NORGES TEKNISK-NATURVITENSKAPELIGE
UNIVERSITET

On the implementation of Lie group methods on the Stiefel manifold

by

Elena Celledoni and Brynjulf Owren

PREPRINT
NUMERICS NO. 9/2001

NORWEGIAN UNIVERSITY OF SCIENCE AND
TECHNOLOGY
TRONDHEIM, NORWAY

This report has URL http://www.math.ntnu.no/preprint/numerics/2001/N9-2001.ps
Address: Department of Mathematical Sciences, Norwegian University of Science and Technology,
N-7491 Trondheim, Norway.

On the implementation of Lie group methods on the Stiefel manifold *

Elena Celledoni† Brynjulf Owren‡

November 23, 2001

Abstract

There are several applications in which one needs to integrate a system of ODEs whose solution is an $n \times p$ matrix with orthonormal columns. In recent papers algorithms of arithmetic complexity order $n \times p^2$ have been proposed. The class of Lie group integrators may seem like a worth while alternative for this class of problems, but it has not been clear how to implement such methods with $\mathcal{O}(np^2)$ complexity. In this paper we show how Lie group methods can be implemented in a computationally competitive way, by exploiting that analytic functions of $n \times n$ matrices of rank $2p$ can be computed with $\mathcal{O}(np^2)$ complexity.

AMS Subject Classification: 65L05
Key Words: time integration, geometric integration, numerical integration of ordinary differential equations on manifolds, numerical analysis, Lie algebras, Lie groups.

1 Introduction

We consider the following initial value problem

$$\begin{cases} Y' = F(t, Y), \\ Y(0) = Y_0 \end{cases} \qquad Y \in \mathbb{V}_p(\mathbb{R}^n) \qquad Y^{\mathrm{T}} Y = I_p \tag{1}$$

where $\mathbb{V}_p(\mathbb{R}^n)$ is the Stiefel manifold, the set of all $n \times p$ matrices with p orthonormal columns. There is a variety of applications which can be modelled by (1) ranging from optimisation problems [4], to the computation of Lyapunov exponents, see Bridges and Reich [1], or Dieci and Van Vleck [9]. The cited references also propose integration schemes for (1) and they all have in common that they exhibit a computational complexity of $\mathcal{O}(np^2)$.

The Lie group integrators as we now know them were introduced by Crouch and Grossman [7] and Munthe-Kaas [13, 14]. In their full generality, they include also the standard Runge-Kutta methods as well as many other types of schemes. In applying Lie group methods to the problem (1), it is natural to make use of the fact the the $\mathbb{V}_p(\mathbb{R}^n)$ is a homogeneous space, acted upon by the Lie group of $n \times n$ orthogonal matrices. By following naively the recipe

*This work was in part sponsored by The Norwegian Research Council under contract no. 111038/410, through the SYNODE project. WWW: `http://www.math.ntnu.no/num/synode/`

†Email: `Elena.Celledoni@math.ntnu.no`, WWW: `http://www.math.ntnu.no/~elenac/`

‡Email: `Brynjulf.Owren@math.ntnu.no`, WWW: `http://www.math.ntnu.no/~bryn/`

proposed by either of [7, 14] one ends up with an integration method which apply several evaluations of the matrix exponential of $n \times n$ skew-symmetric matrices in each step. Using general software for this, one must expect the computational cost to be proportional to n^3 and thus not competitive to the methods in the papers [1, 9] as has also been pointed out by these authors. The aim of this paper is to show how Lie group integrators for approximating the solution of ODEs on the Stiefel manifold can be implemented with computational complexity $\mathcal{O}(np^2)$. Another way of achieving this has been proposed by Krogstad [12] in a recent paper. A key point is to show how one can calculate expressions of type $\psi(A) \cdot Q$ in order np^2 arithmetic complexity. Here ψ is an analytic function, A is an $n \times n$ matrix of rank $2p$, and $Q \in \mathbb{V}_p(\mathbb{R}^n)$.

A special feature of the Lie group integrators is that they are *intrinsic*. Generally, this means that the integration scheme does not depend on how the manifold is embedded into a Euclidean space. In particular, it does not assume that the vector field which defines the differential system is to be evaluated on points not belonging to the manifold. This property of Lie group methods is typically not shared by the projection methods.

2 Background theory and notation

A general idea for constructing intrinsic numerical integrators for

$$y' = F(t, y) \quad y(t) \in \mathcal{M}, \quad y(0) = q \in \mathcal{M}, \tag{2}$$

was described in [3]. It is based on representing a neighbourhood of $q \in \mathcal{M}$ by means of a map $\mathcal{R} : T\mathcal{M} \to \mathcal{M}$ whose restriction to $T_q\mathcal{M}$, is denoted by $\mathcal{R}_q : T_q\mathcal{M} \to \mathcal{M}$ ($\forall q \in \mathcal{M}$). Locally around q the solution $y(t)$ is expressed in terms of a curve in the tangent space $T_q\mathcal{M}$ at q, $\sigma(t)$, i.e.

$$y(t) = \mathcal{R}_q(\sigma(t)), \quad \sigma(t) \in T_q\mathcal{M}. \tag{3}$$

If it is possible to find a differential equation for σ, the numerical integration can be locally carried out on the tangent space with a timestepping procedure for $\sigma(t)$.

In this paper we assume that $\mathcal{R}$ is a retraction map [16], i.e. for each fiber $T_q\mathcal{M}$ of $T\mathcal{M}$ we have:

- $\mathcal{R}_q$ is defined in some open ball $B(0, r_q) \subset T_q\mathcal{M}$ of radius r_q about 0.
- $\mathcal{R}_q(v) = q$ if and only if $v = 0 \in T_q\mathcal{M}$;
- $\mathcal{R}'_q\big|_0 = \mathbb{1}_{T_q\mathcal{M}}$.

Here $\mathcal{R}'_q : TT_q\mathcal{M} \to T\mathcal{M}$ is the tangent of the map $\mathcal{R}_q$. We may identify $TT_q\mathcal{M}$ with $T_q\mathcal{M} \times T_q\mathcal{M}$, and for each pair (u, v) we may fix the first factor, and get the map $R'_q|_u$: $T_q\mathcal{M} \to T_{\bar{q}}\mathcal{M}$ where $\bar{q} = R_q(u)$. We proceed by differentiating (3) with respect to t to get

$$F(t, y) = y'(t) = R'_q|_{\sigma(t)}(\sigma'(t))$$

We may choose $q = y_0$ so that $\sigma(0) = 0$. By the continuity of σ and the fact that $R'_q|_0 = \mathbb{1}_{T_q\mathcal{M}}$ if follows that for small t the map $R'_q|_{\sigma(t)}$ is invertible, and we obtain the following differential equation for σ

$$\sigma'(t) = \left(\mathcal{R}'_q\big|_{\sigma(t)}\right)^{-1}\left(F(t, \mathcal{R}_q(\sigma(t)))\right). \tag{4}$$

The following algorithm summarises the procedure described above:

Algorithm 2.1 *RK based on retractions*

Input y_0, *stepsize* h, *and RK parameters* a_{ij}, b_i.

for $n = 0, 1, \ldots$

for $i = 1, \ldots, s$
$u_i := h \sum_j a_{ij} k_j$
$Y := \mathcal{R}_{y_n}(u_i)$
$m_i := F(t_n + hc_i, Y)$
$k_i := \left(\mathcal{R}_{y_n}^{-1}|_Y\right)' (m_i)$
end for

$v = h \sum_i b_i k_i$
$y_{n+1} = \mathcal{R}_{y_n}(v)$
[Update stepsize h *if desired]*

end for

For this algorithm to be of practical value, it is of course required that both $\mathcal{R}_q$ and $\left(\mathcal{R}_q'\right)^{-1}$ are computationally affordable.

Suppose we are given a Lie group G with identity element e acting on $\mathcal{M}$ and we denote by $\mathfrak{g}$ the Lie algebra of G, with $\Lambda : G \times \mathcal{M} \to \mathcal{M}$ the Lie group action denoted by $\Lambda(g, q) = \Lambda_q(g) = g \cdot q$. Denote by $\mathcal{X}(\mathcal{M})$ the Lie algebra of vector fields on $\mathcal{M}$ under the Lie-Poisson bracket. Let ρ be the Lie algebra homomorphism from $\mathfrak{g}$ to $\mathcal{X}(\mathcal{M})$, that associates to any $v \in \mathfrak{g}$ the vector field $\hat{v}$

$$\hat{v}|_q = \rho_q(v) = \frac{d}{d\tau}\psi(\tau v) \cdot q\bigg|_{\tau=0} = \Lambda_q'|_e \circ \psi'|_0,$$

where ψ is a coordinate map from the Lie algebra to the Lie group and we assume $\psi(0) = e$ (it sends the null element in $\mathfrak{g}$ to the identity in G). We denote by $\rho_q(v) = \hat{v}|_q$.

Provided we can find a map

$$a_q : T_q\mathcal{M} \to \mathfrak{g}, \text{ s.t. } \rho_q \circ a_q = \mathbb{1}_{T_q\mathcal{M}},$$

it is possible to write (2) in the form

$$y' = \rho_y(a_y(F(t, y))), \;\; y(t) \in \mathcal{M}, \;\; y(0) = q \in \mathcal{M}. \tag{5}$$

There is a natural way of defining a retraction in this situation

Proposition 2.2 *Suppose that* $a_q : T_q\mathcal{M} \to \mathfrak{g}$ *is a linear map satisfying* $\rho_q \circ a_q = \mathbb{1}_{T_q\mathcal{M}}$. *Then the map defined as*

$$\mathcal{R}_q(v) = \psi(a_q(v)) \cdot q \tag{6}$$

is a retraction.

Proof. Clearly there must be a neighbourhood including $0 \in \mathrm{T}_q\mathcal{M}$ on which $\mathcal{R}_q$ is defined. We proceed to prove that $\mathcal{R}'_q|_0 = \mathbb{1}_{T_q\mathcal{M}}$. Write $\Lambda_q(g) = \Lambda(g, q) = g \cdot q$ for any $g \in G$. Then $\mathcal{R}_q = \Lambda_q \circ \psi \circ a_q$ and by the chain rule $\mathcal{R}'_q = \Lambda'_q \circ \psi' \circ a'_q$. The linearity of a_q also implies that $a'_q|_u = a_q$ for any $u \in T_q\mathcal{M}$. On the 0-fiber we get $\mathcal{R}'_q|_0 = \Lambda'_q|_e \circ \psi'|_0 \circ a_q = \rho_q \circ a_q = \mathbb{1}_{T_q\mathcal{M}}$. The requirement $\psi(0) = e$ asserts that $\mathcal{R}_q(0) = q$ and by the inverse function theorem, there must be some neighbourhood of $0 \in T_q\mathcal{M}$ in which $\mathcal{R}_q$ is injective. □

We can now use the map (6) and its inverse tangent map in (4) and solve (2) numerically via the retraction algorithm described above.

If $\sigma(t) \in T_q\mathcal{M}$, is the solution of (4) then $\gamma(t) = a_q(\sigma(t))$ is a curve in the Lie algebra. The solution $y(t)$ of the original differential equation can then be represented locally in the alternative way

$$y(t) = \psi(\gamma(t)) \cdot q.$$

It is now possible to obtain a differential equation for γ in the Lie algebra,

$$\gamma'(t) = \mathrm{d}\psi^{-1}_{\gamma(t)}(a_p(F(t, p))), \qquad p = \psi(\gamma(t)) \cdot q.$$

where $\mathrm{d}\psi_u$ is the right trivialised tangent map of ψ at $u \in \mathfrak{g}$, by a slight abuse of notation

$$\mathrm{d}\psi_u \cdot \psi(u) = \left.\frac{d}{d\tau}\right|_{\tau=0} \psi(u + \tau v), \quad v \in \mathfrak{g}.$$

The methods introduced by Munthe-Kaas in [14] are based on a time stepping procedure for this equation in the Lie algebra.

Algorithm 2.3 *RK-MK*

Input y_0*, stepsize* h*, and RK parameters* a_{ij}, b_i.

for $n = 0, 1, \dots$

  for $i = 1, \dots, s$
    $u_i := h \sum_j a_{ij} k_j$
    $Y_i := \psi(u_i) \cdot y_n$
    $m_i := a_{Y_i}(F(t_n + hc_i, Y_i))$
    $k_i := \mathrm{d}\psi^{-1}_{u_i}(m_i)$
  end for

  $v = h \sum_i b_i k_i$
  $y_{n+1} = \psi(v) \cdot y_n$
  [Update stepsize h *if desired]*

end for

In the case $\mathcal{M} = G$ the RK-MK algorithm is very similar to the retraction algorithm since in this case the tangent space at each point to the Lie group can be identified, (for example via a right trivialization), with the Lie algebra.

When $\mathcal{M}$ is an homogeneous space, and in particular in the case of the Stiefel manifold, the Munthe-Kaas methods do not coincide with the retraction methods any longer. Finding the inverse tangent of the retraction map may not be a trivial task.

As it turns out it is easy to express local parametrisations of the solutions of ODEs on manifolds in the framework of retractions using a minimal set of parameters.

In the sequel we will consider both strategies described above and derive computationally competitive implementations of known Lie-group methods. In the next section we define some examples of retractions $\mathcal{R}_q$ and corresponding inverse tangents $\left(\mathcal{R}_q'\right)^{-1}$ on the Stiefel manifold.

3 Retractions on the Stiefel manifold

In order to be able to define a retraction map on the Stiefel manifold we need a convenient description of the tangent space. We here denote by $\mathcal{M} = \mathbb{V}_p(\mathbb{R}^n)$ the Stiefel manifold, and with capital letters its elements and the elements of the tangent space $T_Q\mathcal{M}$, $(Q \in \mathcal{M})$.

Given a curve $Y(t)$ on the Stiefel manifold, $Y(0) = Q$, differentiating $Y(t)^{\mathrm{T}}Y(t) = I$ one gets the following equation for the general tangent vector $\frac{d}{dt}\big|_{t=0} Y(t) = V$ at Q:

$$V^{\mathrm{T}}Q + Q^{\mathrm{T}}V = 0,$$

so that $Q^{\mathrm{T}}V$ is a $p \times p$ skew-symmetric matrix.

We claim that any tangent vector V at Q can be expressed in the form

$$V = (gQ^{\mathrm{T}} - Qg^{\mathrm{T}}) \cdot Q, \tag{7}$$

with g suitable $n \times p$ matrix. In fact, by multiplying from the left by Q^T, we get

$$g^{\mathrm{T}}Q - Q^{\mathrm{T}}g = -Q^{\mathrm{T}}V,$$

thus, $-\frac{1}{2}Q^TV$ is the skew-symmetric part of $g^{\mathrm{T}}Q$. Assuming now that S is the symmetric part of $g^{\mathrm{T}}Q$ we get from (7)

$$g = V + Q\,g^TQ = V + Q(S - \frac{Q^{\mathrm{T}}V}{2}). \tag{8}$$

It is easy to verify by direct calculation that $gQ^{\mathrm{T}} - Qg^{\mathrm{T}}$ is independent of S which therefore can be chosen arbitrarily. We here fix $S = 0$ and finally get

$$g = V - Q\frac{Q^{\mathrm{T}}V}{2}. \tag{9}$$

which proves our claim. We proceed by using the representation (7) of $T_Q\mathcal{M}$ for defining retractions on the Stiefel manifold. Define the maps

$$\begin{aligned} g_Q : T_Q\mathcal{M} \to \mathbb{R}^{n\times p}, &\qquad g_Q(V) = V - Q\frac{Q^{\mathrm{T}}V}{2} \\ a_Q : T_Q\mathcal{M} \to \mathfrak{g}, &\qquad a_Q(V) = g_Q(V)Q^{\mathrm{T}} - Qg_Q(V)^{\mathrm{T}}, \end{aligned}$$

We rewrite (1) in the form (5), such that

$$Y' = a_Y(F(t, Y)) \cdot Y, \tag{10}$$

where "$\cdot$" here signifies matrix-matrix multiplication. In (10) it is now necessary that $\psi'|_0 = \mathbb{1}_{\mathfrak{g}}$ in order to obtain $\rho_q \circ a_q = \mathbb{1}_{T_Q\mathcal{M}}$ as in Proposition 2.2. With this in mind, we may consider retractions of the type defined by equation (6), so that on $T_Q\mathcal{M}$:

$$\mathcal{R}_Q(V) = \psi\big(g_Q(V)Q^{\mathrm{T}} - Qg_Q(V)^{\mathrm{T}}\big) \cdot Q, \tag{11}$$

with ψ a coordinate map from $\mathfrak{so}(n)$ to $\mathrm{SO}(n)$, satisfying $\psi(0) = I$ and $\psi'|_0 = \mathbb{1}_{\mathfrak{g}}$. In this paper we will consider analytic maps like for example the matrix exponential, or the Cayley transformation, $\mathrm{cay}(M) = (I - \frac{1}{2}M)^{-1}(I + \frac{1}{2}M)$.

Both exp and cay are analytic functions of $g_Q(V)Q^{\mathrm{T}} - Qg_Q(V)^{\mathrm{T}}$. The crucial feature that we will exploit in the computation of the map $\mathcal{R}_Q$ is the factorisation

$$g_Q(V)Q^{\mathrm{T}} - Qg_Q(V)^{\mathrm{T}} = (g_Q(V), -Q) \cdot (Q, g_Q(V))^{\mathrm{T}}.$$

In fact, analytic functions of matrices given as a product of two $n \times k$ matrices ($k << n$) can be computed very cheaply. Let ψ be an analytic function such that $\psi(0) = 1$ and suppose that the matrix C can be factorized: $C = \alpha\beta^T$ where α and β are $n \times k$ matrices. Then

$$\psi(C) = I + \alpha\phi(D)\beta^{\mathrm{T}}, \quad D = \beta^{\mathrm{T}}\alpha, \quad \phi(z) = \frac{\psi(z) - 1}{z}. \tag{12}$$

This formula was obtained in [2] for $\psi = \exp$. The cost of computing $\psi(C) \cdot X$ with X $n \times k$ matrix, is $4nk^2 + nk + \mathcal{O}(k^3)$ flops that when $k << n$ amounts to $\mathcal{O}(nk^2)$ flops.

In light of this observation we can rewrite (11) as follows:

$$\mathcal{R}_Q(V) = Q + (g_Q(V), -Q)\,\phi(D)\,(Q, g_Q(V))^{\mathrm{T}} \cdot Q, \qquad D := (Q, g_Q(V))^{\mathrm{T}}\,(g_Q(V), -Q)\,, \tag{13}$$

with $\phi(z)$ as defined in (12), and we may conclude that $\mathcal{R}_Q(V)$ can be computed in $\mathcal{O}(np^2)$ operations.

Based on this map one can choose to construct numerical integrators *a la* Munthe-Kaas following algorithm (2.3) or alternatively find a computationally competitive strategy for computing $\big(\mathcal{R}_Q^{-1}\big)'$ following algorithm (2.1). We will discuss how to achieve a computationally competitive implementation of these strategies in the next sections.

3.1 Cheap implementation of RK-MK algorithms

By looking at the RK-MK algorithm presented in Section 2, we see that for each stage in the Runge-Kutta method the main computational tasks are the map $a_Q : T_Q\mathcal{M} \to \mathfrak{so}(n)$ the coordinate map $\psi : \mathfrak{so} \to \mathrm{SO}$ and the inverse trivialised tangent $\mathrm{d}\psi_u^{-1} : \mathfrak{so}(n) \to \mathfrak{so}(n)$. In the previous section we have shown how to choose and implement a_Q and ψ with a computational cost of $\mathcal{O}(np^2)$ flops. Here, we first address the computation of $\mathrm{d}\psi_u^{-1} : \mathfrak{so}(n) \to \mathfrak{so}(n)$ in the particular case of the Stiefel manifold, and then show how the overall algorithm gets the same computational complexity.

There are known explicit expressions for $\mathrm{d}\psi_u^{-1}(v)$ when ψ is exp or cay like we have assumed, and in particular for exp

$$\mathrm{dexp}_u^{-1}(v) = \varphi(\mathrm{ad}_u)(v) \tag{14}$$

with $\varphi(z) := \frac{z}{e^z - 1}$ and $\mathrm{ad}_a(b) = [a, b]$, with $[\cdot, \cdot]$ the matrix commutator. For cay

$$\mathrm{dcay}_u^{-1}(v) = v - \frac{1}{2}[u, v] - \frac{1}{4}u \cdot v \cdot u = (I - \frac{1}{2}u)v(I + \frac{1}{2}u). \tag{15}$$

Assume that both u and v can be expressed in a factorized form $u = \alpha U \beta^{\mathrm{T}}$ and $v = \tilde{\alpha} V \tilde{\beta}^{\mathrm{T}}$. Here α and β are $n \times k$ matrices, U is $k \times k$, $\tilde{\alpha}$ and $\tilde{\beta}$ are $n \times m$, and V is $m \times m$ with k and m small multiples of p. We may now give a procedure for computing $\mathrm{d}\psi_u^{-1}(v)$ as given by (14) and (15) respectively, at a cost of $\mathcal{O}(np^2)$ flops.

Consider the two $(k+m) \times (k+m)$ matrices

$$\hat{U} := \begin{bmatrix} U & O \\ O & O \end{bmatrix}, \quad \hat{V} := \begin{bmatrix} O & O \\ O & V \end{bmatrix},$$

then u and v can be expressed in the following form:

$$u = \alpha U \beta^{\mathrm{T}} = (\alpha \tilde{\alpha})\, \hat{U} \left(\beta \tilde{\beta}\right)^{\mathrm{T}} \quad v = \tilde{\alpha} V \tilde{\beta}^{\mathrm{T}} = (\alpha \tilde{\alpha})\, \hat{V} \left(\beta \tilde{\beta}\right)^{\mathrm{T}}.$$

The commutator of u and v can be written in the following way

$$[\alpha U \beta^{\mathrm{T}}, \tilde{\alpha} V \tilde{\beta}^{\mathrm{T}}] = (\alpha \tilde{\alpha}) \left[\hat{U}, \hat{V}\right]_S \left(\beta \tilde{\beta}\right)^{\mathrm{T}}$$

with

$$S := \left(\beta \tilde{\beta}\right)^{\mathrm{T}} (\alpha \tilde{\alpha})$$

and

$$[A, B]_S := ASB - BSA,$$

for A, B, S $(k+m) \times (k+m)$ matrices. We now substitute these expressions into $d\exp_u^{-1}(v)$ and $\mathrm{dcay}_u^{-1}(v)$ to get

$$\mathrm{dexp}_{\alpha U \beta^{\mathrm{T}}}^{-1} \left(\tilde{\alpha} V \tilde{\beta}^{\mathrm{T}}\right) = (\alpha \tilde{\alpha})\, \mathrm{dexp}_{S,\hat{U}}^{-1} \left(\hat{V}\right) \left(\beta \tilde{\beta}\right)^{\mathrm{T}}, \quad \mathrm{dexp}_{S,\hat{U}}^{-1} \left(\hat{V}\right) := \varphi\left(\mathrm{ad}_{S,\hat{U}}\right) \left(\hat{V}\right) \tag{16}$$

with

$$\mathrm{ad}_{S,A}(B) := [A, B]_S,$$

and

$$\mathrm{dcay}_{\alpha U \beta^{\mathrm{T}}}^{-1} \left(\tilde{\alpha} V \tilde{\beta}^{\mathrm{T}}\right) = (\alpha \tilde{\alpha})\, \mathrm{dcay}_{S,\hat{U}}^{-1}(\hat{V}) \left(\beta \tilde{\beta}\right)^{\mathrm{T}}, \mathrm{dcay}_{S,\hat{U}}^{-1}(\hat{V}) = \hat{V} - \frac{1}{2}[\hat{U}, \hat{V}]_S - \frac{1}{4}\hat{U} S \hat{V} S \hat{U}. \tag{17}$$

Notice that in the expressions (16) and (17) we compute commutators and matrix-matrix multiplications of $(k+m) \times (k+m)$ matrices instead of $n \times n$. The products $(\alpha \tilde{\alpha})$ and $\left(\beta \tilde{\beta}\right)^{\mathrm{T}}$ in (16) and (17) are never computed explicitly, each factor is applied sequentially to an appropriate $n \times p$ matrix, an operation resulting in a computational cost of $\mathcal{O}(np^2)$ flops.

Given that we can compute $\psi(V) \cdot Q$, $a_Q(V)$ and also $\mathrm{d}\psi_u^{-1}(v)$ in $\mathcal{O}(np^2)$ flops, we now discuss how to implement the RK-MK algorithm in $\mathcal{O}(np^2)$ flops.

We use the notation

$$\mathrm{d}\psi_{S,A}^{-1},$$

to signify $\mathrm{dexp}_{S,A}^{-1}$ and $\mathrm{dcay}_{S,A}^{-1}$ respectively. Of course, when S coincides with the identity matrix we get back the original definitions (14) and (15). As observed in the previous section $F(t, Y) = a_Y \cdot Y$ and a_Y can be written in a factorized form as a product of an $n \times 2p$ and a

$2p \times n$ matrix. In the algorithm we write $F(t_n + c_i h, Y) = \alpha_i \beta_i^{\mathrm{T}} Y$, and we define matrices A_i and B_i $n \times 2ps$ as follows:

$$A_i := (\alpha_1, \ldots, \alpha_i, O, \ldots, O)\,, \qquad B_i := (\beta_1, \ldots, \beta_i, O, \ldots, O)\,,$$

where s is the number of stages of the explicit Runge-Kutta method we consider.

In light of the previous observations we now rewrite algorithm (2.3) in a form based on computations with $2ps \times 2ps$ matrices.

Algorithm 3.1 RK-MK

Input *y_0, stepsize h, and RK parameters a_{ij}, b_i.*

for $n = 0, 1, \ldots$

$U_1 = 0$, $K_1 = I$ *$2ps \times 2ps$ matrices* $A_0 = B_0 = 0$ *$n \times 2ps$ matrices*
for $i = 1, \ldots, s$
$U_i := h \sum_{j=1}^{i-1} a_{ij} K_j$ *with* U_i *and* K_j, $j = 1, \ldots, i-1$, *$2ps \times 2ps$ matrices*
$u_i := A_{i-1} U_i B_{i-1}^{\mathrm{T}}$, A_{i-1} *and* B_{i-1} *$n \times 2ps$ matrices,*
$Y := \psi(u_i) \cdot y_n$,
$F_i := F(t_n + hc_i, Y) = \alpha_i \beta_i^{\mathrm{T}} Y$, , *and* α_i, β_i *$n \times 2p$ matrices*
$A_i := (\alpha_1, \ldots, \alpha_i, O, \ldots, O)$, $\quad B_i := (\beta_1, \ldots, \beta_i, O, \ldots, O)$ *with* A_i *and* B_i *$n \times 2ps$ matrices*
$K_i := \mathrm{d}\psi_{S,U_i}^{-1}(I_i)$, $\quad S := B_i^{\mathrm{T}} A_i$ *and* I_i *such that* $\alpha_i \beta_i^{\mathrm{T}} = A_i I_i B_i^{\mathrm{T}}$
end for

$V = h \sum_i b_i K_i$
$v = A_s V B_s^{\mathrm{T}}$
$y_{n+1} = \psi(v) \cdot y_n$
[Update stepsize h if desired]

end for

In the above algorithm u_i $i = 1, \ldots, s$ is stored in a factorized form and $Y := \psi(u_i) \cdot y_n$, is computed by using (12) at the cost of $\mathcal{O}(np^2)$ flops. Note that due to the sparsity of U_i we have that $u_i = A_k U_i B_k^{\mathrm{T}}$ for $k = i-1, \ldots, s$.

3.2 Inverse tangent map for the retractions

In the retractions algorithm the implementation of the retraction map and of its inverse tangent is needed. We here derive an algorithm to compute the inverse tangent of the retraction defined by (11). Assume $V \in T_Q\mathcal{M}$ and

$$P = \mathcal{R}_Q(V),$$

this means

$$P = \psi(a_Q(V)) \cdot Q, \qquad a_Q(V) = g_Q(V) Q^{\mathrm{T}} - Q g_Q(V)^{\mathrm{T}},$$

and $g_Q(V) = V - Q\frac{Q^{\mathrm{T}}V}{2}$. Since $R_Q = \Lambda_Q \circ \psi \circ a_Q$, with $\Lambda_Q(g) = g \cdot Q$ (the action of $g \in \mathrm{SO}(n)$ on $Q \in \mathcal{M}$), by the chain rule we find that

$$R_Q' = \Lambda_Q' \circ \psi' \circ a_Q',$$

and $R_Q' : TT_Q\mathcal{M} \to T\mathcal{M}$. If we fix $V \in T_Q\mathcal{M}$ and the corresponding fiber of $TT_Q\mathcal{M}$ we have that

$$R_Q'\big|_V = \Lambda_Q'\big|_{\psi(a_Q(V))} \circ \psi'\big|_{a_Q(V)} \circ a_Q'\big|_V .$$

Since a_Q is a linear map $a_Q'\Big|_V = a_Q$, and by using the right trivialised tangent map of ψ at $a_Q(V) \in \mathfrak{g}$ we get that for any $\tilde{V} \in T_Q\mathcal{M}$

$$\mathcal{R}_Q'\big|_V(\tilde{V}) = \mathrm{d}\psi_{a_Q(V)}\left(a_Q(\tilde{V})\right) \cdot \psi(a_Q(V)) \cdot Q = \mathrm{d}\psi_{a_Q(V)}\left(a_Q(\tilde{V})\right) \cdot P. \tag{18}$$

To find the inverse of $\mathcal{R}_Q'\Big|_V$, let us now assume that we are given $W \in T_P\mathcal{M}$ such that

$$\mathcal{R}_Q'\big|_V(\tilde{V}) = W \in T_P\mathcal{M}, \ \ P = \mathcal{R}_Q(V),$$

and we want to find $\tilde{V}$.

By (18) we have that $\tilde{V} = \left(\mathcal{R}_Q'\Big|_V\right)^{-1}(W)$ is the solution of the following equation

$$\mathrm{d}\psi_{a_Q(V)}\left(a_Q(\tilde{V})\right) \cdot P = W. \tag{19}$$

We will show in the next section a cheap way for solving (19) for $\tilde{V}$ in the case $\psi = \mathrm{cay}$.

3.2.1 Cayley methods

Assume now that $\mathrm{d}\psi_a = \mathrm{dcay}_a$, by (15) we have

$$\mathrm{dcay}_a(b) = (I - \frac{1}{2}a)^{-1} b (I + \frac{1}{2}a)^{-1},$$

and by substituting this expression into (19) we get

$$(I - \frac{1}{2}a_Q(V))^{-1} a_Q(\tilde{V})(I + \frac{1}{2}a_Q(V))^{-1}P - W = 0,$$

from which we get that

$$a_Q(\tilde{V})(I + \frac{1}{2}a_Q(V))^{-1}P - (I - a_Q(V))W = 0.$$

By direct calculation after some simple algebra one can verify that

$$P + Q = (I + \frac{1}{2}a_Q(V))^{-1}P,$$

and substituting in the previous equation we get

$$a_Q(\tilde{V})(P + Q) - (I - a_Q(V))W = 0.$$

By substituting the explicit expression for $a_Q(\tilde{V}) = g_Q(\tilde{V})Q^{\mathrm{T}} - Qg_Q(\tilde{V})^{\mathrm{T}}$ we have

$$g_Q(\tilde{V})Q^{\mathrm{T}}(P+Q) - Qg_Q(\tilde{V})^{\mathrm{T}}(P+Q) = (I - a_Q(V))W. \tag{20}$$

We define now $Z := g_Q(\tilde{V})^{\mathrm{T}}(P+Q)(Q^{\mathrm{T}}P+I)^{-1}$ and $G := (I - a_Q(V))W(Q^{\mathrm{T}}P+I)^{-1}$ and substituting in (20) we obtain

$$g_Q(\tilde{V}) \quad = \quad G + QZ. \tag{21}$$

In (21) Q is known and G can be computed in $\mathcal{O}(np^2)$ flops by means of known quatities: W, V, Q, P, while both $g_Q(\tilde{V})$ and Z are unknown, the last one containing implicitly $g_Q(\tilde{V})$. We will show how Z can be computed without knowing explicitely $g_Q(\tilde{V})$.

Infact by multiplying from the left by Q^{T} we get

$$Q^{\mathrm{T}}g_Q(\tilde{V}) \quad = \quad Q^{\mathrm{T}}G + Z,$$

whose left hand side is equal to $\frac{Q^{\mathrm{T}}\tilde{V}}{2}$ and therefore skew-symmetric. From this follows that the symmetric part of Z is

$$\tfrac{Z+Z^{\mathrm{T}}}{2} \quad = \quad \tfrac{Q^{\mathrm{T}}G+G^{\mathrm{T}}Q}{2}. \tag{22}$$

On the other hand by multiplying (21) by $(P+Q)^{\mathrm{T}}$ from the left we have

$$(P+Q)^{\mathrm{T}}g_Q(\tilde{V}) \quad = \quad (P+Q)^{\mathrm{T}}G + (P^{\mathrm{T}}Q+I)Z.$$

and noting that by the definition of Z we have

$$(P+Q)^{\mathrm{T}}g_Q(\tilde{V}) \quad = \quad (P^{\mathrm{T}}Q+I)Z^{\mathrm{T}}$$

we obtain the following expression for the skew-symmetric part of Z

$$\tfrac{Z-Z^{\mathrm{T}}}{2} \quad = \quad -\big(P^{\mathrm{T}}Q+I\big)^{-1}\tfrac{(P+Q)^{\mathrm{T}}G}{2}. \tag{23}$$

Then both the symmetric and skew-symmetric part of Z can be computed by means of known quatities and independently on $g_Q(\tilde{V})$, which means that by computing first the two expressions (22) and (23) we can obtain $g_Q(\tilde{V})$ by substitution into (21). When $g_Q(\tilde{V})$ is known is then easy to compute $\tilde{V}$ by

$$\tilde{V} = g_Q(\tilde{V}) - QQ^{\mathrm{T}}g_Q(\tilde{V}).$$

We summarise the all procedure for computing $\left(R'_Q\big|_V \right)^{-1}(W)$ in the case of the Cayley retraction in the following algorithm.

Algorithm 3.2 $\left(R'_Q\big|_V \right)^{-1}(W)$**-Cayley**

Input $Q,\ V,\ P,\ W.$

$g_Q(V) := V - Q\frac{Q^{\mathrm{T}}V}{2}$

$\tilde{P} := P + Q$

$M := Q^{\mathrm{T}}P + I$

$G := (W - g_Q(V)Q^{\mathrm{T}}W + Qg_Q(V)^{\mathrm{T}}W)M^{-1}$

$Z := \frac{Q^{\mathrm{T}}G+G^{\mathrm{T}}Q}{2} - M^{-\mathrm{T}}\frac{\tilde{P}^{\mathrm{T}}G}{2}$

$g_Q(\tilde{V}) := G + QZ$

$\tilde{V} := g_Q(\tilde{V}) - Qg_Q(\tilde{V})^{\mathrm{T}}Q$

Output $\tilde{V}$.

If we neglect lower order terms this algorithm has a cost of $24np^2$ flops, this cost can be further reduced to $20np^2$ flops if we substitute V and W in input with $g_Q(V)$ and $g_P(W)$ and in output $\tilde{V}$ with $g_Q(\tilde{V})$.

4 Lie group methods along geodesics

A geodesic connecting two points on a Riemannian manifold is the *path of shortest length with respect to a fixed Riemannian metric.* In the rest of this section we adopt the description and results of [10]. One can observe that any $V \in \mathrm{T}_Q\mathbb{V}_p$ can be written in the form

$$V = QA + Q_\perp B$$

where $Q_\perp$ is an $n \times (n-p)$ matrix with orthonormal columns and such that $(Q, Q_\perp)$ is an orthogonal $n \times n$ matrix, A is $p \times p$ and skew-symmetric and B is an $(n-p) \times p$ matrix. Let us consider the following metric on the Stiefel manifold, (canonical metric),

$$d_c(V, V) := \frac{1}{2}\mathrm{tr}A^{\mathrm{T}}A + \mathrm{tr}B^{\mathrm{T}}B.$$

Assume that $\gamma(t)$ is the geodesic curve exiting from the point Q with direction V with respect to this metric, then it is possible to show that

$$\gamma(t) = (Q, Q_\perp)\exp\big(t\begin{bmatrix} A & -B^{\mathrm{T}} \\ B & O \end{bmatrix}\big)I_{n,p},$$

here we denote by $I_{n,p}$ the first p columns of the $n \times n$ identity matrix. In [10] the authors propose two algorithms for computing geodesics on the Stiefel manifold with respect to the canonical metric, in $\mathcal{O}(np^2)$ flops.

We observe here that the retraction

$$\mathcal{R}_Q(V) := \exp(a_Q(V)) \cdot Q$$

is the geodesic $\gamma(t)$ at $t = 1$, infact

$$a_Q(V) = (Q, Q_\perp)\begin{bmatrix} A & -B^{\mathrm{T}} \\ B & O \end{bmatrix}(Q, Q_\perp)^{\mathrm{T}}.$$

Therefore, for this choice of retraction, the integration method discussed in the previous section produces approximations of the solution by composition of geodesic flows.

Earlier work on integrators on the Stiefel manifold based on the composition of geodesic flows can be found in [8]. The authors obtained methods of order up to two by computing correction terms using parallel transport.

5 Crouch–Grossman methods

The Crouch-Grossman (C-G) methods are a generalisation of classical Runge-Kutta methods for the integration of ODEs on manifolds, [7], [15]. In a C-G method with s stages and Runge-Kutta coefficients $(a_{i,j})_{i,j=1,s}$ and $(b_i)_{i=1,s}$, the approximate solution and the intermediate stages are computed by composing flows of elementary vector fields weighted with the corresponding coefficients of the Runge-Kutta method.

The methods rely on the possibility of representing the given vector field in terms of frame vector fields everywhere on the manifold and the ability of computing flows of elementary vector fields.

Vector fields $E_1, \dots, E_d$ for the manifold $\mathcal{M}$ such that $E_1(X), \dots, E_d(X)$ generate the tangent space $\mathrm{T}_X\mathcal{M}$ are a set of frame vector fields. It is then possible to express the vector field $F(Y)$ defining a differential equation that we want to integrate by means of the frame vector fields as follows:

$$F(Y) = \sum_{i=1}^{d} f_i(Y) E_i,$$

with $f_i : \mathcal{M} \to \mathbb{R}$.

The Crouch and Grossman method with appropriate Runge-Kutta coefficients can then be expressed as follows:

Algorithm 5.1 C-G

Input *y_0, stepsize h, and RK parameters a_{ij}, b_i.*

for $k = 0, 1, \dots$
  $P = Y_k$
  for $r = 1, \dots, s$
    $X_r := \exp(h a_{rs} F_P^s) \cdots \exp(h a_{r1} F_P^1) P \qquad r = 1, \dots s$
    $F_P^r := \sum_{i=1}^d f_i(X_r) E_i$
  end for

  $Y_{k+1} = \exp(h b_s F_P^s) \cdots \exp(h b_1 F_P^1) P$
  [Update stepsize h if desired]

end for

Note that in the algorithm above the exponentials indicate flows of vector fields.

In the case of the Stiefel manifold the vector fields F_P^r can be represented as matrices of the type

$$F_P^r := g_{X_r}(F_P^r) X_r^{\mathrm{T}} - X_r g_{X_r}(F_P^r)^{\mathrm{T}} \qquad r = 1, \dots s,$$

and then the corresponding elementary flows are simply matrix exponentials that can be cheaply computed using (12). As a consequence the C-G algorithm implies $\mathcal{O}(np^2)$ per time step.

6 Numerical experiments

With the following numerical experiments we will try to illustrate the computational cost implied by the Lie group methods considered in the previous sections and we will also consider an application to multivariate data analysis. We here indicate the Lie group methods and two other comparison methods with the following abbreviations:

- Runge-Kutta Munthe-Kaas based on the exponential: RKMK-E
- Runge-Kutta Munthe-Kaas based on the Cayley map: RKMK-C
- Runge-Kutta method based on the Cayley retraction: RKR-C
- Runge-Kutta method of Crouch and Grossman type: RKCG
- Runge-Kutta method combined with QR projection: RKDQR
- Runge-Kutta splitting method with stabilised projection [1]: RKSPL

All the methods have order 4. The first three methods are based on the explicit classical Runge-Kutta method of order 4. The RKCG is using two exponentials per stage. All numerical experiments have been performed in Matlab.

6.1 First experiment

In the first experiment we consider the following ODE on the Stiefel manifold as a test case:

$$Y' = S(Y) \cdot Y, \qquad Y_0 \in \mathcal{M},$$

where Y is an $n \times p$ matrix, $S(Y)$ is the banded skew-symmetric matrix with p super and subdiagonals whose nonzero elements are

$$S_{i,i+k} = Y(i,k), \qquad i = 1, \ldots, n-k, \qquad k = 1, \ldots, p.$$

The integration has been carried out on the time interval $[0,1]$, with different values of the step size $h = 1/(2^j)$ and $j = 1, \ldots, 4$. We have taken $n = 1000$ and $p = 4$.

In the following table we report the number of flops per step for the methods. In the first column the number of flops is divided by np^2 and in the second by np^2. For this particular choice of the vector field each function evaluation requires about $4np^2$ flops. In practical applications it is however quite usual that the cost for a function evaluation is $\mathcal{O}(n^2p)$.

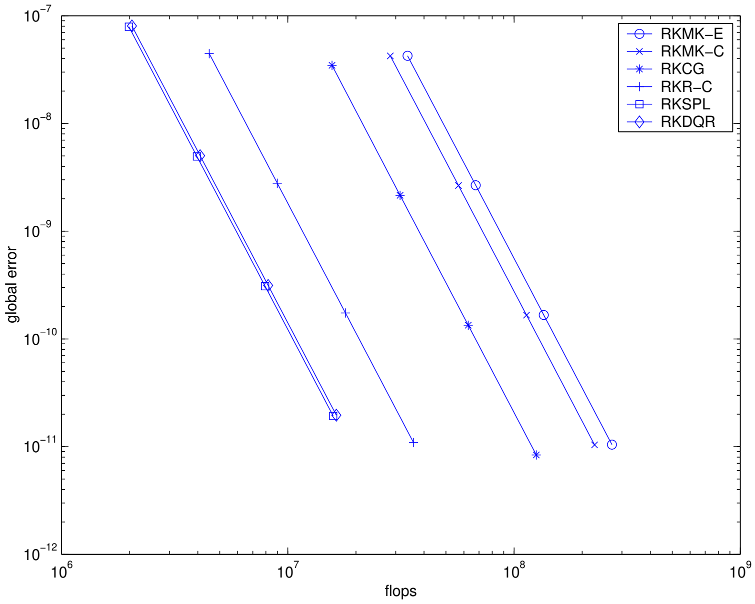

Figure 1: Global error versus the number of flops.

Complexity constants		
methods	flops/n^2p	flops/np^2
RKMK-E	4.2283	1.0571e+03
RKMK-C	3.5475	8.8687e+02
RKR-C	0.5613	1.4033e+02
RKCG	1.9599	4.89972e+02
RKSPL	0.2479	6.1966e+01
RKDQR	0.2558	6.3957e+01

In figure (1) we plotted the number of flops (x-axis) against the norm of the global error (y-axis) for the various methods.

In general the projection methods (RKDQR and RSPL) require less flops per step compared to the Lie group methods. Among the Lie group methods considered in this paper the RKR-C implies the least amount of flops per step, with a cost that is about double the cost of the projection methods. The computational complexity of this method is given by about $10np^2$ for computing the retraction map and about $16 - 20np^2$ for the evaluation of the inverse tangent mapping of the retraction. These complexity counts are comparable to those obtained for the methods in [3] and [12]. The implementations of the Lie group methods proposed in this paper are based on matrix matrix operations that never require more than $\mathcal{O}(np^2)$ flops. A part form the RKR-C these implementations present however in our experiment a complexity that is rather of the type $\mathcal{O}(n^2p)$ flops per time step. While we do not exclude that further optimisation could lower the cost for these methods, we want to emphasise that better qualitative performance can motivate the use of such methods in some cases. We will see in fact in the next experiment how the Lie group methods outperform projection methods in certain applications.

6.2 Second experiment

The second experiment we considered the Penrose regression problem (including the orthonormal Procrustes problem). This problem arises in multivariate data analysis [11], [17] and it can be expressed as follows, given $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{p\times q}$, $B \in \mathbb{R}^{m\times q}$ and $\mathcal{M}$ the Stiefel manifold of matrices with n rows and p columns, find

$$\min_{Y\in\mathcal{M}} \|AYC - B\|.$$

When $q = p$ and C is the identity matrix the above problem reduces to the orthonormal Procrustes problem.

To solve the given problem we here consider the projected gradient method of [6] and [5]. The idea behind this method is to use the ordinary differential equation on the Stiefel manifold

$$Y' = -g(Y) \tag{24}$$

which describes the gradient flow for the Penrose regression problem, and to find the minimiser Y^* following the solution of this equation. The expression for g is

$$g(Y) = \frac{Y}{2}\left(Y^{\mathrm{T}}A^{\mathrm{T}}(AYC - B)C^{\mathrm{T}} - C(AYC - B)^{\mathrm{T}}AY\right) + (I - YY^{\mathrm{T}})A^{\mathrm{T}}(AYC - B)C^{\mathrm{T}}.$$

We integrated equation (24) on the interval $[0, 60]$ with stepsize fixed and equal to $h = 1/2^4$, in the figures we plot the results at time intervals of $T = 5$.

As criteria of convergence to the minimiser we use the *residual* $\|AYC-B\|$ and the norm of the difference of two successive approximations of the minimiser, i.e. $\|Y((k-1)T) - Y(kT)\|$, and $k = 1, \ldots, 12$.

We consider two examples the first is a orthonormal Procrustes problem, and the second is a Penrose regression problem.

6.2.1 Example 1

In the first example we took $C = I$, $n = 4$ and $p = 2$. The matrix $A = \mathrm{diag}(1, 0.7, 0.8, 0.5)$ the minimiser is generated in Matlab as the Q factor of the QR factorisation of a random 4×2 matrix, and we denote it here with Y^*. The matrix B is then obtained by multiplication, $B = AY^*$. The initial value for the integration procedure is

$$Y_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

In figure (2) we plotted time (x-axis) against the norm of the residual. All the integration methods perform similarly and the different curves are not distinguishable.

6.2.2 Example 2

In the next example we consider

$$C = \begin{bmatrix} -7.5 & -2.5 \\ -2.5 & -7.5 \end{bmatrix}.$$

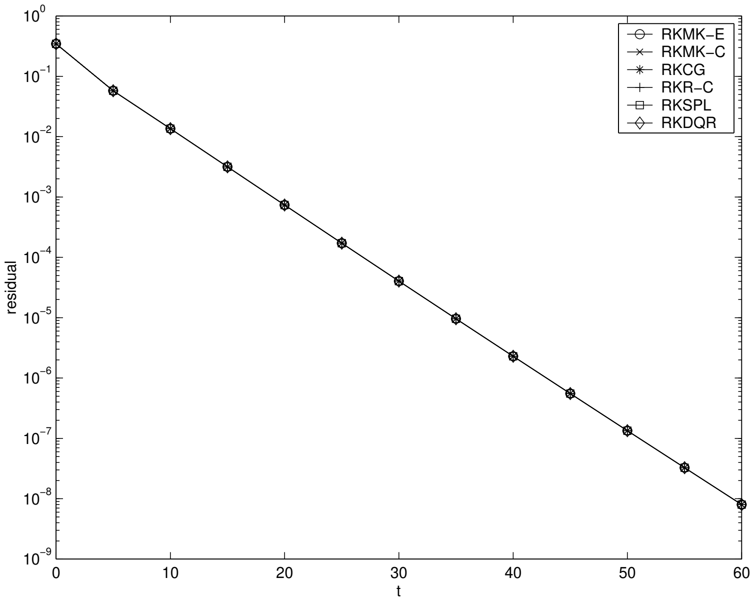

Figure 2: Example 1, convergence to the minimiser: residual

We took A as above and B a random matrix generated in Matlab,

$$B = \begin{bmatrix} 0.21299365889249 & 0.34020344729273 \\ 0.03559996705196 & 0.46615497822012 \\ 0.08116370409078 & 0.91376323973698 \\ 0.85056678324539 & 0.22857670585214 \end{bmatrix}.$$

Note that in this case the norm of the residual of the minimiser is not equal to zero.

In figure (3) we reported the norm of the residual and in figure (4) we reported the norm of the difference of two successive approximations of the minimiser.

The method RKSPL gives no sensible results which are then not reported here, perhaps a reformulation of the expression for g in equation (24) could lead to better results for this method. We observe that the projection method RKDQR fails to find a local minimiser in this time interval, while the Lie group methods have converged to an answer already in an early stage of the integration. The values of the residual are similar to those of figure (4).

The results of the experiments of this section indicate that the Lie group methods present better qualitative behaviour compared to integrators based on the use of projection. The last ones are however in general cheaper to implement.

In this paper we have shown how Lie group methods as those proposed by Munthe-Kaas [14] and Crouch-Grossman [7] can be implemented in $\mathcal{O}(np^2)$ complexity per time step for the Stiefel manifold $\mathbb{V}_p(\mathbb{R}^n)$. The main tool we have used to achieve these results are retraction maps. We also proposed a new method based on the retraction algorithm presented earlier in [3]. Methods of this complexity order are not new, the originality of this work lies in the implementation of Lie group methods for problems of Stiefel manifolds with such complexity. Recently, a similar, but different approach has been proposed by Krogstad [12]. The results of the conducted experiments are promising, and we believe that these methods offer a worthwhile alternative to other approaches for solving ODEs on the Stiefel manifold.

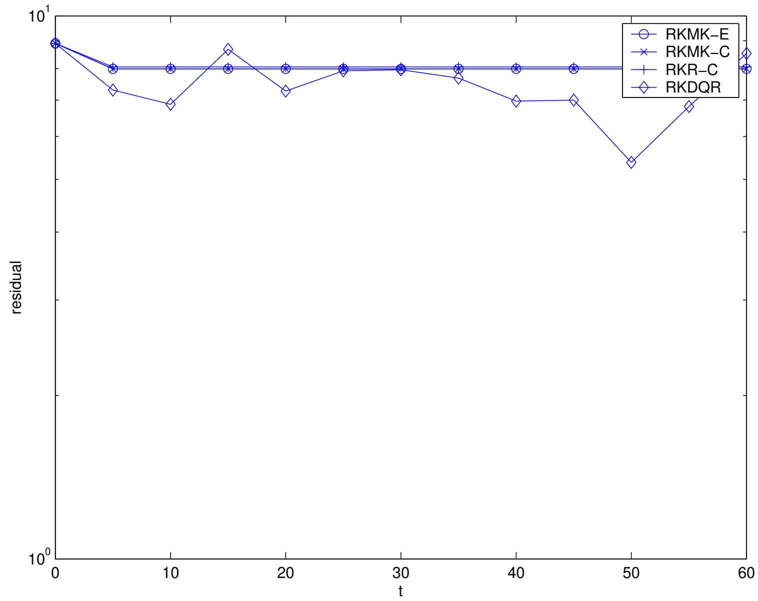

Figure 3: Example 2, convergence to the minimiser: residual

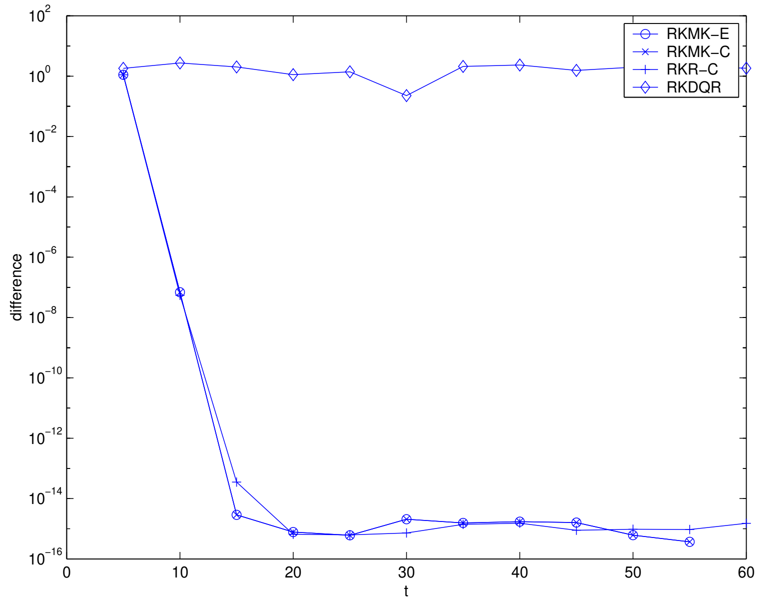

Figure 4: Example 2, convergence to the minimiser: difference

References

[1] T.J. Bridges and S. Reich. Computing Lyapunov exponents on a Stiefel manifold. Technical report, University of Surrey UK, April 2000.

[2] E. Celledoni and A. Iserles. Approximating the exponential from a Lie algebra to a Lie group. *Math. Comp.*, 2000. Posted on March 15, PII S 0025-5718(00)01223-0 (to appear in print).

[3] E. Celledoni and B. Owren. A class of low complexity intrinsic schemes for orthogonal integration. Technical Report Numerics No. 1/2001, The Norwegian University of Science and Technology, Trondheim, Norway, 2001.

[4] M. T Chu and N. Trendafilov. On a differential equation approach to the weighted orthogonal Procrustes problem. *Statistics and Computing*, 8:125–133, 1998.

[5] M. T. Chu and N. T. Trendafilov. The orthogonally constrained regression revisited. Manuscript, 1997.

[6] M.T. Chu and K.R. Driessel. The projected gradient method for least squares matrix approximations with spectral constraints. 27:1050–1060, 1990.

[7] P. E. Crouch and R. Grossman. Numerical integration of ordinary differential equations on manifolds. *J. Nonlinear Sci.*, 3:1–33, 1993.

[8] N. Del Buono and L. Lopez. Runge-Kutta type methods on geodesics for systems of ODEs on the Stiefel manifold. Technical report, Dipartimento Iteruniversitario di Matematica, Univarsita' degli Studi di Bari, Via E. Orabona 4, I-70125 Bari, 2000.

[9] L. Dieci and E. S. Van Vleck. Computation of a few Lyapunov exponents for continuous and discrete dynamical systems. *Applied Numerical Mathematics*, 17:275–291, 1995.

[10] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Applic.*, 20(2):303–353, 1999.

[11] J.C. Gower. Ordination, multidimensional scaling and allied topics. In Emlyn Lloyd, editor, *Handbook of Applicable Mathematics*, volume VI. John Wiley & Son, 1984.

[12] S. Krogstad. A low complexity Lie group method on the Stiefel manifold, 2001.

[13] H. Munthe-Kaas. Lie–Butcher theory for Runge–Kutta methods. *BIT*, 35(4):572–587, 1995.

[14] H. Munthe-Kaas. High order Runge–Kutta methods on manifolds. *Appl. Numer. Math.*, 29:115–127, 1999.

[15] B. Owren and A. Marthinsen. Integration methods based on canonical coordinates of the second kind. *Numer. Math.*, 87:763–790, 2001.

[16] M. Shub. Some remarks on dynamical systems and numerical analysis. In L. Lara-Carrero and J. Lewowicz, editors, *Dynamical systems and partial differential equations*, volume VII ELAM, pages 62–92, 1986.

[17] J.M.F. ten Berge and D.L. Knol. Orthogonal rotations to maximal agreement for two or more matrices of different column orders. *Psychometrika*, 49:49–55, 1984.