Exponential Rosenbrock Methods and Their Application in Visual Computing

Vu Thai Luan and Dominik L. Michels

1 Introduction

Developing numerical models for practical simulations in science and engineering usually results in problems regarding the presence of wide-range time scales. These problems involve both slow and fast components leading to rapid variations in the solution. This gives rise to the so-called *stiffness phenomena*. Typical examples are models in molecular dynamics (see e.g. [36]), chemical kinetics, combustion, mechanical vibrations (mass-spring-damper models), visual computing (specially in computer animation), computational fluid dynamics, meteorology, etc., just to name a few. They are usually formulated as systems of stiff differential equations which can be cast in the general form

$$u'(t) = F(u(t)), \quad u(t_0) = u_0, \tag{1}$$

where $u \in \mathbb{R}^n$ is the state vector and $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ represents the vector field. The challenges in solving this system are due to its stiffness by means of the eigenvalues of the Jacobian matrix of F differing by several orders of magnitude. In the early days of developing numerical methods for ordinary differential equations (ODEs), classical methods such as the explicit Runge–Kutta integrators were proposed. For stiff problems, however, they are usually limited by stability issues due to the CFL condition leading to the use of unreasonable time steps, particularly for large-scale

V. T. Luan (✉)
Department of Mathematics and Statistics, Mississippi State University, Starkville, MS, USA
e-mail: luan@math.msstate.edu

D. L. Michels
Computational Sciences Group, Visual Computing Center, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
e-mail: dominik.michels@kaust.edu.sa

T. Jax et al. (eds.), *Rosenbrock—Wanner-Type Methods*, Mathematics Online First Collections, https://doi.org/10.1007/978-3-030-76810-2_3

applications. The introduction of implicit methods such as semi-implicit, IMEX (see [2]), and BDF methods (see [10, 14]) has changed the situation. Theses standard methods require the solution of nonlinear systems of equations in each step. As the stiffness of the problem increases, considerably computational effort is observed. This can be seen as a shortcoming of the implicit schemes.

In the last 20 years, with the new developments of numerical linear algebra algorithms in computing matrix functions [1, 25, 41], exponential integrators have become an alternative approach for stiff problems (see the survey [24]; next to physics simulations, exponential integrators are nowadays also employed for different applications as for the construction of hybrid Monte Carlo algorithms, see [7]). For the fully nonlinear stiff system (1), we mention good candidates, the so-called *explicit exponential Rosenbrock methods*, which can handle the stiffness of the system in an explicit and very accurate way. This class of exponential integrators was originally proposed in [23] and further developed in [26, 30, 32, 34]. They have shown to be very efficient both in terms of accuracy and computational savings. In particular, the lower-order schemes were recently successfully applied to a number of different applications [8, 15, 17, 46, 49] and very recently the fourth- and fifth-order schemes were shown to be the method of choice for some meteorological models (see [35]).

In this work, we show how the exponential Rosenbrock methods (particularly higher-order schemes) can be also applied efficiently in order to solve problems in computational modeling of elastodynamic systems of coupled oscillators (particle systems) which are often used in visual computing (e.g. for computer animation). In their simplest formulation, their dynamics can be described using Newton's second law of motion leading to a system of second-order ODEs of the form

$$m_i\ddot{x}_i + \sum_{j\in\mathcal{N}(i)} k_{ij}(\|x_i - x_j\| - \ell_{ij})\frac{x_i - x_j}{\|x_i - x_j\|} = g_i(x_i, \dot{x}_i, \cdot), \quad i = 1, 2, \cdots, N, \tag{2}$$

where N is the number of particles, $x_i \in \mathbb{R}^3$, m_i, k_{ij}, ℓ_{ij} denote the position of particle i from the initial position, its mass, the spring stiffness, the equilibrium length of the spring between particles i and j, respectively, and $\mathcal{N}(i)$ denotes the set of indices of particles that are connected to particle i with a spring (the neighborhood of particle i). Finally, g_i represents the external force acting on particle i which can result from an external potential, collisions, etc., and can be dependent of all particle positions, velocities, or external forces set by user interaction.

Our approach for integrating (2) is first to reformulate it in the form of (1) (following a novel approach in [40]). The reformulated system is a very stiff one since the linear spring forces usually possess very high frequencies. Due to the special structure of its linear part (skew-symmetric matrix) and large nonlinearities, we then make use of exponential Rosenbrock methods. Moreover, we propose to use the improved algorithm in [35] for the evaluation of a linear combination of φ-functions acting on certain vectors $v_0, \dots, v_p$, i.e. $\sum_{k=0}^{p} \varphi_k(A)v_k$ which is crucial for implementing exponential schemes. Our numerical results on a number of complex models in visual computing indicate that this approach significantly reduces computational time over the current state-of-the-art techniques while maintaining sufficient levels of accuracy.

This chapter is organized as follows. In Sect. 2, we present a reformulation of systems of coupled oscillators (2) in the form of (1) and briefly review previous approaches used for simulating these systems in visual computing. In Sect. 3, we describe the exponential Rosenbrock methods as an alternative approach for solving large stiff systems (1). The implementation of these methods is discussed in Sect. 4, where we also introduce a new procedure to further improve one of the state-of-the-art algorithms. In Sect. 5 we demonstrate the efficiency of the exponential Rosenbrock methods on a number of complex models in visual computing. In particular, we address the simulation of deformable bodies, fibers including elastic collisions, and crash scenarios including nonelastic deformations. These examples focus on relevant aspects in the realm of visual computing, like stability and energy conservation, large stiffness values, and high fidelity and visual accuracy. We include an evaluation against classical and state-of-the-art methods used in this field. Finally, some concluding remarks are given in Sect. 6.

2 Reformulation of Systems of Coupled Oscillators

We first consider the system of coupled oscillators (2). Let $x(t) \in \mathbb{R}^{3N}$, $M \in \mathbb{R}^{3N\times 3N}$, $D \in \mathbb{R}^{3N\times 3N}$, $K \in \mathbb{R}^{3N\times 3N}$ and $g(x) \in \mathbb{R}^{3N}$ denote the vector of positions, the mass matrix (often diagonal and thus nonsingular), the damping matrix, the spring matrix (stiff), and the total external forces acting on the system, respectively. Using these matrix notations and denoting $A = M^{-1}K$, (2) can be written as a system of second-order ODEs

$$x''(t) + Ax(t) = g(x(t)), \quad x(t_0) = x_0,\ x'(t_0) = v_0. \tag{3}$$

Here x_0, v_0 are some given initial positions and velocities. For simplicity we neglect damping and assume that A is a symmetric, positive definite matrix (this is a reasonable assumption in many models, see [38]). Therefore, there exists a unique positive definite matrix Ω such that $A = \Omega^2$ (and clearly Ω^{-1} exists).

Following our approach in [40], we introduce the new variable

$$u(t) = \begin{bmatrix} \Omega x(t) \\ x'(t) \end{bmatrix}. \tag{4}$$

Using this one can reformulate (3) as a first-order system of ODEs of the form like (1):

$$u'(t) = F(u(t)) = \mathscr{A}u(t) + G(u(t)), \quad u(t_0) = u_0, \tag{5}$$

where

$$\mathscr{A} = \begin{bmatrix} \mathbf{0} & \Omega \\ -\Omega & \mathbf{0} \end{bmatrix}, \quad G(u) = \begin{bmatrix} \mathbf{0} \\ g(x) \end{bmatrix}. \tag{6}$$

Since the linear spring forces usually possess high frequencies (thus $\|K\| \gg 1$ and so is $\|A\|$), (5) becomes a very stiff ODE. Regarding the new formulation (5)–(6), we observe the following two remarks.

Remark 1 Clearly, the new linear part associated with $\mathscr{A}$, that is a skew-symmetric matrix. We note that this significantly differs from the common way of reformulating (3) that is to use the change of variable $X(t) = [x(t),\ x'(t)]^T$ which results in a non-symmetric matrix. The advantage here is that since $\mathscr{A}$ is a skew-symmetric matrix, its nonzero eigenvalues are all pure imaginary and are in pairs $\pm\lambda_k i$. We also observe that $\mathscr{A}$ is an infinitesimal symplectic (or Hamiltonian). This is because, by definition of an infinitesimal symplectic matrix, we check whether $W\mathscr{A} + \mathscr{A}^T W = \mathbf{O}$, where W is the anti-symmetric matrix $W = \begin{bmatrix} \mathbf{0} & I \\ -I & \mathbf{0} \end{bmatrix}$. This can be easily verified since

$$W\mathscr{A} = \begin{bmatrix} -\Omega & \mathbf{0} \\ \mathbf{0} & -\Omega \end{bmatrix},$$

which is clearly a symmetric matrix, i.e., $W\mathscr{A} = (W\mathscr{A})^T$.

Remark 2 If the Jacobian matrix $F'(u) = \mathscr{A} + G'(u)$ is infinitesimal symplectic, (5) is a Hamiltonian system. This can be fulfilled since a typical situation in Hamiltonian systems is that $g(x) = \nabla f(x)$ for some function $f(x)$ and thus $g'(x) = \nabla^2 f(x)$ becomes a Hessian matrix, which is symmetric.

As seen, either using the common way (mentioned in Remark 1) or the new way (4) for reformulating (3), one has to solve the stiff ODE (5). In visual computing it is usually solved by explicit methods such as the fourth-order Runge–Kutta methods, semi-implicit methods such as the Störmer–Verlet methods, the backward differentiation formulas (BDF-1 and BDF-2) methods, or IMEX methods. In this regard, we refer to some contributions in the context of interacting deformable bodies, cloth, solids, and elastic rods, see [3, 4, 12, 16, 19, 47]. For large-scale applications associated with stiff systems, however, both types of these time integration techniques have their own limitations as mentioned in the introduction. In recent years, exponential integrators have shown to be competitive for large-scale problems in physics and for nonlinear parabolic PDEs, as well as for highly oscillatory problems (see [24]). They have attracted much attention by the broad computational mathematics community since mid-1990s. At the time while solving linear systems $(I - \alpha h J)x = v$ with some Jacobian matrix J (required when using implicit methods) is generally only of linear convergence, it was realized that Krylov subspace methods for approximating the action of a matrix exponential on a vector, $\mathrm{e}^{hJ}v$, offer superlinear convergence (see [21]). Unless a good preconditioner is available, this is clearly a computational advantage of exponential integrators over implicit methods. This has been addressed in the visual computing community very recently through a number of interesting work on exponential integrators, see e.g.[37–40]. Inspired by this interest, in the following sections we will show how exponential Rosenbrock methods—one of the popular classes of exponential integrators—can be applied for simulating systems of coupled oscillators.

3 Explicit Exponential Rosenbrock Methods

In this section, based on [23, 26, 29, 32, 34] we present a compact summary of the introduction of exponential Rosenbrock methods and their derivations for methods of order up to 5. We then display some efficient schemes for our numerical experiments for some applications in visual computing.

3.1 Approach

Motivated by the idea of deriving Rosenbrock-type methods, see [18, Chap. IV.7], instead of integrating the fully nonlinear system (1) (which has a large nonlinearity for stiff problems), one can replace it by a sequence of semilinear problems

$$u'(t) = F(u(t)) = J_n u(t) + g_n(u(t)), \tag{7}$$

by linearizing the forcing term $F(u)$ in each time step at the numerical solution u_n (due to [42]) with

$$J_n = F'(u_n), \ g_n(u) = F(u) - J_n u \tag{8}$$

are the Jacobian and the nonlinear remainder, respectively. An advantage of this approach is that $g_n'(u_n) = F'(u_n) - J_n = 0$ which shows that the new nonlinearity $g_n(u)$ has a much smaller Lipschitz constant than that of the original one $F(u)$. The next idea is to handle the stiffness by solving the linear part $J_n u$ exactly and integrating the new nonlinearity $g_n(u)$ explicitly. For that, the representation of the exact solution at time $t_{n+1} = t_n + h$ of (7) using the variation-of-constants formula

$$u(t_{n+1}) = \mathrm{e}^{hJ_n}u(t_n) + \int_0^h \mathrm{e}^{(h-\tau)J_n} g_n(u(t_n + \tau))\mathrm{d}\tau \tag{9}$$

plays a crucial role in constructing this type of integrators. As seen from (9), while the linear part can be integrated exactly by computing the action of the matrix exponential e^{hJ_n} on the vector $u(t_n)$, the integral involving $g_n(u)$ can be approximated by some quadrature. This procedure results in the so-called *exponential Rosenbrock methods*, see [23, 26].

Remark 3 For the system of coupled oscillators (2), the forcing term $F(u)$ has the semilinear form (5), which can be considered as a fixed linearization. Therefore, one can directly apply explicit the exponential Runge–Kutta methods (see [22]) to (5). The advantage of these methods is that the time-step h is not restricted by the CFL condition when integrating the linear part $\mathscr{A}u$. In our applications, however, the nonlinearity $G(u)$ is large in which the CFL condition usually serves as a reference for setting the time-step. In particular, for the stability hL_G should be sufficiently small (L_G is the Lipschitz constant of $G(u)$). In this regard, the dynamic linearization approach (7) applied to (5)

$$u'(t) = F(u) = \mathscr{A}u + G(u) = J_n u + G_n(u) \tag{10}$$

with

$$J_n = \mathscr{A} + G'(u_n), \tag{11}$$

offers a great advantage in improving the stability (in each step) when integrating $G(u)$. This is because instead of integrating the original semilinear problem with large nonlinearity $G(u)$, we only have to deal with a much smaller nonlinearity $G_n(u)$ (as mentioned above). Note that the new linear part $J_n u$ with the Jacobian J_n now incorporates both $\mathscr{A}$ and the Jacobian of the nonlinearity $G(u)$, which can be again solved exactly. It is thus anticipated that this idea of exponential Rosenbrock methods opens up the possibility to take even larger time steps compared to exponential Runge–Kutta methods.

3.2 Formulation of a Second-Order and General Schemes

In this subsection, we will illustrate the approach of exponential Rosenbrock methods by presenting a simple derivation of a second-order scheme and formulating general schemes.

3.2.1 A Second-Order Scheme

First, expanding $u(t_n+\tau)$ in a Taylor series gives $u(t_n+\tau) = u(t_n)+\tau u'(t_n)+\mathcal{O}(\tau^2)$. Then inserting this into $g_n(u(t_n + \tau))$ and again expanding it as a Taylor series around $u(t_n)$ (using $g_n'(u(t_n)) = 0$) leads to

$$g_n(u(t_n + \tau)) = g_n(u(t_n)) + \mathcal{O}(\tau^2)\,. \tag{12}$$

Inserting (12) into the integral of (9) and denoting $\varphi_1(hJ_n) = \frac{1}{h}\int_0^h \mathrm{e}^{(h-\tau)J_n}\,\mathrm{d}\tau$ gives

$$u(t_{n+1}) = \mathrm{e}^{hJ_n}u(t_n) + h\varphi_1(hJ_n)g_n(u(t_n)) + \mathcal{O}(h^3). \tag{13}$$

Neglecting the local error term $\mathcal{O}(h^3)$ results in a second-order scheme, which can be reformulated as

$$u_{n+1} = u_n + h\varphi_1(hJ_n)F(u_n) \tag{14}$$

by replacing $g_n(u(t_n))$ by (8) and using the fact that $\varphi_1(z) = (e^z - 1)/z$. This scheme was derived before and named as *exponential Rosenbrock-Euler method*, see [23, 26] (since when considering the formal limit $J_n \to \mathbf{0}$, (14) is the underlying Euler method). The derivation here, however, shows directly that this scheme has an order of consistency three and thus it is a second-order stiffly accurate method (since the constant behind the Landau notation $\mathcal{O}$ only depends on the regularity assumptions on $u(t)$ and $g_n(u)$, but is independent of $\|J_n\|$).

3.2.2 General Schemes

For the derivation of higher-order schemes, one can proceed in a similar way as for the construction of classical Runge–Kutta methods. Namely, one can approximate the integral in (9) by using some higher-order quadrature rule with nodes c_i in [0, 1] and weights $b_i(hJ_n)$ which are matrix functions of hJ_n, yielding

$$u(t_{n+1}) \approx \mathrm{e}^{hJ_n}u(t_n) + h\sum_{i=1}^{s} b_i(hJ_n)g_n(u(t_n + c_ih)). \tag{15}$$

The unknown intermediate values $u(t_n + c_ih)$ can be again approximated by using (9) (with c_ih in place of h) with another quadrature rule using the same nodes c_j, $1 \le j \le i-1$, (to avoid generating new unknowns) and new weights $a_{ij}(hJ_n)$, leading to

$$u(t_n + c_ih) \approx \mathrm{e}^{c_ihJ_n}u(t_n) + h_n\sum_{j=1}^{i-1} a_{ij}(hJ_n)g_n(u(t_n + c_jh)). \tag{16}$$

Let us denote $u_n \approx u(t_n)$ and $U_{ni} \approx u(t_n + c_ih_n)$. As done for (14), using (12) (with c_ih, h in place of τ, respectively) one can reformulate (15) and (16) in a similar manner, which yields the general format of s-stage explicit exponential Rosenbrock methods

$$U_{ni} = u_n + c_i h\varphi_1(c_i hJ_n)F(u_n) + h\sum_{j=2}^{i-1} a_{ij}(hJ_n)D_{nj}, \tag{17a}$$

$$u_{n+1} = u_n + h\varphi_1(hJ_n)F(u_n) + h\sum_{i=2}^{s} b_i(hJ_n)D_{ni} \tag{17b}$$

with

$$D_{ni} = g_n(U_{ni}) - g_n(u_n), \tag{17c}$$

As in (12), we have $D_{ni} = \mathcal{O}(h^2)$. Thus, the general methods (17) are small perturbations of the exponential Rosenbrock–Euler method (14). Note that the weights $a_{ij}(hJ_n)$ and $b_i(hJ_n)$ are usually linear combinations of $\varphi_k(c_i hJ_n)$ and $\varphi_k(hJ_n)$, respectively, where the φ functions (similar to φ_1) are given by

$$\varphi_k(hZ) = \frac{1}{h^k}\int_0^h \mathrm{e}^{(h-\tau)Z}\frac{\tau^{k-1}}{(k-1)!}\mathrm{d}\tau, \quad k \geq 1 \tag{18}$$

and satisfy the recursion relation

$$\varphi_{k+1}(z) = \frac{\varphi_k(z) - \frac{1}{k!}}{z}, \quad k \geq 1. \tag{19}$$

It is important to note that these functions are bounded (uniformly) independently of $\|J_n\|$ (i.e. the stiffness) so are the coefficients $a_{ij}(hJ_n)$ and $b_i(hJ_n)$ (see e.g. [24]).

Clearly, using exponential Rosenbrock schemes (17) offers some good advantages. First, they do not require the solution of linear or nonlinear systems of equations. Second, as mentioned above, they offer a better stability when solving stiff problems with large nonlinearities and thus allow to use larger time-steps. Third, since the Jacobian of the new nonlinearity vanishes at every step ($g_n'(u_n) = 0$), the derivation of the order conditions and hence the schemes can be simplified considerably. In particular, higher-order stiffly accurate schemes can be derived with only a few stages (see the next section).

The convergence analysis of exponential Rosenbrock methods is usually carried out in an appropriate framework (strongly continuous semigroup) under regularity assumptions on the solution $u(t)$ (sufficiently smooth) and $g_n(u)$ (sufficiently Fréchet differentiable in a neighborhood of the solution) with uniformly bounded derivatives in some Banach space. For more details, we refer to [26, 32].

3.3 Selected Schemes for Numerical Simulations

First, we discuss some important points for the derivation of exponential Rosenbrock schemes. Clearly, the unknown coefficients $a_{ij}(hJ_n)$ and $b_i(hJ_n)$ have to be determined by solving order conditions. For nonstiff problems, where the Jacobian matrix has a small norm, one can expand those matrix functions using classical Taylor series expansions, leading to nonstiff order conditions and in turn classical exponential schemes (see e.g. [9, 27]). For stiff problems, however, one has to be cautious when analyzing the local error to make sure that error terms do not involve powers of J_n (which has a large norm). Recently, Luan and Ostermann [30, 33] derived a new expansion of the local error which fulfills this requirement and thus derived a new stiff order conditions theory for methods of arbitrary order (both for exponential Runge–Kutta and exponential Rosenbrock methods). As expected, with the same order, the number of order conditions for exponential Rosenbrock methods is significant less than those for exponential Runge–Kutta methods. For example, in Table 1, we display the required 4 conditions for deriving schemes up to order 5 in [32] (note that for exponential Runge–Kutta methods, 16 order conditions are required for deriving schemes of order 5, see [31]).

We note that with these order conditions one can easily derive numerous different schemes of order up to 5. Taking the compromise between efficiency and accuracy into consideration, we seek for the most efficient schemes for our applications. Namely, the following two representative fourth-order schemes are selected.

`exprb42` (a fourth-order 2-stage scheme which can be considered as a superconvergent scheme, see [29]):

$$U_{n2} = u_n + \tfrac{3}{4} h\varphi_1(\tfrac{3}{4} hJ_n)F(u_n), \tag{20a}$$

$$u_{n+1} = u_n + h\varphi_1(hJ_n)F(u_n) + h\tfrac{32}{9}\varphi_3(hJ_n)(g_n(U_{n2}) - g_n(u_n)). \tag{20b}$$

Table 1 Stiff order conditions for exponential Rosenbrock methods up to order five. Here Z and K denote arbitrary square matrices and $\psi_{3,i}(z) = \sum_{k=2}^{i-1} a_{ik}(z)\frac{c_k^2}{2!} - c_i^3\varphi_3(c_i z)$

No.	Order condition	Order
1	$\sum_{i=2}^{s} b_i(Z)c_i^2 = 2\varphi_3(Z)$	3
2	$\sum_{i=2}^{s} b_i(Z)c_i^3 = 6\varphi_4(Z)$	4
3	$\sum_{i=2}^{s} b_i(Z)c_i^4 = 24\varphi_5(Z)$	5
4	$\sum_{i=2}^{s} b_i(Z)c_i K\psi_{3,i}(Z) = 0$	5

pexprb43 (a fourth-order 3-stage scheme, which can be implemented in parallel, see [34]):

$$U_{n2} = u_n + \tfrac{1}{2} h\varphi_1(\tfrac{1}{2} hJ_n)F(u_n), \tag{21a}$$

$$U_{n3} = u_n + h\varphi_1(hJ_n)F(u_n), \tag{21b}$$

$$\begin{aligned} u_{n+1} = u_n &+ h\varphi_1(hJ_n)F(u_n) + h\varphi_3(hJ_n)(16D_{n2} - 2D_{n3}) \\ &+ h\varphi_4(hJ_n)(-48D_{n2} + 12D_{n3}). \end{aligned} \tag{21c}$$

Note that the vectors D_{n2} and D_{n3} in (21) are given by (17c), i.e., $D_{n2} = g_n(U_{n2}) - g_n(u_n)$ and $D_{n3} = g_n(U_{n3}) - g_n(u_n)$.

4 Implementation

In this section, we present the implementation of exponential Rosenbrock methods for the new formulation (5) of the system of coupled oscillators. First, we discuss on the computation of the matrix square root Ω needed for the reformulation. We then briefly review some state-of-the-art algorithms for implementing exponential Rosenbrock methods and introduce a new routine which is an improved version of one of these algorithms (proposed very recently in [35]) for achieving more efficiently. Finally, we specifically discuss applying this routine for implementing the selected schemes exprb42 and pexprb43.

4.1 *Computation of the Matrix Square Root* $\boldsymbol{\Omega = \sqrt{A}}$

For the computation of $\Omega = \sqrt{A}$ used in (5), we follow our approach in [40]. Specifically, we use the Schur decomposition for moderate systems. For large systems, the Newton square root iteration (see [20]) is employed in order to avoid an explicit precomputation of Ω. Namely, one can use the following simplified iteration method for approximating the solution of the equation $\Omega^2 = A$:

(i) choose $\Omega_0 = A$ $(k = 0)$,
(ii) update $\Omega_{k+1} = \frac{1}{2}(\Omega_k + \Omega_k^{-1} A)$.

This method offers unconditional quadratic convergence with much less cost compared to the Schur decomposition. We note that Ω^{-1} can be computed efficiently using a Cholesky decomposition since Ω is symmetric and positive definite and it is given by $\Omega^{-1} = \mathbf{S}^{-1}\mathbf{S}^{-\mathsf{T}}$, where $\mathbf{S}$ is an upper triangular matrix with real and positive diagonal entries. For more details, we refer to [20, 40].

With Ω at hand, one can easily compute the Jacobian J_n as in (11) and $F(u)$, $G_n(u)$ as in (10). As the next step, we discuss the implementation of the exponential Rosenbrock schemes.

4.2 Implementation of Exponential Rosenbrock Methods

In view of the exponential Rosenbrock schemes in Sect. 3, each stage requires the evaluation of a linear combination of φ-functions acting on certain vectors $v_0, \ldots, v_p$

$$\varphi_0(M)v_0 + \varphi_1(M)v_1 + \varphi_2(M)v_2 + \cdots + \varphi_p(M)v_p, \tag{22}$$

where the matrix M here could be hJ_n or $c_i hJ_n$. Starting from a seminal contribution by Hochbruck and Lubich [21] (which they analyzed Krylov subspace methods for efficiently computing the action of a matrix exponential (with a large norm) on some vector), many more efficient techniques have been proposed. A large portion of these developments is concerned with computing the expression (22). For example, we mention some of the state-of-the-art algorithms: `expmv` proposed by Al-Mohy and Higham in [1] (using a truncated standard Taylor series expansion), `phipm` proposed by Niessen and Wright in [41] (using adaptive Krylov subspace methods), and `expleja` proposed by Caliari et al. in [5, 6] (using Leja interpolation). With respect to computational time, it turns out that `phipm` offers an advantage. This algorithm utilizes an adaptive time-stepping method to evaluate (22) using only one matrix function (see Sect. 4.2.1 below). This task is carried out in a lower dimensional Krylov subspace using standard Krylov subspace projection methods i.e. the Arnoldi iteration. Moreover, the dimension of Krylov subspaces and the number of substeps are also chosen adaptivity for improving efficiency.

Recently, the `phipm` routine was modified by Gaudreault and Pudykiewicz in [13] (Algorithm 2) by using the incomplete orthogonalization method (IOM) within the Arnoldi iteration and by adjusting the two crucial initial parameters for starting the Krylov adaptivity. This results in the new routine called `phipm/IOM2`. It is shown in [13] that this algorithm reduces computational time significantly compared to `phipm` when integrating the shallow water equations on the sphere.

Very recently, the authors of [35] further improved `phipm/IOM2` which resulted in a more efficient routine named as `phipm_simul_iom2`. For the reader's convenience, we present the idea of the adaptive time-stepping method (originally proposed in [41]) for evaluating (22) and introduce some new features of the new routine `phipm_simul_iom2`.

4.2.1 Computing of Linear φ-Combinations Based on Time-Stepping

It was observed that the following linear ODE

$$u'(t) = Mu(t) + v_1 + tv_2 + \cdots + \frac{t^{p-1}}{(p-1)!}v_p,\ u(0) = v_0, \tag{23}$$

defined on the interval [0, 1] has the exact solution at $t = 1$, $u(1)$ to be the expression (22). The time-stepping technique approximates $u(1)$ by discretizing [0, 1] into subintervals $0 = t_0 < t_1 < \cdots < t_k < t_{k+1} = t_k + \tau_k < \cdots < t_K = 1$ with a substepsize sequence τ_k ($k = 0, 1, \ldots, K-1$) and using the following relation between $u(t_{k+1})$ and its previous solution $u(t_k)$:

$$u(t_{k+1}) = \varphi_0(\tau_k M)u(t_k) + \sum_{i=1}^{p} \tau_k^i \varphi_i(\tau_k M) \sum_{j=0}^{p-i} \frac{t_k^j}{j!} v_{i+j}. \tag{24}$$

Using the recursion relation (19) and (24) can be simplified as

$$u(t_{k+1}) = \tau_k^p \varphi_p(\tau_k M) w_p + \sum_{j=0}^{p-i} \frac{t_k^j}{j!} w_j, \tag{25}$$

where the vectors w_j satisfy the recurrence relation

$$w_0 = u(t_k),\ w_j = Mw_{j-1} + \sum_{\ell=0}^{p-j} \frac{t_k^\ell}{\ell!} v_{j+\ell},\ j = 1, \ldots, p. \tag{26}$$

Equation (25) implies that evaluating $u(t_K) = u(1)$ i.e. the expression (22) requires only one matrix function $\varphi_p(\tau_k A)w_p$ in each substep instead of $(p+1)$ matrix-vector multiplications. As $0 < \tau_k < 1$, this task can be carried out in a Krylov subspace of lower dimension m_k, and in each substep only one Krylov projection is needed. With a reasonable number of substeps K, it is thus expected that the total computational cost of $\mathcal{O}(m_1^2) + \cdots + \mathcal{O}(m_K^2)$ for approximating $\varphi_p(\tau_k M)w_p$ is less than that of $\mathcal{O}(m^2)$ for approximating $\varphi_p(M)v$ in a Krylov subspace of dimension m. If K is too large (e.g. when the spectrum of M is very large), this might be not true. This case, however, is handed by using the adaptive Krylov algorithm in [41] allowing to adjust both the dimension m and the step sizes τ_k adaptivity. This explains the computational advance of this approach compared to standard Krylov algorithms.

4.2.2 New Routine `phipm_simul_iom2` [35]

First, we note that the resulting routine `phipm_simul_iom2` optimizes computational aspects of `phipm/IOM2` corresponding to the following two specific changes:

1. Unlike (22), where each of the φ_k functions is evaluated at the same argument M, the internal stages of exponential Rosenbrock schemes require evaluating the φ functions at fractions of the matrix M:

$$w_k = \sum_{l=1}^{p} \varphi_l(c_k M) v_l, \quad k = 2, \dots, s, \tag{27}$$

   where now the node values $c_2, \dots, c_s$ are scaling factors used for each v_k output. To optimize this evaluation, `phipm_simul_iom2` computes all w_k outputs in (27) simultaneously, instead of computing only one at a time. This is accomplished by first requiring that the entire array $c_2, \dots, c_s$ as an input to the function. Within the substepping process (24), each value c_j is aligned with a substep-size τ_k. The solution vector is stored at each of these moments and on output the full set $\{w_k\}_{k=1}^{s}$ is returned. Note that this approach is similar but differs from [48] that it guarantees no loss of solution accuracy since it explicitly stops at each c_k instead of using interpolation to compute w_k as in [48].
2. In view of the higher-order exponential Rosenbrock schemes (see also from Sect. 3.3), it is realized that they usually use a subset of the φ_l functions. Therefore, multiple vectors in (27) will be zero. In this case, `phipm_simul_iom2` will check whether $w_{j-1} \neq 0$ (within the recursion (26)) before computing the matrix-vector product $M\, w_{j-1}$. While matrix-vector products require $O(n^2)$ work, checking $u \neq 0$ requires only $O(n)$. This can result in significant savings for large n.

4.2.3 Implementation of `exprb42` and `pexprb43`

Taking a closer look at the structures of the two selected exponential Rosenbrock schemes `exprb42` and `pexprb43`, we now make use of `phipm_simul_iom2` for implementing these schemes. For simplicity, let us denote $M = hJ_n$ and $v = hF(u_n)$.
Implementation of `exprb42`: Due to the structure of `exprb42` given in (20), one needs two calls to `phipm_simul_iom2`:

1. Evaluate $y_1 = \varphi_1(\frac{3}{4}M)w_1$ with $w_1 = \frac{3}{4}v$ (so $w_0 = 0$) to get $U_{n2} = u_n + y_1$,
2. Evaluate $w = \varphi_1(M)v_1 + \varphi_3(M)v_3$ (i.e. $v_0 = v_2 = 0$) with $v_1 = v, v_3 = \frac{32}{9}hD_{n2}$ to get $u_{n+1} = u_n + w$.

Implementation of `pexprb43`: Although `pexprb43` is a 3-stage scheme, its special structure (21) allows to use only two calls to `phipm_simul_iom2`:

1. Evaluate both terms $y_1 = \varphi_1(\frac{1}{2}M)v$ and $z_1 = \varphi_1(M)v$ simultaneously to get the two stages $U_{n2} = u_n + \frac{1}{2}y_1$ and $U_{n3} = u_n + z_1$,
2. Evaluate $w = \varphi_3(M)v_3 + \varphi_4(M)v_4$ (i.e. $v_0 = v_1 = v_2 = 0$) with $v_3 = h(16D_{n2} - 2D_{n3})$, $v_4 = h(-48D_{n2} + 12D_{n3})$ to get $u_{n+1} = U_{n3} + w$.

5 Numerical Examples

In this section we present several numerical examples to study the behavior of the presented exponential Rosenbrock-type methods, in particular the fourth-order scheme `exprb42` using two stages and the fourth-order `pexprb43` scheme using three stages implemented in parallel.

In particular, we focus on relevant aspects in the realm of visual computing, like stability and energy conservation, large stiffness, and high fidelity and visual accuracy. A tabular summary of the models that are used throughout this section can be found in Table 2. Furthermore, our simulation includes important aspects like elastic collisions and nonelastic deformations. The presented exponential Rosenbrock-type methods are evaluated against classical and state-of-the-art methods used in visual computing, in particular against the implicit-explicit variational (IMEX) integrator (cf. [44, 45]), the standard fourth-order Runge–Kutta method (see [28, 43]), and the implicit BDF-1 integrator (see [11]). All simulation results visualized here have been computed using a machine with an Intel(R) Xeon E5 3.5 GHz and 32 GB DDR-RAM. For each simulation scenario the largest possible time step size is used which still leads to a desired visually plausible result.

5.1 Simulation of Deformable Bodies

In order to illustrate the accurate energy preservation of the presented exponential Rosenbrock-type methods, we set up an undamped scene of an oscillating coil spring, which is modeled as a deformable body composed of tetrahedra, in particular of 8000 vertices corresponding to $N = 24\,000$ equations of motion, which are derived from a system of coupled oscillators with uniform spring stiffness of $k = 10^6$. Since the coil spring is exposed to an external forces field, it starts to oscillate as illustrated in Fig. 1. It can be seen that the top of the coil spring returns to its initial height periodically during the simulation which can be seen as an indicator for energy conservation. In fact when using the exponential Rosenbrock-type methods `exprb42` and `pexprb43` we observe that the discrete energy is only slightly oscillating around the real energy without increasing oscillations over time. In contrast, the standard fourth order Runge–Kutta method respectively the BDF-1

Table 2 Overview of the test cases used for the numerical experiments. Their complexity N (i.e. the number of the resulting equations of motion), the simulated time, and respective running times for the exponential Rosenbrock-type methods `exprb42` and `pexprb43`, the implicit-explicit variational integrator (V-IMEX), the standard fourth order Runge–Kutta method (RK4), and the BDF-1 integrator are shown

No.	Model	N	Sim. time	`exprb42`	`pexprb43`	V-IMEX	RK 4	BDF-1
1	Coil Spring	$24k$	$60\,s$	55 s	47 s	12 min	46 min	62 min
2	Brushing	$90k$	$15\,s$	52 s	51 s	11 min	53 min	72 min
3	Crash test (moderate)	$360k$	$2\,s$	44 s	44 s	9 min	47 min	58 min
4	Crash test (fast)	$360k$	$2\,s$	47 s	46 s	9 min	47 min	59 min

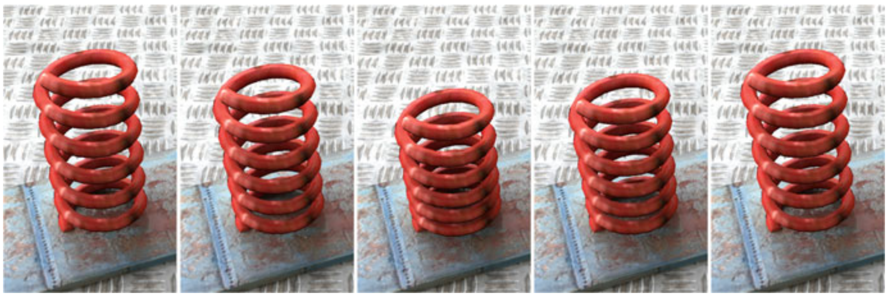

Fig. 1 Simulation of an oscillating coil spring

integrator generate significant numerical viscosity leading to a loss of energy around 22% respectively 40% after 60 s of simulated time.

The exponential Rosenbrock-type methods `exprb42` and `pexprb43` show their advantageous behavior since these methods can be applied with orders of magnitude larger time steps compared to the other integrators. Even with a step size of $h = 0.05$ the relative error is still below 2% for `exprb42` and about a single percent for `pexprb43`.[1] From a point of view of computation time, we achieve a speed up of a factor of around thirteen using `exprb42` and of over fifteen using `pexprb43` compared to the second best method, the variational IMEX integrator as illustrated in Table 2. Compared to the other methods, the exponential Rosenbrock-type methods allow for accurate simulations in real-time.

5.2 *Simulation of Fibers Including Elastic Collisions*

Fibers are canonical examples for complex interacting systems. According to the work of Michels et al. (see [39]), we set up a toothbrush composed of individual bristles. Each bristle consists of coupled oscillators that are connected in such a way that the fiber axis is enveloped by a chain of cuboid elements. For preventing a volumetric collapse during the simulation, additional diagonal springs are used. The toothbrush consists of 1500 bristles, each of 20 particles leading to 90 000 equations of motion. We make use of additional repulsive springs in order to prevent from interpenetrations.[2] Since the approach allows for the direct use of realistic parameters in order to set up the stiffness values in the system of coupled oscillators, we employ a Young's modulus of $3.2 \cdot 10^6\,\mathrm{Ncm}^{-2}$, a torsional modulus of $10^5\,\mathrm{Ncm}^{-2}$, and segment thicknesses of 0.12 mm.

[1] We estimated the error after 60 s of simulated time based on the accumulated Euclidean distances of the individual particles in the position space compared to ground truth values which are computed with a sufficiently small step size.

[2] In order to detect collisions efficiently, we make use of a standard bounding volume hierarchy.

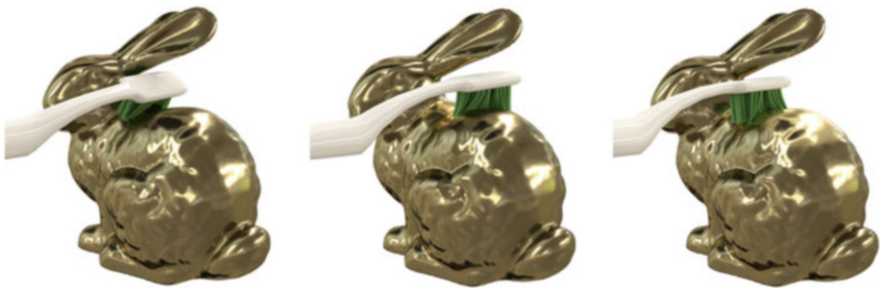

Fig. 2 Simulation of a brush cleaning a bronze-colored paperweight

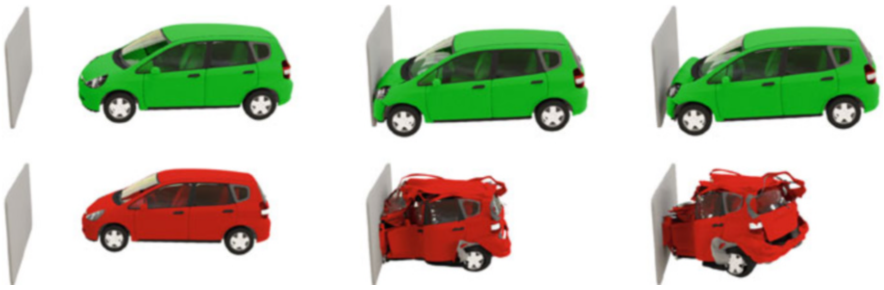

Fig. 3 Simulations of two frontal nonelastic crash scenarios: a car with moderate velocity (top) and high velocity (bottom)

We simulate 15 s of a toothbrush cleaning a paperweight illustrated in Fig. 2. This simulation can be carried out almost in real-time which is not possible with the use of classical methods as illustrated in Table 2.

5.3 *Crash Test Simulation Including Nonelastic Deformations*

As a very complex example with relevance in the context of special effects, we simulate a frontal crash of a car into a wall as illustrated in Fig. 3. The mesh of the car and its interior is composed of 120 000 vertices leading to 360 000 equations of motion. The global motion (i.e. the rebound of the car) is computed by treating the car as a rigid body. Using an appropriate bounding box, this can be easily carried out in real-time. The deformation is then computed using a system of coupled oscillators with structural stiffness values of $k = 10^4$ and bending stiffness values of $k/100$. If the deformation reaches a defined threshold, the rest lengths of the corresponding springs are corrected in a way, that they do not elastically return to their initial shape. Using the exponential Rosenbrock-type methods, the whole simulation can be carried out at interactive frame rates. Such an efficient computation can not be achieved with established methods as illustrated in Table 2.

6 Conclusion

We introduced the class of *explicit exponential Rosenbrock methods* for the time integration of large systems of nonlinear differential equations. In particular, the exponential Rosenbrock-type fourth-order schemes `exprb42` using two stages and `pexprb43` using three stages were discussed and their implementation were addressed. In order to study their behavior, a broad spectrum of numerical examples was computed. In this regard, the simulation of deformable bodies, fibers including elastic collisions, and crash scenarios including nonelastic deformations was addressed focusing on relevant aspects in the realm of visual computing, like stability and energy conservation, large stiffness values, and high fidelity and visual accuracy. An evaluation against classical and state-of-the-art methods was presented demonstrating their superior performance with respect to the simulation of large systems of stiff differential equations.

Acknowledgments The first author has been partially supported by NSF grant DMS–2012022. The second author has been partially supported by King Abdullah University of Science and Technology (KAUST baseline funding).

References

1. A.H. Al-Mohy, N.J. Higham, Computing the action of the matrix exponential, with an application to exponential integrators. SIAM J. Sci. Comput. **33**, 488–511 (2011)
2. U. Ascher, S. Ruuth, B. Wetton, Implicit-explicit methods for time-dependent PDEs. SIAM J. Numer. Anal. **32**(3), 797–823 (1997)
3. D. Baraff, A. Witkin, Large steps in cloth simulation, in *ACM Transactions on Graphics, SIGGRAPH'98* (ACM, New York, 1998), pp. 43–54
4. M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, E. Grinspun, Discrete elastic rods. ACM Trans. Graph. **27**(3), 63:1–63:12 (2008)
5. M. Caliari, A. Ostermann, Implementation of exponential Rosenbrock-type integrators. Appl. Numer. Math. **59**(3–4), 568–581 (2009)
6. M. Caliari, P. Kandolf, A. Ostermann, S. Rainer, The Leja method revisited: backward error analysis for the matrix exponential. SIAM J. Sci. Comput. **38**(3), A1639–A1661 (2016)
7. W.L. Chao, J. Solomon, D. Michels, F. Sha, Exponential integration for Hamiltonian Monte Carlo, in *Proceedings of the 32nd International Conference on Machine Learning*, ed. by F. Bach, D. Blei. Proceedings of Machine Learning Research, vol. 37, pp. 1142–1151 (PMLR, Lille, 2015)
8. Y.J. Chen, U.M. Ascher, D.K. Pai, Exponential Rosenbrock-Euler integrators for elastodynamic simulation. IEEE Trans. Visual. Comput. Graph. **24**(10), 2702–2713 (2018)
9. S.M. Cox, P.C. Matthews, Exponential time differencing for stiff systems. J. Comput. Phys. **176**(2), 430–455 (2002)
10. C. Curtiss, J.O. Hirschfelder, Integration of stiff equations. Proc. Natl. Acad. Sci. **38**(3), 235–243 (1952)
11. C.F. Curtiss, J.O. Hirschfelder, Integration of stiff equations. Proc. Natl. Acad. Sci. USA **38**(3), 235–243 (1952)

12. B. Eberhardt, O. Etzmuß, M. Hauth, Implicit-explicit schemes for fast animation with particle systems, in *Proceedings of the 11th Eurographics Workshop on Computer Animation and Simulation (EGCAS)* (Springer, Berlin, 2000), pp. 137–151
13. S. Gaudreault, J. Pudykiewicz, An efficient exponential time integration method for the numerical solution of the shallow water equations on the sphere. J. Comput. Phys. **322**, 827–848 (2016)
14. C. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations* (Prentice–Hall, Englewood Cliffs, 1971)
15. S. Geiger, G. Lord, A. Tambue, Exponential time integrators for stochastic partial differential equations in 3D reservoir simulation. Comput. Geosci. **16**(2), 323–334 (2012)
16. R. Goldenthal, D. Harmon, R. Fattal, M. Bercovier, E. Grinspun, Efficient simulation of inextensible cloth, in *ACM Transactions on Graphics, SIGGRAPH'07* (2007)
17. M.A. Gondal, Exponential Rosenbrock integrators for option pricing. J. Comput. Appl. Math. **234**(4), 1153–1160 (2010)
18. E. Hairer, G. Wanner, *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems* (Springer, New York, 1996)
19. M. Hauth, O. Etzmuss, A high performance solver for the animation of deformable objects using advanced numerical methods. Comput. Graph. Forum **20**, 319–328 (2001)
20. N.J. Higham, *Functions of Matrices: Theory and Computation* (SIAM, Philadelphia, 2008)
21. M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator. SIAM J. Numer. Anal. **34**, 1911–1925 (1997)
22. M. Hochbruck, A. Ostermann, Explicit exponential Runge–Kutta methods for semilinear parabolic problems. SIAM J. Numer. Anal. **43**, 1069–1090 (2005)
23. M. Hochbruck, A. Ostermann, Explicit integrators of Rosenbrock-type. Oberwolfach Rep. **3**, 1107–1110 (2006)
24. M. Hochbruck, A. Ostermann, Exponential integrators. Acta Numer. **19**, 209–286 (2010)
25. M. Hochbruck, C. Lubich, H. Selhofer, Exponential integrators for large systems of differential equations. SIAM J. Sci. Comput. **19**, 1552–1574 (1998)
26. M. Hochbruck, A. Ostermann, J. Schweitzer, Exponential Rosenbrock-type methods. SIAM J. Numer. Anal. **47**, 786–803 (2009)
27. S. Krogstad, Generalized integrating factor methods for stiff PDEs. J. Comput. Phys. **203**(1), 72–88 (2005)
28. M.W. Kutta, Beitrag zur näherungsweisen Integration totaler Differentialgleichungen. Z. Math. Phys. **46**, 435–453 (1901)
29. V.T. Luan, Fourth-order two-stage explicit exponential integrators for time-dependent PDEs. Appl. Numer. Math. **112**, 91–103 (2017)
30. V.T. Luan, A. Ostermann, Exponential B-series: the stiff case. SIAM J. Numer. Anal. **51**, 3431–3445 (2013)
31. V.T. Luan, A. Ostermann, Explicit exponential Runge–Kutta methods of high order for parabolic problems. J. Comput. Appl. Math. **256**, 168–179 (2014)
32. V.T. Luan, A. Ostermann, Exponential Rosenbrock methods of order five–construction, analysis and numerical comparisons. J. Comput. Appl. Math. **255**, 417–431 (2014)
33. V.T. Luan, A. Ostermann, Stiff order conditions for exponential Runge–Kutta methods of order five, in *Modeling, Simulation and Optimization of Complex Processes - HPSC 2012*, H.B. et al. (ed.) (Springer, Berlin, 2014), pp. 133–143
34. V.T. Luan, A. Ostermann, Parallel exponential Rosenbrock methods. Comput. Math. Appl. **71**, 1137–1150 (2016)
35. V.T. Luan, J.A. Pudykiewicz, D.R. Reynolds, Further development of the efficient and accurate time integration schemes for meteorological models J. Comput. Sci. **376**, 817–837 (2018)
36. D.L. Michels, M. Desbrun, A semi-analytical approach to molecular dynamics. J. Comput. Phys. **303**, 336–354 (2015)
37. D.L. Michels, J.P.T. Mueller, Discrete computational mechanics for stiff phenomena, in *SIGGRAPH ASIA 2016 Courses* (2016), pp. 13:1–13:9

38. D.L. Michels, G.A. Sobottka, A.G. Weber, Exponential integrators for stiff elastodynamic problems. ACM Trans. Graph. **33**(1), 7:1–7:20 (2014)
39. D.L. Michels, J.P.T. Mueller, G.A. Sobottka, A physically based approach to the accurate simulation of stiff fibers and stiff fiber meshes. Comput. Graph. **53B**, 136–146 (2015)
40. D.L. Michels, V.T. Luan, M. Tokman, A stiffly accurate integrator for elastodynamic problems. ACM Trans. Graph. **36**(4), 116 (2017)
41. J. Niesen, W.M. Wright, Algorithm 919: a Krylov subspace algorithm for evaluating the φ-functions appearing in exponential integrators. ACM Trans. Math. Softw. **38**, 3 (2012)
42. D.A. Pope, An exponential method of numerical integration of ordinary differential equations. Commun. ACM **6**, 491–493 (1963)
43. C.D. Runge, Über die numerische Auflösung von Differentialgleichungen. Math. Ann. **46**, 167–178 (1895)
44. A. Stern, M. Desbrun, Discrete geometric mechanics for variational time integrators, in *SIGGRAPH 2006 Courses* (2006), pp. 75–80
45. A. Stern, E. Grinspun, Implicit-explicit variational integration of highly oscillatory problems. Multiscale Model. Simul. **7**, 1779–1794 (2009)
46. A. Tambue, I. Berre, J.M. Nordbotten, Efficient simulation of geothermal processes in heterogeneous porous media based on the exponential Rosenbrock–Euler and Rosenbrock-type methods. Adv. Water Resour. **53**, 250–262 (2013)
47. D. Terzopoulos, J. Platt, A. Barr, K. Fleischer, Elastically deformable models. ACM Trans. Graph. **21**, 205–214 (1987)
48. M. Tokman, J. Loffeld, P. Tranquilli, New adaptive exponential propagation iterative methods of Runge-Kutta type. SIAM J. Sci. Comput. **34**, A2650–A2669 (2012)
49. H. Zhuang, I. Kang, X. Wang, J.H. Lin, C.K. Cheng, Dynamic analysis of power delivery network with nonlinear components using matrix exponential method, in *2015 IEEE Symposium on Electromagnetic Compatibility and Signal Integrity* (IEEE, Piscataway, 2015), pp. 248–252