A Stiffly Accurate Integrator for Elastodynamic Problems

DOMINIK L. MICHELS, KAUST and Stanford University
VU THAI LUAN, UC Merced and VAST
MAYYA TOKMAN, UC Merced

Fig. 1. Visualization of the dynamical simulation of human hair during a woman's head shake carried out with our stiffly accurate integrator.

We present a new integration algorithm for the accurate and efficient solution of stiff elastodynamic problems governed by the second-order ordinary differential equations of structural mechanics. Current methods have the shortcoming that their performance is highly dependent on the numerical stiffness of the underlying system that often leads to unrealistic behavior or a significant loss of efficiency. To overcome these limitations, we present a new integration method which is based on a mathematical reformulation of the underlying differential equations, an exponential treatment of the full nonlinear forcing operator as opposed to more standard partially implicit or exponential approaches, and the utilization of the concept of stiff accuracy which ensures that the efficiency of the simulations is significantly less sensitive to increased stiffness. As a consequence, we are able to tremendously accelerate the simulation of stiff systems compared to established integrators and significantly increase the overall accuracy. The advantageous behavior of this approach is demonstrated on a broad spectrum of complex examples like deformable bodies, textiles, bristles, and human hair. Our easily parallelizable integrator enables more complex and realistic models to be explored in visual computing without compromising efficiency.

CCS Concepts: • **Mathematics of computing → Ordinary differential equations**; *Solvers*; • **Applied computing → Physics**;

Additional Key Words and Phrases: Accurate simulation, efficient simulation, elastodynamic problems, exponential treatment, stiff accuracy, stiff problems.

ACM Reference format:
Dominik L. Michels, Vu Thai Luan, and Mayya Tokman. 2017. A Stiffly Accurate Integrator for Elastodynamic Problems. *ACM Trans. Graph.* 36, 4, Article 116 (July 2017), 14 pages.
DOI: 10.1145/3072959.3073706

This work has been partially supported by the National Science Foundation of the United States or America (grant 1419105), the King Abdullah University of Science and Technology (KAUST baseline funding), the Max Planck Center for Visual Computing and Communication (MPC-VCC) funded by Stanford University and the Federal Ministry of Education and Research of the Federal Republic of Germany (grants FKZ-01IMC01 and FKZ-01IM10001).
Author's addresses: D. Michels, King Abdullah University of Science and Technology, Visual Computing Center and Leland Stanford Junior University, Computer Science Department; V.T. Luan, University of California, Merced, School of Natural Sciences and Vietnam Academy of Science and Technology, Institute of Information Technology; M. Tokman, University of California, Merced, School of Natural Sciences.

1 INTRODUCTION

For most problems, simulating dynamics more accurately invariably means sacrificing computational efficiency. This is especially the case when the underlying differential equations are "stiff". Such systems of equations are characterized by a wide range of time scales present in their evolution. Stiffness arises when the time scale of interest in the dynamics is much slower than the fastest modes of the system. Stiff equations are ubiquitous in a wide range of fields including electromagnetics, fluid dynamics, acoustics, electrodynamics, molecular modeling, computerized tomography and imaging, plasma transport, and celestial mechanics [Engquist et al. 2009]. Prominent examples in visual computing include the dynamics of cloth, fibers, fluids, or solids, and their interaction with each other.

The numerical time integration of stiff systems of differential equations is one of the central problems in numerical analysis. The history of this branch of numerical analysis has been dominated by two classes of time integrators: explicit and implicit. Both types of integrators allow advancing the numerical solution along a discretized time interval, but the numerical properties of these two classes are fundamentally different. Explicit methods require the least amount of computations per time step but suffer severe stability restrictions that limit the allowable size of the time step. Implicit methods possess better stability properties and allow for accurate integration with much larger time steps, but the increase in time step size comes at the expense of significantly more computations required in each time iteration. As the stiffness of the problem grows, integrating equations explicitly over a long period of time becomes impractical and animators turn to implicit methods. However, implicit schemes are not immune to the increase in stiffness and the amount of computation required per time step grows correspondingly.

Recently, exponential methods emerged as a viable alternative to implicit schemes for a number of stiff problems. A range of exponential integrators have been developed including stiffly accurate methods that are particularly suited for the integration of stiff systems. However, ideas of stiffly accurate exponential integration have not been extensively explored in the context of visual computing applications such as the simulation of elastodynamic systems.

In this work we propose a new time integration approach to simulate elastodynamic systems. Our new time integrator for these types of equations is comprised of three key ideas: a new mathematical formulation of the problem, an exponential treatment of the full nonlinear forcing operator, and the application of stiffly accurate exponential methods optimized for adaptive Krylov projection algorithms.

Using a broad spectrum of applications we demonstrate that the new time integrator offers significant computational advantages compared to current state-of-the-art techniques for simulating elastodynamic systems in visual computing.

2 RELATED WORK

In the realm of visual computing certain methods have been established over the past decades to handle numerical difficulties associated with stiff scenarios. Most algorithms are based on implicit integration schemes since they usually come with artificial dissipation which limits numerical instabilities (like energy blow-up effects) and can therefore be used with significantly larger time step sizes compared to simple explicit schemes [Hairer and Wanner 2004]. However, excessive numerical dissipation in an implicit method can severely impact the accuracy of the solution as well as result in a violation of key defining properties of the underlying system as for example the fundamental physical law of energy conservation. A number of improvements to the integration of stiff systems arising in visual computing, particularly elastodynamic models, have been introduced over the past thirty years, for example by Terzopoulos et al. [1987] (interacting deformable bodies), Baraff and Witkin [1998], Hauth and Etzmuss [2001], Goldenthal et al. [2014] (cloth simulation), and Bergou et al. [2008] (simulating elastic rods). Variational integrators were used [Stern and Desbrun 2006] to improve geometric properties of the solution.

The work on time integration in visual computing most closely related to our approach is the use of Gautschi-type integrators [Deuflhard 1979; Gautschi 1961; Hairer and Lubich 1999; Michels and Desbrun 2015] to improve the accuracy and efficiency of the simulation of elastodynamic systems [Michels and Mueller 2016; Michels et al. 2014]. Unlike Gautschi-type methods, where linear spring forces are advanced using trigonometric functions of the linear portion of the Jacobian, our approach involves a reformulation of the problem and the advancing of the full nonlinear forcing using exponential-like functions. It is important to note that most of the implicit integrators as well as Gautschi-type methods developed for problems in visual computing rely on partial implicitness, i.e. so-called implicit-explicit (IMEX) approaches [Ascher et al. 1995]. An IMEX method handles a portion of the forcing operator implicitly while the rest of the forcing operator is integrated explicitly [Eberhardt et al. 2000; Hauth and Etzmuss 2001]. In our method no partitioning of the forcing is used; instead, the full nonlinear Jacobian is used to integrate the motion forward.

Our new approach to time integration of elastodynamic systems is based on ideas from the field of exponential integration. Over the past several decades exponential methods emerged as alternatives to implicit methods in simulating large scale stiff systems. While the first exponential integrators date back to 1960's [Certaine 1960; Lawson 1967; Pope 1963], a broader interest and active development and application of these type of integrators are more recent. Renewed interest in exponential methods was spurred by advances in numerical linear algebra which enabled efficient computation of products of exponential-like functions of matrices with vectors. A number of exponential integrators have been proposed for stiff equations [Hochbruck and Ostermann 2006; Kassam and Trefethen 2005; Krogstad 2005; Luan and Ostermann 2014b; Tokman 2006, 2011]. In particular, the exponential propagation iterative methods of Runge-Kutta-type (EPIRK) framework [Tokman 2006, 2011] used in this work was developed to improve the efficiency of exponential integrators. We focus on stiffly accurate exponential integrators [Hochbruck and Ostermann 2005, 2006; Luan and Ostermann 2013, 2014a,b, 2016; Rainwater and Tokman 2016b] which introduce additional computational savings compared to standard techniques.

The new time integrator presented in this work offers significant computational advantages compared to previously proposed methods for elastodynamic systems. In the following sections, we describe the new method and demonstrate its performance on a number of visual computing problems.

3 TIME INTEGRATION ALGORITHM

Elastodynamic equations are routinely used in visual computing to model the motion of objects such as cloth, fibers, and deformable bodies. Our new time integration approach to simulate the dynamics of such systems is comprised of the following three key ideas: (i) a mathematical reformulation of the equations modeling elastodynamic systems which makes the problem more suitable for exponential integration, (ii) exponential treatment of the full nonlinear forcing operator as opposed to more standard partially implicit or exponential approaches, and (iii) the use of stiffly accurate exponential methods which ensures that the efficiency of the simulations is significantly less sensitive to increased stiffness.

The novelty of our time integration approach stems from both, the design of the algorithm and the application of the stiffly accurate exponential schemes to large scale complex highly oscillatory system such as the elastodynamic equations. Specifically, we construct an exponential integrator based on the evaluation of exponential-like functions of a full Jacobian matrix of the forcing operator. Our new method differs from previously proposed integrators in several important ways. We propose a reformulation of the problem (see Sec. 3.1) which improves the structural properties of the system and makes application of standard exponential integration using matrix exponential-like functions more efficient. The change of variables utilized in this reformulation has been used for modeling linear elastic systems [Chen and Russell 1982], and more recently, it was employed in numerical analysis for studying the error of the rational Krylov approximations to linear second order systems [Göckler and Grimm 2013] and for establishing connections between Gautschi-type and splitting methods [Buchholz et al. 2017]. We utilize this change of variables to improve the structure of the full Jacobian matrix of the full nonlinear forcing operator. Our reformulation enables us to employ exponential-like functions of the Jacobian of the full nonlinear forcing operator and an efficient exponential integrator of EPIRK-type to advance the solution forward in time. Note that while the particular stiffly accurate EPIRK scheme we use appeared in a recent publication [Rainwater and Tokman 2016a], its

performance has only been evaluated in the context of numerical test problems. Our work is the first test of this new scheme on a complex real-world application such as the elastodynamic systems.

In addition to presenting a new time integration approach for nonlinear elastodynamic systems, our work also expands current knowledge of the applicability and computational advantages of stiffly accurate exponential integrators. While theoretical understanding of the structure and efficiency of exponential integration grew over the past decade, questions of their practical use and performance remain to be addressed. This is particularly true with respect to exponential integrators for general large scale nonlinear systems. Applications of the exponential methods to complex real-world problems are just beginning to appear [Einkemmer et al. 2017] and more research is needed to identify areas where exponential methods bring significant computational savings and to quantify these advantages. Unresolved issues also remain regarding the use of stiffly accurate exponential methods. While theoretically methods satisfying stiff order conditions are expected to mitigate computational challenges associated with stiffness, demonstration of these advantages has been limited to numerical test problems. Our work shows that stiff accuracy and exponential integration enable significant computational savings for modeling elastodynamic systems of size and complexity relevant for practical applications.

Below, we detail the structure of the new time integration algorithm and explain why this approach is computationally advantageous.

3.1 Reformulating Elastodynamic Systems

We consider elastodynamic problems relevant to visual computing such as the simulation of systems of $N/3$ coupled oscillators with N degrees of freedom. Each oscillator is represented by its position vector $\mathbf{x}_i(t) \in \mathbb{R}^3$ $(i = 1, \ldots, N/3)$ at time t, with its velocity computed as $\mathbf{x}_i'(t)$. Denoting the vector of positions by $\mathbf{x}(t) = (\mathbf{x}_1(t), \mathbf{x}_2(t), \ldots, \mathbf{x}_{N/3}(t)) \in \mathbb{R}^N$ we can write the final form of the model equations that describe a system of coupled oscillators as

$$\mathbf{M}\mathbf{x}''(t) + \mathbf{D}\mathbf{x}'(t) + \mathbf{K}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)). \tag{1}$$

Note that no changes are needed to our time integration approach based on whether coupled oscillators or finite elements are used to model the system as long as the structure of the formulation can be described by Eq. (1). Here, $\mathbf{M}, \mathbf{D}, \mathbf{K} \in \mathbb{R}^{N\times N}$ are respectively the symmetric mass matrix, the damping matrix, and the symmetric stiffness matrix, and $\mathbf{f}(\mathbf{x}) \in \mathbb{R}^N$ represents the total external forces on the system. We ignore damping here for brevity and set $\mathbf{D} = \mathbf{0}$ in the following derivations. Since the mass matrix $\mathbf{M}$ is always nonsingular (and often even diagonal), Eq. (1) can be re-written in the simpler form

$$\mathbf{x}''(t) + \mathbf{L}\mathbf{x}(t) = \mathbf{g}(\mathbf{x}(t)), \tag{2}$$

where we set $\mathbf{L} = \mathbf{M}^{-1}\mathbf{K} \in \mathbb{R}^{N\times N}$ and $\mathbf{g}(\mathbf{x}(t)) = \mathbf{M}^{-1}\mathbf{f}(\mathbf{x}(t))$, and provide initial conditions $\mathbf{x}(t_0) = \mathbf{x}_0$ and $\mathbf{x}'(t_0) = \mathbf{v}_0$. In this work we assume that the matrix $\mathbf{L}$ is symmetric positive-definite. This is certainly the case when $\mathbf{M}$ is a diagonal matrix and $\mathbf{K}$ is symmetric positive definite as a consequence of the Maxwell-Betti reciprocal work theorem. We are interested in stiff systems where the solutions of Eq. (2) possess a wide range of frequencies including very fast modes that significantly outpace the time scale of the dynamics we want to simulate, e.g. fast appearance of small, imperceptible to the eye, wrinkles in a simulated cloth versus the overall folding of the garment. Our aim is to develop a time integrator which is able to efficiently evolve the system on the slower time scale of interest, e.g. that of the overall movement of the garment, while preserving a sufficient amount of accuracy, e.g. resolving enough wrinkles so that the process of folding looks realistic.

A standard approach to solving Eq. (2) numerically is to write it as a first order system of ordinary differential equations by changing the vector of unknowns to include both positions and velocities of the oscillators

$$\mathfrak{X}(t) = (\mathbf{x}(t), \mathbf{x}'(t))^{\mathsf{T}} \in \mathbb{R}^{2N}.$$

Eq. (2) can then be re-written as

$$\mathfrak{X}'(t) = \mathfrak{A}\mathfrak{X}(t) + \Gamma(\mathfrak{X}(t)), \quad \mathfrak{X}(t_0) = \mathfrak{X}_0 = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{v}_0 \end{bmatrix} \tag{3}$$

with

$$\mathfrak{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{L} & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{2N\times 2N}, \quad \Gamma(\mathfrak{X}) = \begin{bmatrix} \mathbf{0} \\ \mathbf{g}(\mathbf{x}) \end{bmatrix} \in \mathbb{R}^{2N}.$$

It is typically assumed that the source of stiffness in the model are the linear spring forces $\mathbf{K}\mathbf{x}(t)$ in Eq. (1) which corresponds to the term $\mathfrak{A}\mathfrak{X}(t)$ in Eq. (3). Using explicit time integrators for a severely stiff problem Eq. (3) is impractical due to restrictive stability constraints on the size of a time step for such schemes. A common way to address this computational challenge in visual computing and other fields is to use a semi-implicit time discretization which handles $\mathfrak{A}\mathfrak{X}(t)$ implicitly. The implicit treatment of the linear term $\mathfrak{A}\mathfrak{X}(t)$ allows for the integration with a considerably larger time step compared to explicit methods but this approach has two important limitations.

The first issue stems from the fact that matrix $\mathfrak{A}$ is generally non-symmetric. Using an implicit integrator to solve Eq. (3) implies that at each time step during the integration a product of an inverse of a matrix $(\mathbf{I} - ch\mathfrak{A})^{-1}$ (c is a constant and h is the time step size) with some vector $\mathbf{v}$ has to be calculated. Note that $(\mathbf{I} - ch\mathfrak{A})^{-1}$ is a rational function of a matrix $\mathfrak{A}$. Since $\mathfrak{A}$ is a non-symmetric stiff matrix and generally not much is known about its spectrum of eigenvalues, the computation of $(\mathbf{I} - ch\mathfrak{A})^{-1}\mathbf{v}$ is a computationally expensive task. In particular, if an iterative method is used to compute this product, it will likely require the construction of an effective preconditioner to achieve reasonable efficiency.

The second limitation to the common semi-implicit approach is that the forcing term $\mathbf{f}(\mathbf{x}(t))$ in Eq. (1), or $\Gamma(\mathfrak{X}(t))$ in Eq. (3), can also be a source of stiffness. This can occur, for instance, when external forces applied to the system are comprised of spatially and temporally non-uniform time scales, e.g. fabric of a garment can be simultaneously responding to overall motion of a body as well as shaking of a flabby body part. Moreover, $\mathbf{f}(\mathbf{x}(t))$ is usually a source of stiffness when strong impact forces are applied, since these usually unpredictable forces are often notoriously stiff. Since it is difficult, or perhaps even impossible, to partition the external force operator $\mathbf{f}(\mathbf{x}(t))$ into a non-stiff and a stiff part, or extract information about a spectrum of its Jacobian $\mathbf{f}'(\mathbf{x}(t))$, it is also difficult to treat it implicitly.

Exponential integrators emerged as an alternative to explicit and implicit techniques in numerical time integration. Unlike implicit methods which require the approximation of a rational function of a stiff matrix (e.g. matrix $(\mathbf{I}-ch\mathfrak{A})^{-1}$ as explained above), exponential methods involve the computation of an exponential or exponential-like function of this matrix (e.g. matrix $\exp(h\mathfrak{A})$ as explained in the next section). These integrators possess stability properties similar to implicit schemes and allow for an integration with time steps far exceeding those required for explicit methods. At the same time exponential integrators can offer significant computational savings in per-time step computational complexity compared to implicit methods. This is particularly true for stiff problems, for which the construction of an efficient preconditioner is difficult or impossible. However, just like explicit and implicit schemes, exponential methods are not immune to the effects of increased stiffness. While using an iterative method to evaluate a product $\exp(h\mathfrak{A})\mathbf{v}$ can be more efficient than computing a similar product that involves a rational function $(\mathbf{I}-ch\mathfrak{A})^{-1}\mathbf{v}$, the number of required iterations will still increase for both the exponential and the rational functions as the stiffness of the matrix $\mathfrak{A}$ is increased.

Given these considerations we reformulate the problem to improve the numerical properties of the stiff matrix used as an argument for the exponential and exponential-like functions as follows. As in Gautschi-type integrators [Michels et al. 2014] we define the matrix $\Omega=\sqrt{\mathbf{L}}$. Please note, that for the uniqueness of its square root, the structure of $\mathbf{L}=\mathbf{M}^{-1}\mathbf{K}$ has to meet certain criteria, in particular symmetry and positive definiteness, which are in general not fulfilled in cases of an inhomogeneous mass distribution or for a non-diagonally lumped mass matrix $\mathbf{M}$. The appropriate handling of these cases is described in App. A.

We now use the change of variable $\mathbf{X}(t)=(\Omega\mathbf{x}(\mathbf{t}),\mathbf{x}'(\mathbf{t}))^{\mathsf{T}}$, so that Eq. (2) can be reformulated as

$$\mathbf{X}'(t)=\mathbf{F}(\mathbf{X}(t))=\mathscr{A}\mathbf{X}(t)+\mathbf{G}(\mathbf{X}(t)) \tag{4}$$

with $\mathbf{X}(t_0)=\mathbf{X}_0$, where $\mathscr{A}=\operatorname{adiag}(\Omega,-\Omega)$ and $\mathbf{G}(\mathbf{X})=(\mathbf{0},\mathbf{g}(\mathbf{x}))^{\mathsf{T}}$. As noted above, a similar change of variables was used in prior publications [Buchholz et al. 2017; Chen and Russell 1982; Göckler and Grimm 2013] for different purposes. For example, in Göckler and Grimm [2013] this variable change is used to integrate a linear Schrödinger equation using extended Krylov subspaces. Note that in contrast to a generally nonsymmetric matrix $\mathfrak{A}$, matrix $\mathscr{A}$ is skew-symmetric and consequently its spectrum is comprised of purely imaginary pairs of eigenvalues $\pm\lambda_k i$. In addition, $\mathscr{A}$ is also an infinitesimal symplectic (or Hamiltonian) matrix since given $\mathscr{J}=\operatorname{adiag}(\mathbf{I},-\mathbf{I})$, we obtain $\mathscr{J}\mathscr{A}=\operatorname{diag}(-\Omega,-\Omega)=(\mathscr{J}\mathscr{A})^{\mathsf{T}}$. Clearly, since $\mathscr{A}$ is infinitesimal symplectic then $\exp(t\mathscr{A})$ is a symplectic matrix, i.e. $\exp(t\mathscr{A})^{\mathsf{T}}\mathscr{J}\exp(t\mathscr{A})=\mathscr{J}$ for all t.

Whether the complete Jacobian of Eq. (4), $\mathbf{F}'(X)=\mathscr{A}+\mathbf{G}'(\mathbf{X})$, is infinitesimal symplectic, and consequently the matrix $\exp(t\mathbf{F}'(\mathbf{X}))$ is symplectic and the system described by Eq. (4) is Hamiltonian, depends on the matrices $\mathbf{g}'(\mathbf{x})$ and Ω since $\mathscr{J}\mathbf{F}'(\mathbf{X})=\operatorname{diag}(-\Omega+\mathbf{g}'(\mathbf{x})\Omega^{-1},-\Omega)$. Clearly, $\mathscr{J}\mathbf{F}'(\mathbf{X})$ is symmetric if $\mathbf{g}'(x)\Omega^{-1}$ is symmetric which will result in $\mathbf{F}'(\mathbf{X})$ being an infinitesimal symplectic matrix. Symplecticity of the system described by Eq. (4) would, of course, guarantee that propagating the solution with the operator $\exp(t\mathbf{F}'(\mathbf{X}))$ would exactly preserve the Hamiltonian which corresponds to the total energy of the solution. While symplecticity of $\mathbf{F}'(\mathbf{X})$ is not guaranteed, as the numerical experiments presented in Sec. 4.1 demonstrate, our approach of using exponential integration to propagate the solution with a direct high-order of accuracy approximation to $\exp(t\mathbf{F}'(\mathbf{X}))$ results in the energy being preserved within the required tolerance for the time intervals of interest in our applications.

3.2 Exponential Integration

We now describe how an exponential integrator is constructed to integrate the reformulated problem described by Eq. (4). We start with discretizing the time variable $t_n=t_{n-1}+h_n$ $(n=1,\dots,T)$ over the interval $t\in[t_0,t_{\text{final}}]$ and writing Eq. (4) in the form

$$\mathbf{X}'(t)=\mathbf{F}(\mathbf{X}(t_n))+\mathbf{F}'(\mathbf{X}(t_n))(\mathbf{X}(t)-\mathbf{X}(t_n))+\mathbf{R}_n(\mathbf{X}(t)), \tag{5}$$

where

$$\mathbf{R}_n(\mathbf{X}(t))=\mathbf{F}(\mathbf{X}(t))-\mathbf{F}(\mathbf{X}(t_n))-\mathbf{F}'(\mathbf{X}(t_n))(\mathbf{X}(t)-\mathbf{X}(t_n)) \tag{6}$$

is the nonlinear remainder function of the first-order Taylor expansion of $\mathbf{F}(\mathbf{X}(t))$ around $\mathbf{X}(t_n)$ and $\mathbf{J}(\mathbf{X}(t))=\mathbf{F}'((\mathbf{X}(t))=\mathscr{A}+\mathbf{G}'(\mathbf{X}(t))$ is the Jacobian matrix. By integrating Eq. (5) from t_n to t_{n+1} using an integration factor $\exp(-\mathbf{J}(\mathbf{X}(t_n))t)$, we can write the solution of Eq. (5) at t_{n+1} in an integral form

$$\begin{aligned}\mathbf{X}(t_{n+1})&=\mathbf{X}(t_n)+h_n\varphi_1(h_n\mathbf{J}(\mathbf{X}(t_n)))\mathbf{F}(\mathbf{X}(t_n))\\&\quad+\int_{t_n}^{t_{n+1}}\exp((t_{n+1}-t)\mathbf{J}(\mathbf{X}(t_n)))\mathbf{R}_n(\mathbf{X}(t))\,\mathrm{d}t,\end{aligned} \tag{7}$$

where $\varphi_1(z)$ is an exponential-like analytic function defined as $\varphi_1(z)=z^{-1}(\exp(z)-1)$. Making a change of variable in the integral in Eq. (7) to θ with $t=t_n+\theta h_n$ we obtain a slightly simplified integral form

$$\begin{aligned}\mathbf{X}(t_{n+1})&=\mathbf{X}(t_n)+h_n\varphi_1(h_n\mathbf{J}(\mathbf{X}(t_n)))\mathbf{F}(\mathbf{X}(t_n))\\&\quad+h_n\int_0^1\exp((1-\theta)h_n\mathbf{J}(\mathbf{X}(t_n)))\mathbf{R}_n(\mathbf{X}(t_n+\theta h_n))\,\mathrm{d}\theta.\end{aligned} \tag{8}$$

Exponential integrators are then constructed by developing a quadrature for the nonlinear integral in Eq. (8). To simplify the notation let us now define the approximate solution computed by an integrator as $\mathbf{X}_n\approx\mathbf{X}(t_n)$ and respectively $\mathbf{F}_n=\mathbf{F}(\mathbf{X}_n)$ and $\mathbf{J}_n=\mathbf{F}'(\mathbf{X}_n)$. If some polynomial approximation is chosen for

$$\mathbf{R}_n(\mathbf{X}(t_n+\theta h_n))\approx\bar{\mathbf{v}}_1+\theta h_n\bar{\mathbf{v}}_2+\theta^2h_n^2\bar{\mathbf{v}}_3+\cdots+\theta^{p-1}h_n^{p-1}\bar{\mathbf{v}}_p$$

($\bar{\mathbf{v}}_i$ are vectors in $\mathbb{R}^{2N}$), any resulting quadrature rule will involve linear combinations of terms of the form

$$\left(h_n\int_0^1\exp((1-\theta)h_n\mathbf{J}_n)\theta^{k-1}\mathrm{d}\theta\right)\bar{\mathbf{v}}_k.$$

Introducing a rescaling factor $\bar{\mathbf{v}}_k=\mathbf{v}_k/(k-1)!$ we can write these terms as

$$h_n\varphi_k(h_n\mathbf{J}_n)\mathbf{v}_k,$$

where $\varphi_k(z)$ are exponential-like analytic functions defined by

$$\varphi_k(z)=\int_0^1\exp((1-\theta)z)\frac{\theta^{k-1}}{(k-1)!}\,\mathrm{d}\theta,\quad k\in\mathbb{N}^*. \tag{9}$$

3.3 EPIRK Methods

In this paper we make use of EPIRK methods which, in our numerical experiments, allowed for most efficiency given the desired accuracy in the simulations. These methods were originally proposed by Tokman [2006] and then generalized, extended and tested in subsequent publications [Loffeld and Tokman 2013; Rainwater and Tokman 2016b; Tokman 2011]. The EPIRK framework allows for a formulation of general classes of methods with flexible structure. This flexibility is then used to construct specific schemes that are particularly efficient for given classes of problems. Using the definitions of $\mathbf{X}_n$, $\mathbf{F}_n$, $\mathbf{J}_n$ and $\mathbf{R}_n$ from the previous section we can write the general s-stage EPIRK method for the problem $\mathbf{X}'(t) = \mathbf{F}(\mathbf{X}(t))$ in the form

$$\begin{aligned}
\mathbf{X}_{ni} &= \mathbf{X}_n + a_{i1}\psi_{i1}(g_{i1}h_n\mathbf{A}_{i1})h_n\mathbf{F}_n \\
&+ h_n\sum_{j=2}^{i-1} a_{ij}\psi_{ij}(g_{ij}h_n\mathbf{A}_{ij})\Delta^{(j-1)}\mathbf{R}(\mathbf{X}_n), \quad i = 2,\ldots,s, \\
\mathbf{X}_{n+1} &= \mathbf{X}_n + b_1\psi_{s+1,1}(g_{s+1,1}h_n\mathbf{A}_{i1})h_n\mathbf{F}_n \\
&+ h_n\sum_{j=2}^{s} b_j\psi_{s+1,j}(g_{s+1,j}h_n\mathbf{A}_{ij})\Delta^{(j-1)}\mathbf{R}(\mathbf{X}_n),
\end{aligned} \tag{10}$$

where $h_n = t_{n+1} - t_n$ is the time step, a_{ij}, b_i, g_{ij} are constant coefficients, ψ_{ij} are functions, $\mathbf{A}_{ij}$ are matrices and $\mathbf{R}(\mathbf{X})$ is a vector function. The k-th forward difference is denoted by $\Delta^{(k)}$ and $\Delta^{(k)}\mathbf{R}(\mathbf{X}_n)$ denotes the forward difference operator constructed using the stage vectors $\mathbf{X}_n, \mathbf{X}_{n2}, \ldots, \mathbf{X}_{ns}$. The choices of matrices $\mathbf{A}_{ij}$, and functions ψ_{ij} and $\mathbf{R}(\mathbf{X})$ are made depending on the structure of the operator $\mathbf{F}(\mathbf{X}(t))$ in $\mathbf{X}'(t) = \mathbf{F}(\mathbf{X}(t))$ to derive different classes of EPIRK methods [Tokman and Rainwater 2014]. Here we focus on the class of general EPIRK methods which result from the following choices:

$$\mathbf{A}_{ij} = \mathbf{J}_n = \mathbf{F}_n'(\mathbf{X}_n)$$

are the Jacobian of the full nonlinear forcing operator $\mathbf{F}(\mathbf{X})$, $\mathbf{R}(\mathbf{X}) = \mathbf{R}_n(\mathbf{X}(t))$ is the nonlinear remainder function from Eq. (6) and functions $\psi_{ij}(z)$ are chosen to be linear combinations of exponential-like functions (9):

$$\psi_{ij}(z) = \sum_{k=1}^{K} p_{ijk}\varphi_k(z), \quad \varphi_k(z) = \int_0^1 e^{z(1-\theta)}\frac{\theta^{k-1}}{(k-1)!}\,\mathrm{d}\theta.$$

The general EPIRK methods are sometimes called "unpartitioned" because the operator $\mathbf{F}$ is not partitioned into separate terms and its full Jacobian is used in the integrator. Specific EPIRK schemes can be constructed by determining constant coefficients a_{ij}, b_i, g_{ij} and p_{ijk} given considerations of accuracy and efficiency. In Sec. 3.4 we describe how these coefficients can be derived using the concept of stiff accuracy and the corresponding order conditions. Typically either classical or stiff order conditions are underdetermined and additional constraints can be placed on the coefficients. These constraints usually come from considerations of efficiency of the method. For example, in Sec. 3.5 we discuss how minimizing coefficients g_{ij} can result in significant computational savings if Krylov projection-based algorithms are used to evaluate matrix function-vector products $\varphi_k(h_n\mathbf{J}_n)\mathbf{v}_k$ in Eq. (10) [Loffeld and Tokman 2013; Rainwater and Tokman 2014, 2016b].

For the elastodynamic problems considered here we found that the best balance of accuracy versus efficiency is obtained using fourth order three-stage EPIRK methods that can be written in a general form as

$$\begin{aligned}
\mathbf{X}_{n2} &= \mathbf{X}_n + a_{21}\psi_{21}\big(g_{21}h_n\mathbf{J}_n\big)h_n\mathbf{F}_n, \\
\mathbf{X}_{n3} &= \mathbf{X}_n + a_{31}\psi_{31}\big(g_{31}h_n\mathbf{J}_n\big)h_n\mathbf{F}_n \\
&\qquad + a_{32}\psi_{32}\big(g_{32}h_n\mathbf{J}_n\big)h_n\Delta\mathbf{R}_n, \\
\mathbf{X}_{n+1} &= \mathbf{X}_n + b_1\psi_{41}\big(g_{41}h_n\mathbf{J}_n\big)h_n\mathbf{F}_n \\
&\qquad + b_2\psi_{42}\big(g_{42}h_n\mathbf{J}_n\big)h_n\Delta\mathbf{R}_n \\
&\qquad + b_3\psi_{43}\big(g_{43}h_n\mathbf{J}_n\big)h_n\Delta^2\mathbf{R}_n,
\end{aligned} \tag{11}$$

where $\psi_{ij}(z) = \sum_{k=0}^{K_{ij}} p_{ijk}\varphi_k(z)$, and the forward differences Δ and Δ^2 are constructed on the nodes $\mathbf{X}_{n1} = \mathbf{X}_n, \mathbf{X}_{n2}, \mathbf{X}_{n3}$ and defined by (note $\mathbf{R}_n(\mathbf{X}_n) = 0$)

$$\begin{aligned}
\Delta\mathbf{R}_n &= \mathbf{R}_n(\mathbf{X}_n) - \mathbf{R}_n(\mathbf{X}_{n2}) = -\mathbf{R}_n(\mathbf{X}_{n2}), \\
\Delta^2\mathbf{R}_n &= -2\mathbf{R}_n(\mathbf{X}_{n2}) + \mathbf{R}_n(\mathbf{X}_{n3}).
\end{aligned}$$

To derive the three-stage EPIRK scheme of fourth order we need to determine an expansion of the local error of the method over one time step in terms of powers of h_n. Different forms of such an expansion lead to either classically or stiffly accurate methods. In either case the order conditions are derived by setting the coefficients of the powers of h_n to zero and solving the resulting equations with appropriate additional constraints for the constants a_{ij}, b_j, g_{ij}, and p_{ijk}. For very stiff problems stiffly accurate schemes offer significant computational savings compared to classical methods. In the subsequent sections we describe the key idea behind stiff accuracy and demonstrate the efficiency and accuracy of the stiffly accurate exponential integrator.

3.4 Stiff Accuracy

A classical approach to deriving time integration schemes of a given order starts with expanding both the exact solution of Eq. (4) and the numerical solution defined by the numerical scheme (11) in a Taylor series around $\mathbf{X}(t_n)$. The resulting Taylor expansions are comprised of linear combinations of so-called elementary differentials, i.e. products of $\mathbf{F}$ and its derivatives [Butcher 2008; Hairer et al. 2004; Hairer and Wanner 2004]. The exact solution of Eq. (4) expanded in a Taylor series can be written as

$$\begin{aligned}
\mathbf{X}(t_n + h_n) = \mathbf{X}(t_n) &+ h_n\mathbf{X}'(t_n) + \frac{1}{2!}h_n^2\mathbf{X}''(t_n) \\
&+ \frac{1}{3!}h_n^3\mathbf{X}'''(t_n) + \ldots
\end{aligned}$$

or using Eq. (4) itself and defining $\tilde{\mathbf{F}}_n = \mathbf{F}(\mathbf{X}(t_n))$, $\tilde{\mathbf{J}}_n = \tilde{\mathbf{F}}_n'$ as

$$\begin{aligned}
\mathbf{X}(t_n + h_n) &= \mathbf{X}(t_n) + h_n\tilde{\mathbf{F}}_n + \frac{1}{2!}h_n^2\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n \\
&\quad + \frac{1}{3!}h_n^3\Big[\tilde{\mathbf{F}}_n''(\tilde{\mathbf{F}}_n,\tilde{\mathbf{F}}_n) + \tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n\Big] + \ldots \\
&= \mathbf{X}(t_n) + h_n\tilde{\mathbf{F}}_n + \frac{1}{2!}h_n^2\tilde{\mathbf{J}}_n\tilde{\mathbf{F}}_n \\
&\quad + \frac{1}{3!}h_n^3\Big[\tilde{\mathbf{F}}_n''(\tilde{\mathbf{F}}_n,\tilde{\mathbf{F}}_n) + \tilde{\mathbf{J}}_n^2\tilde{\mathbf{F}}_n\Big] + \ldots.
\end{aligned} \tag{12}$$

We now define $\widehat{\mathbf{X}}_{n+1}$ to be the numerical solution computed using Eq. (11) with the approximation at the previous time step $\mathbf{X}_n$ replaced with an exact solution $\tilde{\mathbf{X}}_n = \mathbf{X}(t_n)$. The Taylor expansion of $\widehat{\mathbf{X}}_{n+1}$ also involves powers of h_n and the elementary differentials of $\mathbf{F}$, but has more complicated coefficients c_{ij} that are functions of constants a_{ij}, b_j, g_{ij}, and p_{ijk}:

$$\begin{aligned}\widehat{\mathbf{X}}_{n+1} = \mathbf{X}(t_n) &+ h_n c_{11}\tilde{\mathbf{F}}_n + h_n^2 c_{21}\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n \\ &+ h_n^3\left[c_{31}\tilde{\mathbf{F}}_n''(\tilde{\mathbf{F}}_n,\tilde{\mathbf{F}}_n) + c_{32}\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n\right] + \dots . \end{aligned} \tag{13}$$

Note that in obtaining the expansion (13) functions $\psi_{ij}(z)$ and consequently functions $\varphi_k(z)$ are also expanded in a Taylor series around $\tilde{\mathbf{J}}_n = \mathbf{F}'(\mathbf{X}(t_n))$. The local truncation error of a method is the difference between the Taylor expansions of the exact Eq. (12) and the numerical Eq. (13) solutions:

$$\begin{aligned}\mathbf{e}_n &= \widehat{\mathbf{X}}_{n+1} - \mathbf{X}(t_n + h_n) \\ &= h_n(c_{11}-1)\tilde{\mathbf{F}}_n + h_n^2\left(c_{21}-\tfrac{1}{2!}\right)\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n \\ &+ h_n^3\left[\left(c_{31}-\tfrac{1}{3!}\right)\tilde{\mathbf{F}}_n''(\tilde{\mathbf{F}}_n,\tilde{\mathbf{F}}_n) + \left(c_{32}-\tfrac{1}{3!}\right)\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n'\tilde{\mathbf{F}}_n\right] + \dots \\ &= h_n(c_{11}-1)\tilde{\mathbf{F}}_n + h_n^2\left(c_{21}-\tfrac{1}{2!}\right)\tilde{\mathbf{J}}_n\tilde{\mathbf{F}}_n \\ &+ h_n^3\left[\left(c_{31}-\tfrac{1}{3!}\right)\tilde{\mathbf{F}}_n''(\tilde{\mathbf{F}}_n,\tilde{\mathbf{F}}_n) + \left(c_{32}-\tfrac{1}{3!}\right)\tilde{\mathbf{J}}_n^2\tilde{\mathbf{F}}_n\right] + \dots . \end{aligned} \tag{14}$$

The order conditions are then derived by setting coefficients of the elementary differentials in Eq. (14) to zero up to the desired order of accuracy, e.g. to get a method of order three from Eq. (14) we will solve the order conditions

$$c_{11} = 1,\ c_{21} - 1/2! = 0,\ c_{3i} - 1/3! = 0$$

for $i = 1, 2$. Since c_{ij} are nonlinear algebraic functions in the method's coefficients a_{ij}, b_j, g_{ij}, and p_{ijk}, the values for these coefficients can be obtained by solving these order conditions. The precise form of the functions c_{ij} is complex and presenting them here does not add to the conceptual understanding so we refer an interested reader to Tokman [2011] for these formulas. The order conditions are solved exactly, if possible, or approximately to a high precision to obtain values for a_{ij}, b_j, g_{ij}, and p_{ijk}.

Since in this classical approach functions $\psi_{ij}(z)$ and consequently functions $\varphi_k(z)$ are also expanded in a Taylor series, in a sense, the resulting method is using a finite-order polynomial approximation to these exponential-like functions. The convergence of the method can be proved by showing that the method is stable. Convergence is demonstrated by proving that the global error $\mathbf{E}_n$ is bounded as $||\mathbf{E}_n|| \le Ch^q$, where C is independent of h_n and h is the maximum time step size $h = \max_n\{h_n\}$.

While methods derived in this classical way will exhibit the theoretically expected order of accuracy for non-stiff problems, this might not be the case for stiff systems. For stiff problems the Jacobian matrix $\tilde{\mathbf{J}}_n$ can have a very large norm, so the elementary differentials such as, for example, $\tilde{\mathbf{J}}_n\tilde{\mathbf{F}}_n$ or $\tilde{\mathbf{J}}_n^2\tilde{\mathbf{F}}_n$ in Eq. (14) can potentially be vectors with a very large norm as well. The finite precision nature of practical computations implies that the actual values used for coefficients a_{ij}, b_j, g_{ij}, and p_{ijk} will result in the order conditions being satisfied at best up to the roundoff error but not exactly. Thus the terms in the expansion of the local error (14) can actually be large if the corresponding elementary differentials involving $\tilde{\mathbf{J}}_n$ are very large in the norm. Another way to see this effect is to realize that the formula for coefficient C involved in the bound Ch^q of the classical global error $\mathbf{E}_n$, can depend on the stiff matrix $\tilde{\mathbf{J}}_n$.

For example, a second order method with $c_{11} = 1$ and $c_{21} = 1/2$ applied to a very stiff problem might not actually exhibit full second order in practice because the large norm of $\tilde{\mathbf{J}}_n$ will make the error term

$$(c_{21} - 1/2)h_n^2\tilde{\mathbf{J}}_n\tilde{\mathbf{F}}_n$$

in Eq. (14) large enough even if the value of $c_{21} - 1/2$ is close to a roundoff error. Such an order reduction can result in a significant loss of efficiency.

An approach to derive time integrators that help mitigate this problem is to derive so-called stiff order conditions and to use them to construct specific schemes. Stiff accuracy is the notion originally developed for implicit methods [Hairer and Wanner 2004] and then for exponential Runge-Kutta and exponential Rosenbrock methods [Hochbruck and Ostermann 2005, 2006; Luan and Ostermann 2013, 2014a,b]. Recently, Rainwater and Tokman [2016b] expanded the stiff order theory for exponential Rosenbrock methods to the EPIRK framework and constructed stiffly accurate EPIRK schemes, which we employ in this work.

To understand the notion of stiff accuracy it is important to recall that if the spectrum of a matrix $\mathbf{A}$ lies in the left half of the complex plane or on the imaginary axis (which is the case for stable dynamical systems), then even if $\mathbf{A}$ has a large norm, the norm of matrix $\exp(\mathbf{A})$ is bounded by 1. Similarly bounded will be the norms of matrices involving exponential-like functions $\varphi_k(\mathbf{A})$. In addition, it is reasonable to assume that the solution of Eq. (4) and its derivatives are bounded. To derive stiffly accurate methods the local error $\mathbf{e}_n$ is expressed in terms of these bounded quantities rather than all of the elementary differentials. Using stiff order conditions derived from these expressions, it is then possible to show that the resulting global error $\mathbf{E}_n$ is bounded by $||\mathbf{E}_n|| \le Ch^q$ where the constant C is independent of $\|\tilde{\mathbf{J}}_n\|$ and thus the resulting method is much less sensitive to the stiffness of the problem.

For problem types described by Eq. (4) it has been shown for exponential Rosenbrock schemes [Luan and Ostermann 2014b] and for EPIRK integrators [Rainwater and Tokman 2016b] that the local error can be expressed as

$$\mathbf{e}_n = C_1\left(h_n\tilde{\mathbf{J}}_n\right)h_n\tilde{\mathbf{F}}_n + C_3\left(h_n\tilde{\mathbf{J}}_n\right)h_n^3\tilde{\mathbf{F}}_n''(\tilde{\mathbf{F}}_n,\tilde{\mathbf{F}}_n) + \dots . \tag{15}$$

While coefficients $C_i(h_n\tilde{\mathbf{J}}_n)$ still depend on a_{ij}, b_{ij}, g_{ij}, and p_{ijk}, unlike classical coefficients c_{ij} these are now bounded functions of $h_n\tilde{\mathbf{J}}_n$. This new form of the local error (15) no longer involves explicit powers of $\tilde{\mathbf{J}}_n$ and expresses $\mathbf{e}_n$ in terms of quantities that are assumed bounded such as $\tilde{\mathbf{F}}_n$ and $\tilde{\mathbf{F}}_n''$. For the sake of simplicity we omit the higher order terms here but note that their formulas have the same properties and express the error in terms of matrix functions $\varphi_k(h_n\tilde{\mathbf{J}}_n)$ along with bounded derivatives of $\mathbf{F}$ and $\mathbf{X}$. Once again, the expressions of the stiff order conditions are quite complex. For example, to satisfy $C_3(h_n\tilde{\mathbf{J}}_n) = 0$ coefficients of an s-stage stiffly

accurate EPIRK method Eq. (10) must satisfy the conditions

$$C_3\big(Z\big) = \sum_{i=2}^{s} B_i(Z) a_{i1}^2 \mathbb{P}_{i1}^2 - 2\varphi_3(Z) = 0,$$

$$\mathbb{P}_{i1} = \sum_{k=1}^{s} \frac{p_{i1k}}{k!},$$

$$B_i(Z) = \sum_{k=i}^{s} \binom{k-1}{s} (-1)^{k-s-3} b_k \psi_{s+1,k}(g_{s+1,k} Z)$$

for any square matrix Z. For exact formulas of the stiff order conditions we refer the reader to Rainwater and Tokman [2016b] and note that stability and convergence of such schemes was proved [Luan and Ostermann 2014b; Rainwater and Tokman 2016b]. In addition, in App. B we offer a compact proof of the stability of the method.

The stiff order conditions are not uniquely solvable and many stiffly accurate EPIRK schemes can be constructed. Thus additional constraints can be imposed on the order conditions. To further improve the efficiency of the method we choose to enforce additional constraints based on the approximation of the matrix function-vector products $\varphi_k(gh_n\mathbf{J}_n)\mathbf{v}$ as explained in the next section.

3.5 Adaptive Krylov Evaluations

The main computational cost of an exponential method lies in the evaluations of matrix functions $\varphi_k(\mathbf{A})\mathbf{v}$ ($\mathbf{A}$ is a matrix and $\mathbf{v}$ is a vector). A number of algorithms have been proposed in the literature [Al-Mohy and Higham 2011; Bergamaschi et al. 2004; Caliari et al. 2016; Kassam and Trefethen 2005; Moler and Loan 2003; Niesen and Wright 2012]. Many of these methods, however, either assume that the matrix $\mathbf{A}$ is small or require additional information about the spectral radius or structure of the matrix. The most general and efficient method applicable to general large matrices is the adaptive Krylov algorithm developed by Niesen and Wright [2012]. Since the detailed description of the algorithm is complex and is available along with the pseudocode in their original paper [2012], here we focus on the main idea of the algorithm, particularly highlighting the properties of the method that allow imposing additional constraints on the order conditions and constructing efficient EPIRK schemes.

At the core of the adaptive Krylov algorithm is the Krylov subspace projection method originally proposed by Van der Vorst [1987] for the evaluation of products $f(\mathbf{A})\mathbf{v}$ of arbitrary matrix functions and vectors. The standard Krylov algorithm uses a projection of $\mathbf{A}$ and $\mathbf{v}$ onto a Krylov subspace

$$S_m = \mathrm{span}\{\mathbf{v}, \mathbf{A}\mathbf{v}, \mathbf{A}^2\mathbf{v}, \ldots, \mathbf{A}^{m-1}\mathbf{v}\} = \mathrm{span}\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m\}.$$

An orthonormal basis $\{\mathbf{v}_i\}_{i=1}^m$ of S_m is obtained by using the Arnoldi iteration [Arnoldi 1951], which is based on a Gram-Schmidt orthogonalization procedure. The Arnoldi iteration provides the matrix $\mathbf{V}_m$ which has vectors $\mathbf{v}_i$ as its columns and the $(m \times m)$-matrix $\mathbf{H}_m = \mathbf{V}_m^{\mathsf{H}} \mathbf{A} \mathbf{V}_m$, where $\mathbf{V}_m^{\mathsf{H}}$ is a conjugate transpose of $\mathbf{V}_m$. The product $f(\mathbf{A})\mathbf{v}$ is then approximated as $f(\mathbf{A})\mathbf{v} \approx ||\mathbf{v}||_2 \mathbf{V}_m f(\mathbf{H}_m) e_1$ with $e_1 = (1, 0, \ldots, 0)^{\mathsf{T}}$. If m is small, the evaluation of $f(\mathbf{H}_m)$ is inexpensive and can be done by a number of methods including Taylor expansion, Padé approximation or other algorithms as described for instance by Moler and Loan [2003]. The value of m that ensures the desired accuracy of the approximation, is determined dynamically using residual estimates.

The overall cost of a Krylov subspace projection method is dominated by the Arnoldi iteration to construct $\mathbf{V}_m$ and the number of flops scales as $O(m^2)$. Clearly the number of the Krylov vectors m depends on the spectrum of $\mathbf{A}$ and the magnitude of $\mathbf{v}$. A stiffer matrix with a larger spectral radius will require a larger number of Krylov vectors m. It is well-known that a more clustered spectrum results in a faster convergence of the Krylov projection algorithm and therefore a smaller m. Note that if instead of $f(\mathbf{A})\mathbf{v}$ we need to approximate $f(\tau\mathbf{A})\mathbf{v}$ with a small coefficient $\tau \in (0, 1)$, the spectrum of the matrix $\tau\mathbf{A}$ will be more clustered compared to $\mathbf{A}$. Thus we can expect the number of Krylov vectors to approximate $f(\tau\mathbf{A})\mathbf{v}$ to be smaller.

The adaptive Krylov algorithm is designed to compute the linear combination

$$\varphi_0(\mathbf{A})\mathbf{w}_0 + \varphi_1(\mathbf{A})\mathbf{w}_1 + \varphi_2(\mathbf{A})\mathbf{w}_2 + \cdots + \varphi_M(\mathbf{A})\mathbf{w}_M \qquad (16)$$

for some vectors $\mathbf{w}_i$, $i = 0, \ldots, M$. This is accomplished by using an auxiliary function

$$\mathbf{U}(\rho) = \varphi_0(\rho\mathbf{A})\mathbf{w}_0 + \rho\varphi_1(\rho\mathbf{A})\mathbf{w}_1 + \rho^2\varphi_2(\rho\mathbf{A})\mathbf{w}_2 + \cdots + \rho^M\varphi_M(\rho\mathbf{A})\mathbf{w}_M$$

and realizing that $\mathbf{U}(\rho)$ is the solution of the initial-value problem

$$\mathbf{U}'(\rho) = \mathbf{A}\mathbf{U}(\rho) + \mathbf{w}_1 + \rho\mathbf{w}_2 + \cdots + \frac{\rho^{M-1}}{(M-1)!}\mathbf{w}_M, \quad \mathbf{U}(0) = \mathbf{w}_0.$$

Evaluating $\mathbf{U}(\rho)$ over the interval $\rho \in [0, 1]$, i.e. computing $\mathbf{U}(\rho = 1)$, provides the desired value for (16). If the interval $[0, 1]$ is split into subintervals $0 = \rho_0 < \rho_1 < \cdots < \rho_k < \rho_{k+1} = \rho_k + \tau_k < \cdots < \rho_K = 1$, the solution $\mathbf{U}(\rho_{k+1})$ given a known value of $\mathbf{U}(\rho_k)$ can be calculated from

$$\mathbf{U}(\rho_{k+1}) = \varphi_0(\tau_k\mathbf{A})\mathbf{U}(\rho_k) + \sum_{i=1}^{M} \tau_k^i \varphi_i(\tau_k\mathbf{A}) \sum_{j=0}^{M-i} \frac{\rho_k^j}{j!}\mathbf{w}_{i+j}. \qquad (17)$$

The functions φ_k are connected via a recurrence relation $\varphi_q(\mathbf{A}) = \varphi_{q+1}(\mathbf{A})\mathbf{A} + \mathbf{I}/q!$, or more generally via

$$\varphi_q(\mathbf{A}) = \varphi_M(\mathbf{A})\mathbf{A}^{M-q} + \sum_{j=0}^{M-q-1} \frac{1}{(q+j)!}\mathbf{A}^j, \quad q = 0, 1, \ldots, M-1. \qquad (18)$$

Using Eq. (18) in Eq. (17) we can rewrite the explicit expression for $\mathbf{U}(\rho_{k+1})$ as

$$\mathbf{U}(\rho_{k+1}) = \tau_k^M \varphi_M(\tau_k\mathbf{A})\mathbf{u}_M + \sum_{j=0}^{M-1} \frac{\tau_k^j}{j!}\mathbf{u}_j, \qquad (19)$$

where the vectors $\mathbf{u}_j$ are given by

$$\mathbf{u}_j = \mathbf{A}^j\mathbf{U}(\rho_k) + \sum_{i=1}^{j} \mathbf{A}^{j-i} \sum_{l=0}^{j-i} \frac{\rho_k^l}{l!}\mathbf{w}_{i+l}, \quad j = 0, 1, \ldots, M.$$

Note that the computation of $\mathbf{U}(\rho_{k+1})$ using Eq. (19) requires only one evaluation of a matrix function-vector product $\varphi_M(\tau_k\mathbf{A})w_M$ and this evaluation involves a scaled matrix $\tau_k\mathbf{A}$ with $\tau_k < 1$. The adaptive Krylov algorithm uses estimators of the computational

cost and error to chose the substep sizes τ_k adaptively as described by Niesen and Wright [2012].

In short, the adaptive Krylov algorithm replaces one Krylov projection to evaluate $\varphi_j(\mathbf{A})\mathbf{w}_j$ which requires m iterations, with several Krylov projections to compute $\varphi_j(\tau_k\mathbf{A})\mathbf{w}_j$ each requiring m_k Krylov iterations. Practice shows that for most applications the total cost of the adaptive Krylov step $O(m_1^2) + \cdots + O(m_K^2)$ is, in fact, significantly smaller than the cost of one regular Krylov projection $O(m^2)$ since $m_k \ll m, k = 1, \ldots, K$.

The adaptive Krylov-friendly EPIRK methods are optimized to take advantage of the properties of the adaptive Krylov algorithm. Such EPIRK schemes are constructed in a way to reduce the total number of Krylov projections executed by the algorithm as well as to choose projections that require fewer Krylov vectors. We illustrate this point by using the following three-stage stiffly accurate fourth-order EPIRK4s3 method [Rainwater and Tokman 2016a] employed in our numerical examples:

$$\begin{aligned}
\mathbf{X}_{n2} &= \mathbf{X}_n + \frac{1}{8}\varphi_1\left(\frac{1}{8}h_n\mathbf{J}_n\right)h_n\mathbf{F}_n,\\
\mathbf{X}_{n3} &= \mathbf{X}_n + \frac{1}{9}\varphi_1\left(\frac{1}{9}h_n\mathbf{J}_n\right)h_n\mathbf{F}_n,\\
\mathbf{X}_{n+1} &= \mathbf{X}_n + \varphi_1\left(h_n\mathbf{J}_n\right)h_n\mathbf{F}_n\\
&\quad + \left(1892\varphi_3\left(h_n\mathbf{J}_n\right) - 42336\varphi_4\left(h_n\mathbf{J}_n\right)\right)h_n\mathbf{R}_n(\mathbf{X}_{n2})\\
&\quad + \left(1458\varphi_3\left(h_n\mathbf{J}_n\right) - 34992\varphi_4\left(h_n\mathbf{J}_n\right)\right)\\
&\qquad \cdot h_n\left(\mathbf{R}_n(\mathbf{X}_{n3}) - 2\mathbf{R}_n(\mathbf{X}_{n2})\right). \qquad (20)
\end{aligned}$$

We use the constant time step version of this scheme, i.e. $h_n = h$. We tested several exponential methods and determined that the scheme (20) provided the most efficient way to obtain a numerical solution of our elastodynamic models to the desired accuracy. The coefficients of scheme (20) are derived using stiff order conditions and an additional requirement to reduce the coefficients g_{ij}. As we discussed earlier smaller coefficients g_{ij} translate into a more clustered spectrum of the matrix $g_{ij}\mathbf{J}_n$ compared to $\mathbf{J}_n$. Such a clustering allows for a significant reduction of the required number of Krylov iterations to approximate the products of matrix functions and vectors.

While at first glance the scheme (20) appears to require calculation of five products of type $\varphi_k(\mathbf{A})\mathbf{v}$ for each time step, employing adaptive Krylov to obtain these approximations reduces the number of Krylov projections needed to only two. The first projection is needed to evaluate these products in the first and second stages, i.e. to compute both terms $\varphi_1(h_n/8\,\mathbf{J}_n)h_n\mathbf{F}_n$ and $\varphi_1(h_n/9\,\mathbf{J}_n)h_n\mathbf{F}_n$. Note that the optimization of the $g_{21} = 1/8$ and $g_{31} = 1/9$ coefficients implies that $\mathbf{A} = h_n/8\,\mathbf{J}_n$ which makes the spectrum of $\mathbf{A}$ more clustered compared to the spectrum of $h_n\mathbf{J}_n$. The adaptive Krylov substepping procedure has to be modified slightly to ensure that one of the substepping stops is $\rho_k = 1/9$ where the term $\varphi_1(h_n/9\,\mathbf{J}_n)h_n\mathbf{F}_n$ is computed and stored.

The second adaptive Krylov projection is used to compute the linear combination of all products of type $\varphi_k(\mathbf{A})\mathbf{v}$ in the last stage of the method. It was shown [Rainwater and Tokman 2016b] that a necessary condition for the stiff order conditions to hold is that all g-coefficients in the last stage of a method must be equal to 1. Thus it is not possible to derive a stiffly accurate three-stage EPIRK method with $g_{4i} < 1$.

3.6 Final Algorithm

Summarizing the ideas presented in the previous sections we now provide the outline of the overall stiffly accurate integration method in Alg. 1 for the initial value problem

$$\mathbf{x}''(t) + \bar{\mathbf{D}}\mathbf{x}'(t) + \mathbf{L}\mathbf{x}(t) = \mathbf{g}(\mathbf{x}(t)) \qquad (21)$$

with $\mathbf{x}(t_0) = \mathbf{x}_0$, $\mathbf{x}'(t_0) = \mathbf{v}_0$, and quantities $\mathbf{L}$ and $\mathbf{g}$ as specified in Eq. (2).

ALGORITHM 1: Stiffly accurate integration algorithm for the Cauchy problem $\mathbf{x}''(t) + \bar{\mathbf{D}}\mathbf{x}'(t) + \mathbf{L}\mathbf{x}(t) = \mathbf{g}(\mathbf{x}(t))$, $\mathbf{x}(t_0) = \mathbf{x}_0$, $\mathbf{x}'(t_0) = \mathbf{v}_0$.

- **Input:**
  - $\mathbf{L} \in \mathbb{R}^{N\times N}$; $\mathbf{g}(\mathbf{x})$, $\mathbf{x}_0$, $\mathbf{v}_0 \in \mathbb{R}^N$;
  - time interval $[t_0, t_{\text{final}}]$ and step size h.
- **Initialization:**
  1. Compute $\Omega = \sqrt{\mathbf{L}}$ according to Sec. 3.6.1.
  2. Set $\mathbf{X}_0 = \begin{bmatrix} \Omega\mathbf{x_0} \\ \mathbf{v}_0 \end{bmatrix}$.
  3. Compute $\mathbf{F}_0 = \mathbf{F}(\mathbf{X}_0) = \mathscr{A}\mathbf{X}_0 + \mathbf{G}(\mathbf{X}_0)$ as in Eq. (4) where $\mathscr{A}$ and $\mathbf{G}(\mathbf{X})$ are given by
  $$\mathscr{A} = \begin{bmatrix} \mathbf{0} & \Omega \\ -\Omega & \mathbf{0} \end{bmatrix}, \quad \mathbf{G}(\mathbf{X}) = \begin{bmatrix} \mathbf{0} \\ \mathbf{g}(\mathbf{x}) - \bar{\mathbf{D}}\mathbf{v} \end{bmatrix}.$$
  4. Set nts = round$((t_{\text{final}} - t_0)/h)$.
  5. $n = 0$.
- **Time stepping procedure:**
  (1) **For** $n = 0$ to (nts $- 1$) **do**
  (2) Compute $\mathbf{F}_n = \mathbf{F}(\mathbf{X}_n) = \mathscr{A}\mathbf{X}_n + \mathbf{G}(\mathbf{X}_n)$.
  (3) Use the adaptive Krylov algorithm to compute simultaneously terms $\mathbf{W}_1 = \varphi_1\left(\frac{1}{8}h\mathbf{J}_n\right)h\mathbf{F}_n$ and $\mathbf{W}_2 = \varphi_1\left(\frac{1}{9}h\mathbf{J}_n\right)h\mathbf{F}_n$.
  (4) Compute $\mathbf{X}_{n2} = \mathbf{X}_n + \frac{1}{8}\mathbf{W}_1$ and $\mathbf{X}_{n3} = \mathbf{X}_n + \frac{1}{9}\mathbf{W}_2$ as in Eq. (20).
  (5) Compute $\mathbf{R}_{n2} = \mathbf{R}_n(\mathbf{X}_{n2})$ and $\mathbf{R}_{n3} = \mathbf{R}_n(\mathbf{X}_{n3})$ using $\mathbf{R}_n(\mathbf{X}) = \mathbf{F}(\mathbf{X}) - \mathbf{F}(\mathbf{X}_n) - \mathbf{F}'(\mathbf{X}_n)(\mathbf{X} - \mathbf{X}_n)$.
  (6) Use a single call to the adaptive Krylov algorithm to compute the linear combination $\mathbf{W}$ of terms $\varphi_k(\mathbf{A})\mathbf{v}_k$ in the definition of $\mathbf{X}_{n+1}$ as given by Eq. (20).
  (7) Update the solution at the next time step $\mathbf{X}_{n+1} = \mathbf{X}_n + \mathbf{W}$.
  (8) Update $X_n := X_{n+1}$. $t_n := t_n + h$. $n := n + 1$.
  (9) **End for**
- **Output**
  (1) Compute positions $\mathbf{x}_{\text{nts}}$ by solving the linear system of equations $\Omega\mathbf{x}_{\text{nts}} = \mathbf{X}_{\text{nts}}(\mathbf{1} : \mathbf{N})$.
  (2) Set velocities to $\mathbf{x}'_{\text{nts}} = \mathbf{X}_{n+1}(N + 1 : 2N)$.

3.6.1 Matrix Square Root Computation. The choice of the method for computing a matrix square root $\Omega = \sqrt{\mathbf{L}}$ in Alg. 1 depends on the size of $\mathbf{L}$ and its properties. For example, when dealing with small or moderate systems we can use the Schur decomposition. The cost of this method for a symmetric matrix $\mathbf{L} \in \mathbb{R}^{N\times N}$ is $10N^3$ flops

[Higham 2008]. Therefore, the Schur method is not feasible if the system is large. In this case, to avoid the explicit precomputation of Ω, we suggest using iterative methods such as Newton square root iteration or its variants. The idea is to approximate the solution of the equation $\Omega^2 = \mathbf{L}$ by iteration as follows. First, suppose that $\Omega_\mathbf{k}$ is an approximation to Ω with small error $\mathcal{E}_k = \Omega - \Omega_\mathbf{k}$. We then obtain $\Omega = \Omega_\mathbf{k} + \mathcal{E}_\mathbf{k}$ and thus $(\Omega_\mathbf{k} + \mathcal{E}_\mathbf{k})^2 = \mathbf{L}$, which is equivalent to $\Omega_\mathbf{k}^2 + \Omega_\mathbf{k}\mathcal{E}_\mathbf{k} + \mathcal{E}_\mathbf{k}\Omega_\mathbf{k} + \mathcal{E}_\mathbf{k}^2 = \mathbf{L}$. Since $\mathcal{E}_k$ is small, one can neglect $\mathcal{E}_k^2$ and obtain the following iterative method:

1. given $\Omega_\mathbf{0}$ $(k = 0)$,
2. solve $\Omega_\mathbf{k}\mathcal{E}_\mathbf{k} + \mathcal{E}_\mathbf{k}\Omega_\mathbf{k} = \mathbf{L} - \Omega_\mathbf{k}^2$ for $\mathcal{E}_k$ $(k = 1, 2, \dots)$,
3. update $\Omega_{\mathbf{k+1}} = \Omega_\mathbf{k} + \mathcal{E}_\mathbf{k}$.

A well known result [Higham 2008] shows that if $\Omega_\mathbf{0}$ is chosen to commute with $\mathbf{L}$ then all the $\Omega_\mathbf{k}$ and $\mathcal{E}_k$ commute with $\mathbf{L}$ as well, which allows to reduce the cost significantly and to simplify the iteration scheme. In particular, the use of the common choice $\Omega_\mathbf{0} = \mathbf{L}$ results in the simplified iteration scheme

1. choose $\Omega_\mathbf{0} = \mathbf{L}$ $(k = 0)$,
2. update $\Omega_{\mathbf{k+1}} = \frac{1}{2}(\Omega_\mathbf{k} + \Omega_\mathbf{k}^{-1}\mathbf{L})$,

which offers unconditional quadratic convergence with cost $8/3N^3$ flops. Since Ω is symmetric and positive definite, its inversion can be done efficiently using a Cholesky decomposition $\Omega = \mathbf{S}^\mathsf{T}\mathbf{S}$ with an upper triangular matrix $\mathbf{S}$ with real and positive diagonal entries. The inverse matrix is then given by $\Omega = \mathbf{S}^{-1}\mathbf{S}^{-\mathsf{T}}$.

3.6.2 Damping. Since damping is of importance for several scenarios in visual computing, Alg. 1 already contains the necessary terms in order to handle damping caused by a term $\bar{\mathbf{D}} = \mathbf{M}^{-1}\mathbf{D}$ (with mass- and damping matrices $\mathbf{M}$ and $\mathbf{D}$ as specified in Eq. (1)) numerically.

3.6.3 Nonlinear Elasticity. For phenomena like nonlinear elasticity described by a varying matrix $\mathbf{L}(\mathbf{x}(t), t)$, Eq. (21) can be transformed into

$$\mathbf{x}''(t) + \bar{\mathbf{D}}\mathbf{x}'(t) + \mathbf{L}(\mathbf{x}_0, t_0)\,\mathbf{x}(t) = \tilde{\mathbf{g}}(\mathbf{x}(t)),$$

where $\tilde{\mathbf{g}}(\mathbf{x}(t)) = \mathbf{g}(\mathbf{x}(t)) + (\mathbf{L}(\mathbf{x}_0, t_0) - \mathbf{L}(\mathbf{x}(t), t))\,\mathbf{x}(t)$. That way, $\tilde{\mathbf{g}}$ does not only contain the contact forces, but rather the whole nonlinear part of the elastic forces. A major advantage of Alg. 1 is the exponential handling of the nonlinearity, so that stability is also ensured in such nonlinear cases.

3.6.4 Stiff Accuracy. To illustrate the importance of the stiff accuracy of our method we compare the stiffly accurate method EPIRK4s3 (20) with the Gautschi-type exponential integrator [Michels et al. 2014] and a classically fourth-order accurate two-stage exponential Rosenbrock method [Luan 2017]. The results are discussed in Sec. 4.1, in particular a precision diagram is shown in Fig. 3(a). It can clearly be observed that the stiffly accurate method is much less sensitive to increased stiffness of the problem and provides superior efficiency compared to other methods.

4 NUMERICAL SIMULATIONS

In this section, we present several numerical examples to study the behavior of our new stiffly accurate integrator described in Alg. 1. Our method is evaluated against a classical and a state-of-the-art method for stiff systems in visual computing , in particular against a backward Euler scheme computed with the Newton-Raphson procedure and the Gautschi-type exponential integrator described by Michels et al. [2014]. In order to improve the convergence of the classical Newton-Raphson procedure, we employ a line search scheme and additionally apply diagonal Hessian correction in case of indefinite matrices as described by Nocedal and Wright [2006]. The standard Krylov method described in Sec. 3.5 is used to evaluate the matrix functions required by the Gautschi-type exponential integrator. A tabular summary of the test cases that are used throughout this section and the computation times can be found in Tab. 1. All simulation times are measured on a desktop computer with an Intel(R) Xeon CPU clocked at 3.5GHz and 64 GB DDR-RAM.

#	Model	N	Duration	SAI	(h)	EI	(h)	NR	(h)
0a	Bunny (κ_0)	24K	3 min	126 s	(57.1 ms)	523 s	(17.2 ms)	1 h	(3.9 ms)
0b	Bunny (κ_1)	24K	3 min	127 s	(56.6 ms)	728 s	(12.3 ms)	1.5 h	(2.6 ms)
0c	Bunny (κ_2)	24K	3 min	127 s	(56.7 ms)	946 s	(9.5 ms)	2.3 h	(1.7 ms)
1a	Bunny (κ_3)	3K	3 min	8 s	(72.9 ms)	56 s	(40.5 ms)	7 min	(6.2 ms)
1b	Bunny (κ_3)	12K	3 min	58 s	(69.1 ms)	269 s	(33.4 ms)	43 min	(5.5 ms)
1c	Bunny (κ_3)	24K	3 min	127 s	(56.6 ms)	861 s	(10.4 ms)	2 h	(2.1 ms)
2	Flag	10.5K	6 s	1 s	(0.2 s)	8 s	(37.5 ms)	72 s	(6.6 ms)
3a	Eiffel Tower (k_1)	15K	12 s	3 s	(0.1 s)	26 s	(23.0 ms)	5 min	(3.2 ms)
3b	Eiffel Tower (k_2)	15K	12 s	3 s	(0.1 s)	33 s	(18.1 ms)	7 min	(2.2 ms)
4	Head Shake	6.48M	4 s	54 min	(49.3 ms)	31 h	(1.7 ms)	80 h	(13.1 ms)
5	Tooth Brushing	90K	3 min	16 min	(7.5 ms)	6 h	(0.4 ms)	18 h	(0.22 ms)
6	Bacteria	360	2 s	< 1 s	(0.1 s)	4 s	(25.4 ms)	23 s	(6.95 ms)

Table 1. The table provides an overview of the test cases used throughout this section. The number of degrees of freedom (N), the duration of the scene, and respective computation times for simulating the scene using the stiffly accurate integrator (SAI), the Gautschi-type exponential integrator (EI), and the Newton-Raphson procedure (NR) are shown. All measurements are for "equal quality" and a maximally tolerated error of 10% is enforced to ensure similar visual impressions. The time step values (h) are stated in brackets behind the computation times.

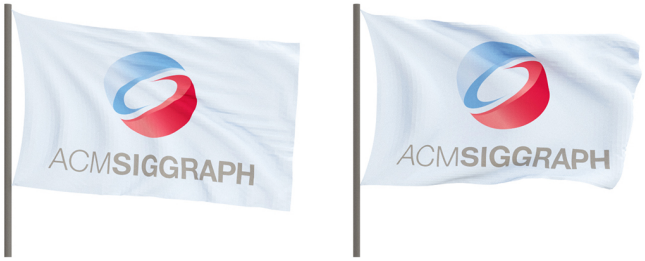

Fig. 2. Visualization of the simulation of two flags waving in the wind computed with the classical Newton-Raphson procedure (left) and the stiffly accurate integrator (right). While the flag on the left suffers from an uncontrollable artificial numerical dissipation, the flag on the right shows a plausible dynamical behavior.

4.1 Deformable Objects

As canonical examples, we set up undamped scenes of wobbling bunnies, which are composed of tetrahedra, in particular each bunny of 8 000 particles corresponding to $N = 24\,000$ degrees of freedom. The original geometry of the bunnies is taken from Stanford's scanning repository [2013], further processed, and its volume is tetrahedrized using the Delaunay-based tetrahedral mesh generator developed by Si [2015]. The underlying equations of motion are

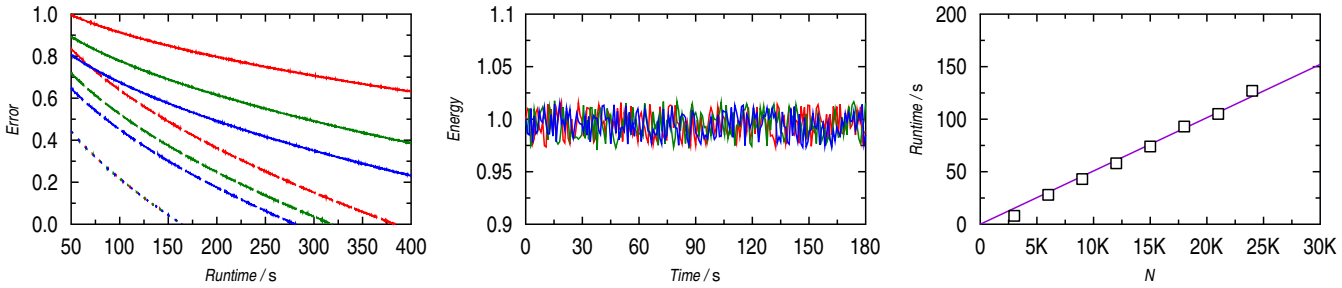

Fig. 3. (a) Precision diagram (left) for illustrating the relative L^2-error in the position and velocity space for the Gautschi-type exponential integrator (continuous curves), the exponential Rosenbrock integrator (dashed curves), and the stiffly accurate integrator (dotted curves). This is shown for different stiffnesses κ_0 (blue), κ_1 (green), and κ_2 (red). Please note, that it is hard to distinguish between the dotted curves since they almost overlap. (b) Energy diagram (middle) showing the temporal evolution of the discrete energy measured during the simulation of the wobbling bunnies with different stiffness ratios κ_0 (blue), κ_1 (green), and κ_2 (red) using the stiffly accurate integrator. (c) Runtime diagram (right) showing the dependence of the computation time on the number of degrees of freedom (N) for the simulation with the stiffly accurate integrator. Structural spring stiffnesses of $k_s = 10^2$ and diagonal spring stiffnesses of $k_d = 10^{11}$ are used leading to a stiffness ratio of $\kappa_3 = 10^9$.

derived from a classical system of coupled oscillators. The structural springs correspond to the edges of the tetrahedra. In order to prevent a volumetric collapse of the tetrahedra, strong diagonal springs connecting the vertices with the center points of the opposed triangles are added. The bunnies are exposed to an external force field pointing downwards in the vertical direction. We always set the stiffness constants of the structural springs equal to $k_s = 10^2$, whereas the stiffness constants of the diagonal springs $k_d \in \{10^8, 10^{10}, 10^{12}\}$ are varied throughout our experiments leading to different stiffness ratios $\kappa := k_d/k_s$, in particular to $\kappa_0 = 10^6$, $\kappa_1 = 10^8$, and $\kappa_2 = 10^{10}$. The results of our experiments are shown in Fig. 3. In particular, the precision diagram Fig. 3(a) shows the performance of our stiffly accurate integrator compared to the classically accurate two stage exponential Rosenbrock integrator [Luan 2017] and the Gautschi-type exponential integrator [Michels et al. 2014] for the simulation of wobbling bunnies over 3 min. Whereas the performance of the classical exponential Rosenbrock and the Gautschi-type integrator strongly decreases when the stiffness is increased, our stiffly accurate integrator is almost immune and does not show a significant decrease of its performance. This leads to dramatic speedups, especially when it comes to large stiffnesses as summarized in Tab. 1(0a-0c), which shows the computation times for the simulation of wobbling bunnies with different stiffnesses and a maximally tolerated relative L^2-error of 10% in the position and velocity space (although chosen arbitrarily ensuring a sufficient degree of visual accuracy) compared to a ground truth determined using the classical Runge-Kutta method (RK4) with sufficiently tiny time step sizes $h = 10^{-10}$ s. When it comes to high stiffness values, in particular $\kappa = 10^{10}$ here, our stiffly accurate integrator significantly speeds up the simulation of the wobbling bunny by a factor of over seven compared to the Gautschi-type exponential integrator, and by a factor of around 65 compared to the classical Newton-Raphson procedure.

We observe that the tips of the bunnies' ears return to their initial position periodically during the simulation which can be considered as an indicator for energy conservation. To confirm this assumption, the temporal evolutions of the total energies of the systems are measured and plotted in Fig. 3(b) for different stiffnesses. It can be observed that the discrete energy is oscillating around the real energy without increasing oscillations over time showing long-term stability of the stiffly accurate integrator. Moreover, the oscillations are not significantly increasing when the stiffness is increased, which demonstrates that the stiffly accurate integrator handles the numerical stiffness in a sophisticated way while preserving energy.

Finally, we carry out the simulation with different resolutions of the bunny geometry. The runtime is measured and illustrated in Fig. 3(c) for the stiffly accurate integrator. Moreover, corresponding runtimes for the Gautschi-type exponential integrator and the Newton-Raphson procedure can be found in Tab. 1(1a-1c). For our stiffly accurate integrator, we observe an almost linear trend.

4.2 Nonlinear Textiles

At least since the early work of Terzopoulos and Fleischer [1987], the simulation of cloth and textiles is within the scope of visual computing. Textiles can be modeled using a flat system of coupled oscillators connected with structural, diagonal (shear), and bending springs. The highly nonlinear load-deflection behavior of textiles could be better approximated using cubic springs instead of linear ones. According to that, we set up simulations of flags waving in the wind each composed of 50×70 particles (corresponding to $N = 10\,500$ degrees of freedom) connected with cubic springs. The structural spring stiffnesses are set up to $k_s = 10^4$, the diagonal spring stiffnesses to $k_d = 10^3$, and the bending spring stiffnesses to $k_d = 10^2$. According to the ground truth computed using the classical Runge-Kutta method (RK4) with sufficiently tiny time step sizes $h = 10^{-10}$ s, the total area of the textile (i.e. the sum of the areas of the triangles describing the flag's surface) is not varying by more than 2.5%. Tab. 1(2) shows the computation times for the simulations with a maximally tolerated error which ensures that the change of the area of the textile is below a 3%($\approx$ 2.5%) threshold. While the flag simulated with the stiffly accurate integrator shows a realistic dynamical behavior, the flag simulated with the Newton-Raphson procedure suffers from an uncontrollable artificial numerical dissipation, see Fig. 2. The stiffly accurate integrator

achieves a clearly better result while speeding up the simulation by a factor of eight compared to the Gautschi-type exponential integrator and by a factor of 72 compared to the Newton-Raphson procedure.

4.3 Co-rotational Deformations

The co-rotational formulation for finite elements was originally introduced by Wempner [1969]: geometrically nonlinear deformations of a continuum are decomposed into rigid rotations and contributions due to pure strain. The strain fraction typically remains small so that it can be treated by means of a linearized form of Green's strain tensor which is not invariant under rotation in general and is thus only valid for infinitesimal small deformations. The computationally costly application of the nonlinear form of Green's strain tensor for the simulation of large deformations is therefore elegantly circumvented. This concept has been widely adopted by the visual computing community where the simulation of large deformations is a major concern [Müller et al. 2002]. Although the approach presented by Müller et al. [2002] has well known shortcomings [Chao et al. 2010], we embed our stiffly accurate integrator in this formulation in order to demonstrate its applicability in such kind of setups as done by Michels et al. [2014] for the Gautschi-type exponential integrator. For that, we consider the positions $\mathbf{x} = \mathbf{X} - \mathbf{X}_0$ as displacements from the original positions $\mathbf{X}_0$. The matrix $\mathbf{K}$ in Eq. (1) is then replaced by

$$\mathbf{KX} \mapsto \mathbf{RK}(\mathbf{R}^{-1}\mathbf{X} - \mathbf{X}_0),$$

in which $\mathbf{R}$ denotes a sparse rotation matrix [Müller et al. 2002]. We set up the simulation of an elastic deformation of the Eiffel Tower with stiffness constants $k_0 = 10^8$ in the lower part. The stiffness constants in the upper part are set up to $k_1 = 10^2$ in a first simulation, and to $k_2 = 10^4$ in a second simulation. The girder network describing the topology and geometry of the Eiffel Tower contains 5 000 nodes ($N = 15\,000$). The runtimes needed by the different methods in order to obtain a L^2-error in position space which is less than 10% are shown in Tab. 1(3a-3b). The ground truth is computed using the classical Runge-Kutta method (RK4) with sufficiently tiny time step sizes $h = 10^{-10}$ s. The stiffly accurate integrator clearly shows an advantageous behavior over the Gautschi-type exponential integrator and the Newton-Raphson procedure. In contrast to the other methods, the runtime of the stiffly accurate integrator does not significantly increase when the stiffness is increased from k_1 to k_2.

4.4 Complex Stiff Systems

Densely packed fiber assemblies are canonical examples for complex and stiff systems. They are ubiquitous complex interacting mechanical systems like human hair, pillow infills made of wool, spaghetti, cable looms, the bristles of a tooth or bog brush, fur, algae carpets, or even tree roots. Consequently, their modeling and simulation has been of interest in the visual computing community; see e.g. the work of Kaufman et al. [2014] and references therein. In particular, the problem of the numerical simulation of human hair has gained special interest in the community. In this context, the Cosserat rod model was introduced to the community by Pai [2002], and later

Fig. 4. Visualization of a co-rotational elastic deformation of the Eiffel Tower simulated with the stiffly accurate integrator.

Bertails-Descoubes et al. [2006] presented the sophisticated Super-Helix approach for the simulation of fibers. Another line of work is based on systems of coupled oscillators. In this context, Selle et al. [2008] was the first who simulated the dynamics of a thinned out hair style with several thousand individual fibers. Although, such coupled oscillator models are rather popular because of their efficiency, it was complicated to produce physically accurate results since realistic material parameters are hard to integrate without running into numerical issues. This was recently addressed using a cuboidal fiber model [Michels et al. 2015].

We employ the approach of Michels et al. [2015] here, since it allows for the use of systems of coupled oscillators and realistic material parameters. We employ a Young's modulus of $3.6 \cdot 10^5\,\mathrm{Ncm}^{-2}$, a torsional modulus of $10^5\,\mathrm{Ncm}^{-2}$, and segment thicknesses between 0.04 mm (tip) and 0.09 mm (root), which are typical parameters for female caucasian hair [Robbins 2012]. The hairstyle contains 60 000 individual fibers, each composed of between six and ten cuboidal segments summing up to 480 000 cubes corresponding to 2 160 000 particles respectively $N = 6\,480\,000$ degrees of freedom. This results in a complex, interacting, and inherently stiff mechanical system. To avoid interpenetrations between the individual fibers among themselves and with the head geometry, correction impulses of appropriate magnitudes are applied [Bridson et al. 2002]. The system is dampened using linear Rayleigh damping [Liu and Gorman 1995]. The dynamical behavior of the hair during a head shake simulated with the stiffly accurate integrator is shown in Fig. 1. The corresponding computational times can be found in Tab. 1(4). Whereas the classical Newton-Raphson procedure requires around 80 h and the Gautschi-type exponential integrator around 6 h to simulate the hair dynamics of four seconds, the stiffly accurate integrator requires only 54 min in order compute a result of similar visual quality. Please note, that all these computation times include around 23 min for the collision detection, which is accelerated using a hierarchy of axis-aligned bounding boxes.

Similarly, in a second scene, we simulate a tooth brushing scenario of three minutes. The scene contains 1 500 bristles, each containing four cuboidal segment respectively 20 particles leading to $N = 90\,000$ degrees of freedom. 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

Fig. 5. Visualization of the simulation of tooth brushing carried out with the stiffly accurate integrator. The plaque removal efficacy is stained using the sequence of colors "blue (low), cyan, green, yellow, red (high)".

thicknesses of 0.12 mm. The result computed with the stiffly accurate integrator is shown in Fig. 5. The runtimes needed by the different methods in order to obtain a L^2-error in position space which is less than 5% are shown in Tab. 1(5). The ground truth is computed using the classical Runge-Kutta method (RK4) with sufficiently tiny time step sizes $h = 10^{-10}$ s. The stiffly accurate integrator clearly outperforms the Gautschi-type exponential integrator by a factor of over 22 and the Newton-Raphson procedure by a factor of almost 70.

4.5 Coupled Systems

Finally, we demonstrate the application of the our stiffly accurate integrator in the context of a coupled system. For that a flagellated microswimmer is set up by a fiber representing the centerline of the flagellum. The fiber is modeled as described in the previous experiments using a system of coupled oscillators. A constant torque perpendicular to the flagellum's base is applied to emulate the rotation of a motor. As described by Cortez et al. [2004], we couple the mechanical simulation of the flagellum with a fluid simulation. Since biologically microswimmers are usually located in highly viscous fluid environments, it is sufficient to model the fluid flow as Stokes flow, which can be solved analytically using the grid-free method of regularized Stokeslets [Cortez 2001]. A monotrichous bacteria swimming in a viscous fluid is simulated as illustrated in Fig. 6. The stiffly accurate integrator speeds up the simulation by a factor of over four compared to the Gautschi-type exponential integrator and by a factor of over 23 compared to the Newton-Raphson procedure, see Tab. 1(6). This example should be understood as a demonstration of the applicability of the stiffly accurate method in the context of two-way coupled fiber-fluid systems.

5 CONCLUSION

We have devised a new stiffly accurate integrator for the solution of stiff elastodynamic problems governed by the second-order ordinary differential equations of structural mechanics. This is based on a mathematical reformulation of the underlying differential equations, an exponential treatment of the full nonlinear forcing operator as opposed to more standard partially implicit or exponential approaches like Gautschi-type exponential integrators [Michels et al. 2014], and the utilization of the concept of stiff accuracy. As in the case of Gautschi-type exponential integrators, the presented method requires the evaluation of matrix functions, which is realized efficiently by employing adaptive Krylov subspace projections. The final method is highly competitive with respect to accuracy and computation times as demonstrated with significant accelerations on a broad spectrum of complex examples like deformable bodies, textiles, girders, bristles, and human hair. The method is easily parallelizable in order to exploit the power of modern massively parallel hardware. Moreover, its applicability was demonstrated in the context of co-rotational elasticity and two-way coupled fiber-fluid systems. Whereas the performance of past methods is highly dependent on the numerical stiffnesses of the underlying systems, the new integrator is much more robust with respect to stiffness increases. Moreover, these gains are not the result of nonphysical ad hoc manipulations of the underlying integration scheme. Instead, improved performance for highly stiff systems is an intrinsic property of the presented stiffly accurate integrator. Next to the relevance of the presented integrator for simulation scenarios where high accuracy and speed are mandatory, its strong robustness with respect to stiffness increases can be considered as one of its key features.

6 LIMITATIONS AND FUTURE WORK

Our work can be extended in various ways. A fine-grained parallel implementation on the GPU is likely to bring dramatically improved performance, since carefully hand-coded routines can significantly enhance speedup of such integrators. Although the presented stiffly accurate integrator preserves energy with high accuracy, it is not formally symplectic. The construction of efficient symplectic exponential integrators is an open question and further research is needed to determine if such methods can be built for problems relevant to visual computing and whether they will bring further improvements to the integration of these systems. We plan to investigate whether further efficiency improvements of the new stiffly accurate integrator are possible. A variable time step version of the algorithm can result in computational savings. It is possible that further optimizations of the coefficients of the stiffly accurate exponential integrators can yield more efficiency. In addition, since the development of algorithms for the efficient evaluation of matrix function and vector products is currently an active area of research, further efficiency gains could result from improvements to the adaptive Krylov algorithm portion of the method or from the utilization of novel algorithms for these computations.

A INHOMOGENEOUS MASS DISTRIBUTIONS AND NON-DIAGONALLY LUMPED MASS MATRICES

For the uniqueness of the square root $\sqrt{\mathbf{L}}$, the matrix $\mathbf{L} = \mathbf{M}^{-1}\mathbf{K}$ must be symmetric and positive semidefinite. However, although $\mathbf{M}$ is symmetric, an inhomogeneous mass distribution or a non-diagonally lumped mass matrix $\mathbf{M}$ can violate this condition.

Therefore, in such cases, we integrate according to Michels et al. [2014] instead of Eq. (2), the equivalent system

$$\hat{\mathbf{x}}''(t) + \hat{\mathbf{L}}\hat{\mathbf{x}}(t) = \hat{\mathbf{g}}(\hat{\mathbf{x}}(t)),$$

in which scaled coordinates, matrix, and nonlinearity are given by

$$\hat{\mathbf{x}}(t) = \sqrt{\mathbf{M}}\,\mathbf{x}(t),\ \hat{\mathbf{L}} = \sqrt{\mathbf{M}}^{-1}\mathbf{K}\sqrt{\mathbf{M}}^{-1},\ \hat{\mathbf{g}}(\hat{\mathbf{x}}) = \sqrt{\mathbf{M}}^{-1}\mathbf{M}\,\mathbf{g}\left(\sqrt{\mathbf{M}}^{-1}\hat{\mathbf{x}}\right).$$

B STABILITY AND CONVERGENCE ANALYSIS

In the following, we prove the stability of the stiffly accurate scheme (20). First, we rewrite the nonlinear system (5) in a more compact

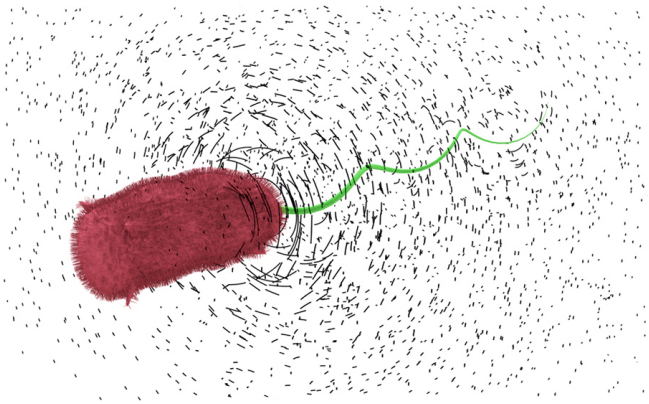

Fig. 6. Visualization of the simulation of a monotrichous bacteria swimming in a viscous fluid carried out with the stiffly accurate integrator. The rotation of the motor located at the back side of the bacteria's head causes the characteristic motion of the flagellum leading to a movement of the bacteria.

form

$$\mathbf{X}'(t) = \mathbf{F}(\mathbf{X}(t)) = \mathbf{J}_n\mathbf{X}(t) + \mathbf{N}_n(\mathbf{X}(t)) \tag{22}$$

with nonlinearity

$$\mathbf{N}_n(\mathbf{X}) = \mathbf{F}(\mathbf{X}) - \mathbf{J}_n\mathbf{X} = \mathbf{R}_n(\mathbf{X}) + \mathbf{F}(\mathbf{X}) - \mathbf{J}_n\mathbf{X}_n.$$

Using this form one can show that scheme (20) belongs to the class of three-stage parallel exponential Rosenbrock methods [Luan and Ostermann 2016], particularly when nodes $c_2 = 1/8$ and $c_3 = 1/9$ are used. We represent $\mathbf{X}_{n+1}$ as

$$\begin{aligned}\mathbf{X}_{n+1} = {} & \mathrm{e}^{h\mathbf{J}_n}\mathbf{X}_n + h\mathbf{b}_1(h\mathbf{J}_n)\mathbf{N}_n(\mathbf{X}_n) \\ & + h\mathbf{b}_2(h\mathbf{J}_n)\mathbf{N}_n(\mathbf{X}_{n2}) + h\mathbf{b}_3(h\mathbf{J}_n)\mathbf{N}_n(\mathbf{X}_{n3})\end{aligned} \tag{23}$$

where $\mathbf{b}_1(h\mathbf{J}_n) = \varphi_1(h\mathbf{J}_n)$, $\mathbf{b}_2(h\mathbf{J}_n) = 27648\varphi_4(h\mathbf{J}_n) - 1024\varphi_3(h\mathbf{J}_n)$, and $\mathbf{b}_3(h\mathbf{J}_n) = 1458\varphi_3(h\mathbf{J}_n) - 34992\varphi_4(h\mathbf{J}_n)$ by inserting

$$\mathbf{F}_n = \mathbf{J}_n\mathbf{X}_n + \mathbf{N}_n(\mathbf{X}_n)$$

into scheme (20). The convergence proof (implying the stability) of this scheme can be thus carried out in a similar way as done by Hochbruck et al. [2009] or Luan and Ostermann [2014b]. However, we now present a much shorter proof compared to these previous contributions.

Let $\mathbf{E}_{n+1} = \mathbf{X}_{n+1} - \mathbf{X}(t_{n+1})$ denote the global error, i.e. the difference between the numerical and the exact solution of Eq. (22) at time t_{n+1}. Let $\mathbf{e}_{n+1} = \widehat{\mathbf{X}}_{n+1} - \mathbf{X}(t_{n+1})$ denote the local error, i.e. the difference between the numerical solution $\widehat{\mathbf{X}}_{n+1}$ after one step with initial value being on the exact solution $\mathbf{X}(t_n)$ and the corresponding exact solution of Eq. (22) at time t_{n+1}. Since our method satisfies the stiff order conditions for methods of order four, $\mathbf{e}_{n+1} \in O(h^5)$.

Next, we note that the global error can be estimated by the sum of the propagated local errors $\mathbf{e}_k = \widehat{\mathbf{X}}_k - \mathbf{X}(t_k)$ $(k = 1, \ldots, n+1)$. Therefore, in order to show that the numerical scheme (20) is stable, the remaining task is to show the stability of the error propagation. The main difficulty here is that $\mathbf{J}_n$ changes from step to step.

Clearly, $\mathbf{E}_{n+1} = (\mathbf{X}_{n+1} - \widehat{\mathbf{X}}_{n+1}) + \mathbf{e}_{n+1}$. Taking a closer look at Eq. (23) one can consider $\mathbf{X}_{n+1}$ as a result of applying the discrete operator $D(h\mathbf{J}_n)$ to $\mathbf{X}_n$, where

$$D(h\mathbf{J}_n)(\mathbf{X}_n) = \mathrm{e}^{h\mathbf{J}_n}\mathbf{X}_n + h\sum_{i=1}^{3}\mathbf{b}_i(h\mathbf{J}_n)\mathbf{N}_n(\mathbf{X}_{ni})$$

(with $\mathbf{X}_{n1} = \mathbf{X}_n$) and thus

$$\mathbf{X}_{n+1} = D(h\mathbf{J}_n)(\mathbf{X}_n) = \prod_{j=0}^{n} D(h\mathbf{J}_{n-j})\mathbf{X}_0. \tag{24}$$

Similarly, we obtain

$$\begin{aligned}\widehat{\mathbf{X}}_{n+1} &= D(h\tilde{\mathbf{J}}_n)(\mathbf{X}(t_n)) \\ &= D(h\tilde{\mathbf{J}}_n)(\widehat{\mathbf{X}}_n) - D(h\tilde{\mathbf{J}}_n)(\mathbf{e}_n).\end{aligned} \tag{25}$$

Solving the recursion (25) and inserting the result and Eq. (24) into $\mathbf{E}_{n+1}$ gives

$$\begin{aligned}\mathbf{E}_{n+1} = {} & \prod_{j=0}^{n}\left(D(h\mathbf{J}_{n-j}) - D(h\tilde{\mathbf{J}}_{n-j})\right)\mathbf{X}_0 \\ & + \sum_{i=0}^{n}\prod_{j=0}^{n-i-1} D(h\tilde{\mathbf{J}}_{n-j})\mathbf{e}_{i+1}.\end{aligned}$$

Thus the stability of the error propagation requires the boundedness of $\prod_{j=0}^{n-i-1} D(h\tilde{\mathbf{J}}_{n-j})$. This can be shown using the stability bound for the discrete evolution operators given in Hochbruck et al. [2009],

$$\left\|\prod_{j=0}^{n-\nu}\mathrm{e}^{hJ_{n-j}}\right\| \le C_{\mathrm{S}}$$

for $t_0 \le t_\nu \le t_n \le t_{\mathrm{end}}$ with a constant C_{S} uniform in k and n despite the fact that $\mathbf{J}_n$ varies from step to step. Finally, it is clear that

$$\left\|\prod_{j=0}^{n}(D(h\mathbf{J}_{n-j}) - D(h\tilde{\mathbf{J}}_{n-j}))\mathbf{X}_0\right\| \le C\|\mathbf{E}_n\|.$$

Combining this, we obtain

$$\|\mathbf{E}_{n+1}\| \le C\|\mathbf{E}_n\| + \sum_{i=0}^{n} Ch^5.$$

Using the discrete Grönwall inequality, the error bound

$$\|\mathbf{E}_{n+1}\| \le Ch^4$$

with a constant C independent of n and h can be derived.

ACKNOWLEDGEMENTS

The authors are grateful to Franziska Lissel for helpful discussions and Stefan Feess for his kind help with the visualizations. Moreover, the reviewers' valuable comments that improved the manuscript are gratefully acknowledged.

REFERENCES

Awad H. Al-Mohy and Nicholas J. Higham. 2011. Computing the Action of the Matrix Exponential, with an Application to Exponential Integrators. *SIAM Journal on Scientific Computing* 33 (2011), 488–511.

Walter E. Arnoldi. 1951. The Principle of Minimized Iteration in the Solution of the Matrix Eigenvalue Problem. *Quarterly of Applied Mathematics* 9 (1951), 17–29.

Uri M. Ascher, Steven J. Ruuth, and Brian T.R. Wetton. 1995. Implicit-explicit Methods for Time-dependent Partial Differential Equations. *SIAM Journal on Numerical Analysis* 32, 3 (1995), 797–823.

David Baraff and Andrew Witkin. 1998. Large Steps in Cloth Simulation. In *Proceedings of SIGGRAPH 98*. Annual Conference Series, 43–54.

Luca Bergamaschi, Marco Caliari, and Marco Vianello. 2004. The ReLPM Exponential Integrator for FE Discretizations of Advection-Diffusion Equations. *International Conference on Computational Science* 3039 (2004), 434–442.

Miklós Bergou, Max Wardetzky, Stephen Robinson, Basile Audoly, and Eitan Grinspun. 2008. Discrete Elastic Rods. *ACM Transactions on Graphics* 27, 3 (2008), 63:1–63:12.

Florence Bertails, Basile Audoly, Marie-Paule Cani, Bernard Querleux, Frédéric Leroy, and Jean-Luc Lévêque. 2006. Super-helices for Predicting the Dynamics of Natural Hair. *ACM Transactions on Graphics* 25, 3 (2006), 1180–1187.

Robert Bridson, Ronald Fedkiw, and John Anderson. 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. *ACM Transactions on Graphics* 21, 3 (2002), 594–603.

Simone Buchholz, Ludwig Gauckler, Volker Grimm, Marlis Hochbruck, and Tobias Jahnke. 2017. Closing the Gap between Trigonometric Integrators and Splitting Methods for Highly Oscillatory Differential Equations. *SIAM Journal on Numerical Analysis* (2017), 1–18.

John C. Butcher. 2008. *Numerical Methods for Ordinary Differential Equations* (2nd ed.). Wiley.

Marco Caliari, Peter Kandolf, Alexander Ostermann, and Stefan Rainer. 2016. The Leja Method Revisited: Backward Error Analysis for the Matrix Exponential. *SIAM Journal on Scientific Computation* 38, 3 (2016), 1639–1661.

John Certaine. 1960. The Solution of Ordinary Differential Equations with Large Time Constants. In *Mathematical Methods for Digital Computers*. Wiley, 128–132.

Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. 2010. A Simple Geometric Model for Elastic Deformations. *ACM Transactions on Graphics* 29, 4 (2010), 38:1–38:6.

G. Chen and D. L. Russell. 1982. A Mathematical Model for Linear Elastic Systems with Structural Damping. *Quarterly of Applied Mathematics* 39, 4 (1982), 433–454.

Ricardo Cortez. 2001. The Method of Regularized Stokeslets. *SIAM Journal on Scientific Computing* 23, 4 (2001), 1204–1225.

Ricardo Cortez, Lisa Fauci, Nathaniel Cowen, and Robert Dillon. 2004. Simulation of Swimming Organisms: Coupling Internal Mechanics with External Fluid Dynamics. *Computing in Science and Engineering* 6, 3 (2004), 38–45.

Peter Deuflhard. 1979. A Study of Extrapolation Methods based on Multistep Schemes without Parasitic Solutions. *Journal of Applied Mathematics and Physics* 30 (1979), 177–189.

Bernd Eberhardt, Olaf Etzmuß, and Michael Hauth. 2000. Implicit-Explicit Schemes for Fast Animation with Particle Systems. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation*. 137–151.

Lukas Einkemmer, Mayya Tokman, and John Loffeld. 2017. On the Performance of Exponential Integrators for Problems in Magnetohydrodynamics. *Journal of Computational Physics* 330 (2017), 550–565.

Björn Engquist, Athanasios Fokas, Ernst Hairer, and Arieh Iserles. 2009. *Highly Oscillatory Problems*. Cambridge University Press.

Walter Gautschi. 1961. Numerical Integration of Ordinary Differential Equations based on Trigonometric Polynomials. *Numerische Mathematik* 3 (1961), 381–397.

Tanja Göckler and Volker Grimm. 2013. Convergence Analysis of an Extended Krylov Subspace Method for the Approximation of Operator Functions in Exponential Integration. *SIAM Journal on Numerical Analysis* 51, 4 (2013), 2189–2213.

Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. 2014. Efficient Simulation of Inextensible Cloth. *ACM Transactions on Graphics* 26, 3 (2014).

Ernst Hairer and Christian Lubich. 1999. Long-time Energy Conservation of Numerical Methods for Oscillatory Differential Equations. *SIAM Journal on Numerical Analysis* 38 (1999), 414–441.

Ernst Hairer, Syvert P. Nørsett, and Gerhard Wanner. 2004. *Solving Ordinary Differential Equations I: Nonstiff problems* (2nd ed.). Springer.

Ernst Hairer and Gerhard Wanner. 2004. *Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems* (2nd ed.). Springer.

Michael Hauth and Olaf Etzmuss. 2001. A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods. *Computer Graphics Forum* 20 (2001), 319–328.

Nicholas J. Higham. 2008. *Functions of Matrices: Theory and Computation*. SIAM.

Marlis Hochbruck and Alexander Ostermann. 2005. Explicit Exponential Runge–Kutta Methods for Semilinear Parabolic Problems. *SIAM Journal on Numerical Analysis* 43 (2005), 1069–1090.

Marlis Hochbruck and Alexander Ostermann. 2006. Exponential Integrators of Rosenbrock-type. *Oberwolfach Reports* 3 (2006), 1107–1110.

Marlis Hochbruck, Alexander Ostermann, and Julia Schweitzer. 2009. Exponential Rosenbrock-type Methods. *SIAM Journal on Numerical Analysis* 47 (2009), 786–803.

Aly-Khan Kassam and Lloyd N. Trefethen. 2005. Fourth-order Time Stepping for Stiff PDEs. *SIAM Journal on Scientific Computing* 26, 4 (2005), 1214–1233.

Danny M. Kaufman, Rasmus Tamstorf, Breannan Smith, Jean-Marie Aubry, and Eitan Grinspun. 2014. Adaptive Nonlinearity for Collisions in Complex Rod Assemblies. *ACM Transactions on Graphics* 33, 4 (2014), 123:1–123:12.

Stein Krogstad. 2005. Generalized Integrating Factor Methods for Stiff PDEs. *Journal of Computational Physics* 203 (2005), 72–88.

John D. Lawson. 1967. Generalized Runge–Kutta Processes for Stable Systems with Large Lipschitz Constants. *SIAM Journal on Numerical Analysis* 4, 3 (1967), 372–380.

Man Liu and Dadiv G. Gorman. 1995. Formulation of Rayleigh Damping and its Extensions. *Computers & Structures* 57, 2 (1995), 277–285.

John Loffeld and Mayya Tokman. 2013. Comparative Performance of Exponential, Implicit, and Explicit Integrators for Stiff Systems of ODEs. *Journal of Computational and Applied Mathematics* 241 (2013), 45–67.

Vu Thai Luan. 2017. Fourth-order Two-stage Explicit Exponential Integrators for Time-dependent PDEs. *Applied Numerical Mathematics* 112 (2017), 91–103.

Vu Thai Luan and Alexander Ostermann. 2013. Exponential B-series: The Stiff Case. *SIAM Journal on Numerical Analysis* 51 (2013), 3431–3445.

Vu Thai Luan and Alexander Ostermann. 2014a. Explicit Exponential Runge–Kutta Methods of High Order for Parabolic Problems. *Journal of Computational and Applied Mathematics* 256 (2014), 168–179.

Vu Thai Luan and Alexander Ostermann. 2014b. Exponential Rosenbrock Methods of Order Five – Construction, Analysis and Numerical Comparisons. *Journal of Computational and Applied Mathematics* 255 (2014), 417–431.

Vu Thai Luan and Alexander Ostermann. 2016. Parallel Exponential Rosenbrock Methods. *Computers & Mathematics with Applications* 71 (2016), 1137–1150.

Dominik L. Michels and Mathieu Desbrun. 2015. A Semi-analytical Approach to Molecular Dynamics. *Journal of Computational Physics* 303 (2015), 336–354.

Dominik L. Michels and J. Paul T. Mueller. 2016. Discrete Computational Mechanics for Stiff Phenomena. In *SIGGRAPH ASIA 2016 Courses*. 13:1–13:9.

Dominik L. Michels, J. Paul T. Mueller, and Gerrit A. Sobottka. 2015. A Physically Based Approach to the Accurate Simulation of Stiff Fibers and Stiff Fiber Meshes. *Computers & Graphics* 53B (2015), 136–146.

Dominik L. Michels, Gerrit A. Sobottka, and Andreas G. Weber. 2014. Exponential Integrators for Stiff Elastodynamic Problems. *ACM Transactions on Graphics* 33, 1 (2014), 7:1–7:20.

Cleve Moler and Charles Van Loan. 2003. Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-five Years Later. *SIAM Review* 45, 1 (2003), 3–49.

Matthias Müller, Julie Dorsey, Leonard McMillan, Robert Jagnow, and Barbara Cutler. 2002. Stable Real-Time Deformations. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 49–54.

Jitse Niesen and Will M. Wright. 2012. Algorithm 919: A Krylov Subspace Algorithm for Evaluating the Functions Appearing in Exponential Integrators. *ACM Transactions on Mathematical Software* 38, 3 (2012), 22:1–22:19.

Jorge Nocedal and Stephen J. Wright. 2006. *Numerical Optimization* (2nd ed.). Springer.

Dinesh K. Pai. 2002. STRANDS: Interactive Simulation of Thin Solids using Cosserat Models. *Comp. Graph. Forum* 21, 3 (2002), 347–352.

David A. Pope. 1963. An Exponential Method of Numerical Integration of Ordinary Differential Equations. *Communications of the ACM* 6, 8 (1963), 491–493.

Greg Rainwater and Mayya Tokman. 2014. A new Class of Split Exponential Propagation Iterative Methods of Runge-Kutta Type (sEPIRK) for Semilinear Systems of ODEs. *Journal of Computational Physics* 269 (2014), 40–60.

Greg Rainwater and Mayya Tokman. 2016a. Designing Efficient Exponential Integrators with EPIRK Framework. In *AIP Conference Proceedings of ICNAAM*.

Greg Rainwater and Mayya Tokman. 2016b. A new Approach to Constructing Efficient Stiffly Accurate EPIRK Methods. *Journal of Computational Physics* 323 (2016), 283–309.

Clarence R. Robbins. 2012. *Chemical and Physical Behavior of Human Hair* (5th ed.). Springer.

Andrew Selle, Michael Lentine, and Ronald Fedkiw. 2008. A Mass Spring Model for Hair Simulation. *ACM Transactions on Graphics* 27, 3 (2008), 64:1–64:11.

Hang Si. 2015. TetGen, a Delaunay-based Quality Tetrahedral Mesh Generator. *ACM Transactions on Mathematical Software* 41, 2 (2015), 11:1–11:36.

Stanford University. 2013. The Stanford 3D Scanning Repository. (2013).

Ari Stern and Mathieu Desbrun. 2006. Discrete Geometric Mechanics for Variational Time Integrators. In *SIGGRAPH 2006 Courses*. 75–80.

Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. 1987. Elastically Deformable Models. In *Computer Graphics*, Vol. 21. 205–214.

Mayya Tokman. 2006. Efficient Integration of Large Stiff Systems of ODEs with Exponential Propagation Iterative (EPI) Methods. *Journal of Computational Physics* 213 (2006), 748–776.

Mayya Tokman. 2011. A new Class of Exponential Propagation Iterative Methods of Runge–Kutta Type (EPIRK). *Journal of Computational Physics* 230 (2011), 8762–8778.

Mayya Tokman and Greg Rainwater. 2014. Four Classes of Exponential EPIRK Integrators. *Oberwolfach Reports* 14 (2014), 855–858.

Henk A. van der Vorst. 1987. An Iterative Solution Method for Solving $f(A)x = b$, using Krylov Subspace Information obtained for the Symmetric Positive Definite Matrix A. *Journal of Computational and Applied Mathematics* 18, 2 (1987), 249–263.

Gerald A. Wempner. 1969. Finite Elements, Finite Rotations and Small Strains of Flexible Shells. *International Journal of Solids and Structures* 5, 2 (1969), 117–153.