SIAM J. MATRIX ANAL. APPL.
Vol. 23, No. 3, pp. 840–862

GENERALIZED POLAR DECOMPOSITIONS FOR THE APPROXIMATION OF THE MATRIX EXPONENTIAL*

A. ZANNA† AND H. Z. MUNTHE-KAAS†

Abstract. In this paper we describe the use of the theory of generalized polar decompositions [H. Munthe-Kaas, G. R. W. Quispel, and A. Zanna, *Found. Comput. Math.*, 1 (2001), pp. 297–324] to approximate a matrix exponential. The algorithms presented have the property that, if $Z \in \mathfrak{g}$, a Lie algebra of matrices, then the approximation for $\exp(Z)$ resides in G, the matrix Lie group of $\mathfrak{g}$. This property is very relevant when solving Lie-group ODEs and is not usually fulfilled by standard approximations to the matrix exponential.

We propose algorithms based on a splitting of Z into matrices having a very simple structure, usually one row and one column (or a few rows and a few columns), whose exponential is computed very cheaply to machine accuracy.

The proposed methods have a complexity of $\mathcal{O}(\kappa n^3)$, with constant κ small, depending on the order and the Lie algebra $\mathfrak{g}$. The algorithms are recommended in cases where it is of fundamental importance that the approximation for the exponential resides in G, and when the order of approximation needed is not too high. We present in detail algorithms up to fourth order.

Key words. matrix exponential, Lie algebra, Lie-group integrator

AMS subject classification. 65M06

PII. S0895479800377551

1. Introduction.

With the recent developments in the theory of Lie-group integration schemes for ordinary differential equations (ODEs) [10], the problem of approximating the matrix exponential has lately received renewed attention. Most Lie-group methods require a number of computations of matrix exponentials from a Lie algebra $\mathfrak{g} \subseteq \mathbb{R}^{n\times n}$ to a Lie group $G \subseteq \mathrm{GL}(n,\mathbb{R})$ that usually constitutes a bottleneck in the numerical implementation of the schemes [5].

The matrix exponentials need to be approximated to the order of the underlying ODE method (hence exact computation is not an issue); however, it is of fundamental importance that such approximations reside in G. In general, this property is not fulfilled by many standard approximations to the exponential function [14] unless the exponential is evaluated exactly.

In some few cases (usually for small dimension) the exponential of a matrix can be evaluated exactly. This happens, for instance, for 3×3 skew-symmetric matrices, whose exponential can be calculated exactly by means of the well-known *Euler–Rodriguez* formula

$$\exp(Z) = I + \frac{\sin\alpha}{\alpha} Z + \frac{1}{2}\left(\frac{\sin(\alpha/2)}{\alpha/2}\right)^2 Z^2, \tag{1.1}$$

where

$$Z = \begin{pmatrix} 0 & z_3 & -z_2 \\ -z_3 & 0 & z_1 \\ z_2 & -z_1 & 0 \end{pmatrix}, \qquad \alpha = (z_1^2 + z_2^2 + z_3^3)^{1/2}$$

*Received by the editors September 5, 2000; accepted for publication (in revised form) by A. Edelman August 24, 2001; published electronically January 23, 2002.
http://www.siam.org/journals/simax/23-3/37755.html
†Institutt for Informatikk, University of Bergen, Høyteknologisenteret, Thormøhlensgate 55, N-5020 Bergen, Norway (hans@ii.uib.no, anto@ii.uib.no).

[13]. Exact formulas for skew-symmetric matrices and matrices in $\mathfrak{so}(p,q)$ can be derived up to dimension eight making use of the *Cayley–Hamilton* theorem [9] with significant savings with respect to approximation techniques [1, 12]. However, for several reasons the algorithms are not practical for larger dimensions. First, they require high powers of the matrix in question (and each matrix-matrix multiplication amounts to $\mathcal{O}(n^3)$ computations). Second, it is well known that the direct use of the characteristic polynomial, for large-scale matrices, may lead to computational instabilities.

The problem of approximating the exponential of a matrix from a Lie algebra to its corresponding Lie group has been recently considered by [4, 3]. In the first paper, the authors construct the approximation by first splitting the matrix $X \in \mathfrak{g}$ as the sum of bordered matrices. Strang-type splittings of order 2 are considered, so that one could apply a Yoshida technique [24], based on a symmetric composition of a basic scheme whose error locally expands in odd powers of time only, to increase the order. In the second paper, the authors consider techniques based on canonical coordinates of the second kind (CCSK) [22]. To follow that approach, it is necessary to choose a basis of the Lie algebra $\mathfrak{g}$. The choice of the basis plays a significant role in the computational complexity of the algorithms [19], and, by choosing *Chevalley bases* [2] which entail a large number of zero structure constants, it is possible to significantly reduce the cost of the methods.

In this paper we consider the problem of approximating to a given order of accuracy

$$F(t,Z) \approx \exp(tZ) \in G, \qquad Z \in \mathfrak{g}, \tag{1.2}$$

so that $F(t,Z) \in G$, where $\mathfrak{g} \subseteq \mathfrak{gl}(\mathbb{R},n)$ and $G \subseteq \mathrm{GL}(\mathbb{R},n)$. The techniques we introduce consist of a Lie-algebra splitting of the matrix Z by means of an iterated *generalized polar decomposition* induced by an appropriate involutive automorphism $\sigma : G \to G$, as discussed in [16]. We introduce a general technique for approximations of arbitrary high order and discuss practical algorithms of order 2, 3, and 4. For large n, these algorithms are very competitive with standard approximations of the exponential function (for example, diagonal Padé approximants).

The paper is organized as follows. In section 2 we discuss the background theory of the polar decomposition on Lie groups and its symmetric version. Such polar decomposition can be used to induce a splitting in the Lie algebra $\mathfrak{g}$. As long as this splitting is *practical* to compute, together with the exponential of each "split" part, it leads to splitting methods for the approximation of the exponential of practical interest.

In section 3 we use the theory developed in section 2 to derive approximations of the exponential function for some relevant matrix Lie groups as $\mathrm{SO}(\mathbb{R},n)$ and $\mathrm{SL}(\mathbb{R},n)$. Methods of order 2, 3, and 4 are discussed in greater detail, together with their computational complexity. The methods are based on splittings in bordered matrices, whose exact exponentials are very easy to compute.

Section 4 is devoted to some numerical experiments in which we illustrate the results derived in this paper; in section 5 we discuss the relation between our approach and another method for the approximation of the exponential in terms of eigenspace and Schur decompositions; and finally section 6 is devoted to some concluding remarks.

2. Background theory. It is usual in differential geometry to denote Lie-group elements with lowercase letters and Lie-algebra elements with uppercase letters,

whether they represent matrices, vectors, or scalars [7]. We adopt this convention throughout this section.

Let G be a Lie group with Lie algebra $\mathfrak{g}$. We restrict our attention to matrix groups, i.e., to the case when $G \subseteq \mathrm{GL}(\mathbb{R}, n)$.

It is known that, provided $\sigma : G \to G$ is an involutive automorphism of G, every element $z \in G$ sufficiently close to the identity can be decomposed in the product

$$z = xy, \tag{2.1}$$

where $y \in G^\sigma = \{w \in G : \sigma(w) = w\}$, the subgroup of elements of G fixed under σ, and $x \in G_\sigma = \{w : \sigma(w) = w^{-1}\}$ is the subset of antifixed points of σ [11, 16]. The set G_σ has the structure of a *symmetric space* [7] and is closed under the product

$$x_1 \cdot x_2 = x_1 x_2^{-1} x_1,$$

as can be easily verified by application of σ to the right-hand side of the above relation. The decomposition (2.1) is called the *generalized polar decomposition* of z in analogy with the case of real matrices with the choice of automorphism $\sigma(z) = z^{-\mathrm{T}}$.

Next set $z = \exp(tZ)$ with $Z \in \mathfrak{g}$. The automorphism σ induces an involutive automorphism $\mathrm{d}\sigma$ on $\mathfrak{g}$ in a natural manner,

$$\mathrm{d}\sigma(Z) = \frac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} \sigma(\exp(tZ)),$$

and it defines a splitting of the algebra $\mathfrak{g}$ into the sum of two linear spaces,

$$\mathfrak{g} = \mathfrak{p} \oplus \mathfrak{k}, \tag{2.2}$$

where $\mathfrak{k} = \{Z \in \mathfrak{g} : \mathrm{d}\sigma(Z) = Z\}$ is a subalgebra of $\mathfrak{g}$, while $\mathfrak{p} = \{Z \in \mathfrak{g} : \mathrm{d}\sigma(Z) = -Z\}$ has the structure of a Lie-triple system, a set closed under the double commutator,

$$A, B, C \in \mathfrak{p} \Longrightarrow [A, [B, C]] \in \mathfrak{p}.$$

To keep our presentation relevant to the argument matter of this paper, we refer the reader to [16, 15] and the references therein for a more extensive treatment of such decompositions. However, it is of fundamental importance to note that the sets $\mathfrak{k}$ and $\mathfrak{p}$ possess the following properties:

$$\begin{aligned} [\mathfrak{k}, \mathfrak{k}] &\subseteq \mathfrak{k}, \\ [\mathfrak{k}, \mathfrak{p}] &\subseteq \mathfrak{p}, \\ [\mathfrak{p}, \mathfrak{p}] &\subseteq \mathfrak{k}. \end{aligned} \tag{2.3}$$

We denote by $\Pi_\mathfrak{p} : \mathfrak{g} \to \mathfrak{p}$ the canonical projection onto the subspace $\mathfrak{p}$ and by $\Pi_\mathfrak{k} : \mathfrak{g} \to \mathfrak{k}$ the projection onto $\mathfrak{k}$. Then,

$$Z = \Pi_\mathfrak{p} Z + \Pi_\mathfrak{k} Z = P + K,$$

where

$$\Pi_\mathfrak{p} Z = \frac{1}{2}(Z - \mathrm{d}\sigma(Z)), \qquad \Pi_\mathfrak{k} Z = \frac{1}{2}(Z + \mathrm{d}\sigma(Z)).$$

Assume that x and y in (2.1) are of the form $x = \exp(X(t))$ and $y = \exp(Y(t))$. Then $X(t) \in \mathfrak{p}$, $Y(t) \in \mathfrak{k}$ and they can be expanded in series

$$X(t) = \sum_{i=1}^{\infty} X_i t^i, \qquad Y(t) = \sum_{i=1}^{\infty} Y_i t^i,$$

where the X_i and Y_i can be explicitly calculated by means of the following recurrence relations:

$$\begin{aligned} X_1 &= P, \\ (i+1)X_{i+1} &= -[X_i, K] + \sum_{\substack{\ell \geq 1 \\ 2\ell \leq i}} c_{2\ell} \sum_{\substack{\ell_1, \ldots, \ell_{2\ell} > 0 \\ \ell_1 + \cdots + \ell_{2\ell} = i}} [X_{\ell_1}, [X_{\ell_2}, \ldots, [X_{\ell_{2\ell}}, P]]], \quad i = 1, 2, \ldots, \end{aligned} \tag{2.4}$$

and

$$\begin{aligned} Y_1 &= K, \\ Y_{2i} &= O, \qquad i = 0, 1, 2, \ldots, \\ 2(2i+1)Y_{2i+1} &= -2\sum_{q=1}^{i} \sum_{\substack{k \geq 1 \\ k \leq q}} \frac{1}{(2k+1)!} \sum_{\substack{k_1, \ldots, k_{2k} > 0 \\ k_1 + \cdots + k_{2k} = 2q}} [Y_{k_1}, \ldots, [Y_{k_{2k}}, Y_{2(i-q)+1}] \ldots] \\ &\quad - \sum_{m=1}^{i} \frac{2(i-m)+1}{(2m)!} \mathrm{ad}_Z^{2m} Y_{2(i-m)+1} \\ &\quad - \sum_{q=0}^{2(i-1)} \sum_{j=0}^{2(i-1)-q} \frac{(-1)^{2i-q-j-1}(j+1)}{(2i-q-j-1)!} \mathrm{ad}_Z^{2i-j-q-1} \\ &\qquad \sum_{\substack{k \geq 1 \\ k \leq q+1}} \frac{1}{(k+1)!} \sum_{\substack{j_1, \ldots, j_k > 0 \\ j_1 + \cdots + j_k = q+1}} [Y_{j_1}, \ldots, [Y_{j_k}, Y_{j+1}] \ldots] \\ &\quad - \sum_{\substack{\ell \geq 1 \\ \ell \leq i}} \frac{1}{(2\ell)!} \sum_{\substack{\ell_1, \ldots, \ell_{2\ell} > 0 \\ \ell_1 + \cdots + \ell_{2l} = 2i}} [Y_{\ell_1}, \ldots, [Y_{\ell_{2\ell}}, P - K] \ldots] \end{aligned} \tag{2.5}$$

(see [25]). Note that $Y(t)$ expands in odd powers of t only. The first terms in the expansions of $X(t)$ and $Y(t)$ are

$$\begin{aligned} X = {} & Pt - \tfrac{1}{2}[P, K]t^2 - \tfrac{1}{6}[K, [P, K]]t^3 \\ & + \Big(\tfrac{1}{24}[P, [P, [P, K]]] - \tfrac{1}{24}[K, [K, [P, K]]] \Big) t^4 \\ & + \Big(\tfrac{7}{360}[K, [P, [P, [P, K]]]] - \tfrac{1}{120}[K, [K, [K, [P, K]]]] - \tfrac{1}{180}[[P, K], [P, [P, K]]] \Big) t^5 \\ & + \mathcal{O}\big(t^6\big), \end{aligned}$$

$$\begin{aligned} Y = {} & Kt - \tfrac{1}{12}[P, [P, K]]t^3 + \Big(\tfrac{1}{120}[P, [P, [P, [P, K]]]] \\ & + \tfrac{1}{720}[K, [K, [P, [P, K]]]] - \tfrac{1}{240}[[P, K], [K, [P, K]]] \Big) t^5 + \mathcal{O}\big(t^7\big). \end{aligned} \tag{2.6}$$

We also consider a symmetric-type generalized polar decomposition,

$$z = xyx, \qquad z = \exp(tZ), \quad x = \exp(X(t)), \quad y = \exp(Y(t)), \tag{2.7}$$

where, as above, $X(t) \in \mathfrak{p}$ and $Y(t) \in \mathfrak{k}$. To compute $X(t)$, we apply σ to both sides of (2.7) to obtain

$$\sigma(z) = \exp(-X(t)) \exp(Y(t)) \exp(-X(t)). \tag{2.8}$$

Isolating the y term in (2.8) and (2.7) and equating the result, we obtain

$$\exp(tZ) = \exp(2X(t)) \exp(tW) \exp(2X(t)), \qquad W = \mathrm{d}\sigma(Z). \tag{2.9}$$

This leads to a differential equation for X which is very similar to the one obeyed by Y in (2.5) [25]. Using the recursions in [25] we obtain recursions for $X(t)$ and $Y(t)$. The first terms are given as

$$\begin{aligned} X(t) = {} & \tfrac{1}{2}Pt + \tfrac{1}{24}[K,[P,K]]t^3 - \Big(\tfrac{1}{1440}[K,[P,[P,[P,K]]]] \\ & + \tfrac{1}{240}[K,[K,[P,K]]]] + \tfrac{1}{360}[[P,K],[P,[P,K]]]\Big)t^5 + \cdots, \end{aligned} \tag{2.10}$$

$$\begin{aligned} Y(t) = {} & Kt + \tfrac{1}{24}[P,[P,K]]t^3 + \Big(\tfrac{1}{1920}[P,[P,[P,[P,K]]]] \\ & - \tfrac{13}{1440}[K,[K,[P,[P,K]]]] - \tfrac{1}{240}[[P,K],[K,[P,K]]]\Big)t^5 + \cdots \end{aligned} \tag{2.11}$$

and both $X(t)$ and $Y(t)$ expand in odd powers of t only.

3. Generalized polar decomposition and its symmetric version for the approximation of the exponential. Assume now that we wish to approximate $\exp(tZ)$ for some $Z \in \mathfrak{g}$, and that σ_1 is an involutive automorphism so that the exponential of terms in $\mathfrak{p}_1 = \{X \in \mathfrak{g} : \mathrm{d}\sigma_1 X = -X\}$ as well as analytic functions of $\mathrm{ad}_P = [P, \cdot]$ are easy to compute. Then $\mathfrak{g} = \mathfrak{p}_1 \oplus \mathfrak{k}_1$, and we can approximate

$$\exp(tZ) \approx \exp(X^{[1]}(t)) \exp(Y^{[1]}(t)), \tag{3.1}$$

where $X^{[1]}$ and $Y^{[1]}$ obey the order conditions (2.4)–(2.6) to suitable order.

Alternatively, we can approximate

$$\exp(tZ) \approx \exp(X^{[1]}(t)) \exp(Y^{[1]}(t)) \exp(X^{[1]}(t)), \tag{3.2}$$

where $X^{[1]}$ and $Y^{[1]}$ now obey the order conditions (2.10)–(2.11) to given accuracy.

The same mechanism can be applied to split $\mathfrak{k}_1$ in $\mathfrak{p}_2 \oplus \mathfrak{k}_2$ by means of a suitable automorphism σ_2. The procedure can be iterated and, provided that the exponential of $\mathfrak{k}_m$ is easy to compute, we have an algorithm to approximate $\exp(tZ)$ to a given order of accuracy. In this circumstance, (3.1) will read

$$\exp(tZ) \approx F(t,Z) = \exp(X^{[1]}(t)) \cdots \exp(X^{[m]}(t)) \exp(Y^{[m]}(t)), \tag{3.3}$$

while the analogue of (3.2) is

$$\begin{aligned} \exp(tZ) & \approx F(t,Z) \\ & = \exp(X^{[1]}(t)) \cdots \exp(X^{[m]}(t)) \exp(Y^{[m]}(t)) \exp(X^{[m]}(t)) \cdots \exp(X^{[1]}(t)), \end{aligned} \tag{3.4}$$

both corresponding to the algebra splitting

$$\mathfrak{g} = \mathfrak{p}_1 \oplus \cdots \oplus \mathfrak{p}_m \oplus \mathfrak{k}_m. \tag{3.5}$$

3.1. On the choice of the automorphisms σ_i. In what follows, we will consider automorphisms σ of the form

$$\sigma(z) = SzS, \qquad z \in G, \tag{3.6}$$

where S is an idempotent matrix, i.e., $S^2 = I$ [18]. Clearly,

$$\mathrm{d}\sigma(Z) = SZS,$$

and, for simplicity, we will abuse notation by writing σZ in place of $\mathrm{d}\sigma Z$, given that all our computations take place in the space of matrices.

Since $S^2 = I$, all the eigenvalues of S are either $+1$ or -1. Thus, powers of matrices $P = \Pi_{\mathfrak{p}}(Z)$ as well as powers of ad_P are easy to evaluate by means of the $(+1)$- and (-1)-eigenspace of S [18].

Note that automorphisms of the type (3.6) are defined a priori, with respect to a fixed basis chosen independently of the data in Z. In section 5 we shall discuss automorphisms based on approximate eigenspace decompositions of the matrix Z, a case in which the splitting depends dynamically on the given matrix.

3.2. Automorphisms that lead to bordered matrix splittings. Let $Z \in \mathfrak{gl}(n, \mathbb{R})$ be an $n \times n$ matrix and consider the automorphism

$$\sigma_1 Z = S_1 Z S_1,$$

where S_1 is the idempotent matrix

$$S_1 = \left(\begin{array}{c|ccc} -1 & 0 & \cdots & 0 \\ \hline 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{array}\right).$$

It is easy to verify that

$$\Pi_{\mathfrak{p}_1} Z = \frac{1}{2}(Z - S_1 Z S_1) = \left(\begin{array}{c|ccc} 0 & z_{1,2} & \cdots & z_{1,n} \\ \hline z_{2,1} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ z_{n,1} & 0 & \cdots & 0 \end{array}\right), \tag{3.7}$$

while

$$\Pi_{\mathfrak{k}_1} Z = \frac{1}{2}(Z + S_1 Z S_1) = \left(\begin{array}{c|ccc} z_{1,1} & 0 & \cdots & 0 \\ \hline 0 & z_{2,2} & \cdots & z_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & z_{n,2} & \cdots & z_{n,n} \end{array}\right). \tag{3.8}$$

In general, assume that, at the jth step, the space $\mathfrak{k}_{j-1}$ consists of matrices of the form

$$W = \left(\begin{array}{ccc|ccc} w_{1,1} & & O & & & \\ & \ddots & & & O & \\ O & & w_{j-1,j-1} & & & \\ \hline & & & w_{j,j} & \cdots & w_{j,n} \\ & O & & \vdots & \ddots & \vdots \\ & & & w_{n,j} & \cdots & w_{n,n} \end{array}\right). \tag{3.9}$$

Then, the obvious choice is

$$S_j = \left(\begin{array}{c|c} I_{j-1} & O \\ \hline O & \tilde{S}_j \end{array}\right), \qquad \tilde{S}_j = \begin{pmatrix} -1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \tag{3.10}$$

where I_{j-1} denotes the $(j-1)\times(j-1)$ identity matrix and $\tilde{S}_j$ is an $(n-j+1)\times(n-j+1)$ block, so that the subspace $\mathfrak{p}_j$ consists of matrices of the form

$$\Pi_{\mathfrak{p}_j} W = \left(\begin{array}{c|c} O_{j-1} & O \\ \hline O & \tilde{P}_j \end{array}\right), \qquad \tilde{P}_j = \left(\begin{array}{c|ccc} 0 & w_{j,j+1} & \cdots & w_{j,n} \\ \hline w_{j+1,j} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ w_{n,j} & 0 & \cdots & 0 \end{array}\right), \tag{3.11}$$

where O_{j-1} denotes the $(j-1)\times(j-1)$ zero matrix.

Exponentials of matrices of the form (3.11) are very easy to compute: in effect,

$$\exp\left(\begin{array}{c|c} O_{j-1} & O \\ \hline O & \tilde{P}_j \end{array}\right) = \left(\begin{array}{c|c} I_{j-1} & O \\ \hline O & \exp(\tilde{P}_j) \end{array}\right),$$

where $\exp(\tilde{P}_j)$ can be computed *exactly* with a formula analogous to the Euler–Rodriguez formula (1.1): denote $\boldsymbol{a}_j = [w_{j+1,j}, \ldots, w_{n,j}]^{\mathrm{T}}$ and $\boldsymbol{b}_j = [w_{j,j+1}, \ldots w_{j,n}]^{\mathrm{T}}$. Then,

$$\exp(\tilde{P}_j) = \begin{cases} I_{n-j+1} + \frac{\sinh\alpha_j}{\alpha_j}\tilde{P}_j + \frac{1}{2}\left(\frac{\sinh(\alpha_j/2)}{\alpha_j/2}\right)^2 \tilde{P}_j^2 & \text{if } \boldsymbol{a}_j^{\mathrm{T}}\boldsymbol{b}_j > 0, \alpha_j = \sqrt{\boldsymbol{a}_j^{\mathrm{T}}\boldsymbol{b}_j}, \\ I_{n-j+1} + \tilde{P}_j + \frac{1}{2}\tilde{P}_j^2 & \text{if } \boldsymbol{a}_j^{\mathrm{T}}\boldsymbol{b}_j = 0, \\ I_{n-j+1} + \frac{\sin\alpha_j}{\alpha_j}\tilde{P}_j + \frac{1}{2}\left(\frac{\sin(\alpha_j/2)}{\alpha_j/2}\right)^2 \tilde{P}_j^2 & \text{if } \boldsymbol{a}_j^{\mathrm{T}}\boldsymbol{b}_j < 0, \alpha_j = \sqrt{-\boldsymbol{a}_j^{\mathrm{T}}\boldsymbol{b}_j}. \end{cases} \tag{3.12}$$

Note that

$$\tilde{P}_j^2 = \left(\begin{array}{c|c} \alpha_j^2 & \mathbf{0}^{\mathrm{T}} \\ \hline \mathbf{0} & \boldsymbol{a}_j\boldsymbol{b}_j^{\mathrm{T}} \end{array}\right).$$

Another alternative for the exact exponential of $\tilde{P}_j$ is the one proposed in [4]:

$$\exp(\tilde{P}_j) = I_{n-j+1} + [\boldsymbol{k}, \boldsymbol{e}_1]\varphi(D)\begin{bmatrix} \boldsymbol{e}_1^{\mathrm{T}} \\ \boldsymbol{l}^{\mathrm{T}} \end{bmatrix}, \tag{3.13}$$

where

$$\boldsymbol{k} = \begin{bmatrix} 0 \\ \boldsymbol{a}_j \end{bmatrix}, \quad \boldsymbol{l} = \begin{bmatrix} 0 \\ \boldsymbol{b}_j \end{bmatrix}, \quad D = \begin{pmatrix} 0 & 1 \\ \boldsymbol{b}_j^{\mathrm{T}}\boldsymbol{a}_j & 0 \end{pmatrix},$$

$\boldsymbol{e}_1$ is the vector $[1, 0, \ldots, 0]^{\mathrm{T}} \in \mathbb{R}^{n-j+1}$ and finally $\varphi(z) = (\mathrm{e}^z - 1)/z$. The latter formula (3.13), as we shall see in what follows, leads to significant savings in the computation and assembly of the exponentials.

Moreover, given that

$$\Pi_{\mathfrak{k}_j} W = \left(\begin{array}{c|c} D_{j-1} & O \\ \hline O & \tilde{K}_j \end{array}\right), \qquad D_{j-1} = \operatorname{diag}(w_{1,1}, \ldots, w_{j-1,j-1}), \tag{3.14}$$

where

$$\tilde{K}_j = \left(\begin{array}{c|c} w_{j,j} & \mathbf{0}^{\mathrm{T}} \\ \hline \mathbf{0} & \bar{K}_j \end{array}\right),$$

then

$$[\tilde{P}_j, \tilde{K}_j] = \left(\begin{array}{c|c} 0 & \boldsymbol{b}_j^{\mathrm{T}} \bar{K}_j - w_{j,j} \boldsymbol{b}_j^{\mathrm{T}} \\ \hline w_{j,j} \boldsymbol{a}_j - \bar{K}_j \boldsymbol{a}_j & O \end{array}\right). \tag{3.15}$$

Next, if $Z \in \mathfrak{g}$, to obtain an approximation of the exponential in G by these automorphisms we shall require that the σ_i's, defined by the above matrices S_i, map $\mathfrak{g}$ into $\mathfrak{g}$. Clearly, this is the case for

- $\mathfrak{so}(n, \mathbb{R})$, since $\sigma_i Z = S_i Z S_i = \mathrm{Ad}_{S_i} Z$ is a map from $\mathfrak{so}(n) \to \mathfrak{so}(n)$, given that each S_i is an orthogonal matrix;
- $\mathfrak{sl}(n, \mathbb{R})$, since σ_i leaves the diagonal elements of Z (hence its trace) unchanged;
- quadratic Lie algebras $\mathfrak{g} = \{Z : ZJ + JZ^{\mathrm{T}} = O, J \text{ nonsingular}\}$ provided that J and the S_i's commute. This is, for instance, the case when J is diagonal; hence our formulas are valid for $\mathfrak{so}(p, q)$, $p+q = n$, but not for the symplectic algebra $\mathfrak{sp}(n, \mathbb{R})$. In the latter situation, we consider different choices for the automorphisms σ_i, discussed at a greater length in [18].

3.3. Splittings of order 2 to 4, their implementation and complexity. In this section we describe in more detail the algorithms, the implementation, and the complexity of the splittings induced by the automorphisms described above. Note that the complexity counts refer to the computation of the splitting alone and depends on the desired order of approximation, and further work is required for the assembly of the approximation to the exponential. The total cost of some schemes (splitting and assembly of the approximated exponential) is therefore presented in section 4.

The cases of a polar-type representation, $z = xy$, or a symmetric polar-type representation, $z = xyx$, are discussed separately.

ALGORITHM 1 (polar-type splitting, order 2). *Based on the iterated generalized polar decomposition* (3.3). Note that the $\Pi_{\mathfrak{p}_j}$ and $\Pi_{\mathfrak{k}_j}$ projections need not be stored in separate matrices but can be stored in places of the rows and columns of the matrix Z. We truncate the expansions (2.6) to order 2, and hence at each step only the $\mathfrak{p}_j$-part needs correction. Taking in mind (3.3), the matrices $X^{[j]}$ are low-rank matrices with nonzero entries only on the jth row, column $j+1$ to n, and jth column, row $j+1$ to n, for $j = 1, \ldots, n-2$, which are stored in place of the corresponding Z entries. The matrix $Y^{[n-1]}$ is diagonal and is stored in the diagonal entries of Z.

% Purpose: second-order approximation of the splitting (3.3)
% In: $n \times n$ matrix Z
% Out: Z overwritten with the nonzero elements of $X^{[i]}$ and $Y^{[m]}$ as:
% $Z(i+1:n, i) = X^{[i]}(i+1:n, i), \quad Z(i, i+1:n) = X^{[i]}(i, i+1:n),$
% $\operatorname{diag}(Z) = \operatorname{diag}(Y^{[m]})$

```
%
for j = 1 : n − 1
   a_j := Z(j + 1 : n, j),
   b_j := Z(j, j + 1 : n)^T,
   K̄_j := Z(j + 1 : n, j + 1 : n),
   c_j := z_{j,j} a_j − K̄_j a_j,
   d_j := −z_{j,j} b_j + K̄_j^T b_j,
   Z(j + 1 : n, j) := a_j − (1/2) c_j,
   Z(j, j + 1 : n) := (b_j − (1/2) d_j)^T
end
```

The computation of the splitting requires at each step two matrix-vector multiplications, each amounting to $\mathcal{O}\big(2(n-j+1)^2\big)$ floating point operations (we count both multiplications and additions), as well as two vector updates, which are $\mathcal{O}(n-j+1)$ operations. Hence, for large n, the cost of computing the splitting is of the order

- $\frac{4}{3}n^3$ for $\mathfrak{so}(p,q)$, $p+q=n$ and $\mathfrak{sl}(n)$,
- $\frac{2}{3}n^3$ for $\mathfrak{so}(n)$, taking into account that $\boldsymbol{b}_j = -\boldsymbol{a}_j$.

Note that for both $\mathfrak{so}(p,q)$ and $\mathfrak{so}(n)$ the matrix $Y^{[n-1]}$ is the zero matrix.

ALGORITHM 2 (symmetric polar-type splitting, order 2). *Based on the iterated generalized polar decomposition* (3.4). We truncate the expansions (2.10)–(2.11) to order 2. The storing of the entries is as above.

```
% Purpose: second-order approximation of the splitting (3.4)
% In: n × n matrix Z
% Out: Z overwritten with the nonzero elements of X^[i] and Y^[m] as:
%    Z(i + 1 : n, i) = X^[i](i + 1 : n, i),    Z(i, i + 1 : n) = X^[i](i, i + 1 : n),
%    diag(Z) = diag(Y^[m])
%
% Computation of the splitting
for j = 1 : n − 1
   a_j := Z(j + 1 : n, j),
   b_j := Z(j, j + 1 : n)^T,
   Z(j + 1 : n, j) := (1/2) a_j,
   Z(j, j + 1 : n) := (1/2) b_j^T
end
```

This splitting costs only

- $n(n-1)$ for $\mathfrak{so}(p,q)$, $p+q=n$ and $\mathfrak{sl}(n)$,
- $\frac{n(n-1)}{2}$ for $\mathfrak{so}(n)$, because of skew-symmetry.

ALGORITHM 3 (polar-type splitting, order 3). We truncate (2.6)–(2.7) to include $\mathcal{O}\big(t^3\big)$ terms. Note that the term $[K,[P,K]]$ is of the form (3.15). We need to include also the term of the form $[P,[P,K]]$. We observe that

$$[\tilde{P}_j,[\tilde{P}_j,\tilde{K}_j]] = \left(\begin{array}{c|c} 2\boldsymbol{b}_j^{\mathrm{T}}(z_{j,j}I-\bar{K}_j)\boldsymbol{a}_j & \mathbf{0}^{\mathrm{T}} \\ \hline \mathbf{0} & -\boldsymbol{a}_j\boldsymbol{b}_j^{\mathrm{T}}(z_{j,j}I-\bar{K}_j)-(z_{j,j}I-\bar{K}_j)\boldsymbol{a}_j\boldsymbol{b}_j^{\mathrm{T}} \end{array}\right). \tag{3.16}$$

```
% Purpose: third-order approximation of the splitting (3.3)
% In: n × n matrix Z
% Out: Z overwritten with the nonzero elements of X^[i] and Y^[m] as:
```

% $Z(i+1:n,i) = X^{[i]}(i+1:n,i), \quad Z(i,i+1:n) = X^{[i]}(i,i+1:n),$
% $\operatorname{diag}(Z) = \operatorname{diag}(Y^{[m]})$
%
% Computation of the splitting
for $j = 1:n-1$
$\boldsymbol{a}_j := Z(j+1:n,j)$,
$\boldsymbol{b}_j := Z(j,j+1:n)^{\mathrm{T}}$,
$\bar{K}_j := Z(j+1:n,j+1:n)$,
$\boldsymbol{c}_j := z_{j,j}\boldsymbol{a}_j - \bar{K}_j\boldsymbol{a}_j$,
$\boldsymbol{d}_j := -z_{j,j}\boldsymbol{b}_j + \bar{K}_j^{\mathrm{T}}\boldsymbol{b}_j$,
$Z(j+1:n,j) := \boldsymbol{a}_j - \frac{1}{2}\boldsymbol{c}_j + \frac{1}{6}(z_{j,j}I - \bar{K}_i)\boldsymbol{c}_j$,
$Z(j,j+1:n) := (\boldsymbol{b}_j - \frac{1}{2}\boldsymbol{d}_j - \frac{1}{6}(z_{j,j}I - \bar{K}_j)\boldsymbol{d}_j)^{\mathrm{T}}$,
$Z(j+1:n,j+1:n) := Z(j+1:n,j+1:n) + \frac{1}{12}(-\boldsymbol{a}_j\boldsymbol{d}_j^{\mathrm{T}} + \boldsymbol{c}_j\boldsymbol{b}_j^{\mathrm{T}})$,
$Z(j,j) := Z(j,j) - \frac{1}{6}\boldsymbol{b}_j^{\mathrm{T}}\boldsymbol{c}_j$
end

Analyzing the computations involved, we find that the most costly part is constituted by the matrix-vector products in the computations in $\boldsymbol{c}_j, \boldsymbol{d}_j$, $Z(j+1:n,j)$, $Z(j,j+1:n)$ and vector-vector products in the update of $Z(j+1:n,j+1:n)$ and $z(j,j)$. The computation of $\boldsymbol{c}_j, \boldsymbol{d}_j, Z(j+1:n,j)$, and $Z(j,j+1:n)$ amounts to $\frac{8}{3}n^3$ in the whole process. For the update of $Z(j+1:n,j+1:n)$, we need to compute two vector-vector products ($\mathcal{O}\big((n-j+1)^2\big)$ each) plus $3(n-j+1)^2$ operations to update the elements of the matrix. Thus, the whole cost of updating the matrix $Z(j+1:n,j+1:n)$ is $\frac{5}{3}n^3$. The update of $z_{j,j}$ requires $2(n-j+1)^2$ operations per step, which give a $\frac{2}{3}n^3$ contribution to the total cost of the splitting.

In summary, the total cost of the splitting is

- $5n^3$ for $\mathfrak{so}(p,q)$ and $\mathfrak{sl}(n)$,
- for $\mathfrak{so}(n)$, note that $\boldsymbol{d}_j$ need not be calculated as well as $z_{j,j} = 0$. Similarly, we take into account that $\boldsymbol{b}_j = -\boldsymbol{a}_j$ and that only half of the elements of $Z(j+1:n,j+1:n)$ need to be updated. The total amounts to $2\frac{1}{2}n^3$ operations.

It is easy to modify the splitting above to obtain order 4. Note that

$$[\tilde{P}_j,[\tilde{P}_j,[\tilde{P}_j,\tilde{K}_j]]] = \boldsymbol{b}_j^{\mathrm{T}}\boldsymbol{a}_j[\tilde{P}_j,\tilde{K}_j] + 3(\boldsymbol{b}_j^{\mathrm{T}}(z_{j,j}I - \bar{K}_j)\boldsymbol{a}_j)\tilde{S}_j\tilde{P}_j, \tag{3.17}$$

which requires the computation of the scalar $\boldsymbol{b}_j^{\mathrm{T}}\boldsymbol{a}_j$ only, costing $2/3n^3$ operations in the whole process. However, all the other powers $\operatorname{ad}_{\tilde{P}_j}^i \tilde{K}_j$ for $i = 4,5,\ldots$ require no further computation. The next term, $[\tilde{K}_j,[\tilde{K}_j,[\tilde{P}_j,\tilde{K}_j]]]$, can be computed with just two (one) extra matrix-vector computations for $\mathfrak{sl}(n)$ (resp., $\mathfrak{so}(n)$), which contribute $\frac{4}{3}n^3$ (resp., $\frac{2}{3}n^3$) to the cost of the splitting, so that the splitting of order 4 costs a total of $7n^3$ operations for $\mathfrak{sl}(n)$ (resp., $4n^3$ for $\mathfrak{so}(n)$).

ALGORITHM 4 (symmetric polar-type splitting, order 4). We truncate (2.10)–(2.11) to include $\mathcal{O}\big(t^3\big)$ terms. Also in this case, the term $[K,[P,K]]$ is of the form (3.15), while the term $[P,[P,K]]$ is computed according to (3.16).

% Purpose: fourth-order approximation of the splitting (3.4)
% In: $n \times n$ matrix Z
% Out: Z overwritten with the nonzero elements of $X^{[i]}$ and $Y^{[m]}$ as:
% $Z(i+1:n,i) = X^{[i]}(i+1:n,i), \quad Z(i,i+1:n) = X^{[i]}(i,i+1:n),$

```
%   diag(Z) = diag(Y^[m])
%
for j = 1 : n − 1
   a_j := Z(j + 1 : n, j),
   b_j := Z(j, j + 1 : n)^T,
   K̄_j := Z(j + 1 : n, j + 1 : n),
   c_j := (z_{j,j} I − K̄_j)^2 a_j,
   d_j := (z_{j,j} I − K̄_j^T)^2 b_j,
   Z(j + 1 : n, j) := (1/2) a_j − (1/24) c_j,
   Z(j, j + 1 : n) := ((1/2) b_j − (1/24) d_j)^T,
   Z(j + 1 : n, j + 1 : n) := Z(j + 1 : n, j + 1 : n) − (1/24)(a_j b_j^T (z_{j,j} I − K̄_j),
                                  + (z_{j,j} I − K̄_j) a_j b_j^T),
   z(j, j) := z(j, j) + (1/12) b_j^T (z_{j,j} I − K̄_j) a_j
end
```

We need to compute a total of four matrix-vector products, yielding $\frac{8}{3}n^3$ operations. The update of the block $Z(j+1:n, j+1:n)$ costs $\frac{5}{3}n^3$ operations, while the update of $z(j,j)$ costs $\frac{2}{3}n^3$ operations, for a total of

- $5n^3$ operations for $\mathfrak{sl}(n)$ and $\mathfrak{so}(p+q)$,
- $2\frac{1}{2}n^3$ operations for $\mathfrak{so}(n)$.

3.4. On higher-order splittings. The costs of implementing splittings following (3.3) or (3.4) depend on the type of commutation involved: commutators of the form $[P, K]$ and $[P_1, P_2]$, $P, P_1, P_2 \in \mathfrak{p}$, $K \in \mathfrak{k}$, contribute as an $\mathcal{O}(n^3)$ term to the total complexity of the splitting; however, commutators of the form $[K_1, K_2]$ for $K_1, K_2 \in \mathfrak{k}$ can easily contribute an $\mathcal{O}(n^4)$ to the total complexity of the splittings if the special structure of the terms involved is not taken into consideration. If carefully implemented, these terms can also be computed with only matrix-vector and vector-vector products, contributing $\mathcal{O}(n^3)$ operations to the total cost of the splitting. For example, let us consider the term $[\tilde{K}_j, [\tilde{K}_j, [\tilde{P}_j, [\tilde{P}_j, \tilde{K}_j]]]]$, which appears in the $\mathcal{O}(t^5)$ contribution in the expansion of the Y part, for both the polar-type and symmetric polar-type splittings. One has

$$
\begin{aligned}
[\tilde{K}_j, [\tilde{K}_j, [\tilde{P}_j, [\tilde{P}_j, \tilde{K}_j]]]] &= \left(\begin{array}{c|c} 0 & \mathbf{0}^{\mathrm{T}} \\ \hline \mathbf{0} & \bar{Q}_j \end{array}\right), \\
\bar{Q}_j &= -\Big((\bar{K}_j(\bar{K}_j \boldsymbol{a}_j))(\boldsymbol{b}_j^{\mathrm{T}} \Delta_j)\Big) - \Big((\bar{K}_j(\bar{K}_j(\Delta_j \boldsymbol{a}_j)))\boldsymbol{b}_j^{\mathrm{T}}\Big) \\
&\quad +2\Big(((\bar{K}_j \boldsymbol{a}_j)((\boldsymbol{b}_j^{\mathrm{T}} \Delta_j)\bar{K}_j)) + ((\bar{K}_j(\Delta_j \boldsymbol{a}_j))(\boldsymbol{b}_j^{\mathrm{T}} \bar{K}_j))\Big) \\
&\quad -\Big(\boldsymbol{a}_j(((\boldsymbol{b}_j^{\mathrm{T}} \Delta_j)\bar{K}_j)\bar{K}_j)\Big) - \Big((\Delta_j \boldsymbol{a}_j)(\boldsymbol{b}_j^{\mathrm{T}} \bar{K}_j)\bar{K}_j\Big),
\end{aligned} \tag{3.18}
$$

where Δ_j denotes the matrix $z_{j,j} I - \bar{K}_j$. The parentheses indicate the correct order in which the operations should be executed to obtain the right complexity ($\mathcal{O}\big((n-j+1)^2\big)$ per step, hence a total of $\mathcal{O}(n^3)$ for the splitting). Many of the terms are already computed for the lower-order conditions, yet the complexity arises significantly. Therefore we recommend these splitting-type techniques when a moderate order of approximation is required.

To construct higher-order approximations with these splitting techniques, one

could use our symmetric polar-type splittings together with a Yoshida-type symmetric combination.

3.5. Assembly of the approximation $F(t, Z)$ to the exponential. For each algorithm that computes the approximation to the exponential, we distinguish two cases: when the approximation is applied to a vector $\boldsymbol{v}$, and when instead the matrix exponential $\exp(Z)$ is required. Since the matrices $X^{[j]}$ are never constructed explicitly and are stored as vectors, computation of the exponentials $\exp(X^{[j]})$ is also never performed explicitly, but is implemented as in the case of the *Householder reflections* [6] when applied to a vector.

First, let us return to (3.13). It is easy to verify that, if we denote by $\alpha_j = \sqrt{\boldsymbol{b}_j^T \boldsymbol{a}_j}$, then $\exp(D)$ has the exact form

$$\exp(D) = \left(1 + 2\sinh^2\left(\frac{\alpha_j}{2}\right)\right) I + \frac{\sinh \alpha_j}{\alpha_j} D,$$

where I is the 2×2 identity matrix. Similar remarks hold about the matrix D^{-1}. Thus, the computation of $\varphi(D) = D^{-1}(\exp(D) - I)$ can be done in very few flops that "do not contribute" to the total cost of the algorithm. Next, if $\boldsymbol{v}, \boldsymbol{k}, \boldsymbol{l}, \boldsymbol{e}_1 \in \mathbb{R}^j$, the assembly of $\exp(\tilde{P}_j)\boldsymbol{v}$ according to (3.13) can be computed in $6j$ operations. If we let j vary between 1 and n, the total cost of the multiplications is hence $3n^2$. This is precisely the complexity of the assembly of the exponential for polar-type splittings, which has the form as in (3.3).

ALGORITHM 5 (polar-type approximation).

% Purpose: Computing the approximant (3.3) applied to a vector $\boldsymbol{v}$
% In: $\boldsymbol{v}$: n-vector
% Z: $n \times n$ matrix containing the nonzero elements of $X^{[i]}$ and $Y^{[m]}$ as:
% $Z(i+1:n, i) = X^{[i]}(i+1:n, i), \quad Z(i, i+1:n) = X^{[i]}(i, i+1:n)$,
% $\operatorname{diag}(Z) = \operatorname{diag}(Y^{[m]})$
% Out: $\boldsymbol{v} := \exp(X^{[1]}) \cdots \exp(X^{[m]}) \exp(Y^{[m]})\boldsymbol{v}$
%
for $k = 1 : n$
  $v_k := \exp(z_{k,k}) v_k$
end
for $j = n - 1 : -1 : 1$
  $\boldsymbol{a}_j := [0; Z(j+1:n, j)]$,
  $\boldsymbol{b}_j := [0; Z(j, j+1:n)^{\mathrm{T}}]$,
  $\boldsymbol{v}_{\text{old}} := \boldsymbol{v}(j:n)$,
  $\alpha_j := \sqrt{\boldsymbol{b}_j^{\mathrm{T}} \boldsymbol{a}_j}$, $\beta_j = \frac{\sinh \alpha_j}{\alpha_j}$ and $\gamma_j = 2\sinh^2(\alpha_j/2)$,
  $D := \left(\begin{smallmatrix} 0 & 1 \\ \alpha_j^2 & 0 \end{smallmatrix}\right)$,
  $\varphi(D) := \gamma_j D^{-1} + \beta_j I$,
  $\boldsymbol{w} := \varphi(D) \left[\begin{smallmatrix} v_{\text{old},1} \\ \boldsymbol{b}_j^{\mathrm{T}} \boldsymbol{v}_{\text{old}} \end{smallmatrix}\right]$,
  $\boldsymbol{v}_{\text{new}} := [\boldsymbol{a}_j, \boldsymbol{e}_1]\boldsymbol{w} = w_1 \boldsymbol{a}_j + w_2 \boldsymbol{e}_1$,
  $\boldsymbol{v}(j:n) := \boldsymbol{v}_{\text{old}} + \boldsymbol{v}_{\text{new}}$
end

In the case in which the output needs to be applied to an $n \times n$ matrix B, we can apply the above algorithm to each column of B, for a total of $3n^3$ operations. This complexity can be reduced to about $2n^3$ taking into account that the vector

$\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_{n-1}]^{\mathrm{T}}$ can be calculated once and for all, depending only on the splitting of the matrix Z and not in any manner on the columns of B. Also $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_{n-1}]^{\mathrm{T}}$ and $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_{n-1}]^{\mathrm{T}}$ can be computed once and stored for later use.

ALGORITHM 6 (symmetric polar-type approximation). The approximation to the exponential is carried out in a manner very similar to that described above in Algorithm 5, except that, since (3.4) is based on a Strang-type splitting, the assembly is also performed in reverse order.

% Purpose: Computing the approximant (3.4) applied to a vector $\boldsymbol{v}$
% In: $\boldsymbol{v}$: n-vector
% Z: $n \times n$ matrix containing the nonzero elements of $X^{[i]}$ and $Y^{[m]}$ as:
% $Z(i+1:n,i) = X^{[i]}(i+1:n,i), \quad Z(i,i+1:n) = X^{[i]}(i,i+1:n),$
% $\mathrm{diag}(Z) = \mathrm{diag}(Y^{[m]})$
% Out: $\boldsymbol{v} := \exp(X^{[1]}) \cdots \exp(X^{[m]}) \exp(Y^{[m]}) \exp(X^{[m]}) \cdots \exp(X^{[1]}) \boldsymbol{v}$
%

for $j = 1 : n-1$
  $\boldsymbol{a}_j := [0; Z(j+1:n, j)]$,
  $\boldsymbol{b}_j := [0; Z(j, j+1:n)^{\mathrm{T}}]$,
  $\boldsymbol{v}_{\mathrm{old}} := \boldsymbol{v}(j:n)$,
  $\alpha_j := \sqrt{\boldsymbol{b}_j^{\mathrm{T}} \boldsymbol{a}_j}, \beta_j = \frac{\sinh \alpha_j}{\alpha_j}, \gamma_j = 2 \sinh^2(\alpha_j/2)$,
  $D := \left(\begin{smallmatrix} 0 & 1 \\ \alpha_j^2 & 0 \end{smallmatrix}\right)$,
  $\varphi(D) := \gamma_j D^{-1} + \beta_j I$,
  $\boldsymbol{w} := \varphi(D) \left[\begin{smallmatrix} v_{\mathrm{old},1} \\ \boldsymbol{b}_j^{\mathrm{T}} \boldsymbol{v}_{\mathrm{old}} \end{smallmatrix}\right]$,
  $\boldsymbol{v}_{\mathrm{new}} := [\boldsymbol{a}_j, \boldsymbol{e}_1] \boldsymbol{w} = w_1 \boldsymbol{a}_j + w_2 \boldsymbol{e}_1$,
  $\boldsymbol{v}(j:n) := \boldsymbol{v}(j:n) + \boldsymbol{v}_{\mathrm{new}}$
end

for $k = 1 : n$
  $v_k := \exp(z_{k,k}) v_k$
end

for $j = n-1 : -1 : 1$
  $\boldsymbol{a}_j := [0; Z(j+1:n, j)]$,
  $\boldsymbol{b}_j := [0; Z(j, j+1:n)^{\mathrm{T}}]$,
  $\boldsymbol{v}_{\mathrm{old}} := \boldsymbol{v}(j:n)$,
  $D := \left(\begin{smallmatrix} 0 & 1 \\ \alpha_j^2 & 0 \end{smallmatrix}\right)$,
  $\varphi(D) := \gamma_j D^{-1} + \beta_j I$,
  $\boldsymbol{w} := \varphi(D) \left[\begin{smallmatrix} v_{\mathrm{old},1} \\ \boldsymbol{b}_j^{\mathrm{T}} \boldsymbol{v}_{\mathrm{old}} \end{smallmatrix}\right]$,
  $\boldsymbol{v}_{\mathrm{new}} := [\boldsymbol{a}_j, \boldsymbol{e}_1] \boldsymbol{w} = w_1 \boldsymbol{a}_j + w_2 \boldsymbol{e}_1$,
  $\boldsymbol{v}(j:n) := \boldsymbol{v}(j:n) + \boldsymbol{v}_{\mathrm{new}}$
end

The vectors $\boldsymbol{\alpha}, \boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ need to be calculated only once and stored for later use in the reverse-order multiplication. The cost of the assembly is roughly twice the cost of the assembly in Algorithm 1; hence it amounts to $5n^2$ operations. (We save n^2 operations omitting the computation of $\boldsymbol{\alpha}$.)

When the result is applied to a matrix B, again we apply the same algorithm to each column of B, which yields n^3 operations. Also in this case the vector $\boldsymbol{\alpha}$ does not depend on B and can be computed once and for all, reducing the cost to $4n^3$

operations. The same remark holds for the vectors $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$.

It is important to mention that the matrix D might be singular or close to singular (for example, when $\boldsymbol{a}_j$ and $\boldsymbol{b}_j$ are close to be orthogonal); hence the computation of $\exp(\tilde{P}_j)$ according to (3.13) may lead to instabilities. In this case, it is recommended to use (3.12) instead of (3.13) or to compute (3.13) by means of singular value decompositions. The cost of (3.12) is twice the cost of (3.13) ($5n^2$ for polar-type assemblies and $9n^2$ for symmetric assemblies for $F(t, Z)$ applied to a vector).

4. Numerical experiments.

4.1. Nonsymmetric polar-type approximations to the exponential. We commence comparing the polar-type order-2 splitting of Algorithm 1 combined with the assembly of the exponential in Algorithm 5 with the $(1, 1)$-Padé approximant for matrices in $\mathfrak{sl}(n)$ and $\mathfrak{so}(n)$, with corresponding groups $\mathrm{SL}(n)$ and $\mathrm{SO}(n)$. We choose diagonal Padé approximants as a benchmark because they are easy to implement, they are the rational approximant with highest order of approximation at the origin, and it is well known that they map quadratic Lie algebras into quadratic Lie groups (but not necessarily other Lie algebras into the corresponding Lie groups). Furthermore, there exists a well-established error analysis for diagonal Padé approximants, and this makes them the standard against which other methods are compared. For using the Padé approximant, we refer to [23].

Table 1 reports the complexity of the method 1+5. A $(1, 1)$-Padé approximant costs $\mathcal{O}\big(2/3n^3\big)$ floating point operations when applied to a vector (essentially the cost of LU-factorizing a linear system) and $\mathcal{O}\big(2\frac{2}{3}n^3\big)$ operations when applied to $n \times n$ matrices. (Note that $(I - Z/2)^{-1}(I + Z/2) = -I + 2(I - Z/2)^{-1}$; hence we reduce to solve a single linear system: $\frac{2}{3}n^3$ flops come from the LU factorization and $2n^3$ from the n forward and backward solution of triangular systems.)

In Figure 1 we compare the number of floating point operations scaled by n^3 for matrices Z up to size 500 as obtained in MATLAB for our polar-type order-2 algorithm (method 1+5) and the $(1, 1)$-Padé approximant both applied to a matrix. We consider the cases when Z is in $\mathfrak{sl}(n)$ and $\mathfrak{so}(n)$. The costs of computing both approximations clearly converges to the theoretical estimates (which in the plot are represented by solid lines) given in Table 1 for large n.

In Figure 2 we compare the accuracy of the two approximations (top plot) for the exponential of a random 10×10 traceless matrix hZ, where Z is normalized so that $\|Z\|_2 = 1$ and $h = \frac{1}{2}, \frac{1}{4}, \ldots, \frac{1}{64}$. Both methods show a local truncation error of $\mathcal{O}\big(h^3\big)$, revealing that the order of approximation to the exact exponential is 2. The bottom plot shows the error in the determinant as a function of h: the Padé approximant has an error that behaves like h^3, while our method preserves the determinant equal to 1 to machine accuracy.

TABLE 1
Complexity for a polar-type order-2 approximant.

Algorithm	$\mathfrak{sl}(n), \mathfrak{so}(p, q)$		$\mathfrak{so}(n)$	
1+5	Vector	Matrix	Vector	Matrix
Splitting	$1\frac{1}{3}n^3$	$1\frac{1}{3}n^3$	$\frac{2}{3}n^3$	$\frac{2}{3}n^3$
Assembly exp	$3n^2$	$2n^3$	$3n^2$	$2n^3$
Total	$1\frac{1}{3}n^3$	$3\frac{1}{3}n^3$	$\frac{2}{3}n^3$	$2\frac{2}{3}n^3$

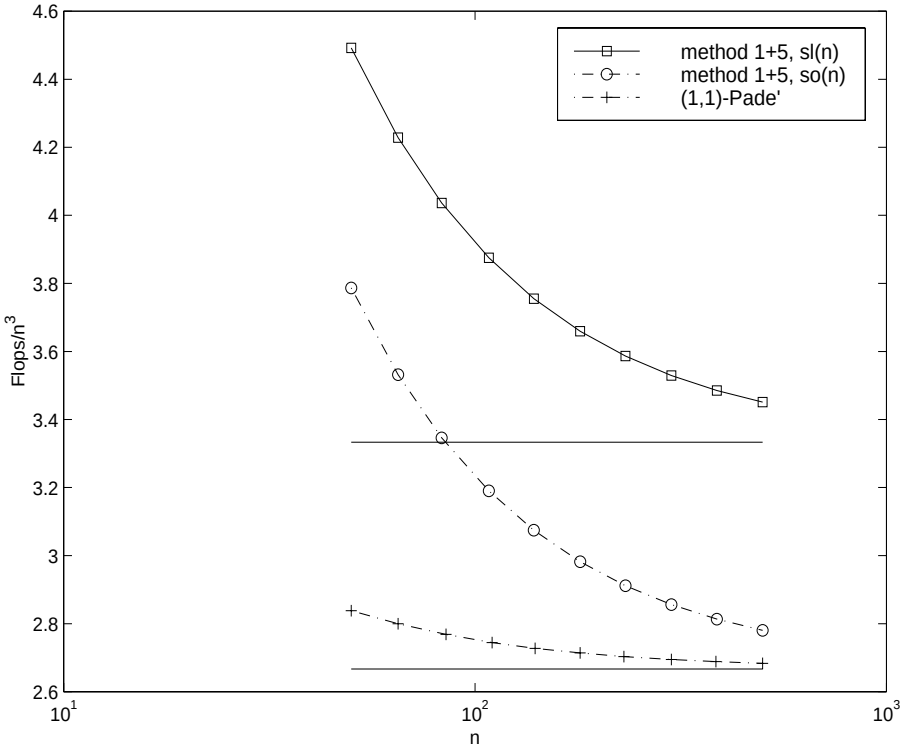

FIG. 1. *Floating point operations (scaled by n^3) versus size for the approximation of the exponential of a matrix in $\mathfrak{sl}(n)$ and in $\mathfrak{so}(n)$ applied to a matrix with the order-2 polar-type algorithm (method $1+5$) and $(1,1)$-Padé approximant.*

In Table 2 we report the complexity of the method 3+5, which yields an approximation to the exponential of order 3. The numbers in parentheses refer to the cost of the algorithm with order-4 corrections.

4.2. Symmetric polar-type approximations to the exponential. We commence comparing our method 2+6, yielding an approximation of order 2, with the $(1,1)$-Padé approximant. Table 3 reports the complexity of the method 2+6.

Clearly, in the matrix-vector case, our methods are one order of magnitude cheaper than the Padé approximant and are definitively to be preferred (see Figure 3 for matrices in $\mathfrak{sl}(n)$). Furthermore, our method maps the approximation in $\mathrm{SL}(n)$, while the Padé approximant does not. When comparing approximations of the matrix exponential applied to a vector, it is a must to consider Krylov subspace methods [20]. We compare the method 2+6 with a Krylov subspace method when Z is a matrix in $\mathfrak{sl}(n)$, normalized so that $\|Z\|_2 = 1$ and $\boldsymbol{v} \in \mathbb{R}^n$ is a vector of unit norm. The Krylov subspaces are obtained by Arnoldi iterations, whose computational cost amounts to about $2mn^2 + 2nm(m-1)$ operations, counting both multiplications and additions. Here m is the dimension of the subspace $K_m \equiv \mathrm{span}\{\boldsymbol{v}, Z\boldsymbol{v}, \ldots, Z^{m-1}\boldsymbol{v}\}$. To obtain the total cost of a Krylov method, we have to add $\mathcal{O}(m^3)$ computations arising from the evaluation of the exponential of the Hessenberg matrix obtained with the Arnoldi iteration, plus $2nm$ operations arising from the multiplication of the latter with the orthogonal basis. However, when n is large and $m \ll n$, these costs are

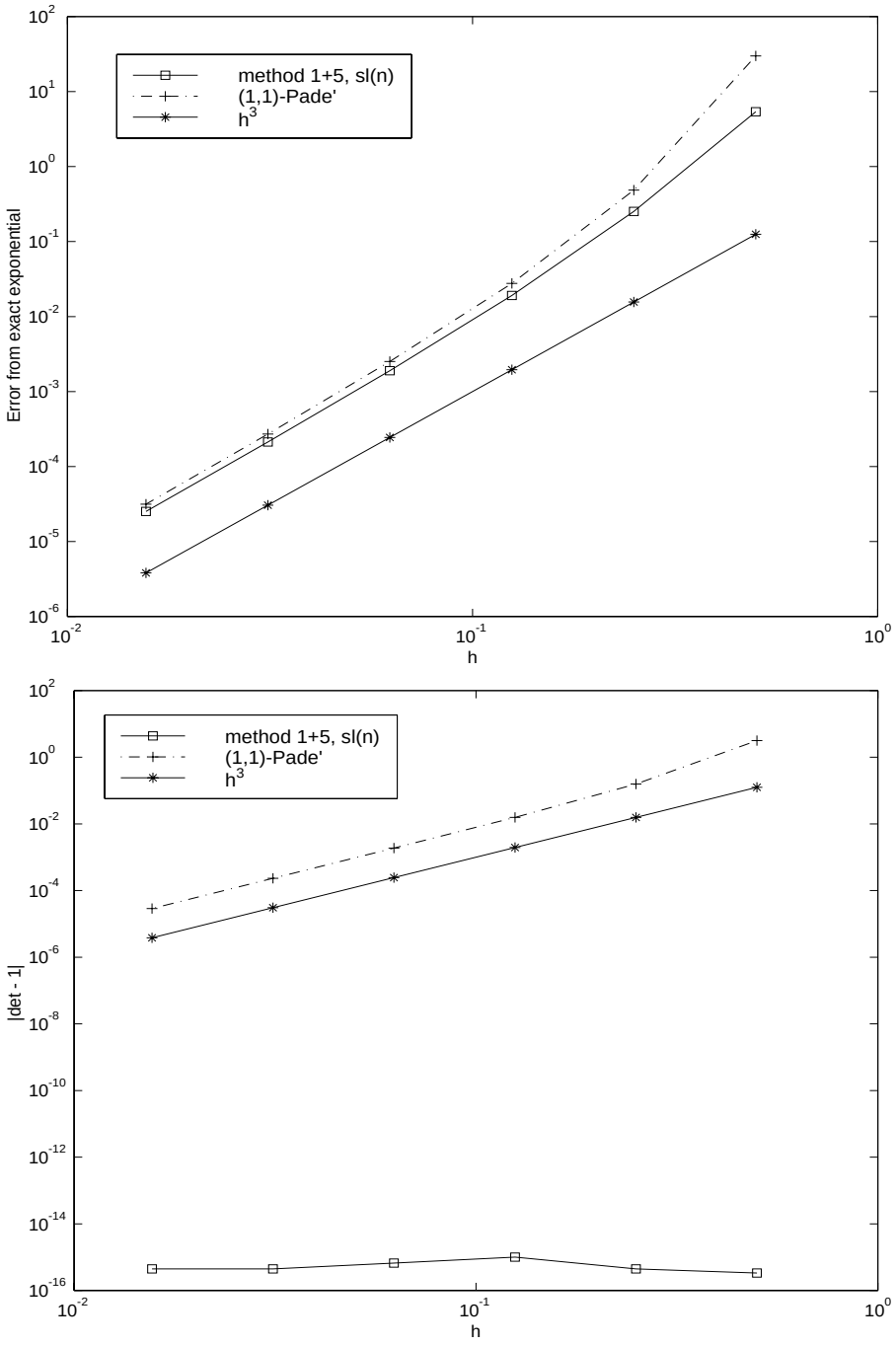

FIG. 2. *Error in the approximation (top) and in the determinant (bottom) versus* h *for the approximation of the exponential of a traceless* 10×10 *matrix of unit norm with the order-*2 *polar-type algorithm (method* $1+5$*) and* $(1,1)$*-Padé approximant.*

TABLE 2
Complexity for a polar-type order-3 approximant. The numbers in parentheses correspond to the coefficients for an order-4 approximation.

Algorithm	$\mathfrak{sl}(n), \mathfrak{so}(p,q)$		$\mathfrak{so}(n)$	
3+5	Vector	Matrix	Vector	Matrix
Splitting	$5(7)n^3$	$5(7)n^3$	$2\frac{1}{2}(4)n^3$	$2\frac{1}{2}(4)n^3$
Assembly exp	$3n^2$	$2n^3$	$3n^2$	$2n^3$
Total	$5(7)n^3$	$7(9)n^3$	$2\frac{1}{2}(4)n^3$	$4\frac{1}{2}(6)n^3$

TABLE 3
Complexity for a symmetric polar-type order-2 approximant.

Algorithm	$\mathfrak{sl}(n), \mathfrak{so}(p,q)$		$\mathfrak{so}(n)$	
2+6	Vector	Matrix	Vector	Matrix
Splitting	n^2	n^2	$\frac{1}{2}n^2$	$\frac{1}{2}n^2$
Assembly exp	$5n^2$	$4n^3$	$5n^2$	$4n^3$
Total	$6n^2$	$4n^3$	$5\frac{1}{2}n^2$	$4n^3$

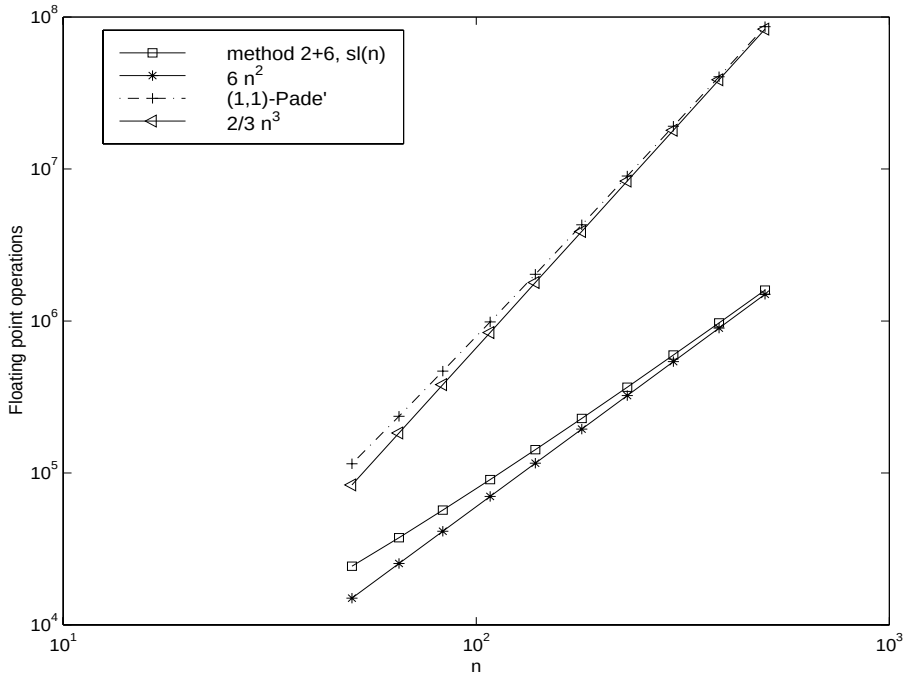

FIG. 3. *Floating point operations versus size for the approximation of the exponential of a matrix in $\mathfrak{sl}(n)$ applied to a vector with the order-2 symmetric polar-type algorithm (method $2+6$) and $(1,1)$-Padé approximant.*

subsumed in that of the Arnoldi iteration, and the leading factor is $2mn^2$ ($2mn^3$ for matrices).

The error, computed as $\|F(1,Z)\boldsymbol{v} - \exp(Z)\boldsymbol{v}\|_2$, and the floating point operations of both approximations for $n = 100, 200, 300$ are given in Table 4. The Krylov method

TABLE 4
Krylov subspace approximations versus the method $2+6$ *for the approximation of* $\exp(Z)\mathsf{v}$.

	Krylov			2+6	
Size n	Error	m	Flops	Error	Flops
100	0.74	1	21041	0.05	66239
	0.01	2	42887		
	4.4e-15	9	219123		
200	0.79	1	82041	0.05	242589
	0.01	2	165477		
	7.1e-15	8	690653		
300	0.75	1	183041	0.06	528939
	0.01	2	368077		
	7.7e-15	8	1510653		

converges very fast: in all three cases 89 iterations are sufficient to obtain almost machine accuracy, while two iterations yield an error which is of the order of method 2+6, at about two-thirds $(0.64, 0.68, 0.69$, respectively) the cost. On the other hand, Krylov methods do not produce an $\mathrm{SL}(n)$ approximation to the exponential unless the computation is performed to machine accuracy, which, in our particular example, is 3.30, 2.84, and 2.85—about three times more costly than the 2+6 algorithm. For the $\mathrm{SO}(n)$ case, it should be noted that if $Z \in \mathfrak{so}(n)$, then the approximation $\boldsymbol{w} \approx \exp(Z)\boldsymbol{v}$ produced by the Krylov method has the feature that $\|w\|_2 = \|v\|_2$ independently of the number m of iterations: in this case, the Hessenberg matrix produced by the Arnoldi iterations is tridiagonal and skew-symmetric, hence its exponential orthogonal. Thus, Krylov methods are the method of choice for actions of $\mathrm{SO}(n)$ on $\mathbb{R}^n$ [17]. One might extrapolate that, if we wish to compute the exponential $\exp(Z)Q$, where $Q \in \mathrm{SO}(n)$, one could perform only a few iterations of the Krylov method to compute $\boldsymbol{w}_i \approx \exp(Z)\boldsymbol{q}_i$, for $i = 1, 2, \ldots, n$, the $\boldsymbol{q}_i$'s being columns of Q. Unfortunately, the approximation $[\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n]$ ceases to be orthogonal: although $\|\boldsymbol{w}_i\|_2 = 1$, the vectors $\boldsymbol{w}_i$ cease to be linearly independent and the final approximation is not in $\mathrm{SO}(n)$. Similar analysis yields for Stiefel manifolds, unless Krylov methods are implemented to approximate the exponential to machine accuracy.

In passing, we recall that our methods based on a symmetric polar-type decomposition are time-symmetric. Hence it is possible to compose a basic scheme in a symmetric manner, following a technique introduced by Yoshida [24], to obtain higher-order approximations: two orders of accuracy can be obtained at three times the cost of the basic method. For instance, we can use the method 2+6 as a basic algorithm to obtain an approximation of order 4. Thus an approximation of order 4 applied to a vector can be obtained in $17n^2$ operations for $\mathfrak{sl}(n)$ (two splittings and three assemblies), compared to $\mathcal{O}(n^3)$ operations required by the method 4+6.

To conclude our gallery, we compare the method 4+6, an order-4 scheme, whose complexity is described in Table 5, with a $(2,2)$-Padé approximant, which requires $2\frac{2}{3}n^3$ floating point operations when applied to vectors ($2n^3$ for the assembly and $\frac{2}{3}n^3$ for the LU factorization) and $6\frac{2}{3}n^3$ for matrices (since we have to resolve for multiple right-hand sides). The figures obtained by numerical simulations for matrices in $\mathfrak{sl}(n)$ and $\mathrm{SO}(n)$ clearly agree with the theoretical asymptotic values (plotted as solid lines), as shown in Figure 4. The costs of both methods are very similar, as is the error from the exact exponential, although, in the $\mathrm{SL}(n)$ case, the 4+6 scheme preserves the determinant to machine accuracy, while the Padé scheme does not (see Figure 5). Note that a Krylov method iterated to convergence would cost $\approx 18n^3$ operations (assuming that 9 iterations are sufficient to obtain machine accuracy), but it is a

TABLE 5
Complexity for a symmetric polar-type order-4 approximant.

Algorithm	$\mathfrak{sl}(n), \mathfrak{so}(p,q)$		$\mathfrak{so}(n)$	
4+6	Vector	Matrix	Vector	Matrix
Splitting	$5n^3$	$5n^3$	$2\frac{1}{2}n^3$	$2\frac{1}{2}n^3$
Assembly exp	$5n^2$	$4n^3$	$5n^2$	$4n^3$
Total	$5n^3$	$9n^3$	$2\frac{1}{2}n^3$	$6\frac{1}{2}n^3$

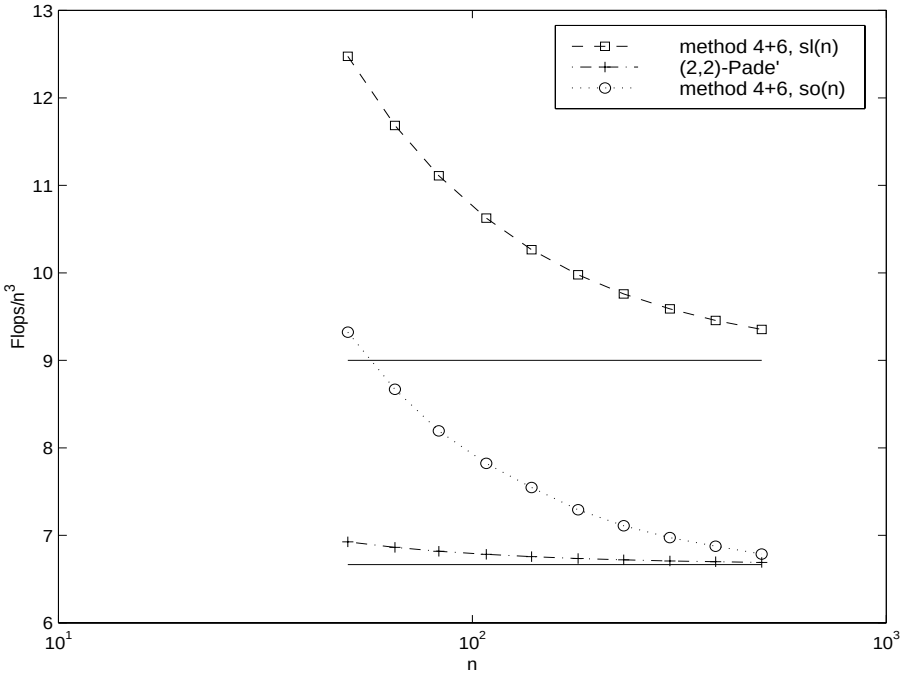

FIG. 4. *Floating point operations (scaled by n^3) versus size for the approximation of the exponential of a matrix in $\mathfrak{sl}(n)$ applied to an $n \times n$ matrix with the order-4 symmetric polar-type algorithm (method $4+6$) and $(2,2)$-Padé approximant.*

clear winner when the exponential is applied to a vector, since our method is an $\mathcal{O}(n^3)$ scheme even in the vector case.

5. Automorphisms based on approximate eigenspace and Schur decompositions. One of the anonymous referees of this paper pointed to possible connections between our theory and the family of splitting methods proposed by Stickel in [21]. Although the original formulation of Stickel does not exactly fit into the framework of this paper, we will see that a modification along the lines of this paper leads to splittings with very interesting properties.

Stickel's approach is based on a commutative splitting derived from the *matrix sign function* $\chi(Z)$. The matrix $\chi(Z)$ has the same eigenvectors as Z, and its eigenvalues are ± 1 or 0 according to whether the corresponding eigenvalues of Z have negative or positive real part, or are purely imaginary.

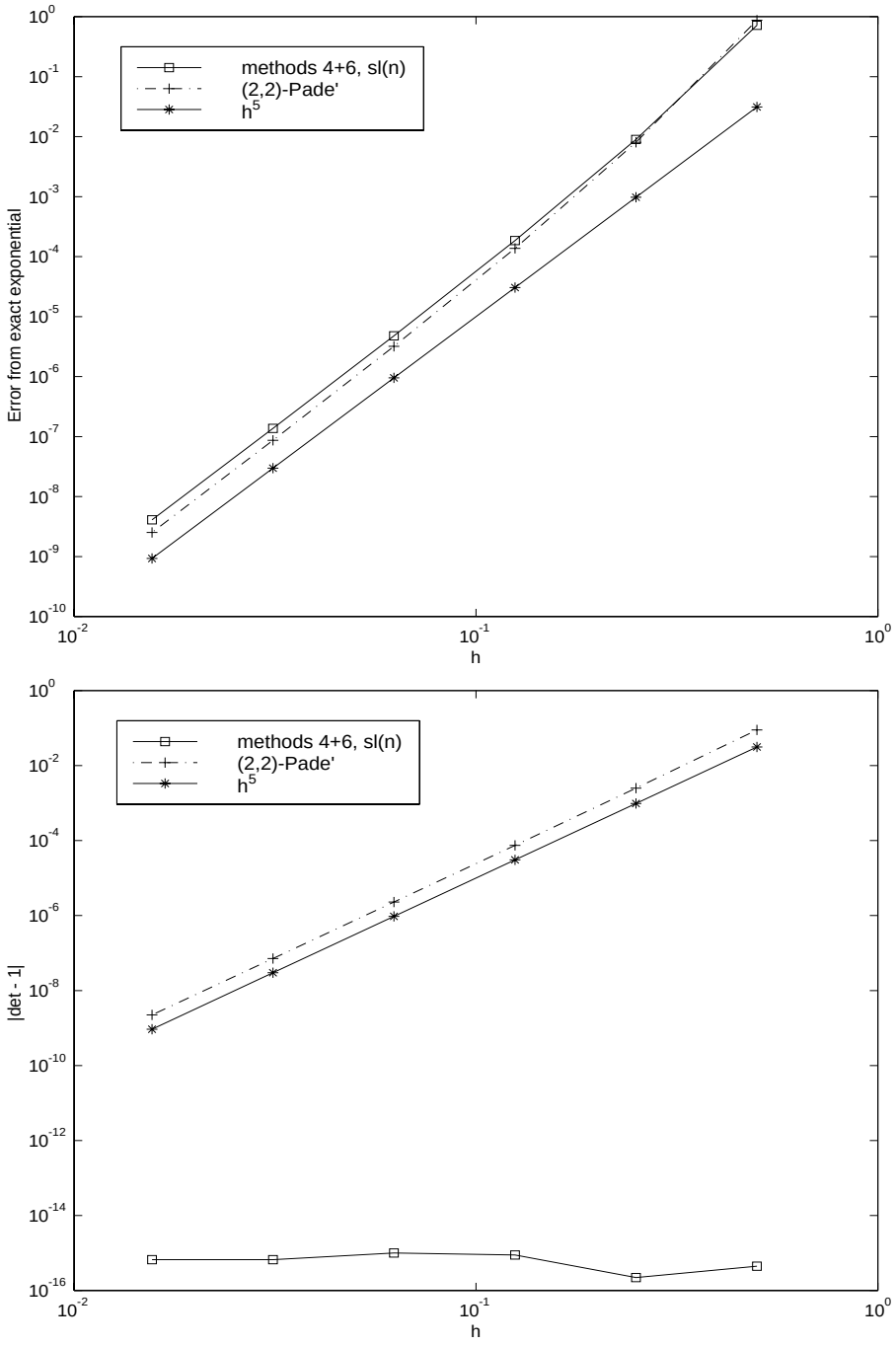

FIG. 5. *Error in the approximation (top) and in the determinant (bottom) versus* h *for the approximation of the exponential of a traceless* 10×10 *matrix of unit norm with the order-*4 *symmetric polar-type algorithm (method* $4+6$*) and* $(2,2)$*-Padé approximant.*

Suppose Z has no purely imaginary eigenvalues.[1] Stickel considers the splitting $Z = Z_1 + Z_2$, where $Z_1 = \frac{1}{2}(Z - Z\chi(Z))$ and $Z_2 = \frac{1}{2}(Z + Z\chi(Z))$. Since $Z\chi(Z) = \chi(Z)Z$, it is clear that Z_1 and Z_2 commute, and hence $\exp(Z) = \exp(Z_1)\exp(Z_2)$ exactly. The same approach may now be employed recursively on Z_1 and Z_2. Note that the map $Z \mapsto Z\chi(Z)$ is an involution, but it is not an involutive automorphism at the algebra level, since it is not linear. Hence the theory of our paper does not directly apply to the splitting methods of Stickel.

A major difficulty in the approach described above is the cost of computing the matrix sign function. It would be of interest to perform similar splittings using approximations S of the sign function. This may lead to a destruction of the commutativity of Z and S, and it should be useful to correct the result by the technique we study in this paper.

Given a matrix S such that $S^2 = I$, the map $Z \mapsto ZS$ is *not* an involutive algebra automorphism. (It is an involution and a linear map; however, in general it is not true that $[Z_1, Z_2]S = [Z_1S, Z_2S]$.) On the other hand, one can consider the map $Z \mapsto SZS$, where S is some approximation of $\chi(Z)$ such that $S^2 = I$, and hence a splitting as we have studied in this paper. Thus, $Z = P + K$, where $P = \frac{1}{2}(Z - SZS)$ and $K = \frac{1}{2}(Z + SZS)$. We can compute $\exp(Z)$ from either of the expansions (3.1) and (3.2), where X and Y are given in (2.4)–(2.6). Note that $XS = -SX$ and $YS = SY$, and that if S is exactly the sign of Z, then $X = 0$. If S is an approximation to $\chi(Z)$, then X and Y are close to commuting, and the expansions converge fast. The exponential of X reduces essentially to a matrix problem of size $p = \operatorname{rank}(S - I)$, while the exponential of Y reduces to a problem of size $n - p$.

How can we produce a suitable approximation $S \approx \chi(Z)$? Let ξ_i and η_j denote right and left eigenvectors of Z, $Z\xi_i = \lambda_i\xi_i$ and $\eta_j^{\mathrm{T}}Z = \lambda_j\eta_j^{\mathrm{T}}$, normalized such that $\eta_j^{\mathrm{T}}\xi_i = \delta_{i,j}$. Then the matrix sign function can be written as

$$\chi(Z) = 2 \sum_{\mathrm{Re}\,(\lambda_i)>0} \xi_i\eta_i^{\mathrm{T}} - I.$$

Given p approximate eigenvectors $\tilde{\xi}_i$ and $\tilde{\eta}_i$, we can use

$$S = 2\sum_{i=1}^{p} \tilde{\xi}_i\tilde{\eta}_i^{\mathrm{T}} - I.$$

Note that if we know p eigenvectors *exactly*, we may use deflation techniques to reduce the size of the problem. Instead, if the eigenvectors are not exactly known, we may use commutators to produce a problem that still can be deflated.

For numerical reasons, a similar approach based on Schur decompositions may be even more attractive. From approximate knowledge of the first p Schur vectors $\tilde{q}_1, \ldots, \tilde{q}_p$ one can employ splittings based on the involutive matrix

$$S = 2\sum_{i=1}^{p} \tilde{q}_i\tilde{q}_i^{\mathrm{T}} - I.$$

The numerical performance of these methods is presently unknown and will have to be addressed in future research.

[1] Stickel introduces shifts $Z - \alpha I$ in the general case, but to keep the exposition simple we avoid the discussion of shifts.

6. Conclusions. In this paper we have introduced numerical algorithms for approximating the matrix exponential. The methods discussed possess the feature that if $Z \in \mathfrak{g}$, then the output is in G, the Lie group of $\mathfrak{g}$, a property that is fundamental to the integration of ODEs by means of Lie-group methods.

The proposed methods have a complexity of $\mathcal{O}(\kappa n^3)$, where n denotes the size of the matrix whose exponential we wish to approximate. Typically, for moderate order (up to order 4), the constant κ is less than 10, whereas the exact computation of a matrix exponential in MATLAB (which employs the scaling and squaring with a Padé approximant) generally costs between $20n^3$ and $30n^3$.

We compare methods of the same order of accuracy applied to a vector $\boldsymbol{v} \in \mathbb{R}^n$ and to a matrix $B \in G$:

- *For the case $F(t,Z)\boldsymbol{v} \approx \exp(tZ)\boldsymbol{v}$, where $\boldsymbol{v}$ is a vector.* Symmetric polar-type methods are slightly cheaper than their nonsymmetric variant. For the $\mathrm{SO}(n)$ case, the complexity of symmetric methods is very comparable to that of diagonal Padé approximants of the same order.

  The complexity of the method 2+6 is $\mathcal{O}(n^2)$, while for the rest of our methods it is $\mathcal{O}(n^3)$. Krylov subspace methods do, however, have the complexity $\mathcal{O}(n^2)$ if the number of iterations is independent of n. Thus, if it is important to stay on the group, we recommend Krylov methods with iteration to machine accuracy for this kind of problem. If convergence of Krylov methods is slow, our methods might be good alternatives. See [8] for accurate bounds on the number m of iterations of Krylov methods.
- *For the case $F(t,Z)B \approx \exp(tZ)B$, with B an $n \times n$ matrix.* Nonsymmetric polar-type methods are marginally cheaper than their symmetric counterpart; however, the latter should be preferred when the underlying ODE scheme is time-symmetric. The proposed methods have a complexity very comparable to that of diagonal Padé approximants of the same order (asymptotically they require slightly less operations in the $\mathrm{SO}(n)$ case); in addition they map $\mathfrak{sl}(n)$ to $\mathrm{SL}(n)$, a property that is shared by neither Padé approximants nor Krylov iterations not carried to convergence.

  For these classes of problems our proposed methods seem to be very competitive.

It should also be noted that significant advantages arise when Z is a banded matrix. For instance, the cost of method 2+6 scales as $\mathcal{O}(nr)$ for $F(t,Z)$ applied to a vector and $\mathcal{O}(n^2 r)$ for $F(t,Z)$ applied to a matrix when Z has bandwidth $2r+1$. The savings are less striking for higher-order methods since commutation usually causes fill-in in the splitting.

As mentioned earlier in the paper, [3] recently proposed similar splitting methods that also produce an output in G when $Z \in \mathfrak{g}$. With respect to their schemes, ours display a slight computational gain: for the $\mathrm{SO}(n)$ case, Celledoni and Iserles propose an order-4 scheme whose complexity is $11\frac{1}{2}n^3$, while our order-4 schemes (method 3+5 with order-4 corrections and method 4+6) cost $6n^3$, $6\frac{1}{2}n^3$ operations—very comparable to the diagonal Padé approximant of the same order.

However, the novelty of our approach lies in the fact that it is based on a rather general theory that can include very different splitting methods already existent in literature (for instance, the method 2+6 is nothing else than a Strang-type splitting), obtaining in an elegant and neat way the otherwise complicated order conditions.

REFERENCES

[1] A. O. Barut, J. R. Zeni, and A. Laufer, *The exponential map for the conformal group* O(2,4), J. Phys. A, 27 (1994), pp. 5239–5250.
[2] R. Carter, G. Segal, and I. Macdonald, *Lectures on Lie Groups and Lie Algebras*, London Math. Soc. Stud. Texts 32, Cambridge University Press, Cambridge, UK, 1995.
[3] E. Celledoni and A. Iserles, *Methods for the approximation of the matrix exponential in a Lie-algebraic setting*, IMA J. Numer. Anal., 21 (2001), pp. 463–488.
[4] E. Celledoni and A. Iserles, *Approximating the exponential from a Lie algebra to a Lie group*, Math. Comp., 69 (2000), pp. 1457–1480.
[5] E. Celledoni, A. Iserles, S. P. Nørsett, and B. Orel, *Complexity Theory for Lie-Group Solvers*, Technical Report NA1999/20, DAMTP, University of Cambridge, Cambridge, UK, 1999.
[6] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins University Press, Baltimore, 1989.
[7] S. Helgason, *Differential Geometry, Lie Groups and Symmetric Spaces*, Academic Press, New York, 1978.
[8] M. Hochbruck and C. Lubich, *On Krylov subspace approximations to the matrix exponential operator*, SIAM J. Numer. Anal., 34 (1997), pp. 1911–1925.
[9] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1985.
[10] A. Iserles, H. Munthe-Kaas, S. P. Nørsett, and A. Zanna, *Lie-group methods*, Acta Numer., 9 (2000), pp. 215–365.
[11] J. D. Lawson, *Polar and Ol'shanskii decompositions*, J. Reine Angew. Math., 448 (1994), pp. 191–219.
[12] F. Silva Leite and P. Crouch, *Closed forms for the exponential mapping on matrix Lie groups based on Putzer's method*, J. Math. Phys., 40 (1999), pp. 3561–3568.
[13] J. E. Marsden and T. S. Ratiu, *Introduction to Mechanics and Symmetry*, Springer-Verlag, New York, 1994.
[14] C. Moler and C. F. Van Loan, *Nineteen dubious ways to compute the exponential of a matrix*, SIAM Rev., 20 (1978), pp. 801–836.
[15] H. Munthe-Kaas, G. R. W. Quispel, and A. Zanna, *Application of Symmetric Spaces and Lie Triple Systems in Numerical Analysis*, Technical Report 217, Department of Informatics, University of Bergen, Norway, 2001.
[16] H. Munthe-Kaas, G. R. W. Quispel, and A. Zanna, *Generalized polar decompositions on Lie groups with involutive automorphisms*, Found. Comput. Math., 1 (2001), pp. 297–324.
[17] H. Munthe-Kaas and A. Zanna, *Numerical integration of differential equations on homogeneous manifolds*, in Foundations of Computational Mathematics, F. Cucker, ed., Springer-Verlag, Berlin, 1997, pp. 305–315.
[18] H. Munthe-Kaas and A. Zanna, *Lie-Group Integrators Based on Generalized Polar Coordinates*, manuscript.
[19] B. Owren and A. Marthinsen, *Integration Methods Based on Canonical Coordinates of the Second Kind*, Technical Report Numerics 5/1999, Department of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway, 1999.
[20] Y. Saad, *Analysis of some Krylov subspace approximations to the matrix exponential operator*, SIAM J. Numer. Anal., 29 (1992), pp. 209–228.
[21] E. U. Stickel, *A splitting method for the calculation of the matrix exponential*, Analysis, 14 (1994), pp. 103–112.
[22] V. S. Varadarajan, *Lie Groups, Lie Algebras, and Their Representation*, Grad. Texts in Math. 102, Springer-Verlag, New York, 1984.
[23] R. C. Ward, *Numerical computation of the matrix exponential with accuracy estimate*, SIAM J. Numer. Anal., 14 (1977), pp. 600–610.
[24] H. Yoshida, *Construction of higher order symplectic integrators*, Phys. Lett. A, 150 (1990), pp. 262–268.
[25] A. Zanna, *Recurrence Relation for the Factors in the Polar Decomposition on Lie Groups*, Technical Report 192, Department of Informatics, University of Bergen, Norway, 2000.