aligned}&\mathrm{ad}_X^0 Y = Y,\\ &\mathrm{ad}_X^k Y = [\mathrm{ad}_X^{k-1}, Y], \qquad X, Y \in \mathrm{M}_n(\mathbb{R}).\end{aligned}$$

We conclude the chapter by considering the following symmetric product

$$e^{-\frac{t}{2}A} e^{t(A+E)} e^{-\frac{t}{2}A}, \tag{2}$$

where $A, E \in \mathfrak{so}(n)$ and $t > 0$ is a real parameter. We restrict E to be small enough so that for $t = 1$ the product (2) is in a small neighborhood of the identity matrix in the group $\mathrm{SO}(n)$. We seek to construct a function $L_A : \mathfrak{so}(n) \longrightarrow \mathfrak{so}(n)$ such that

$$e^{-\frac{t}{2}A} e^{t(A+E)} e^{-\frac{t}{2}A} = e^{L_A(tE)} + R(t, \|E\|^m, \|A\|)$$

with $m \geq 2$. In this representation the higher powers of the norm of E should reduce the influence of $\|A\|$ on the remainder term R. We intend to use the *Magnus expansion* (ME) [12] to obtain a closed expression for the function L_A. This formula is analogous to the *symmetric BCH formula* [8] in a sense that it is obtained by some means of truncating the BCH expansion. The method of truncation, however, is not apparent from the classical form of the symmetric BCH formula. We use ME since it gives a tool to collect the terms which have the same order in powers of $\|E\|$. Furthermore, the application of ME implies the convergence condition, which, as we will see later, does not depend on the size of the matrix A, whereas from the general theory for the BCH formula we know that it is convergent only for matrices A and E of small norm.

Let us regard t as a variable and differentiate the function $Y(t) = e^{-\frac{t}{2}A} e^{t(A+E)} e^{-\frac{t}{2}A}$ with respect to it. Then

$$Y'(t) = U(t)Y(t) - Y(t)W(t), \quad Y(0) = I, \tag{3}$$

where $U(t) = \frac{1}{2} e^{-\frac{t}{2}A} E e^{\frac{t}{2}A} = \frac{1}{2} e^{-t\mathrm{ad}_{\frac{1}{2}A}} E$ and $W(t) = -\frac{1}{2} e^{\frac{t}{2}A} E e^{-\frac{t}{2}A} = -\frac{1}{2} e^{t\mathrm{ad}_{\frac{1}{2}A}} E$.
We follow the method suggested in [8] to solve the above equation in terms of ME. Using the same notation, we let $P(t) = U(t) - W(t) = \cosh(t\mathrm{ad}_{\frac{t}{2}A})E$ and $Q(t) = U(t) + W(t) = -\sinh(t\mathrm{ad}_{\frac{1}{2}A})E$.

When writing $f(\mathrm{ad}_X)$, where f is some function, we will understand this expression as a power series expansion $f(\mathrm{ad}_X) = \sum_{k=0}^{\infty} \alpha_k \mathrm{ad}_X^k$, where $f(z) = \sum_{k=0}^{\infty} \alpha_k z^k$ is the Taylor expansion of a function in its domain of analyticity. Also $f(\mathrm{ad}_X)$ is understood as a function of a matrix, since a commutation operator ad_X is linear and therefore can be represented by a matrix. We refer the reader to [5], [6] for thorough discussion on the topic of functions of matrices.

According to [8] we seek a solution to (3) in the exponential form $Y(t) = e^{\Omega(t)}$, which leads to the differential equation for Ω,

$$\Omega' = \mathrm{dexp}_{\Omega}^{-1} P - \frac{1}{2}\mathrm{ad}_{\Omega} Q, \qquad \Omega(0) = O.$$

The solution of the above equation can be expressed in terms of the series

$$\Omega(t) = \sum_{k=1}^{\infty} \Omega_k(t), \tag{4}$$

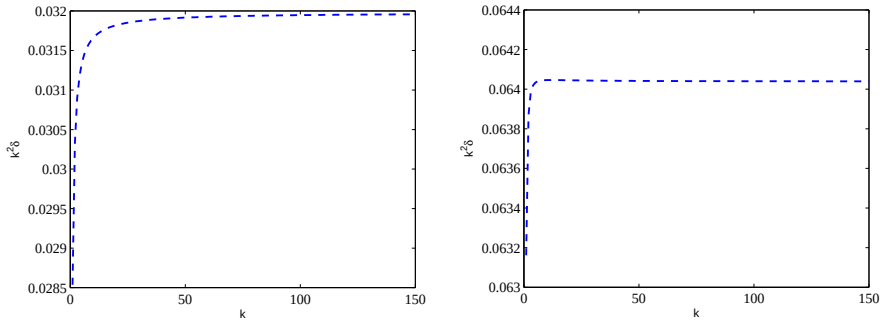

Figure 1: Here $\delta\|E\|^2 = \left\| e^{-\frac{1}{2}A} e^{A+\frac{1}{k}E} e^{-\frac{1}{2}A} - \exp\left(\frac{1}{k}\frac{\sinh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}}E\right)\right\|$. In the case of the left-hand side plot $\|A\| = 5.8528$ and $\|E\| = 0.7388$. In the case of the right-hand side plot $\|A\| = 14.011$ and $\|E\| = 1.1934$.

similarly to the classical Magnus expansion, where Ω_k are nested commutators of the functions P, Q. Note that

$$P(t) = \cosh(t\mathrm{ad}_{\frac{1}{2}A})E = \sum_{k=0}^{\infty} \frac{t^{2k}}{(2k)!}\mathrm{ad}_{\frac{1}{2}A}^{2k}E.$$

Then by definition

$$\begin{aligned}\Omega_1 = \int_0^t P(x)dx &= \sum_{k=0}^{\infty} \frac{1}{(2k)!}\int_0^t x^{2k}dx\,\mathrm{ad}_{\frac{1}{2}A}^{2k}E \\ &= \sum_{k=0}^{\infty} \frac{t^{2k+1}}{(2k+1)!}\mathrm{ad}_{\frac{1}{2}A}^{2k}E = \frac{\sinh(t\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}}E.\end{aligned}$$

Likewise, the second term is

$$\begin{aligned}\Omega_2 &= -\frac{1}{2}\int_0^t \left[\int_0^x P(\xi)d\xi, Q(x)\right] dx \\ &= \frac{1}{2}\int_0^t \left[\sum_{k=0}^{\infty} \frac{x^{2k+1}}{(2k+1)!}\mathrm{ad}_{\frac{1}{2}A}^{2k}E, \sinh(x\mathrm{ad}_{\frac{1}{2}A})E\right] dx \\ &= \frac{1}{2}\sum_{k=0}^{\infty} \frac{1}{(2k+1)!}\left[\mathrm{ad}_{\frac{1}{2}A}^{2k}E, \int_0^t x^{2k+1}\sinh(x\mathrm{ad}_{\frac{1}{2}A})E dx\right].\end{aligned}$$

It follows that $\Omega_2 = \mathcal{O}(t^3)$. Since Ω_k, $k \geq 2$ are generated recursively, we deduce that

$$e^{-\frac{t}{2}A}e^{t(A+E)}e^{-\frac{t}{2}A} = \exp\left(\frac{\sinh(t\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}}E\right) + \mathcal{O}(t^3). \tag{5}$$

If the parameter t is set to 1, then $\Omega_1 = \frac{\sinh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}}E$. With a little more effort (see Appendix A for details), which includes applying the Taylor series expansion for the sinh function and taking the integral, we can compute

$$\Omega_2 = \frac{1}{2}\sum_{k=0}^{\infty}\sum_{l=0}^{\infty} \varphi_{k,l}[\mathrm{ad}_{\frac{1}{2}A}^{2k}E, \mathrm{ad}_{\frac{1}{2}A}^{2l+1}E],$$

where $\varphi_{k,l} = \sum_{m=0}^{l} \left(\frac{1}{(2m+1)!(2l+2k+2-2m)!} - \frac{1}{(2m)!(2l+2k+3-2m)!} \right)$. Note that when l, k tend to infinity, the coefficients decay faster than $1/(k+l+1)!$.

Again due to the recursive nature of higher order terms in the expansion (4) we conclude that

$$e^{-\frac{1}{2}A} e^{A+E} e^{-\frac{1}{2}A} = \exp\left(\frac{\sinh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}} E \right) + \mathcal{O}(\|E\|^2). \tag{6}$$

For the numerical examples (Figure 1) illustrating formula (6), we consider $e^{-\frac{1}{2}A} e^{A+\frac{1}{k}E} e^{-\frac{1}{2}A} - \exp\left(\frac{1}{k} \frac{\sinh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}} E \right)$. The graphs display the remainder scaled by $k^2/\|E\|^2$, which is surprisingly small even when the matrix $\|A\|$ is of large norm.

Regarding the condition for existence of $\Omega(t)$ as well as convergence of the series (4), both could be deduced from the classical case of Magnus series expansion for the fundamental solution of a linear differential system, presented in exponential form [9], and should look like

$$\int_0^t \|P(x)\| dx, \int_0^t \|Q(x)\| dx \leq C,$$

for some constant $C > 0$. However, to our knowledge the optimal value of C, unlike in the classical case [14], is not presently available. Nevertheless, using the fact that $\|X\|_F = \|\nu(X)\|_2$, where $\|\cdot\|_F$, $\|\cdot\|_2$ are the Frobenius and Euclidean norms respectively, and ν is a natural embedding which "stretches" a matrix into a vector, we can derive the following estimate for the function $P(x)$,

$$\begin{aligned} \|P(x)\|_F &= \|\cosh(\mathrm{ad}_{\frac{x}{2}A}) E\|_F \\ &= \|\cosh(C_{\frac{x}{2}A}) \nu(E)\|_2 \leq \|\cosh(C_{\frac{x}{2}A})\|_2 \|\nu(E)\|_2 \leq \|\nu(E)\|_2, \end{aligned}$$

where $C_{\frac{x}{2}A}$ is the matrix corresponding to the adjoint operator in the embedded space. The last estimate is due to the fact that the spectrum of $C_{\frac{x}{2}A}$ in case of a skew-symmetric matrix A is purely imaginary [3], so the hyperbolic cosine is just normal cosine and consequently $\|\cosh(C_{\frac{x}{2}A})\|_2 \leq 1$. It is easy to see that the same result is valid for $Q(x)$. We conclude that existence of the exponential solution $Y(t) = e^{\Omega}$ for (3) for some fixed t and convergence of the series expansion for Ω are predicated by the size of the matrix E only. This is also justified by the numerical examples above.

3 Naive approach to curve interpolation

An obvious way to generate an orthogonal matrix from an arbitrary matrix is to take its QR factorization and leave the upper-triangular component out. So in order to generate a curve satisfying (1), we can simply interpolate the corresponding coefficients of the given matrices for example by splines and then take QR factorization of the resulting matrix function. We realize this approach on a spiral curve $P(t)v$, where $v \in S^2$ and $P(t) \in \mathrm{SO}(3)$ (Figure 2). It is easy to see from the plot that disregarding the topology of SO(3), which is what we do when taking QR factorization of $P(t) = Q(t)R(t)$, leads to large distortion of the interpolating curve $Q(t)v$ from the original curve $P(t)v$ and therefore provides poor approximation results, as well as lacks smoothness. Superior alternative is to make use of tools specific to $\mathrm{SO}(n)$ and the obvious choice is the exponential map. Since the exponential map $\exp : \mathfrak{so}(n) \longrightarrow \mathrm{SO}(n)$ is surjective, for any $X \in \mathrm{SO}(n)$, there exists

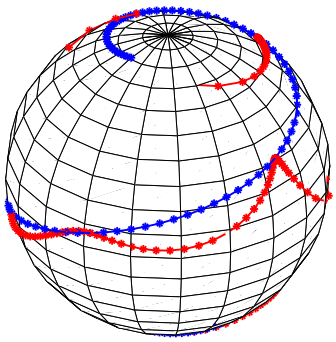

Figure 2: The spiral curve is $P(t)v \in S^2$, where $P(t)$ is a curve in SO(3). QR factorization is applied to $P(t) = Q(t)R(t)$ and the curve $Q(t)v$ is plotted.

$Y \in \mathfrak{so}(n)$ such that $e^Y = X$. Letting $Q_r = e^{A_r}$ for $r = 1, \ldots, m$, we can interpolate the $A_r s$ in $\mathfrak{so}(n)$ by a function $R : [0,1] \longrightarrow \mathfrak{so}(n)$. This can be accomplished easily since $\mathfrak{so}(n)$ is an Euclidean space. Therefore $Q(s) = \exp(R(s))$ is a suitable interpolation function.

Having the exponential trivialization $X = e^Y$ at hand, it seems obvious that in order to find Y, given X, we need to take the logarithm. However, a logarithm is a multi-valued function, so to render it single-valued we can require that it resides in the principal branch (this is, for instance, the approach used by the `logm` function in MATLAB). For the time being we assume that -1 does not belong to the spectrum $\sigma(X)$ of a matrix X for any element $X \in \mathrm{SO}(n)$ we will be considering, in which case the principal logarithm $\log(X)$ is well-defined.

Consider a curve $Q(t) = e^{tA}$ for some $A \in \mathfrak{so}(n)$. Let t_1 be such that $t_1\rho(A) < \pi$ and t_2 such that $t_2\rho(A) > \pi$, and let $L(t)$ be a line segment in the algebra, interpolating $\log(\exp(t_1 A))$ and $\log(\exp(t_2 A))$ with log being the principal logarithm. We can see from the plot (Figure 3) that the interpolating curve $e^{L(t)}$ deviates significantly from the curve e^{tA} within the chosen interval and is useless for applications. In order to have meaningful approximation results we need to be able to follow the correct branch of the logarithm.

One could argue that with a suitable parametrization such that $Q_1 = e^{t_1 A}$, $Q_2 = e^{t_2 A}$ are within one chart, we can avoid the above problem. This can be easily done via the left translation on the group, using the fact that geodesic curves are invariant under this action. In other words, the curve segment $Q_1 e^{t \log Q_1^T Q_2}$ for $t \in [0,1]$ will coincide with e^{tA} for $t \in [t_1, t_2]$. Indeed, but we can obtain this only locally, meaning that the remaining challenge is how to smoothly patch up different approximations for distinct segments.

This is a manifestation of a deeper problem at the heart of our research: to reconcile approximation, which by its very nature is a global concept, with the topology of a group or, with greater generality, a smooth manifold.

4 Following the correct branch: an iterative algorithm

We focuss our attention on developing a constructive approach to interpolate an ordered set of points $Q_1, Q_2, \ldots, Q_m$ in $\mathrm{SO}(n)$. Recall that simply taking principal logarithms of the

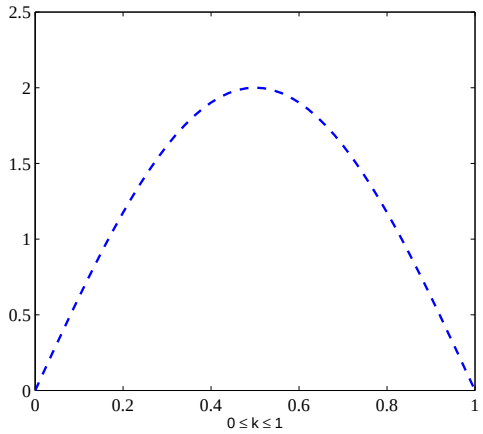

Figure 3: The curve represents the distance $d\left(e^{(1-k)t_1A+kt_2A}, e^{(1-k)t_1A+k\log(e^{t_2A})}\right)$, where $t_2 - t_1 = 1/10$. The maximum of the distance is equal 2.

points in $\mathrm{SO}(n)$ renders approximation of the true solution by the interpolation function next to useless. Consequently, we want to be able to move away from the principal branch in a smooth manner to the "correct" branch whenever necessary.

We impose a restriction that any two neighboring elements Q_r, Q_{r+1} in the group are sufficiently close to each other, by which we mean that $e^{-\frac{1}{2}A_r}Q_{r+1}e^{-\frac{1}{2}A_r}$ is in a small neighborhood of the identity in the group $\mathrm{SO}(n)$, and consequently its logarithm resides in the principal branch. Assume that for some r we already know A_r such that $e^{A_r} = Q_r$. (Without loss of generality we may assume that $Q_1 = I$, whereby $A_1 = O$.) We want to find $A_{r+1} = A_r + E_{r+1}$ such that $Q_{r+1} = e^{A_{r+1}}$ and the difference $A_{r+1} - A_r = E_{r+1}$ has small norm. Observe that formula (6) gives an implicit equation for the unknown E_{r+1}. Whereas we cannot solve the equation, we can use it to approximate E_{r+1}. Consider the following expression:

$$\widetilde{E}_{r+1} = \mathrm{T}_{A_r} \log(e^{-\frac{1}{2}A_r}Q_{r+1}e^{-\frac{1}{2}A_r}), \tag{7}$$

where $\mathrm{T}_{A_r} = \frac{\frac{1}{2}\mathrm{ad}_{A_r}}{\sinh(\frac{1}{2}\mathrm{ad}_{A_r})}$ is a linear operator. The operator T_{A_r} is defined by the values of the function $f(z) = \frac{z/2}{\sinh z/2}$ on the spectrum of ad_{A_r} which is purely imaginary [3], so $f(ix) = \frac{ix/2}{\sinh(ix/2)} = \frac{x/2}{\sin(x/2)}$. Let us assume that the spectrum of the adjoint operator does not contain the points $2\pi ik,\ k \in \mathbb{Z}\backslash\{0\}$, at which the function $f(z) = \frac{z/2}{\sinh z/2}$ is singular (see Figure 4). Then the operator T_{A_r} is bounded.

Theorem 1 *For $\widetilde{E}_{r+1}$ defined by (7) we have*

$$\widetilde{E}_{r+1} = E_{r+1} + \mathcal{O}(\|\mathrm{T}_{A_r}\|\|E_{r+1}\|^2),$$

where E_{r+1} is such that $e^{A_r+E_{r+1}} = Q_{r+1}$.

Proof The proof is straightforward and is a variation of the formula (6),

$$\frac{\frac{1}{2}\mathrm{ad}_{A_r}}{\sinh(\frac{1}{2}\mathrm{ad}_{A_r})} \log(e^{-\frac{1}{2}A_r}Q_{r+1}e^{-\frac{1}{2}A_r}) = E_{r+1} + \mathcal{O}(\|\frac{\frac{1}{2}\mathrm{ad}_{A_r}}{\sinh(\frac{1}{2}\mathrm{ad}_{A_r})}\|\|E_{r+1}\|^2).$$

□

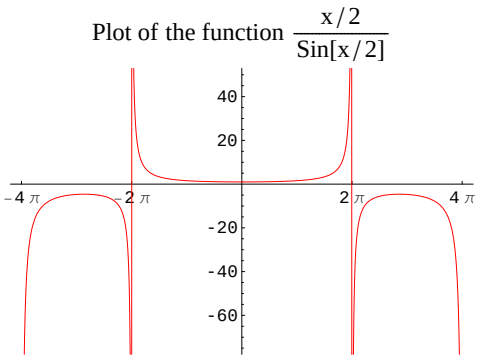

Figure 4: The function is singular at points $2\pi k$, $k \in \mathbb{Z}\backslash\{0\}$. The absolute value of the function $|\frac{x/2}{\sin x/2}| \geq 1$ for any x.

Corollary 1 *The following estimate holds*

$$\|e^{A_r+E_{r+1}} - e^{A_r+\widetilde{E}_{r+1}}\| \leq C\|\widetilde{E}_{r+1}\|^2,$$

where $C > 0$ is a small constant, compared to the norm of A_r.

Proof Since $e^{-\frac{1}{2}A_r}$ is an orthogonal matrix, the following equality holds

$$\|e^{A_r+E_{r+1}} - e^{A_r+\widetilde{E}_{r+1}}\| = \|e^{-\frac{1}{2}A_r}e^{A_r+E_{r+1}}e^{-\frac{1}{2}A_r} - e^{-\frac{1}{2}A_r}e^{A_r+\widetilde{E}_{r+1}}e^{-\frac{1}{2}A_r}\|. \tag{8}$$

It remains to apply the formula (6) to the term $e^{-\frac{1}{2}A_r}e^{A_r+\widetilde{E}_{r+1}}e^{-\frac{1}{2}A_r}$ and then substitute the formula (7) for $\widetilde{E}_{r+1}$ into the expression for the exponential. The difference (8) therefore reduces to the remainder term $\mathcal{O}(\|\widetilde{E}_{r+1}\|^2)$ from the formula (6) and so the proof is complete. □

It follows from the corollary that when the norm of $\widetilde{E}_{r+1}$ is small we can take the principal logarithm of

$$e^{-\frac{1}{2}(A_r+\widetilde{E}_{r+1})}Q_{r+1}e^{-\frac{1}{2}(A_r+\widetilde{E}_{r+1})}$$

and repeat the same construction with $A_r + \widetilde{E}_{r+1}$. In other words we can generate an iterative process in order to approximate A_{r+1}:

$$\begin{aligned} &A_{r+1}^{[1]} = A_r \\ &A_{r+1}^{[k+1]} = A_{r+1}^{[k]} + E_{r+1}^{[k]}, \\ &\text{where } E_{r+1}^{[k]} = \frac{\frac{1}{2}\mathrm{ad}_{A_{r+1}^{[k]}}}{\sinh(\frac{1}{2}\mathrm{ad}_{A_{r+1}^{[k]}})}\log\big(e^{-\frac{1}{2}A_{r+1}^{[k]}}Q_{r+1}e^{-\frac{1}{2}A_{r+1}^{[k]}}\big),\ k = 1, 2, \ldots. \end{aligned} \tag{9}$$

Let us omit the subscript $r+1$ to simplify the notation. Since

$$\begin{aligned} e^{-\frac{1}{2}(A^{[1]}+E^{[1]})}Qe^{-\frac{1}{2}(A^{[1]}+E^{[1]})} &= e^{-\frac{1}{2}(A^{[1]}+E^{[1]})}e^{A^{[1]}+E^{[1]}+(E-E^{[1]})}e^{-\frac{1}{2}(A^{[1]}+E^{[1]})} \\ &= \exp\left(\frac{\sinh(\frac{1}{2}\mathrm{ad}_{A^{[1]}+E^{[1]}})}{\frac{1}{2}\mathrm{ad}_{A^{[1]}+E^{[1]}}}(E-E^{[1]})\right) + \mathcal{O}(\|E-E^{[1]}\|^2), \end{aligned}$$

applying Lemma 1 we obtain

$$E^{[2]} = E - E^{[1]} + \mathcal{O}(\|T_{A^{[2]}}\|\|E-E^{[1]}\|^2).$$

Inductively,

$$E^{[k]} = E - \sum_{l=1}^{k-1} E^{[l]} + \mathcal{O}\left(\|\mathrm{T}_{A^{[k]}}\| \|E - \sum_{l=1}^{k-1} E^{[l]}\|^2 \right) \quad \text{for} \quad k \geq 2.$$

Hence,

$$A_{r+1} - A_{r+1}^{[k+1]} = \mathcal{O}(\|\mathrm{T}_{A_{r+1}^{[k]}}\| \|A_{r+1} - A_{r+1}^{[k]}\|^2). \tag{10}$$

If we have a uniform bound on the norms of the operator $T_{A_{r+1}^{[k]}}$ for $k \geq 1$, the above relation indicates quadratic convergence of the iterative algorithm and is an analogue to the formula for error growth of the famous Newton–Raphson method for finding roots of functions. Unless we hit a point of singularity of the function $f(z) = \frac{z/2}{\sinh z/2}$, we can still achieve the convergence by choosing the interpolation points sufficiently close to each other. The quadratic convergence will be illustrated on the numerical examples presented in the next chapter.

5 Implementation and numerical experiments

Implementation of the iterative algorithm (9) involves computation of the matrix of the commutator ad_{A_r} and then evaluation of the function $f(z) = \frac{z/2}{\sinh z/2}$ from this matrix. This can be done by definition, i.e. by computing the eigenvalue-eigenvector decomposition of the corresponding matrix and then evaluating f on the eigenvalues [6]. However, this approach becomes computationally expensive when n is large. An alternative is first to use a so-called *reduced commutator matrix*, introduced in [3], and then the *diagonal Padé approximation* to the function f [2]. The reduced commutator matrix $C_A \in \mathrm{M}_m(\mathbb{R})$ is obtained if we consider a restricted embedding $\nu : \mathfrak{so}(n) \longrightarrow \mathbb{R}^m$, where $m = \frac{1}{2}n(n-1)$ is the dimension of the Lie algebra $\mathfrak{so}(n)$. In other words, C_A is defined by the relation

$$\nu(\mathrm{ad}_A B) = C_A \nu(B), \qquad B \in \mathfrak{so}(n).$$

In [3] the authors developed an effective algorithm for assembling the reduced commutator matrix C_A of an arbitrary dimension m using directed graphs. The function $f = \frac{z/2}{\sinh z/2}$ is even and analytic in the circle of radius 2π about the origin, so it can be expanded into power series $f(z) = \sum_{k=0}^{\infty} \alpha_{2k} z^{2k}$. Then $f(\mathrm{ad}_A) = \sum_{k=0}^{\infty} \alpha_{2k} \mathrm{ad}_A^{2k}$ and for $B \in \mathfrak{so}(n)$,

$$\begin{aligned} \nu(f(\mathrm{ad}_A)B) &= \nu(\sum_{k=0}^{\infty} \alpha_{2k} \mathrm{ad}_A^{2k} B) = \sum_{k=0}^{\infty} \alpha_{2k} \nu(\mathrm{ad}_A^{2k} B) \\ &= \sum_{k=0}^{\infty} \alpha_{2k} C_A^{2k} \nu(B) = f(C_A)\nu(B). \end{aligned}$$

Going back to the iterative algorithm (9), let $B^{[k]} = \log(e^{-\frac{1}{2}A^{[k]}} Q e^{-\frac{1}{2}A^{[k]}})$ and $N(C_{A^{[k]}})/P(C_{A^{[k]}})$ be the (p,p)-Padé approximation to $f(C_{A^{[k]}})$, were N, P are corresponding polynomials of degree p. Then at each iteration step finding the correction matrix $E^{[k]}$ reduces to solving the linear system

$$P(C_{A^{[k]}})\nu(E^{[k]}) = N(C_{A^{[k]}})\nu(B^{[k]}),$$

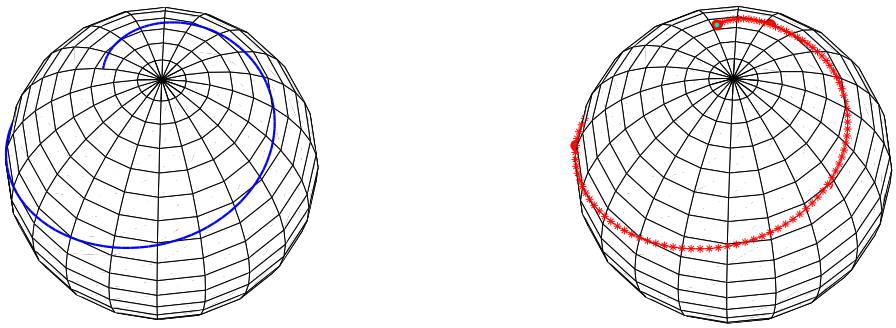

Figure 5: The curve constructed using the iterative algorithm (the right plot) coincides with the original curve (the left plot).

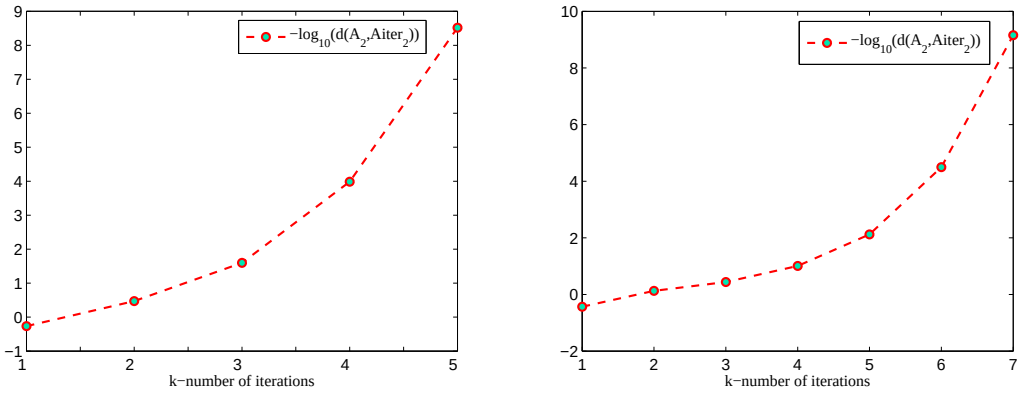

Figure 6: $\sigma(A_1)/\pi = \{\pm 9.14i, \pm 2.92i, \pm 6.32i\}$, and $\sigma(A_2)/\pi = \{\pm 9.23i, \pm 3.08i, \pm 5.95i\}$ for the left plot. $\sigma(A_1)/\pi = \{\pm 7.03i, \pm 4.03i, \pm 3.06i, \pm 1.37i, \pm 0.37i\}$, and $\sigma(A_2)/\pi = \{\pm 7.20i, \pm 3.99i, \pm 2.99i, \pm 1.69i, \pm 0.13i\}$ for the right plot.

with respect to the coefficients of the vector $\nu(E^{[k]})$.

In case of the rotation group SO(3) the logarithm as well as the operator $\mathrm{T}_{A_{r+1}^{[k]}}$ can be derived in closed form, so the above simplifications are not necessary. Specifically, for $Q \in \mathrm{SO}(3)$ such that $\mathrm{Tr}(Q) \neq 1$ we have

$$\log Q = \frac{\theta}{2\sin\theta}(Q - Q^T),$$

where θ satisfies $1 + 2\cos\theta = \mathrm{Tr}(Q)$. For $A \in \mathfrak{so}(3)$,

$$T_A = \frac{\mathrm{ad}_A}{\sinh(\mathrm{ad}_A)} = I + \left(\frac{1}{\theta\sin\theta} - \frac{1}{\theta^2}\right)\mathrm{ad}_A^2,$$

where $\|A\|^2 = \theta^2$.

Our numerical examples are organized as follows. To illustrate the performance of the algorithm (9), we use a sphere S^2 in the Euclidian space $\mathbb{R}^3$. We sample a curve $P(t)v$ on the sphere to obtain the discrete set of data in SO(3). Subsequently, we apply the iterative process (9) to find the corresponding elements in the algebra $\mathfrak{so}(3)$, then interpolate these points by splines and project the spline curve $S(t)$ back to the group

using the exponential map. In Figure 5 the left-hand side plot displays the original curve $P(t)v$ and the right-hand side plot displays the interpolating curve $\exp(S(t))v$, which in this case is indistinguishable from the original curve.

For the examples in Figure 6 we deliberately generated two matrices of large norm such that some of the eigenvalues jump over singularity points, since for these cases the norm of the operator $\mathrm{T}_{A_{r+1}^{[k]}}$ can be large. Given the matrix A_1 we apply the iterative scheme to approximate the matrix A_2. The algorithm still exhibits quadratic convergence.

6 Conclusion and further remarks

Given any two elements Q_r, Q_{r+1} in a Lie group $\mathrm{SO}(n)$, sufficiently close to each other, and a matrix A_r in the corresponding Lie algebra $\mathfrak{so}(n)$ such that $e^{A_r} = Q_r$, we have introduced a numerical algorithm for approximating A_{r+1}, which is a pre-image of Q_{r+1} under the exponential map and is the closest to A_r among other pre-images. Analysis shows that the algorithm exhibits quadratic convergence. The algorithm has immediate application to the interpolation problem in $\mathrm{SO}(n)$. Using the exponential trivialization of the elements of the group we can project the points to the corresponding Lie algebra via the introduced algorithm and then apply standard techniques of interpolation theory for linear spaces. Practical implementation of the algorithm involves computation of the function of a commutation matrix $\frac{\mathrm{ad}_{\frac{1}{2}A_{r+1}^{[k]}}}{\sinh(\mathrm{ad}_{\frac{1}{2}A_{r+1}^{[k]}})}$. A possible way to reduce the complexity of the algorithm is to use diagonal Padé approximation to the function $\frac{x/2}{\sinh x/2}$. In this case one needs to solve a linear system of equations of dimension $\frac{1}{2}n(n+1)$ with respect to the coefficients of the unknown matrix $E_{r+1}^{[k]}$.

As part of future work we intend to investigate the distance between an arbitrary (sufficiently smooth) function sampled at some points on a Lie group G and an interpolation function in G, constructed via the above algorithm, using the intrinsic geometry of the group. A related problem is minimization of this distance with respect to different norms.

A broad field for research is a generalization of the interpolation problem to other Lie groups. In particular for compact semisimple Lie groups it is known that the set of polynomials $p_n(z) = \sum_{-n}^{n} P_k z^k$ with values in a Lie group G, where $z \in S^1$ and $P_k \in \mathrm{M}_p(\mathbb{C})$, is dense in the set the functions $f(z) = \sum_{-\infty}^{+\infty} F_k z^k$ with values in G [17]. In this regime an interesting problem would be to find a minimal degree polynomial $p_n(z)$ passing through a given set of points in G.

We would like to point out that the interpolation method developed in this work can be used for more general interpolation problem when the tangent vectors at interpolation points are prescribed (Hermite interpolation). Let $\{Q_i\}$ and $\{V_i\}$, $i = 1, 2, \ldots, r$ be given. We want to construct an interpolation function $e^{A(t)}$ such that $e^{A(t_i)} = Q_i$ and $\frac{\mathrm{d}}{\mathrm{d}t} e^{A(t)}\big|_{t_i} = V_i$ for the partition $0 = t_1 < t_2 < \ldots < t_0 = 1$. The values $A(t_i) = A_i$ are found via the iterative algorithm (9). From the formula for the derivative of the exponential map it follows immediately that

$$A'(t_i) = \mathrm{dexp}_{A_i}^{-1}(V_i e^{-A_i}),\ i = 1, 2 \ldots, r.$$

Consequently, the interpolation problem is reduced to the Hermite interpolation problem on the corresponding Lie algebra. The situation when higher order derivatives are prescribed at the interpolation points requires further investigation.

References

[1] F. Adams. *Lectures on Lie Groups*. The University of Chicago Press, 1982.

[2] G. A. Baker Jr. *Essentials of Padé Approximants*. Academic Press, 1975.

[3] A. M. Bloch and A. Iserles. Commutators of skew-symmetric matrices. *International Journal of Bifurcation and Chaos*, 15:793–801, 2005.

[4] P. Crouch, G. Kun, and F. S. Leite. The De Casteljau algorithm on Lie groups and spheres. *Journal of Dynamical and Control Systems*, 3:397–429, 1999.

[5] N. J. Higham. Functions of matrices. In L. Hogben, editor, *Handbook of Linear Algebra*. Chapman/CRC Press, 2006.

[6] R. A. Horn and C. R. Johnson. *Topics in Matrix Analysis*. Cambridge Univ. Press, 1991.

[7] K. Hüper and F. S. Leite. On the geometry of rolling and interpolation curves on S^n, $SO(n)$ and Graßmann manifolds. Technical Report SISSA 56/2005/M, International School for Advanced Studies, Trieste, Italy, 2005.

[8] A. Iserles. A Magnus expansion for the equation $Y' = AY - YB$. *J. Comput. Math.*, 19:15–26, 2001.

[9] A. Iserles, H. Munthe-Kaas, S. P. Nørsett, and A. Zanna. Lie-group methods. *Acta Numerica*, 9:215–365, 2000.

[10] I. G. Kang and F. C. Park. Cubic spline algorithms for orientation interpolation. *Int. J. Numer. Engng.*, 46:45–64, 1999.

[11] D. Lewis and P. J. Olver. Geometric integration algorithms on homogeneous manifolds. *Found. Comput. Math.*, 2:363–392, 2002.

[12] W. Magnus. On the exponential solution of differential equations for a linear operator. *Comm. Pure Appl. Math.*, 7:649–767, 1954.

[13] A. Marthinsen. Interpolation in Lie groups. *SIAM J. Numer. Anal.*, 37:269–285, 1999.

[14] P. C. Moan, J. Oteo, and J. Ros. On the existence of the exponential solution of linear differential systems. *Journal of Physics A: Mathematical and General*, 32:5133–5139, 1999.

[15] P. J. Olver. *Equivalence, Invariants, and Symmetry*. Cambridge Univ. Press, 1995.

[16] F. Park and B. Ravani. Bézier curves on Riemannian manifolds and Lie groups with kinematic applications. *ASME J. Mechan. Design*, 117:36–40, 1995.

[17] A. Presley and G. Segal. *Loop Groups*. Oxford Univ. Press, 1986.

[18] R. C. Rodrigues, F. S. Leite, and J. Jakubiak. A new geometric algorithm to generate smooth interpolating curves on Riemannian manifolds. *LMS J. Comput. Math.*, 8:251–266, 2005.

[19] V. S. Varadarajan. *Lie Groups, Lie Algebras, and Their Representations*. Springer-Verlag, 1984.

Appendix A

By expanding $\sinh(x\mathrm{ad}_{\frac{1}{2}A})$ into power series and computing the integral term by term it is possible to verify that

$$\begin{aligned}
&\frac{1}{(2k+1)!}\int_0^1 x^{2k+1}\sinh(x\mathrm{ad}_{\frac{1}{2}A})dx \\
&\qquad = \frac{\cosh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}^{2k+2}}\sum_{l=0}^{k}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2l+1}}{(2l+1)!} - \frac{\sinh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}^{2k+2}}\sum_{l=0}^{k}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2l}}{(2l)!} \\
&\qquad = \frac{\sinh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}^{2k+2}}\sum_{l=k+1}^{\infty}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2l}}{(2l)!} - \frac{\cosh(\mathrm{ad}_{\frac{1}{2}A})}{\mathrm{ad}_{\frac{1}{2}A}^{2k+2}}\sum_{l=k+1}^{\infty}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2l+1}}{(2l+1)!} \\
&\qquad = \sinh(\mathrm{ad}_{\frac{1}{2}A})\sum_{l=0}^{\infty}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2l}}{(2l+2k+2)!} \\
&\qquad\qquad - \cosh(\mathrm{ad}_{\frac{1}{2}A})\sum_{l=0}^{\infty}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2l+1}}{(2l+2k+3)!}
\end{aligned}$$

We substitute the above expression into the formula for Ω_2 and interchange the order of summation. Then

$$\begin{aligned}
\Omega_2 = \frac{1}{2}\sum_{l=0}^{\infty}\Bigg(&\Bigg[\sum_{k=0}^{\infty}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2k}}{(2l+2k+2)!}E,\ \mathrm{ad}_{\frac{1}{2}A}^{2l}\sinh(\mathrm{ad}_{\frac{1}{2}A})E\Bigg] \\
&- \Bigg[\sum_{k=0}^{\infty}\frac{\mathrm{ad}_{\frac{1}{2}A}^{2k}}{(2l+2k+3)!}E,\ \mathrm{ad}_{\frac{1}{2}A}^{2l+1}\cosh(\mathrm{ad}_{\frac{1}{2}A})E\Bigg]\Bigg).
\end{aligned}$$

Substituting the Taylor expansions for sinh and cosh, we obtain

$$\begin{aligned}
\Omega_2 &= \sum_{m=0}^{\infty}\sum_{l=0}^{\infty}\sum_{k=0}^{\infty}\left(\frac{1}{(2m+1)!(2l+2k+2)!} - \frac{1}{(2m)!(2l+2k+3)!}\right)[\mathrm{ad}_{\frac{1}{2}A}^{2k}E, \mathrm{ad}_{\frac{1}{2}A}^{2m+2l+1}E] \\
&= \sum_{k=0}^{\infty}\sum_{m=0}^{\infty}\sum_{l=m}^{\infty}\Bigg(\frac{1}{(2m+1)!(2l+2k+2-2m)!} \\
&\qquad\qquad - \frac{1}{(2m)!(2l+2k+3-2m)!}\Bigg)[\mathrm{ad}_{\frac{1}{2}A}^{2k}E, \mathrm{ad}_{\frac{1}{2}A}^{2l+1}E] \\
&= \sum_{k=0}^{\infty}\sum_{l=0}^{\infty}\varphi_{k,l}[\mathrm{ad}_{\frac{1}{2}A}^{2k}E, \mathrm{ad}_{\frac{1}{2}A}^{2l+1}E],
\end{aligned}$$

and $\varphi_{k,l} = \sum_{m=0}^{l}\left(\frac{1}{(2m+1)!(2l+2k+2-2m)!} - \frac{1}{(2m)!(2l+2k+3-2m)!}\right)$.