Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Efficient exponential time integration for simulating nonlinear coupled oscillators[☆]

Vu Thai Luan [a,*], Dominik L. Michels [b]

[a] Department of Mathematics and Statistics, Mississippi State University, 410 Allen Hall, 175 President's Circle, Mississippi State, MS, 39762, USA
[b] Computational Sciences Group, Visual Computing Center, KAUST, Bldg 1, 23955, Kingdom of Saudi Arabia

ARTICLE INFO

Article history:
Received 12 September 2020
Received in revised form 15 January 2021

Keywords:
Systems of coupled oscillators
Exponential Rosenbrock integrators
Accurate and efficient simulation
Visual computing

ABSTRACT

In this paper, we propose an advanced time integration technique associated with explicit exponential Rosenbrock-based methods for the simulation of large stiff systems of nonlinear coupled oscillators. In particular, a novel reformulation of these systems is introduced and a general family of efficient exponential Rosenbrock schemes for simulating the reformulated system is derived. Moreover, we show the required regularity conditions and prove the convergence of these schemes for the system of coupled oscillators. We present an efficient implementation of this new approach and discuss several applications in scientific and visual computing. The accuracy and efficiency of our approach are demonstrated through a broad spectrum of numerical examples, including a nonlinear Fermi–Pasta–Ulam–Tsingou model, elastic and nonelastic deformations as well as collision scenarios focusing on relevant aspects such as stability and energy conservation, large numerical stiffness, high fidelity, and visual accuracy.

1. Introduction

Many computational models in science and engineering are developed to predict the behavior of multiphysics systems, where complex interactions between these components can result in dynamics over a wide range of time and spatial scales. Typical examples are models in meteorology e.g., [1–3], visual computing and molecular dynamics e.g., [4–6], ocean modeling [7], geophysics [8,9], power systems [10], just to name a few. Using the first principles of physics to model different physical phenomenon of these systems often leads to systems of nonlinear differential equations, e.g., evolution or partial differential equations (PDEs). In time domain simulation of these systems, the major challenge is due to the presence of vastly different time scales in their dynamics (e.g., some components evolve at significantly faster rates than others).

In this work, we are concerned with developing a new approach for computational modeling of systems of nonlinear coupled oscillators (particle systems), which are often used in visual computing (specially in computer animation). Their dynamics can be described by using Newton's second law of motion, leading to a large system of second-order differential equations of the form

$$m_i\ddot{x}_i + \sum_{j\in\mathcal{N}(i)} k_{ij}(\|x_i - x_j\| - \ell_{ij})\frac{x_i - x_j}{\|x_i - x_j\|} = g_i(x_i, \dot{x}_i, \cdot), \quad i = 1, 2, \ldots, N, \tag{1.1}$$

☆ The first author is supported by National Science Foundation, USA grant DMS–2012022. The second author is supported by KAUST, Saudi Arabia baseline funding.
* Corresponding author.
E-mail addresses: luan@math.msstate.edu (V.T. Luan), dominik.michels@kaust.edu.sa (D.L. Michels).

https://doi.org/10.1016/j.cam.2021.113429
0377-0427/© 2021 Elsevier B.V. All rights reserved.

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. For example, the model (1.1) is usually used to describe dynamics of coupled oscillators such as fibers, fluids, or flexible solids, and their interaction with each other. This has many applications in animation, movie industry, and medical imaging. Real-time simulation of these systems is challenging since the dynamics of the fabric occurs over a wide range of temporal scales, e.g. from appearance of small wrinkles to overall folding of the fabric. This is usually referred to the so-called *stiffness*. A common approach for integrating (1.1) is to reduce it to the first order initial value problem, which can be cast in the general form

$$u'(t) = F(u(t)), \quad u(t_0) = u_0, \tag{1.2}$$

where $u \in \mathbb{R}^n$ is the state vector and $F : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ represents the forcing field. In fact, this resulting system is usually very stiff in the sense that the eigenvalues of the Jacobian matrix of F can differ by several orders of magnitude. In the early days of developing numerical methods for solving (1.2), classical methods such as the explicit Runge–Kutta integrators were proposed. For stiff systems, however, these classical methods often show the lack of stability due to the CFL condition, leading to the use of unreasonable time steps particularly for large-scale applications. In this regard, the development of implicit methods such as semi-implicit, IMEX e.g. [11], and BDF methods e.g. [12,13] has changed the situation. These standard methods, however, require the solution of nonlinear systems of equations in each step. As the stiffness of the problem increases, considerable computational effort is required. This can be considered as a shortcoming of the implicit schemes.

Our approach for integrating (1.1) is first to introduce a novel reformulation of it in the form of (1.2), in which the forcing term has a special structure involving an infinitesimal symplectic and skew-symmetric matrix for the linear part and a large nonlinearity. The reformulated system is also very stiff since the linear spring forces possess very high frequencies. Due to the large nonlinearity, we then suggest the use of exponential Rosenbrock-based approach and derive a general family of fourth-order efficient schemes for handling this system (It is worth mentioning that exponential Rosenbrock methods have shown to be efficient for integrating the fully nonlinear stiff system (1.2) since they can handle the stiffness of the system in an explicit and very accurate way. This class of exponential integrators was originally proposed in [14] and further developed in [15–18]. The lower-order schemes were recently successfully applied to a number of different applications [5,8–10,19] and very recently high-order schemes were shown to be very promising for some meteorological models [3]). Moreover, we find regularity conditions posed on (1.1) and prove the fourth-order convergence of the new family of efficient exponential Rosenbrock schemes for the system of coupled oscillators (1.1). Additionally, we propose to use the improved algorithm in a recent work [3] for the evaluation of a linear combination of φ-functions acting on certain vectors $v_0, \ldots, v_p$, i.e. $\sum_{k=0}^{p} \varphi_k(A)v_k$, which is crucial for implementing these exponential schemes. Altogether, 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.

The paper is organized as follows. In Section 2, we present a novel reformulation of the system of coupled oscillators (1.1) in the form of (1.2) (see Lemma 2.1) and briefly review previous approaches used for simulating these systems in visual computing. In Section 3, we describe the exponential Rosenbrock methods as an alternative approach for solving large stiff systems (1.2) and derive a general family of efficient fourth-order schemes called pexprb43(c_2, c_3). The convergence analysis of these schemes and the selected (superconvergent) scheme exprb42 for the system of coupled oscillators (1.1) is presented in Theorem 3.1. The implementation of these methods is discussed in Section 4, where we also introduce a new procedure to further improve one of the state-of-the-art algorithms. The main result of this section is Algorithm 4.1. In Section 5, we demonstrate the accuracy and efficiency of our proposed technique on a nonlinear Fermi–Pasta–Ulam–Tsingou (FPUT) model and 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 Section 6.

2. A novel reformulation of the system of coupled oscillators

For the system of coupled oscillators (1.1), let us denote $x(t) \in \mathbb{R}^{3N}$, $M \in \mathbb{R}^{3N\times 3N}$, $K \in \mathbb{R}^{3N\times 3N}$ and $g(x) \in \mathbb{R}^{3N}$ be the vector of positions, the mass matrix (often diagonal and thus nonsingular), the spring matrix (stiff), and the total external forces acting on the system, respectively. Using these matrix notations and additionally setting $A = M^{-1}K$, (1.1) can be rewritten in a more compact form as

$$x''(t) + Ax(t) = g(x(t)), \quad x(t_0) = x_0, \ x'(t_0) = v_0 \tag{2.1}$$

with some given initial positions included in x_0 and velocities included in v_0. (Note that, for simplicity, here we have neglected the damping).

Simulating models described by (2.1) in visual computing is challenging due to the presence of extremely large number of particles N and the linear spring forces usually possess very high frequencies, i.e., $\|K\| \gg 1$ and so is $\|A\| \gg 1$, meaning that (2.1) is a very large stiff system. To handle this situation, we first introduce a novel (and equivalent) transformation of (2.1) into a first-order system which has a special structure.

Lemma 2.1. *Suppose that A is a symmetric and positive definite matrix, the system (2.1) can be reformulated as (1.2) with the forcing term $F(u(t))$ and the initial value u_0 are given as*

$$u'(t) = F(u(t)) = \mathscr{A}u(t) + G(u(t)), \quad u(t_0) = u_0 = [\sqrt{A}\, x_0, v_0]^T, \tag{2.2}$$

where $\sqrt{A}$ is the matrix square root of A and $\mathscr{A}$ is a skew-symmetric and an infinitesimal symplectic (or Hamiltonian) matrix given by

$$\mathscr{A} = \begin{bmatrix} \mathbf{0} & \sqrt{A} \\ -\sqrt{A} & \mathbf{0} \end{bmatrix}, \quad \text{and} \quad G(u) = \begin{bmatrix} \mathbf{0} \\ g(x) \end{bmatrix}. \tag{2.3}$$

Moreover, if $g(x)$ is a differentiable conservative vector field in $\mathbb{R}^{3N}$ and that $g'(x)$ and $\sqrt{A}$ commute, (2.2) is a Hamiltonian system.

Proof. Since A is a symmetric and positive definite matrix, it is well known that $\sqrt{A}$ exists and is a unique positive definite matrix such that $\sqrt{A}^2 = A$. Thus, $\sqrt{A}$ is also symmetric and clearly $\sqrt{A}^{-1}$ exists. With this, we introduce

$$u(t) = \begin{bmatrix} \sqrt{A}x(t) \\ x'(t) \end{bmatrix} \tag{2.4}$$

as a new change of variable. Note that a similar idea was used for elastic systems, see, e.g., [20]. By differentiating (2.4) with respect to t and using (2.1), one can easily verify (2.2)–(2.3). Clearly, $\mathscr{A}$ given in (2.3) is a skew-symmetric matrix. Next, let us consider the antisymmetric matrix $W = \begin{bmatrix} \mathbf{0} & I \\ -I & \mathbf{0} \end{bmatrix}$. We have

$$W\mathscr{A} = \begin{bmatrix} -\sqrt{A} & \mathbf{0} \\ \mathbf{0} & -\sqrt{A} \end{bmatrix} = (W\mathscr{A})^T,$$

which is a symmetric matrix. This implies that $W\mathscr{A} + \mathscr{A}^T W = \mathscr{A}^T(W^T + W) = \mathbf{0}$. Therefore, $\mathscr{A}$ is an infinitesimal symplectic matrix (by definition).

Finally, we note that (2.2) is a Hamiltonian system if the Jacobian matrix

$$J(u) = F'(u) = \mathscr{A} + G'(u) = \begin{bmatrix} \mathbf{0} & \sqrt{A} \\ -\sqrt{A} + g'(x)\sqrt{A}^{-1} & \mathbf{0} \end{bmatrix} \tag{2.5}$$

is infinitesimal symplectic. As done for $\mathscr{A}$, one needs to show that $WJ(u)$ is a symmetric matrix. In fact, under the assumptions on $g(x)$ stated in Lemma 2.1, there exists some differentiable function $f(x)$ such that $g(x) = \nabla f(x)$ and $g'(x) = \nabla^2 f(x)$ becomes a Hessian matrix, which is symmetric. Since $g'(x)$ and $\sqrt{A}$ commute, $g'(x)\sqrt{A}^{-1}$ is a symmetric matrix. Therefore, it is clear that

$$WJ(u) = \begin{bmatrix} -\sqrt{A} + g'(x)\sqrt{A}^{-1} & 0 \\ 0 & -\sqrt{A} \end{bmatrix}$$

is also a symmetric matrix. □

Remark 2.1. The condition that A is a symmetric and positive definite matrix is a reasonable assumption in many models, see [21]. The new formulation (2.2)–(2.3) for systems of coupled oscillators modeled by (2.1) significantly differs from the common way of changing of variable $u(t) = [x(t),\ x'(t)]^T$, leading to a coefficient matrix of the linear part that is non-symmetric. Here $\mathscr{A}$ (given by (2.3)) is a skew-symmetric matrix, so all of its nonzero eigenvalues are pure imaginary and are in pairs of the form $\pm\lambda_k i$, and thus $\|e^{t\mathscr{A}}\| = 1$. Also, this formulation allows a Hamiltonian-like structure, which helps to improve the preservation of the total energy of the system.

Clearly, either using the common way (mentioned in Remark 2.1) or the new way (2.4) for reformulating (2.1), one has to solve a stiff system of the form (2.2). In visual computing it is usually integrated by explicit methods such as the fourth-order Runge–Kutta scheme (RK4), semi-implicit methods such as the Störmer–Verlet schemes, 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 [22–27]. 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 [28]). They have attracted much attention by the broad computational mathematics community since mid-1990s. At the time while solving linear systems $(I-\alpha hJ)x=v$ 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 [29]). 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 works on exponential integrators, see e.g. [4,21,30,31]. Inspired by this interest, in the following sections we will show how exponential Rosenbrock methods can be applied for simulating the reformulated system of coupled oscillators.

3. Exponential Rosenbrock methods

In this section, we present a compact theory of exponential Rosenbrock methods based on [14–17,32]. We then derive a family of efficient schemes for our experiments with application models in visual computing.

3.1. Ideas and observations

Motivated by [33, Chap. IV.7], the first idea is replace the integration of the fully nonlinear system (1.2) by integrating a sequence of semilinear problems

$$u'(t)=F(u(t))=J_nu(t)+g_n(u(t)), \tag{3.1}$$

resulted from a dynamic linearization of the forcing term $F(u)$ in each time step at the numerical solution u_n (due to [34]) with

$$J_n=F'(u_n),\;\; g_n(u)=F(u)-J_nu \tag{3.2}$$

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 now to handle the stiffness by solving the linear part J_nu 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 (3.1) using the variation-of-constants formula

$$u(t_{n+1})=\mathrm{e}^{hJ_n}u(t_n)+\int_0^h \mathrm{e}^{(h-\tau)J_n}g_n(u(t_n+\tau))\mathrm{d}\tau \tag{3.3}$$

plays a crucial role in constructing this type of integrators. As seen from (3.3), while the linear part can be integrated exactly by 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 [14,15]. We note that these exponential methods can be easily applied to non-autonomous problems $u'(t)=F(t,u)$ by adding the equation $t'=1$, see [15,16] (in this case, the nonlinear remainder, upon the dynamic linearization, would be $g_n(t,u)$ instead of $g_n(u)$).

Remark 3.1. For the system of coupled oscillators (1.1), the forcing term $F(u)$ has the semilinear form (2.2), which can be considered as a fixed linearization, i.e., $J_n=\mathscr{A}$. Therefore, one can directly apply explicit exponential Runge–Kutta methods (see [35]) to (2.2). 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 usually large in which its CFL condition usually serves as a reference for setting the time-step. In particular, hL_G should be sufficiently small (L_G is the Lipschitz constant of $G(u)$) to ensure the stability. With respect to this regard, the dynamic linearization approach (3.1) applied to (2.2)

$$u'(t)=F(u)=\mathscr{A}u+G(u)=J_nu+G_n(u) \tag{3.4}$$

with

$$J_n=\mathscr{A}+G'(u_n), \tag{3.5}$$

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_nu with the Jacobian J_n as in (3.5) 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 the classical and exponential Runge–Kutta methods.

3.2. Formulation of a second-order scheme and general schemes

We now illustrate this approach 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

$$g_n(u(t_n+\tau))=g_n(u(t_n))+\mathcal{O}(\tau^2)\,. \tag{3.6}$$

Inserting (3.6) into (3.3) and denoting $\varphi_1(hJ_n)=\frac{1}{h}\int_0^h \mathrm{e}^{(h-\tau)J_n}\,\mathrm{d}\tau$ gives

$$u(t_{n+1})=\mathrm{e}^{hJ_n}u(t_n)+h\varphi_1(hJ_n)g_n(u(t_n))+\mathcal{O}(h^3). \tag{3.7}$$

Neglecting the local error term $\mathcal{O}(h^3)$ results in a second-order scheme which will be called `exprb2`:

$$u_{n+1}=\mathrm{e}^{hJ_n}u_n+h\varphi_1(hJ_n)g_n(u_n)=u_n+h\varphi_1(hJ_n)F(u_n). \tag{3.8}$$

Here the second equality in (3.8) is due to replacing $g_n(u(t_n))$ by (3.2) and using the fact that $\varphi_1(z)=(e^z-1)/z$. Note that this scheme was already derived before and named as *exponential Rosenbrock–Euler method*, see [14,15] (since when considering the formal limit $J_n\to\mathbf{0}$, (3.8) is the underlying Euler method). Our derivation here, however, shows directly that it has an order of consistency three and thus is a second-order stiffly accurate method.

3.2.2. General schemes

For the derivation of higher-order schemes, one can approximate the integral in (3.3) by using some higher-order quadrature rule with nodes c_i in $[0,1]$ and weights $b_i(hJ_n)$ (matrix functions of hJ_n), yielding

$$u(t_{n+1})\approx \mathrm{e}^{hJ_n}u(t_n)+h\sum_{i=1}^{s}b_i(hJ_n)g_n(u(t_n+c_ih)). \tag{3.9}$$

The unknown intermediate values $u(t_n+c_ih)$ can be again approximated by using (3.3) (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

$$u(t_n+c_ih)\approx \mathrm{e}^{c_ihJ_n}u(t_n)+h\sum_{j=1}^{i-1}a_{ij}(hJ_n)g_n(u(t_n+c_jh)). \tag{3.10}$$

Let us now denote $u_n\approx u(t_n)$ and $U_{ni}\approx u(t_n+c_ih)$. As done for (3.8), using (3.6) (with c_ih, h in place of τ, respectively) one can reformulate (3.9) and (3.10) in a similar manner, which yields the general format of s-stage explicit exponential Rosenbrock methods

$$U_{ni}=u_n+c_ih\varphi_1(c_ihJ_n)F(u_n)+h\sum_{j=2}^{i-1}a_{ij}(hJ_n)D_{nj}, \tag{3.11a}$$

$$u_{n+1}=u_n+h\varphi_1(hJ_n)F(u_n)+h\sum_{i=2}^{s}b_i(hJ_n)D_{ni} \tag{3.11b}$$

with

$$D_{ni}=g_n(U_{ni})-g_n(u_n), \tag{3.11c}$$

As in (3.6), we have $D_{ni}=\mathcal{O}(h^2)$. Thus, the general methods (3.11) are small perturbations of `exprb2` above. Note that the unknown weights $a_{ij}(hJ_n)$ and $b_i(hJ_n)$ can be determined by solving the order conditions. They are usually chosen as linear combinations of the well-known $\varphi_k(z)$ functions (see a justification as follows).

The family of φ-functions. Note that, in (3.6) if we further expand $g_n(u(t_n+\tau))$ in a Taylor series at $u(t_n)$, one gets

$$g_n(u(t_n+\tau))=g_n(u(t_n))+\sum_{k=3}^{p}\frac{\tau^{k-1}}{(k-1)!}g_n^{(k-1)}(u(t_n))(\underbrace{V,\ldots,V}_{k\ \text{times}})+\mathcal{O}(\tau^p)$$

with $V=\frac{1}{\tau}(u(t_n+\tau)-u(t_n))=\sum_{j\ge 1}^{q}\frac{\tau^{j-1}}{j!}u^{(j)}(t_n)+\mathcal{O}(\tau^q)$. Inserting this into (3.3) gives

$$\begin{aligned}u(t_{n+1})&=\mathrm{e}^{hJ_n}u(t_n)+h\varphi_1(hJ_n)g_n(u(t_n))\\&\quad+\sum_{k=3}^{p}h^k\Big(\frac{1}{h^k}\int_0^h \mathrm{e}^{(h-\tau)J_n}\frac{\tau^{k-1}}{(k-1)!}\,\mathrm{d}\tau\Big)g_n^{(k-1)}(u(t_n))(\underbrace{V,\ldots,V}_{k\ \text{times}})+\mathcal{O}(h^{p+1}).\end{aligned} \tag{3.12}$$

This motivates the introduction of the family of φ_k-functions

$$\varphi_0(z) = \mathrm{e}^z, \quad \varphi_k(z) = \frac{1}{h^k}\int_0^h \mathrm{e}^{(h-\tau)\frac{z}{h}} \frac{\tau^{k-1}}{(k-1)!}\,\mathrm{d}\tau, \quad k \geq 1 \tag{3.13}$$

They satisfy the recursion relation

$$\varphi_{k+1}(z) = \frac{\varphi_k(z) - \frac{1}{k!}}{z}, \quad k \geq 1. \tag{3.14}$$

Using (3.13), it is clear that the exact solution $u(t_{n+1})$ in (3.12) can be represented as a linear combination of the product of φ-functions with vectors

$$\begin{aligned} u(t_{n+1}) &= \varphi_0(hJ_n)u(t_n) + h\varphi_1(hJ_n)g_n(u(t_n)) \\ &\quad + \sum_{k=3}^{p} h^k \varphi_k(hJ_n) g_n^{(k-1)}(u(t_n))(\underbrace{V,\ldots,V}_{k \text{ times}}) + \mathcal{O}(h^{p+1}). \end{aligned}$$

This observation suggests that, for the numerical solution in (3.11), it is reasonable to choose (by construction) the coefficients $a_{ij}(hJ_n)$ and $b_i(hJ_n)$ as linear combinations of $\varphi_k(c_i hJ_n)$ and $\varphi_k(hJ_n)$, respectively. It is well-known that these $\varphi_k(hJ_n)$ functions (and thus the coefficients $a_{ij}(hJ_n)$ and $b_i(hJ_n)$) are (uniformly) bounded independently of $\|J_n\|$, i.e., the stiffness, see, e.g., [28].

Clearly, using (3.11) offers several advantages. First, they are fully explicit and 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 thus methods can be simplified considerably. In particular, higher-order stiffly accurate schemes can be derived by using only a few stages (see the next section).

3.3. Local error and stiff order conditions

A standard approach to derive order conditions for exponential Rosenbrock methods is to compare the expansion of the exact solution $u(t_n + h)$ with that of the numerical solution u_{n+1} (obtained after one step starting from $u(t_n)$ with the local assumption that $u(t_n) = u_n$) to obtain the local error $LE_n = u_{n+1} - u(t_n + h)$. For this, a common way is to expand the coefficients $a_{ij}(hJ_n)$ and $b_i(hJ_n)$ of the general scheme (3.11) as $a_{ij}(hJ_n) = \sum_{k\geq 0} \alpha_{ij}^{(k)}(hJ_n)^k$, $b_i(hJ_n) = \sum_{k\geq 0} \beta_i^{(k)}(hJ_n)^k$ (e.g., using classical Taylor series expansions) to determine order conditions. This approach, however, involves powers of the Jacobian J_n, resulting in *nonstiff* or *classical* order conditions since they are only valid for problems where $h\|J_n\|$ is small, i.e., either the problem is nonstiff ($\|J_n\|$ is small) or the time step size h must be very small. As a result, methods derived based on this approach (called non-stiffly accurate or classical exponential methods, see, e.g., [36,37]) may suffer from order reduction when applied to stiff problems. Therefore, for stiff systems ($\|J_n\|$ is usually large or is even unbounded), one has to be cautious when analyzing the local error to ensure that error terms do not involve powers of J_n. Recently, Luan and Ostermann [16] derived a new expansion of the local error

$$\begin{aligned} LE_n &= h^3\psi_3(hJ_n)g_n''(u_n)(u_n', u_n') + h^4\psi_4(hJ_n)\frac{\mathrm{d}^3}{\mathrm{d}t^3}g_n(u(t))\Big|_{t=t_n} \\ &\quad + h^5\sum_{i=2}^{s} b_i(hJ_n)c_i g_n''(u_n)\big(u_n', \psi_{3,i}(hJ_n)g_n''(u_n)(u_n', u_n')\big) \\ &\quad + h^5\psi_5(hJ_n)\frac{\mathrm{d}^4}{\mathrm{d}t^4}g_n(u(t))\Big|_{t=t_n} + \mathcal{O}(h^6) \end{aligned} \tag{3.15}$$

with

$$\begin{aligned} \psi_j(Z) &= \sum_{i=2}^{s} b_i(Z)\frac{c_i^{j-1}}{(j-1)!} - \varphi_j(Z), \quad j = 3, 4, 5 \\ \psi_{3,i}(Z) &= \sum_{k=2}^{i-1} a_{ik}(Z)\frac{c_k^2}{2!} - c_i^3\varphi_3(c_i Z), \end{aligned} \tag{3.16}$$

which fulfills this desirable property (all the terms multiplying with the powers of h in (3.15) are now bounded independently of $\|J_n\|$, i.e., the stiffness). This novel local error expansion thus provides a pathway to derive a new stiff order conditions theory for exponential Rosenbrock/Runge–Kutta methods of arbitrary order [18]. For example, by zeroing the terms in (3.15), one can obtain the stiff order conditions for exponential Rosenbrock methods of orders up to 5, which require only 4 conditions (see Table 1). As expected, the number of order conditions for exponential Rosenbrock methods is significantly less than those for exponential Runge–Kutta methods of the same order (which require 16 order conditions for order 5, see [38,39]).

Table 1
Stiff order conditions for exponential Rosenbrock methods up to order 5. Here Z and K denote arbitrary square matrices.

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	$\sum_{i=2}^{s} b_i(Z)c_i K\big(\sum_{k=2}^{i-1} a_{ik}(Z)\frac{c_k^2}{2!} - c_i^3\varphi_3(c_iZ)\big) = 0$	5

3.4. Derivation of new schemes for numerical simulations

By solving the stiff order conditions in Table 1, we note that one can easily construct various methods of orders up to 5. Taking the compromise between efficiency and accuracy into consideration, however, we propose the two ideas below for building a new family of fourth-order efficient schemes with parallel stages and presenting a superconvergent scheme for our applications.

Our first idea, motivated by [17], is to derive a general family of fourth-order exponential Rosenbrock schemes with parallel stages whose implementation can be done simultaneously or in parallel. This requires solving conditions 1 and 2 of Table 1 in a special manner. Due to the fact that the matrix functions $\{\varphi_3(Z), \varphi_4(Z)\}$ are linearly independent, it is clear that one needs at least 3 stages to satisfy these stiff order conditions in their strong forms (i.e. holding for arbitrary square matrix Z). A scheme with $s = 3$ reads as

$$U_{n2} = u_n + c_2 h\varphi_1(c_2 hJ_n)F(u_n), \tag{3.17a}$$

$$U_{n3} = u_n + c_3 h\varphi_1(c_3 hJ_n)F(u_n) + ha_{32}(hJ_n)D_{n2}, \tag{3.17b}$$

$$u_{n+1} = u_n + h\varphi_1(hJ_n)F(u_n) + hb_2(hJ_n)D_{n2} + hb_3(hJ_n)D_{n3}. \tag{3.17c}$$

and the required conditions 1 and 2 now become:

$$b_2(hJ_n)c_2^2 + b_3(hJ_n)c_3^2 = 2\varphi_3(hJ_n), \tag{3.18a}$$

$$b_2(hJ_n)c_2^3 + b_3(hJ_n)c_3^3 = 6\varphi_4(hJ_n). \tag{3.18b}$$

For positive nodes $c_2 \neq c_3$, (3.18) has a unique solution

$$b_2(hJ_n) = \frac{2c_3\varphi_3(hJ_n) - 6\varphi_4(hJ_n)}{c_2^2(c_3 - c_2)}, \quad b_3(hJ_n) = \frac{2c_2\varphi_3(hJ_n) - 6\varphi_4(hJ_n)}{c_3^2(c_2 - c_3)}.$$

Since (3.18) does not depend on the coefficient $a_{32}(hJ_n)$ in (3.17b), one can consider it as a free parameter and choose it to be the zero matrix to make U_{n3} independent of U_{n2}. Putting altogether and rearranging terms in (3.17c) leads to the following two-parameter fourth-order parallel stage scheme which will be called pexprb43(c_2, c_3):

$$U_{n2} = u_n + c_2 h\varphi_1(c_2 hJ_n)F(u_n), \tag{3.19a}$$

$$U_{n3} = u_n + c_3 h\varphi_1(c_3 hJ_n)F(u_n), \tag{3.19b}$$

$$\begin{aligned} u_{n+1} = u_n &+ h\varphi_1(hJ_n)F(u_n) + h\varphi_3(hJ_n)\big(\tfrac{2c_3}{c_2^2(c_3-c_2)}D_{n2} + \tfrac{2c_2}{c_3^2(c_2-c_3)}D_{n3}\big) \\ &+ h\varphi_4(hJ_n)\big(\tfrac{-6}{c_2^2(c_3-c_2)}D_{n2} - \tfrac{6}{c_3^2(c_2-c_3)}D_{n3}\big), \end{aligned} \tag{3.19c}$$

where the vectors $D_{ni} (i = 2, 3)$ are given by (3.11c).

Although pexprb43(c_2, c_3) is a 3-stage scheme, it has the pair of parallel stages $\{U_{n2}, U_{n3}\}$ which can be implemented in parallel (since they are independent of one another) or simultaneously (since they have the same format $U_{ni} = u_n + c_i h\varphi_1(c_i hJ_n)F(u_n)$ $(i = 2, 3)$, making it behaves like a 2-stage method. For its implementation, this requires only two sequential evaluations in each step (see Section 4.2.3).

Our second idea is to search for a superconvergent exponential Rosenbrock method which uses a minimal number of stages while achieving high-order accuracy. Thanks to a recent work [32], such a method (2-stage 4th-order) was constructed by weakening condition 2 of Table 1. There, it was denoted by exprb42:

$$U_{n2} = u_n + \tfrac{3}{4}h\varphi_1(\tfrac{3}{4}hJ_n)F(u_n), \tag{3.20a}$$

$$u_{n+1} = u_n + h\varphi_1(hJ_n)F(u_n) + h\tfrac{32}{9}\varphi_3(hJ_n)(g_n(U_{n2}) - g_n(u_n)). \tag{3.20b}$$

In the current work, we will now show convergence of this scheme for the system of coupled oscillators (1.1), present an efficient implementation of it in Section 4, and compare its performance in Section 5.

3.5. Convergence of the proposed schemes

The stability and convergence analysis of exponential Rosenbrock methods for nonlinear time-dependent PDEs are usually carried out in the framework of strongly continuous semigroup in some Banach space, see [15,16]. Here, we find regularity conditions posed on (2.1) to apply these analyses and prove the fourth-order convergence of `pexprb43`$(\mathtt{c}_2, \mathtt{c}_3)$ and `exprb42` for the system of coupled oscillators (1.1).

Theorem 3.1. *Consider the system of coupled oscillators (1.1) that can be rewritten as the initial value problem (2.1). Under the assumptions that the solution $x : [0, T] \to \mathbb{R}^{3N}$ is sufficiently smooth, $A \in \mathbb{R}^{3N\times 3N}$ is a symmetric, positive definite matrix with (uniformly) bounded $\sqrt{A}^{-1}$, and that the nonlinearity $g : \mathbb{R}^{3N} \to \mathbb{R}^{3N}$ is sufficiently differentiable (with bounded derivatives) in a strip along the exact solution $x(t)$, the scheme* `pexprb43`$(\mathtt{c}_2, \mathtt{c}_3)$ *converges with order 4 when applied to the reformulated system (2.2). Furthermore, if*

$$\sqrt{A}\frac{\mathrm{d}^3}{\mathrm{d}t^3}g(x(t))\Big|_{t=t_n} \quad \text{and} \quad \sqrt{A}\, g'(x(t_n))x^{(3)}(t_n) \tag{3.21}$$

are uniformly bounded in $\mathbb{R}^{3N}$, then the scheme `exprb42` *also converges with order 4. In particular, the numerical solution x_n of (2.1) obtained by either* `pexprb43`$(\mathtt{c}_2, \mathtt{c}_3)$ *or* `exprb42` *satisfies the error bounds*

$$\|x_n - x(t_n)\| \le Ch^4, \quad \|x'_n - x'(t_n)\| \le C'h^4 \tag{3.22}$$

uniformly on $t_0 \le t_n \le T$. Here, $\|\cdot\|$ denotes the maximum norm, and the constants C and C' are independent of n and h. This means that the global errors in (3.22) satisfy the bounds uniformly on $[t_0, T]$ with the constants C and C' that depend on the interval width $(T - t_0)$ but that are independent of $\|A\|$, i.e., the stiffness of the system.

Proof. First, we show, under the assumptions of Theorem 3.1, that the analytical framework (see the Appendix) for the convergence of exponential Rosenbrock methods (3.19) applied to (2.2) is fulfilled. As stated in Lemma 2.1, $\sqrt{A}$ and its inverse exist (and are unique). Thus, $\mathscr{A}$ (a $6N \times 6N$ matrix given in (2.3)) is well-defined, and that (2.1) is equivalent to its reformulated system (2.2). Using the assumption on $g(x)$ and the boundedness of $\sqrt{A}^{-1}$, it is easy to see that $G(u)$ (given in (2.3)) is also sufficiently differentiable in a strip along $u(t)$ given in (2.4), and that

$$G'(u) = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ g'(x)\sqrt{A}^{-1} & \mathbf{0} \end{bmatrix} \tag{3.23}$$

is bounded. Therefore, we have that the matrix exponentials $\mathrm{e}^{tJ(u)}$ $(t \ge 0, J(u)$ given in (2.5)) and e^{tJ_n} are also well-defined.

Next, we are ready to prove the error bounds (3.22) for `pexprb43`$(\mathtt{c}_2, \mathtt{c}_3)$. Since `pexprb43`$(\mathtt{c}_2, \mathtt{c}_3)$ satisfies the stiff order conditions for methods of order 4 (consistency), i.e., $LE_n = \mathcal{O}(h^5)$, its convergence follows directly from applying [16, Thm. 4.1] with the help of the stability bound

$$\Big\| \prod_{j=0}^{n-\nu} \mathrm{e}^{hJ_{n-j}} \Big\| \le C_S, \qquad t_0 \le t_\nu \le t_n \le T \tag{3.24}$$

where the constant C_S is uniform in ν and n, see [15, Sect. 3.3]. In particular, the global error for (2.2) can be expressed as

$$e_n = u_n - u(t_n) = h\sum_{\nu=0}^{n-1}\prod_{j=1}^{n-\nu-1} \mathrm{e}^{hJ_{n-j}}\Big(P_\nu + \mathcal{O}(h^4)\Big), \tag{3.25}$$

where P_ν is given by [16, Eqs. (4.5)] and satisfies the bound $\|P_\nu\| \le C\|e_\nu\| + C\|e_\nu\|^2 + Ch^6$, where C only depends on values that are uniformly bounded by the assumptions of Theorem 3.1 (excluding the additional assumption (3.21)), see [16, Lemma 4.5]. Using (3.24) and an application of a discrete Gronwall lemma (see [40]) to (3.25), one can show that

$$u_n - u(t_n) = \begin{bmatrix} \sqrt{A}\big(x_n - x(t_n)\big) \\ x'_n - x'(t_n) \end{bmatrix} = \mathcal{O}(h^4) = \begin{bmatrix} \mathcal{O}(h^4) \\ \mathcal{O}(h^4) \end{bmatrix}, \tag{3.26}$$

in which the remainder terms hidden behind the Landau notation $\mathcal{O}(\cdot)$ are bounded by Ch^4, $C'h^4$ with constants C and C' that are independent of n and h. Using (3.26) and the boundedness of $\sqrt{A}^{-1}$, it is straightforward to verify the error bounds in (3.22).

Finally, we verify (3.22) for `exprb42`. As mentioned in Section 3.4, `exprb42` given in (3.20) also fulfills the stiff order conditions for fourth-order methods in Table 1 but with condition 2 held in its weakened form $\sum_{i=2}^{2} b_i(0)c_i^3 = 6\varphi_4(0) = 1/4$. Using [32, Thm. 4.1], we deduce that it still guarantees fourth-order convergence for stiff system (2.2) if

$$\mathscr{A}\Big(G^{(3)}(u(t_n))(u'(t_n), u'(t_n), u'(t_n)) + 3G''(u(t_n))(u'(t_n), u''(t_n)\Big) \tag{3.27}$$

is uniformly bounded. We now show that this condition is satisfied under the assumptions of Theorem 3.1. Indeed, after performing some calculations, one realizes that (3.27) is equal to

$$\begin{aligned}
&\mathscr{A}\Big(\frac{\mathrm{d}^3}{\mathrm{d}t^3}G(u(t))\Big|_{t=t_n} - G'(u(t_n))u^{(3)}(t_n)\Big)\\
&= \begin{bmatrix} \mathbf{0} & \sqrt{A} \\ -\sqrt{A} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \frac{\mathrm{d}^3}{\mathrm{d}t^3}g(x(t))\Big|_{t=t_n} - g'(x(t_n))x^{(3)}(t_n) \end{bmatrix}\\
&= \begin{bmatrix} \sqrt{A}(\frac{\mathrm{d}^3}{\mathrm{d}t^3}g(x(t))\Big|_{t=t_n} - g'(x(t_n))x^{(3)}(t_n)) \\ \mathbf{0} \end{bmatrix},
\end{aligned} \tag{3.28}$$

which is clearly uniformly bounded due to (3.21). The error bounds (3.22) can be now verified following the same techniques as done for pexprb43(c_2, c_3) (see (3.24)–(3.26)). □

We note that the boundedness of $\sqrt{A}^{-1}$ can be considered as a reasonable assumption since $\|\sqrt{A}^{-1}\|$ is usually (uniformly) bounded independently of the discretization in many models of the form (2.2). Moreover, if $g'(x)$ and $\sqrt{A}$ commute (as assumed in Lemma 2.1), one can show that the additional assumption (3.21) is automatically satisfied under the assumptions of Theorem 3.1. For instance, from (2.1), one derives, for all integers $k \geq 1$, that $\sqrt{A}\, x^{(k)}(t) = \sqrt{A}^{-1}\big(\frac{\mathrm{d}^k}{\mathrm{d}t^k}g(x(t)) - x^{(k+2)}(t)\big)$, which are bounded.

4. Implementation

In this section, we present an efficient implementation of our proposed approach for the simulation of the system of coupled oscillators. First, we discuss on the computation of the matrix square root $\sqrt{A}$ needed for the new formulation (2.2). Then, we briefly review some state-of-the-art algorithms for implementing exponential Rosenbrock methods and introduce a new routine (proposed very recently in [3]) for achieving more efficiency for the proposed schemes pexprb43(c_2, c_3) and exprb42. Finally, we summarize the whole procedure by an algorithm.

4.1. Computation of the matrix square root $\sqrt{A}$

Let us denote $\Omega = \sqrt{A}$. For the computation of $\sqrt{A}$ used in (2.2), we use the Schur decomposition for moderate systems. For large systems, the Newton square root iteration (see [41]) can be employed 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 $\Omega^{-1} = \mathbf{S}^{-1}\mathbf{S}^{-\top}$, where $\mathbf{S}$ is an upper triangular matrix with real and positive diagonal entries. For more details, we refer to [4,41].

With Ω in hand, we compute the Jacobian J_n as in (3.5) and $F(u)$, $G_n(u)$ as in (3.4). As the next step, we discuss the implementation of the exponential Rosenbrock schemes.

4.2. Implementation of exponential rosenbrock schemes

In view of the schemes in Section 3, one can see that each stage requires computing a linear combination of φ-functions acting on certain vectors $v_0, \ldots, v_p$

$$\varphi_0(M)v_0 + \varphi_1(M)v_1 + \varphi_2(M)v_2 + \cdots + \varphi_p(M)v_p, \tag{4.1}$$

where the matrix M here could be hJ_n or $c_i hJ_n$. Starting from an early contribution by Hochbruck and Lubich [29], where the action of a matrix exponential on some vector is efficiently computed by using Krylov subspace methods, many more efficient techniques have been proposed. A large portion of these developments is concerned with computing the expression (4.1). For example, we mention some of the state-of-the-art algorithms: expmv proposed by Al-Mohy and Higham in [42] (using a truncated standard Taylor series expansion), phipm proposed by Niessen and Wright in [43] (using adaptive Krylov subspace methods), and expleja proposed by Caliari et al. in [44,45] (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 (4.1) using only one matrix function (see Section 4.2.1). 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 adaptively for improving efficiency.

Recently, the phipm routine was modified by Gaudreault and Pudykiewicz in [2] (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 [2] that this algorithm reduces computational time significantly compared to phipm when integrating the shallow water equations on the sphere.

Very recently, the first author [3] 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 [43]) for evaluating (4.1) and introduce some new features of the new routine phipm_simul_iom2.

4.2.1. *Time-stepping method for computing linear combinations of φ-functions*

By using the variation-of-constants formula (3.3) and the φ- functions' formula (3.13), it was observed that (see, e.g, [43, Lemma 2.1]) the expression (4.1) is the exact solution of the following linear ODE

$$u'(t) = Mu(t) + v_1 + tv_2 + \cdots + \frac{t^{p-1}}{(p-1)!}v_p, \;\; u(0) = v_0, \tag{4.2}$$

at $t = 1$, i.e. $u(1)$. 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 $u(t_k)$:

$$u(t_{k+1}) = \varphi_0(\tau_k M)u(t_k) + \sum_{i=1}^{p} \tau_k^i \varphi_i(\tau_k M) \sum_{j=0}^{p-i} \frac{t_k^j}{j!} v_{i+j}. \tag{4.3}$$

Using the recursion relation (3.14), (4.3) can be simplified as

$$u(t_{k+1}) = \tau_k^p \varphi_p(\tau_k M) w_p + \sum_{j=0}^{p-i} \frac{t_k^j}{j!} w_j, \tag{4.4}$$

where the vectors w_j satisfy the recurrence relation

$$w_0 = u(t_k), \;\; w_j = Mw_{j-1} + \sum_{\ell=0}^{p-j} \frac{t_k^\ell}{\ell!} v_{j+\ell}, \;\; j = 1, \ldots, p. \tag{4.5}$$

Eq. (4.4) implies that evaluating $u(t_K) = u(1)$, i.e., the expression (4.1) 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 [43] 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

The resulting routine phipm_simul_iom2 optimizes computational aspects of phipm/IOM2 corresponding to the following changes:

(i) Unlike (4.1), 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:

$$w_k = \sum_{l=1}^{p} \varphi_l(c_k M) v_l, \quad k = 2, \ldots, s, \tag{4.6}$$

where now the node values $c_2, \ldots, c_s$ are scaling factors used for each v_k output. To optimize this evaluation, phipm_simul_iom2 computes all w_k outputs in (4.6) simultaneously, instead of computing only one at a time. This is accomplished by first requiring that the entire array $c_2, \ldots, c_s$ as an input to the function. Within the substepping process (4.3), 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.

(ii) In view of the higher-order exponential Rosenbrock schemes (see also from Section 3.4), it is realized that they usually use a subset of the φ_l functions. Therefore, multiple vectors in (4.6) will be zero. In this case, phipm_simul_iom2 will check whether $w_{j-1} \neq 0$ (within the recursion (4.5)) before computing the matrix–vector product $M\, w_{j-1}$. While matrix–vector products require $\mathcal{O}(n^2)$ work, checking $u \neq 0$ requires only $\mathcal{O}(n)$. This can result in significant savings for large n.

4.2.3. *Implementation of* `pexprb43(c2, c3)` *and* `exprb42`

Taking a closer look at the structures of schemes `pexprb43(c2, c3)` and `exprb42`, we now make use of `phipm_simul_iom2` for their implementations. For simplicity, let us denote $M = hJ_n$ and $v = hF(u_n)$.

As discussed in Section 3.4, `pexprb43(c2, c3)` can be implemented like a 2-stage method by evaluating the following 2 evaluations, corresponding to 2 calls to `phipm_simul_iom2`:

(i) Compute both terms $y_1 = \varphi_1(c_2M)v$ and $z_1 = \varphi_1(c_3M)v$ in parallel or simultaneously to obtain $U_{n2} = u_n + c_2y_1$ and $U_{n3} = u_n + c_3z_1$.
(ii) Compute $w = \varphi_1(M)v + \varphi_3(M)v_3 + \varphi_4(M)v_4$ (i.e. $v_0 = v_2 = 0$, $v_1 = v$) with $v_3 = h(\frac{2c_3}{c_2^2(c_3-c_2)}D_{n2} + \frac{2c_2}{c_3^2(c_2-c_3)}D_{n3})$, $v_4 = h(\frac{-6}{c_2^2(c_3-c_2)}D_{n2} - \frac{6}{c_3^2(c_2-c_3)}D_{n3})$ to obtain $u_{n+1} = u_n + w$.

In view of (3.20), one needs two calls to `phipm_simul_iom2` for the implementation of `exprb42`:

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

For our numerical experiments below, we choose $c_2 = \frac{1}{3}$ and $c_3 = \frac{3}{4}$. It turns out that `pexprb43`$(\frac{1}{3}, \frac{3}{4})$ and `exprb42` are the most efficient schemes for our application models in visual computing presented in Section 5.

4.3. *The overall algorithm*

We now summarize our implementation by Algorithm 4.1.

Algorithm 4.1 Advanced time integration algorithm for simulating (1.1)

- **Input:** $A \in \mathbb{R}^{3N\times 3N}$; $g(x), x_0, v_0 \in \mathbb{R}^{3N}$; $[t_0, T]$; and step size h.
- **Initialization:**
  1. Compute $\sqrt{A}$ as in described in Section 4.1.
  2. As in (2.2), set $u_0 = [\sqrt{A}\, x_0, v_0]^T$ and compute $F(u_0) = \mathscr{A}u_0 + G(u_0)$, where $\mathscr{A}$ and $G(u)$ are given by (2.3).
- **Time integration:** Set $n = 0$; $u_n := u_0$.
  While $t_n < t_{\text{end}} = T$
  1. Compute the solution at the next time step u_{n+1} using `pexprb43`$(\frac{1}{3}, \frac{3}{4})$ or `exprb42` as described in Section 4.2.3 (with the help of `phipm_simul_iom2`).
  2. Update $u_n := u_{n+1}$.
     $t_n := t_n + h$.
     $n := n + 1$.
- **Output** Approximate values at time $t = t_n$ for
  - positions x_n by solving the linear system $\sqrt{A}\, x_n = u_n(1 : 3N)$.
  - velocities $x'_n = v_n = u_n(3N + 1 : 6N)$.

5. Numerical examples

In this section, we present a broad spectrum of numerical examples to demonstrate the accuracy and efficiency of the new approach (i.e. the new formulation (2.2)–(2.3) combined with the new proposed parallel stages scheme `pexprb43`$(\frac{1}{3}, \frac{3}{4})$ and the selected superconvergent scheme `exprb42`) for simulating large stiff systems of nonlinear coupled oscillators.

In the first part, we consider a nonlinear Fermi–Pasta–Ulam–Tsingou (FPUT) model including stiff springs (which is a simple model for mechanics simulations similarly to Hairer et al. [46]) in order to verify numerically the convergence rate as well as to show the efficiency of the proposed schemes even for this small system.

In the second part, we test our proposed approach on various application models focusing 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 for our test can be found in Table 2. Furthermore, our simulation includes important aspects like elastic collisions and nonelastic deformations. The presented exponential Rosenbrock methods

Table 2
Overview of the test cases used for application models in visual computing. Their complexity N (i.e. the number of the resulting equations of motion), the simulated time, and respective running times for the exponential Rosenbrock methods exprb42 and pexprb43($\frac{1}{3}$, $\frac{3}{4}$), the implicit–explicit variational integrator (V-IMEX), the standard RK4 method, and the BDF-1 integrator are shown. We note that different integrators use different values of h.

Model	N	Simul. Time	exprb42	pexprb43($\frac{1}{3}$, $\frac{3}{4}$)	V-IMEX	RK4	BDF1
Coil spring	24k	60 s	55 s	47 s	12 min	46 min	62 min
Dragon	450k	10 s	272 s	243 s	72 min	4 h	5 h
Brushing	90k	15 s	52 s	51 s	11 min	53 min	72 min
Crash (moderate)	360k	2 s	44 s	44 s	9 min	47 min	58 min
Crash (fast)	360k	2 s	47 s	46 s	9 min	47 min	59 min

are evaluated against classical and state-of-the-art methods used in visual computing, in particular against the implicit–explicit variational (IMEX) integrator (cf. [47,48]), the standard fourth-order Runge–Kutta method (see [49,50]), and the implicit BDF-1 integrator (see [51]). 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. A nonlinear FPUT model

Consider a nonlinear FPUT model with m stiff springs, in which the motion is described by a Hamiltonian system with the total energy

$$H(x', x) = \frac{1}{2}\|x'\|^2 + \frac{1}{2}\|\sqrt{A}x\|^2 + U(x), \tag{5.1}$$

where

$$A = \begin{bmatrix} I_m & \mathbf{0} \\ \mathbf{0} & \omega^2 I_m \end{bmatrix}_{2m\times 2m} \tag{5.2}$$

and

$$U(x) = \frac{1}{4}\Big[(x_{0,1} - x_{1,1})^4 + \sum_{i=1}^{m}(x_{0,i+1} - x_{1,i+1} - x_{0,i} - x_{1,i})^4 + (x_{0,m} + x_{1,m})^4\Big] \tag{5.3}$$

with $x_{0,i}$ represents for positions of the ith stiff spring.

Clearly, the Hamiltonian (5.1) can be formulated as a second-order system of DEs as (2.1) with $g(x) = -\nabla U(x)$. One can also check that this system has A with

$$\sqrt{A} = \begin{bmatrix} I_m & \mathbf{0} \\ \mathbf{0} & \omega I_m \end{bmatrix}_{2m\times 2m}$$

and $g(x)$ fulfill the assumptions of Lemma 2.1 and Theorem 3.1. For our experiments below we choose the same initial positions and velocities as in Hairer et al. [46] ($m = 3, x_{0,1} = x'_{0,1} = x'_{1,1} = 1, x_{1,1} = \omega^{-1}$ and zero for the remaining initial values) with the exception that the value of frequency ω is taken even two times larger, namely, $\omega = 100$. This yields a stiff system (with $\|A\|_\infty = 10^4$).

We then integrate the resulting system on the time interval $[0, 100]$ by using methods exprb2, pexprb43($\frac{1}{3}$, $\frac{3}{4}$), exprb42, and the classical method RK4. The reference solution is computed by using the standard stiff solver ode15s with $TOL = 10^{-14}$. The errors at the final time $T = 100$ are measured in the maximum norm.

Verification of the order of convergence: In Fig. 1, we plot orders for all the considered schemes. The diagrams clearly show that the all considered schemes fully achieve their orders. This is a perfect agreement with Theorem 3.1 regarding the convergence of pexprb43($\frac{1}{3}$, $\frac{3}{4}$) and exprb42. We note, however, while the set of different step sizes $h = 0.02, 0.01, 0.005, 0.0025, 0.00125$ is used for exponential schemes (exprb2, pexprb43($\frac{1}{3}$, $\frac{3}{4}$), and exprb42), RK4 requires a much smaller set of step sizes (40 times smaller—see the right diagram) in order to obtain about the same level of accuracy (when using the "large" set of time steps above, RK4 either failed to converge or did not show the right order of convergence due to the stability issue caused by the stiffness). This confirms the benefit of using the exponential Rosenbrock-based schemes which allow to take much larger time steps. From the left diagram one can see that pexprb43($\frac{1}{3}$, $\frac{3}{4}$) and exprb42 give about the same accuracies for this model.

Demonstration of the efficiency: In Fig. 2, we display the efficiency plot of the fourth-order schemes pexprb43($\frac{1}{3}$, $\frac{3}{4}$), exprb42, and RK4. Given about the same level of accuracy, the precision diagram clearly shows that both pexprb43($\frac{1}{3}$, $\frac{3}{4}$)

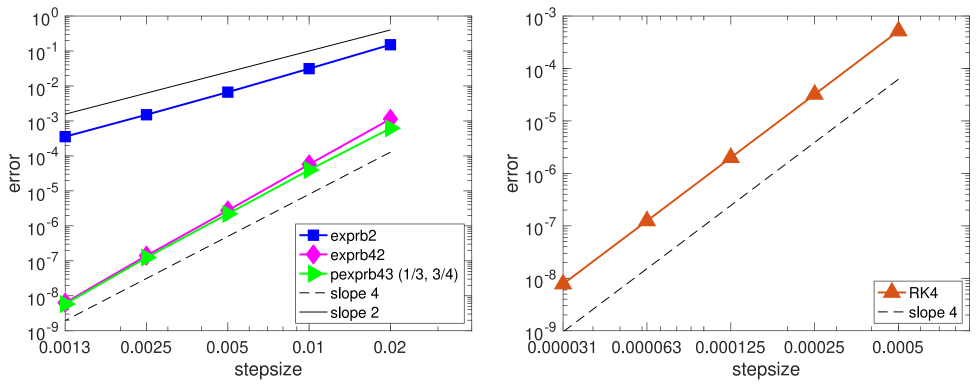

Fig. 1. Order plots of exprb2, pexprb43($\frac{1}{3}$, $\frac{3}{4}$), exprb42, and RK4 when applied to the nonlinear FPUT model ($m = 3$). For confirmation, straight lines with slopes 2 and 4 are added.

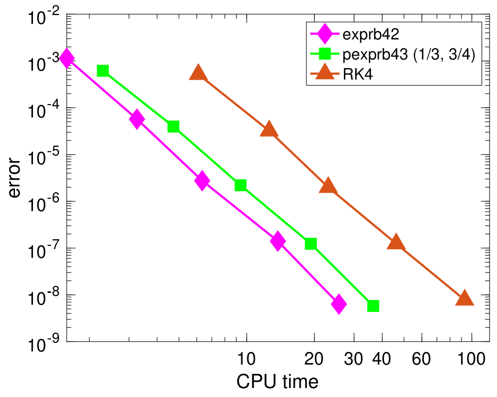

Fig. 2. Total CPU times (in second) of the fourth-order schemes pexprb43($\frac{1}{3}$, $\frac{3}{4}$), exprb42, and RK4 when applied to the nonlinear FPUT model ($m = 3$).

and exprb42 are much faster than RK4 even for this stiff system of small size (In fact, these two schemes achieve speedup factors of around 3 and 4, respectively). It is thus anticipated that, for larger stiff systems of higher frequency, these exponential schemes can significantly increase more the speedup factor compared to RK4 and other standard methods (see the results for other application models in Table 2).

Performance on conservation of the total energy: With the given initial data, one can see from (5.1) that the exact total energy is $H = H(x'(0), x(0)) = \frac{1}{2}.2 + \frac{1}{2}.2 + 0.500300005 = 2.500300005$. In Figs. 3 and 4, we perform evaluations on the conservation of the total energy for the considered schemes. In the left diagram of Fig. 3, the second-order (exprb2) and the high-order exponential Rosenbrock schemes (4th-order, pexprb43($\frac{1}{3}$, $\frac{3}{4}$), exprb42) are compared. In the right diagram of Fig. 3, the classical and standard fourth-order scheme RK4 is compared with the two 4th-order exponential schemes.

As seen, over the considered time interval [0, 100], both pexprb43($\frac{1}{3}$, $\frac{3}{4}$) and exprb42 preserve the total energy quite well and outperform exprb2 (which uses the same time stepsize $h = 0.01$) and RK4. It should be mentioned that RK4 totally fails to conserve the total energy H at this "large" stepsize. However, it does improve the conservation of H when much smaller time stepsizes are employed. This can be also confirmed by viewing the right diagram of Fig. 4. In particular, one can see that RK4 has to use $h = 0.00025$ (which is 40 times smaller than the time stepsize $h = 0.01$ used for pexprb43($\frac{1}{3}$, $\frac{3}{4}$) and exprb42) in order to get about the same quality of conservation of H. Similarly, from the left diagram of Fig. 4, we see that exprb2 also requires a much smaller stepsize to maintain a good level of preserving H.

5.2. Some application models in visual computing

5.2.1. Simulation of deformable bodies

In order to illustrate the accurate energy preservation of the presented exponential Rosenbrock schemes, we set up an undamped scene of an oscillating coil spring, which is modeled as a deformable body composed of tetrahedra, in particular of 8 000 vertices corresponding to $N = 24\,000$ equations of motion, which are derived from a system of coupled

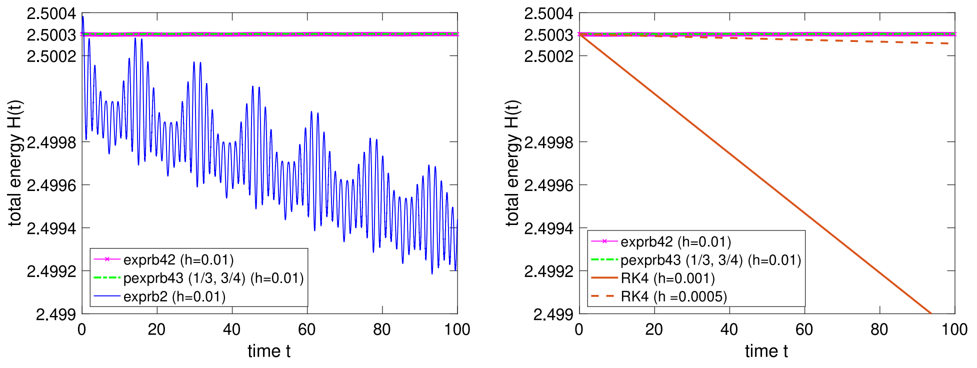

Fig. 3. Conservation of the total energy for the considered schemes when applied to the nonlinear FPUT model ($m = 3$).

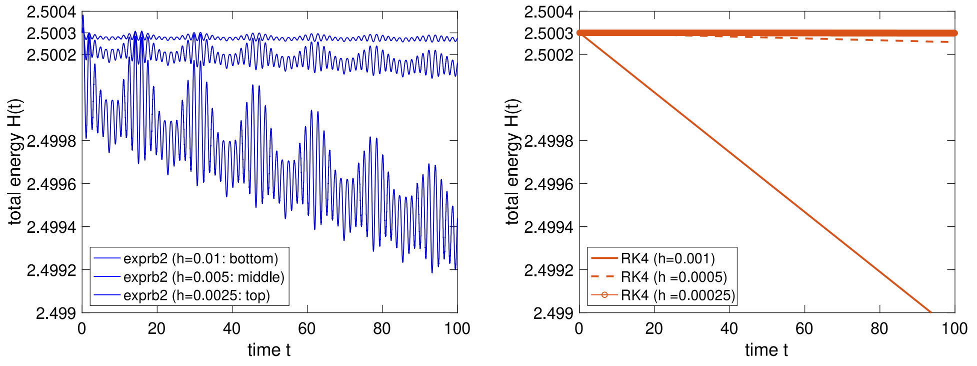

Fig. 4. Conservation of the total energy for exprb2 (left) and RK4 (right) using different time step sizes when applied to the nonlinear FPUT model ($m = 3$).

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. 5. 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 schemes exprb42 and pexprb43($\frac{1}{3}$, $\frac{3}{4}$) we observe that the discrete energy is only slightly oscillating around the real energy without increasing oscillations over time. In contrast, the standard RK4 method and respectively the BDF-1 integrator generate significant numerical viscosity leading to a loss of energy around 22% respectively 40% after 60 s of simulated time. To demonstrate that such a speedup is also scalable, we additionally simulate the elastic deformation of a dragon which is composed of tetrahedra, in particular of 150 000 vertices corresponding to $N = 450\,000$ equations of motion. A uniform spring stiffness of $k = 10^6$ is applied. The deformation of the dragon is shown Fig. 6.

The exponential Rosenbrock schemes exprb42 and pexprb43($\frac{1}{3}$, $\frac{3}{4}$) 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($\frac{1}{3}$, $\frac{3}{4}$).[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($\frac{1}{3}$, $\frac{3}{4}$) compared to the second best method, the variational IMEX integrator as illustrated in Table 2. Compared to the other methods, the exponential Rosenbrock methods allow for accurate simulations in real-time.

5.2.2. *Simulation of fibers including elastic collisions*

Fibers are canonical examples for complex interacting systems. According to the work of Michels et al. (see [30]), 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 1 500 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

[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.

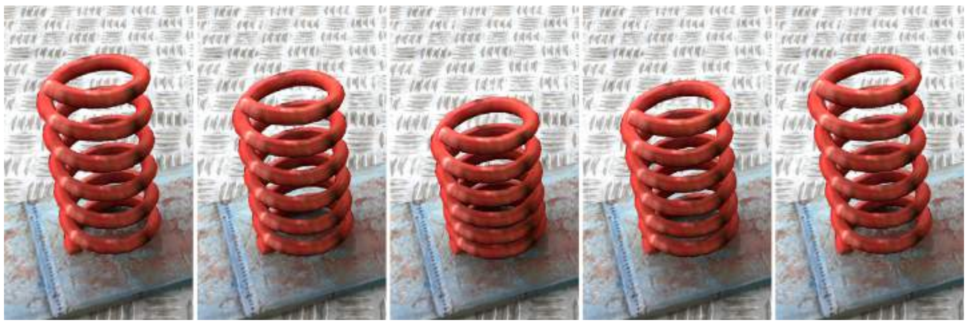

Fig. 5. Simulation of an oscillating coil spring.

Fig. 6. Elastic deformation of a dragon model.

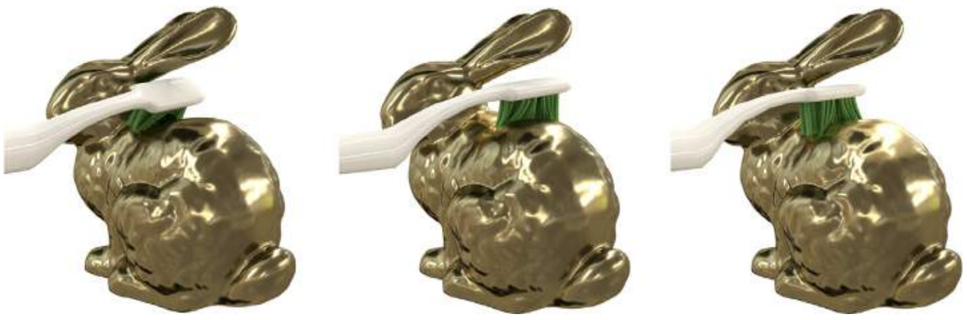

Fig. 7. Simulation of a brush cleaning a bronze-colored paperweight.

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.

We simulate 15 s of a toothbrush cleaning a paperweight illustrated in Fig. 7. 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.

5.2.3. 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. 8. 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 methods, the whole simulation can be carried out at interactive frame rates. Such an efficient computation cannot be achieved with established methods as illustrated in Table 2.

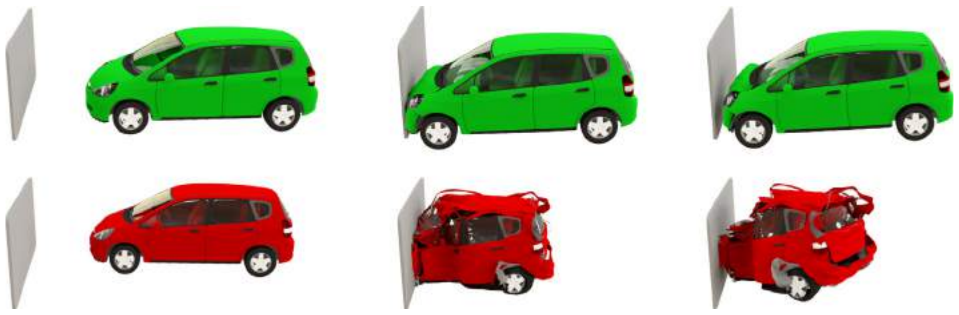

Fig. 8. Simulations of two frontal nonelastic crash scenarios: a car with moderate velocity (top) and high velocity (bottom).

6. Conclusion

We have derived a novel time integration technique for the simulation of large systems of nonlinear coupled oscillators used in visual computing. In particular, a new formulation of these systems is introduced and a general family of efficient fourth-order exponential Rosenbrock schemes `pexprb43`(c_2, c_3) (new scheme) and `exprb42` (superconvergent scheme) are proposed to apply to the resulting system. Furthermore, an efficient implementation of this approach is addressed. In order to study the efficiency and accuracy of this new approach, 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 authors would like to thank the anonymous referees for their valuable comments and useful suggestions.

The first author gratefully acknowledges the financial support of the National Science Foundation, USA under grant NSF DMS–2012022. The second author acknowledges the financial support of KAUST, Saudi Arabia baseline funding.

We also thank Mississippi State University's Center for Computational Science (CCS) for providing computing resources at the High Performance Computing Collaboratory (HPCC). Moreover, the use of the resources of KAUST's Supercomputing Laboratory is gratefully acknowledged.

Appendix. Analytical framework and convergence results for exponential Rosenbrock methods [16]

Consider stiff problems of the general form (1.2) written as abstract time-dependent PDEs (or resulted upon their spatial discretizations) in some Banach space X (we note that in the current work $X = \mathbb{R}^{3N}$). Many of these systems naturally exhibit a semilinear form

$$u'(t) = F(u(t)) = Au(t) + g(u(t)), \qquad u(t_0) = u_0. \tag{A.1}$$

For the error analysis of exponential Rosenbrock methods for (A.1), an abstract framework of strongly continuous semigroups in X with norm $\|\cdot\|$ is usually employed. Let

$$J = J(u) = F'(u) = A + g'(u)$$

be the Fréchet derivative of F. We suppose that F (and hence g) is differentiable in a neighborhood of the exact solution of (A.1). The following assumptions are made.

Assumption 1. The linear operator A is the generator of a strongly continuous semigroup e^{tA} in X.

Assumption 2. Suppose that (A.1) possesses a sufficiently smooth solution $u : [0, T] \to X$, with derivatives in X and that the nonlinearity $g : X \to X$ is sufficiently often Fréchet differentiable in a strip along the exact solution. All occurring derivatives are assumed to be bounded.

Using a standard perturbation result (see [52, Chap. 3.1]), Assumption 1 implies that the Jacobian (A.2) generates a strongly continuous semigroup e^{tJ}, and that there exist constants C and ω such that

$$\|\mathrm{e}^{tJ}\|_{X\leftarrow X} \le C\mathrm{e}^{\omega t}, \quad t \ge 0$$

holds uniformly in a neighborhood of the exact solution. This further implies that $\varphi_k(hJ)$ and thus the coefficients $a_{ij}(hJ)$ and $b_i(hJ)$ of the exponential Rosenbrock methods are bounded operators. Moreover, under Assumption 2, one can verify that $g_n(u) = F(u) - J_n u$ is also sufficiently often Fréchet differentiable as well as satisfies the Lipschitz condition in a strip along the exact solution.

These are crucial for the convergence proof of exponential Rosenbrock methods for stiff problems. Below we recall such a result from [16, Thm. 4.1].

Theorem A.1 (*Convergence Results for Constant Step Sizes*)**.** *Let the initial value problem (A.1) satisfy Assumptions 1 and 2. Consider for its numerical solution an explicit exponential Rosenbrock method (3.11) that fulfills the order conditions of Table 1 up to order p for some $3 \le p \le 5$. Then, under the stability bound (3.24), the method converges with order p. In particular, the numerical solution satisfies the error bound*

$$\|u_n - u(t_n)\| \le Ch^p \tag{A.2}$$

uniformly on $t_0 \le t_n \le T$. Here, the constant C is independent of n and h.

We further note that the global error constant C in (A.2) depends on the interval width $(T - t_0)$ and values that are uniformly bounded by Assumptions 1 and 2, but that is independently of $\|J_n\|$, i.e., the stiffness of the problem.

References

[1] C. Clancy, J. Pudykiewicz, On the use of exponential time integration methods in atmospheric models, Tellus A 65 (2013).

[2] 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 (2016) 827–848.

[3] V.T. Luan, J.A. Pudykiewicz, D.R. Reynolds, Further development of efficient and accurate time integration schemes for meteorological models, J. Comput. Phys. 376 (2019) 817–837.

[4] D.L. Michels, V.T. Luan, M. Tokman, A stiffly accurate integrator for elastodynamic problems, ACM Trans. Graph. 36 (4) (2017) 116.

[5] Y.J. Chen, U. Ascher, D. Pai, Exponential Rosenbrock–Euler integrators for elastodynamic simulation, IEEE Trans. Vis. Comput. Graphics (2017).

[6] D.L. Michels, M. Desbrun, A semi-analytical approach to molecular dynamics, J. Comput. Phys. 303 (2015) 336–354.

[7] K. Pieper, K.C. Sockwell, M. Gunzburger, Exponential time differencing for mimetic multilayer ocean models, J. Comput. Phys. 398 (2019) 817–837.

[8] M.A. Gondal, Exponential Rosenbrock integrators for option pricing, J. Comput. Appl. Math. 234 (4) (2010) 1153–1160.

[9] 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 (2013) 250–262.

[10] 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: Electromagnetic Compatibility and Signal Integrity, 2015 IEEE Symposium on, IEEE, 2015, pp. 248–252.

[11] U. Ascher, S. Ruuth, B. Wetton, Implicit-explicit methods for time-dependent PDEs, SIAM J. Numer. Anal. 32 (3) (1997) 797–823.

[12] C. Curtiss, J.O. Hirschfelder, Integration of stiff equations, Proc. Natl. Acad. Sci. 38 (3) (1952) 235–243.

[13] C. Gear, Numerical Initial Value Problems in Ordinary Differential Equations, Prentice–Hall, Englewood Cliffs, NJ, 1971.

[14] M. Hochbruck, A. Ostermann, Explicit integrators of Rosenbrock-type, Oberwolfach Rep. 3 3 (2006) 1107–1110.

[15] M. Hochbruck, A. Ostermann, J. Schweitzer, Exponential Rosenbrock-type methods, SIAM J. Numer. Anal. 47 (2009) 786–803.

[16] V.T. Luan, A. Ostermann, Exponential Rosenbrock methods of order five–construction, analysis and numerical comparisons, J. Comput. Appl. Math. 255 (2014) 417–431.

[17] V.T. Luan, A. Ostermann, Parallel exponential Rosenbrock methods, Comput. Math. Appl. 71 (2016) 1137–1150.

[18] V.T. Luan, A. Ostermann, Exponential B-series: The stiff case, SIAM J. Numer. Anal. 51 (2013) 3431–3445.

[19] S. Geiger, G. Lord, A. Tambue, Exponential time integrators for stochastic partial differential equations in 3D reservoir simulation, Comput. Geosci. 16 (2) (2012) 323–334.

[20] G. Chen, D.L. Russell, A mathematical model for linear elastic systems with structural damping., Quart. Appl. Math. 39 (4) (1982) 433–454.

[21] D.L. Michels, G.A. Sobottka, A.G. Weber, Exponential integrators for stiff elastodynamic problems, ACM Trans. Graph. 33 (1) (2014) 7:1–7:20.

[22] D. Terzopoulos, J. Platt, A. Barr, K. Fleischer, Elastically deformable models, in: ACM Transactions on Graphics, Vol. 21, 1987, pp. 205–214.

[23] D. Baraff, A. Witkin, Large steps in cloth simulation, in: ACM Transactions on Graphics, SIGGRAPH '98, ACM, New York, NY, USA, 1998, pp. 43–54.

[24] 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, 2000, pp. 137–151.

[25] M. Hauth, O. Etzmuss, A high performance solver for the animation of deformable objects using advanced numerical methods, Comput. Graph. Forum 20 (2001) 319–328.

[26] R. Goldenthal, D. Harmon, R. Fattal, M. Bercovier, E. Grinspun, Efficient simulation of inextensible cloth, in: ACM Transactions on Graphics, SIGGRAPH '07, 2007.

[27] M. Bergou, M. Wardetzky, S. Robinson, B. Audoly, E. Grinspun, Discrete elastic rods, ACM Trans. Graph. 27 (3) (2008) 63:1–63:12.

[28] M. Hochbruck, A. Ostermann, Exponential integrators, Acta Numer. 19 (2010) 209–286.

[29] M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 34 (1997) 1911–1925.

[30] 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 (2015) 136–146.

[31] D.L. Michels, J.P.T. Mueller, Discrete computational mechanics for stiff phenomena, in: SIGGRAPH ASIA 2016 Courses, 2016, pp. 13:1–13:9.

[32] V.T. Luan, Fourth-order two-stage explicit exponential integrators for time-dependent PDEs, Appl. Numer. Math. 112 (2017) 91–103.

[33] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, Springer, New York, 1996.

[34] D.A. Pope, An exponential method of numerical integration of ordinary differential equations, Commun. ACM 6 (1963) 491–493.

[35] M. Hochbruck, A. Ostermann, Explicit exponential Runge–Kutta methods for semilinear parabolic problems, SIAM J. Numer. Anal. 43 (2005) 1069–1090.

[36] S.M. Cox, P.C. Matthews, Exponential time differencing for stiff systems, J. Comput. Phys. 176 (2) (2002) 430–455.

[37] S. Krogstad, Generalized integrating factor methods for stiff PDEs, J. Comput. Phys. 203 (1) (2005) 72–88.

[38] V.T. Luan, A. Ostermann, Explicit exponential Runge–Kutta methods of high order for parabolic problems, J. Comput. Appl. Math. 256 (2014) 168–179.
[39] V.T. Luan, Efficient exponential Runge–Kutta methods of high order: construction and implementation, BIT Numer. Math. (2021) in press.
[40] E. Emmrich, Stability and error of the variable two-step BDF for semilinear parabolic problems, J. Appl. Math. Comput. 19 (1) (2005) 33–55.
[41] N.J. Higham, Functions of Matrices : Theory and Computation, SIAM, 2008.
[42] 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 (2011) 488–511.
[43] 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 (Article 22) (2012) 3.
[44] M. Caliari, A. Ostermann, Implementation of exponential Rosenbrock-type integrators, Appl. Numer. Math. 59 (3–4) (2009) 568–581.
[45] 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) (2016) A1639–A1661.
[46] E. Hairer, C. Lubich, G. Wanner, Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations, Vol. 31, Springer Science & Business Media, 2006.
[47] A. Stern, M. Desbrun, Discrete geometric mechanics for variational time integrators, in: SIGGRAPH 2006 Courses, 2006, pp. 75–80.
[48] A. Stern, E. Grinspun, Implicit-explicit variational integration of highly oscillatory problems, Multiscale Model. Simul. 7 (2009) 1779–1794.
[49] C.D. Runge, Über die numerische Auflösung von differentialgleichungen, Math. Ann. 46 (1895) 167–178.
[50] M.W. Kutta, Beitrag zur näherungsweisen integration totaler differentialgleichungen, Zeit. Math. Phys. 46 (1901) 435–453.
[51] C.F. Curtiss, J.O. Hirschfelder, Integration of stiff equations, Proc. Natl. Acad. Sci. USA 38 (3) (1952) 235–243.
[52] A. Pazy, Semigroups of Linear Operators and Applications to Partial Differential Equations, Springer, New York, 1983.